

Oscar Castillo
Janusz Kacprzyk
Witold Pedrycz (Eds.)

Soft Computing for Intelligent Control and Mobile Robotics

Oscar Castillo, Janusz Kacprzyk, and Witold Pedrycz (Eds.)

Soft Computing for Intelligent Control and Mobile Robotics

Studies in Computational Intelligence, Volume 318

Editor-in-Chief

Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw
Poland
E-mail: kacprzyk@ibspan.waw.pl

Further volumes of this series can be found on our homepage: springer.com

Vol. 294. Manuel F.M. Barros, Jorge M.C. Guilherme, and Nuno C.G. Horta
Analog Circuits and Systems Optimization based on Evolutionary Computation Techniques, 2010
ISBN 978-3-642-12345-0

Vol. 295. Roger Lee (Ed.)
Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2010
ISBN 978-3-642-13264-3

Vol. 296. Roger Lee (Ed.)
Software Engineering Research, Management and Applications, 2010
ISBN 978-3-642-13272-8

Vol. 297. Tania Tronco (Ed.)
New Network Architectures, 2010
ISBN 978-3-642-13246-9

Vol. 298. Adam Wierzbicki
Trust and Fairness in Open, Distributed Systems, 2010
ISBN 978-3-642-13450-0

Vol. 299. Vassil Sgurev, Mincho Hadjiski, and Janusz Kacprzyk (Eds.)
Intelligent Systems: From Theory to Practice, 2010
ISBN 978-3-642-13427-2

Vol. 300. Baoding Liu (Ed.)
Uncertainty Theory, 2010
ISBN 978-3-642-13958-1

Vol. 301. Giuliano Armano, Marco de Gemmis, Giovanni Semeraro, and Eloisa Vargiu (Eds.)
Intelligent Information Access, 2010
ISBN 978-3-642-13999-4

Vol. 302. Bijaya Ketan Panigrahi, Ajith Abraham, and Swagatam Das (Eds.)
Computational Intelligence in Power Engineering, 2010
ISBN 978-3-642-14012-9

Vol. 303. Joachim Diederich, Cengiz Gunay, and James M. Hogan
Recruitment Learning, 2010
ISBN 978-3-642-14027-3

Vol. 304. Anthony Finn and Lakhmi C. Jain (Eds.)
Innovations in Defence Support Systems, 2010
ISBN 978-3-642-14083-9

Vol. 305. Stefania Montani and Lakhmi C. Jain (Eds.)
Successful Case-Based Reasoning Applications-1, 2010
ISBN 978-3-642-14077-8

Vol. 306. Tru Hoang Cao
Conceptual Graphs and Fuzzy Logic, 2010
ISBN 978-3-642-14086-0

Vol. 307. Anupam Shukla, Ritu Tiwari, and Rahul Kala
Towards Hybrid and Adaptive Computing, 2010
ISBN 978-3-642-14362-4

Vol. 308. Roger Nkambou, Jacqueline Bourdeau, and Riichiro Mizoguchi (Eds.)
Advances in Intelligent Tutoring Systems, 2010
ISBN 978-3-642-14362-5

Vol. 309. Isabelle Bichindaritz, Lakhmi C. Jain, Sachin Vaidya, and Ashlesha Jain (Eds.)
Computational Intelligence in Healthcare 4, 2010
ISBN 978-3-642-14463-9

Vol. 310. Dipti Srinivasan and Lakhmi C. Jain (Eds.)
Innovations in Multi-Agent Systems and Applications - 1, 2010
ISBN 978-3-642-14434-9

Vol. 311. Juan D. Velásquez and Lakhmi C. Jain (Eds.)
Advanced Techniques in Web Intelligence, 2010
ISBN 978-3-642-14460-8

Vol. 312. Patricia Melin, Janusz Kacprzyk, and Witold Pedrycz (Eds.)
Soft Computing for Recognition based on Biometrics, 2010
ISBN 978-3-642-15110-1

Vol. 313. Imre J. Rudas, János Fodor, and Janusz Kacprzyk (Eds.)
Computational Intelligence in Engineering, 2010
ISBN 978-3-642-15219-1

Vol. 314. Lorenzo Magnani, Walter Carnielli, and Claudio Pizzi (Eds.)
Model-Based Reasoning in Science and Technology, 2010
ISBN 978-3-642-15222-1

Vol. 315. Mohammad Essaadi, Michele Malgeri, and Costin Badica (Eds.)
Intelligent Distributed Computing IV, 2010
ISBN 978-3-642-15210-8

Vol. 316. Philipp Wolfrum
Information Routing, Correspondence Finding, and Object Recognition in the Brain, 2010
ISBN 978-3-642-15253-5

Vol. 317. xxxx

Vol. 318. Oscar Castillo, Janusz Kacprzyk, and Witold Pedrycz (Eds.)
Soft Computing for Intelligent Control and Mobile Robotics, 2010
ISBN 978-3-642-15533-8

Oscar Castillo, Janusz Kacprzyk, and
Witold Pedrycz (Eds.)

Soft Computing for Intelligent Control and Mobile Robotics

 Springer

Prof. Oscar Castillo
Tijuana Institute of Technology
Department of Computer Science,
Tijuana, Mexico
Mailing Address
P.O. Box 4207
Chula Vista CA 91909, USA
E-mail: ocastillo@tectijuana.mx

Prof. Witold Pedrycz
University of Alberta,
Dept. Electrical and
Computer Engineering
Edmonton, Alberta T6J 2V4,
Canada
E-mail: pedrycz@ece.ualberta.ca

Prof. Janusz Kacprzyk
Polish Academy of Sciences,
Systems Research Institute,
Newelska 601-447 Warszawa Poland
E-mail: kacprzyk@ibspan.waw.pl

ISBN 978-3-642-15533-8

e-ISBN 978-3-642-15534-5

DOI 10.1007/978-3-642-15534-5

Studies in Computational Intelligence

ISSN 1860-949X

Library of Congress Control Number: 2010934864

© 2010 Springer-Verlag Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typeset & Cover Design: Scientific Publishing Services Pvt. Ltd., Chennai, India.

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

We describe in this book, hybrid intelligent systems using soft computing techniques for intelligent control and mobile robotics. Soft Computing (SC) consists of several intelligent computing paradigms, including fuzzy logic, neural networks, and bio-inspired optimization algorithms, which can be used to produce powerful hybrid intelligent systems. The book is organized in five main parts, which contain a group of papers around a similar subject. The first part consists of papers with the main theme of theory and algorithms, which are basically papers that propose new models and concepts, which can be the basis for achieving intelligent control and mobile robotics. The second part contains papers with the main theme of intelligent control, which are basically papers using bio-inspired techniques, like evolutionary algorithms and neural networks, for achieving intelligent control of non-linear plants. The third part contains papers with the theme of optimization of fuzzy controllers, which basically consider the application of bio-inspired optimization methods to automate the design process of optimal type-1 and type-2 fuzzy controllers. The fourth part contains papers that deal with the application of SC techniques in times series prediction and intelligent agents. The fifth part contains papers with the theme of computer vision and robotics, which are papers considering soft computing methods for applications related to vision and robotics.

In the part of theory and algorithms there are 5 papers that describe different contributions that propose new models and concepts, which can be the considered as the basis for achieving intelligent control and mobile robotics. The first paper, by Ramon Zatarain et al., deals with applying intelligent systems for modeling students' learning styles used for mobile and web-based systems. The second paper, by Luis Martinez et al., deals with a fuzzy model for RAMSET: Role Assignment Methodology for Software Engineering Teams. The third paper, by Jorge Soria-Alcaraz et al., describes an academic timetabling design using hyper-heuristics. The fourth paper, by Alberto Ochoa et al., describes a logistics optimization service improved with artificial intelligence. The fifth paper, by Francisco Arce and Mario Garcia-Valdez, describes an accelerometer-based hand gesture recognition system using artificial neural networks.

In the part of intelligent control there are 5 papers that describe different contributions on achieving control using hybrid intelligent systems based on soft computing techniques. The first paper, by Ieroham Baruch et al., describes a direct and indirect neural identification and control of a continuous bioprocess via Marquardt learning. The second paper, by Eduardo Gomez-Ramirez et al., deals with a method for simple tuning of type-2 fuzzy controllers. The third paper, by Leocundo Aguilar et al., proposes an increasing energy efficiency of a preamble sampling MAC protocol for wireless sensor networks using a fuzzy logic

approach. The fourth paper, by Arnulfo Alanis et al., describes a multi-agent system based on psychological models for mobile robots. The fifth paper, by Fevrier Valdez et al., proposes the use of fuzzy logic to control parameters in bio-inspired optimization methods.

In the part of optimization of fuzzy controllers there are 5 papers that describe different contributions of new algorithms for optimization and their application to designing optimal fuzzy logic controllers. The first paper by Ricardo Martinez et al., describes the optimization of type-2 fuzzy logic controllers using PSO applied to linear plants. The second paper, by Yazmin Maldonado et al., deals with an approach for the optimization of membership functions for an incremental fuzzy PD control based on genetic algorithms. The third paper, by Leticia Cervantes and Oscar Castillo, describes a new method for the design of a fuzzy system for the longitudinal control of an F-14 airplane. The fourth paper by Abraham Melendez et al., describes a fuzzy reactive controller of a mobile robot. The fifth paper, by Arnulfo Alanis et al., describes a multi-agent system with personality profiles and preferences and learning for autonomous mobile robot, with fuzzy logic support.

In the part of time series prediction and intelligent agents several contributions are described on the development of new models and algorithms relevant to time series analysis and forecasting, as well as the application of intelligent agents in real-world applications. The first paper, by Pilar Gomez et al., describes composite recurrent neural networks for long-term prediction of highly-dynamic time series supported by wavelet decomposition. The second paper, by Juan R. Castro et al., describes an interval type-2 fuzzy neural network for chaotic time series prediction with cross-validation and the Akaike test. The third paper, by Jesus Soto et al., deals with chaotic time series prediction using Ensembles of ANFIS. The fourth paper, by Lucila Morales et al., describes the modeling of facial expression of intelligent virtual agents. The fifth paper, by Ivan Espinoza et al., describes agent communication using semantic networks. The sixth paper, by Cecilia Leal-Ramirez et al., describes a fuzzy cellular model for predator-prey interaction applied to the control of plagues in a peppers cropping.

In the part of computer vision and robotics several contributions on models and algorithms are presented, as well as their applications to different real-world problems. The first paper, by Rogelio Salinas-Gutierrez et al., describes the use of Gaussian copulas in supervised probabilistic classification. The second paper, by Pablo Rivas et al., proposes subjective co-localization analysis with fuzzy predicates. The third paper, by Jesus David Teran et al., describes an iterated local search algorithm for the linear ordering problem with cumulative costs. The fourth paper, by Nohe Cazarez et al., describes an observer for the type-1 fuzzy control of a servomechanism with backlash using only motor measurements. The fifth paper, by Selene Cardenas et al., proposes a neuro-fuzzy based output feedback controller design for biped robot walking. The sixth paper, by Oscar Montiel et al., describes a fuzzy system to control the movement of a wheeled mobile robot. The seventh paper, by Oscar Montiel et al., proposes an approach for embedding a fuzzy locomotion pose controller for a wheeled mobile robot into an FPGA.

In conclusion, the edited book comprises papers on diverse aspects of bio-inspired models, soft computing and hybrid intelligent systems for control and mobile robotics. There are theoretical aspects as well as application papers.

May 31, 2010

Oscar Castillo
Tijuana Institute of Technology, Mexico

Witold Pedrycz
University of Alberta, Canada

Janusz Kacprzyk
Polish Academy of Sciences, Poland

Contents

Part I: Theory and Algorithms

Applying Intelligent Systems for Modeling Students' Learning Styles Used for Mobile and Web-Based Systems	3
<i>Ramón Zatarain, Lucía Barrón-Estrada, Carlos Alberto Reyes-García, Orion Fausto Reyes-Galaviz</i>	
Towards a Fuzzy Model for RAMSET: Role Assignment Methodology for Software Engineering Teams	23
<i>Luis G. Martínez, Juan R. Castro, Guillermo Licea, Antonio Rodríguez-Díaz, Carlos Álvarez</i>	
Academic Timetabling Design Using Hyper-Heuristics	43
<i>Soria-Alcaraz Jorge A., Carpio-Valadez J. Martín, Terashima-Marin Hugo</i>	
Logistics Optimization Service Improved with Artificial Intelligence	57
<i>Alberto Ochoa, Yazmani Garcia, Javier Yañez</i>	
Accelerometer-Based Hand Gesture Recognition Using Artificial Neural Networks	67
<i>Francisco Arce, José Mario García Valdez</i>	

Part II: Intelligent Control

Direct and Indirect Neural Identification and Control of a Continuous Bioprocess via Marquardt Learning	81
<i>Ieroham Baruch, Carlos-Roman Mariaca-Gaspar, Josefina Barrera-Cortes, Oscar Castillo</i>	
Simple Tuning of Type-2 Fuzzy Controllers	103
<i>Eduardo Gómez-Ramírez, Patricia Melin, Oscar Castillo</i>	

Increasing Energy Efficiency of a Preamble Sampling MAC Protocol for Wireless Sensor Networks Using a Fuzzy Logic Approach	125
<i>Leocundo Aguilar, Oscar Castillo, J. Antonio García-Macías, Guillermo Licea</i>	
Multi-Agent System Based on Psychological Models for Mobile Robots	143
<i>Arnulfo Alanis Garza, Oscar Castillo, José Mario García Valdez</i>	
Fuzzy Control for Dynamical Parameter Adaptation in a Parallel Evolutionary Method Combining Particle Swarm Optimization and Genetic Algorithms	161
<i>Fevrier Valdez, Patricia Melin, Oscar Castillo</i>	
Part III: Optimization of Fuzzy Controllers	
Optimization of Type-2 Fuzzy Logic Controllers Using PSO Applied to Linear Plants	181
<i>Ricardo Martínez, Oscar Castillo, Luis T. Aguilar, Antonio Rodríguez</i>	
Optimization of Membership Functions for an Incremental Fuzzy PD Control Based on Genetic Algorithms	195
<i>Yazmín Maldonado, Oscar Castillo, Patricia Melin</i>	
Design of a Fuzzy System for the Longitudinal Control of an F-14 Airplane	213
<i>Leticia Cervantes, Oscar Castillo</i>	
A Fuzzy Reactive Controller of a Mobile Robot	225
<i>Abraham Meléndez, Oscar Castillo, Arnulfo Alanis Garza</i>	
Multi-Agent System with Personality Profiles and Preferences and Learning for Autonomous Mobile Robot with Fuzzy Logic Support	233
<i>Arnulfo Alanis Garza, Oscar Castillo, José Mario García Valdez</i>	
Part IV: Time Series Prediction and Intelligent Agents	
Composite Recurrent Neural Networks for Long-Term Prediction of Highly-Dynamic Time Series Supported by Wavelet Decomposition	253
<i>Pilar Gomez-Gil, Angel Garcia-Pedrero, Juan Manuel Ramirez-Cortes</i>	

An Interval Type-2 Fuzzy Neural Network for Chaotic Time Series Prediction with Cross-Validation and Akaike Test	269
<i>Juan R. Castro, Oscar Castillo, Patricia Melin, Olivia Mendoza, Antonio Rodríguez-Díaz</i>	
Chaotic Time Series Prediction Using Ensembles of ANFIS	287
<i>Jesus Soto, Oscar Castillo, José Soria</i>	
Modeling Facial Expression of Intelligent Virtual Agents	303
<i>María Lucila Morales-Rodríguez, Fabián Medellín-Martínez, Juan J. González B.</i>	
Agent Communication Using Semantic Networks	315
<i>Iván Espinoza-Hernández, Dora-Luz Flores, Antonio Rodríguez-Díaz, Manuel Castañón-Puga, Carelia Gaxiola</i>	
Fuzzy Cellular Model for Predator-Prey Interaction Applied to the Control of Plagues in a Peppers Cropping	329
<i>Cecilia Leal-Ramírez, Oscar Castillo, Antonio Rodríguez-Díaz</i>	
 Part V: Vision and Robotics	
Using Gaussian Copulas in Supervised Probabilistic Classification	355
<i>Rogelio Salinas-Gutiérrez, Arturo Hernández-Aguirre, Mariano J.J. Rivera-Meraz, Enrique R. Villa-Diharce</i>	
Subjective Colocalization Analysis with Fuzzy Predicates	373
<i>Pablo Rivas-Perea, Jose Gerardo Rosiles, Wei Qian</i>	
Iterated Local Search Algorithm for the Linear Ordering Problem with Cumulative Costs (LOPCC)	395
<i>Jesús David Terán Villanueva, Héctor Joaquín Fraire Huacuja, Rodolfo Pazos Rangel, Juan Martín Carpio Valadez, Héctor José Puga Soberanes, Juan Javier González Barbosa</i>	
An Observer for the Type-1 Fuzzy Control of a Servomechanism with Backlash Using Only Motor Measurements	405
<i>Nohe R. Cazarez-Castro, Luis T. Aguilar, Oscar Castillo</i>	
Neuro-Fuzzy Based Output Feedback Controller Design for Biped Robot Walking	423
<i>Selene L. Cardenas-Maciel, Oscar Castillo, Luis T. Aguilar</i>	

Fuzzy System to Control the Movement of a Wheeled Mobile Robot	445
<i>Oscar Montiel Ross, Jesús Camacho, Roberto Sepúlveda, Oscar Castillo</i>	
Embedding a Fuzzy Locomotion Pose Controller for a Wheeled Mobile Robot into an FPGA	465
<i>Oscar Montiel, Jesús Camacho, Roberto Sepúlveda, Oscar Castillo</i>	
Author Index	483

Applying Intelligent Systems for Modeling Students' Learning Styles Used for Mobile and Web-Based Systems

Ramón Zatarain¹, Lucía Barrón-Estrada¹, Carlos Alberto Reyes-García²,
and Orion Fausto Reyes-Galaviz³

¹ Instituto Tecnológico de Culiacán, México

² Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), México,

³ Universidad Autónoma de Tlaxcala

{rzatarain, lbarron, vponce, rcabada}@itculiacan.edu.mx,
kargaxxi@inaoep.mx, orionfrg@yahoo.com

Abstract. The identification of the best learning style in an Intelligent Tutoring System must be considered essential as part of the success in the teaching process. This research work presents a set of three different approaches applying intelligent systems for automatic identification of learning styles in order to provide an adapted learning scheme under different software platforms. The first approach uses a neuro-fuzzy network (NFN) to select the best learning style. The second approach combines a NFN to classify learning styles with a genetic algorithm for weight optimization. The learning styles are based on Gardner's Pedagogical Model of Multiple Intelligences. The last approach implements a self-organising feature map (SOM) for identifying learning styles under the Felder-Silverman Model. The three approaches are used by an author tool for building Intelligent Tutoring Systems running under a Web 2.0 collaborative learning platform. The tutoring systems together with the neural networks can also be exported to mobile devices. We present results of three different tutoring systems produced by three implemented authoring tools.

Keywords: Intelligent Tutoring System, Web 2.0, Authoring Tool, M-Learning.

1 Introduction

In an e-learning environment, teachers can design various alternatives of content for each possible configuration of learning style [1]. The characteristics of e-learning allow teachers to work collaboratively with students, and even with other teachers interested in designing learning materials. Thus, responsibility for the design of teaching material does not lie with one person, but on a community of designers [2]. The time devoted to teaching every student is not a problem in electronic learning environments. Each student can have access to materials virtually

at the time and place he wants [3]. An implementation of e-learning, in which materials are selected according to student's learning style, can be defined in the context of an Intelligent Tutoring System (ITS) or in an Adaptive Hypermedia System. An ITS refers to any computer system that provides a personalized educational process directly to the student. The ITS works automatically and requires no intervention by the teacher (or student) to perform the customization of the learning material. To identify the learning style of students there are several solutions. One solution is the application of questionnaires designed by teachers specialized in the field. Another more accurate solution is to use artificial intelligence techniques for carrying out this process automatically.

This work addresses the above problems through design and construction of a neural network, which after training, automatically and dynamically identify student learning styles. This neural network provides the means for customizing learning materials. The neural network can be used in any e-learning or m-learning environment, whether for identification of learning style in the context of an Intelligent Tutoring System, or manually with the student consulting its own learning styles.

The arrangement of the paper is as follows: Section 2 gives a general structure of the tool. Section 3 presents the neural network and predictive engine used in the tool. Results are shown in Section 4. Discussions and conclusions are given in Section 5.

2 Learning Style Models

Models of learning styles categorized both the ways in which students learn and how teachers teach. Its main objective is that in each category holding the model, the learning needs of students are met [4]. Taking into account the majority of existing models concerning learning styles, there are five major families [5]. Figure 1 shows the first four classes of models, which relate to theories of learning styles, the fifth family, which is omitted in the figure, contains models that deviate from the concept of learning styles and propose other theories.

The model of learning styles of Gregorc was presented by Anthony Gregorc and Kathleen Butler. The objective of the model is to provide an organized structure of how the mind works. In this model there are four learning styles: Concrete Sequential, Abstract Random, Abstract Sequential and Concrete Random. Howard Gardner maintains the idea that every student is equipped with different types of intelligence. Gardner's theory is known as Multiple Intelligences, and provides that each individual has different intelligences at different levels. Gardner also states that every person has a unique cognitive profile. According to multiple intelligence theory, there are nine basic types of intelligence: Visual-spatial, Verbal-linguistic, Logical-mathematical, Bodily-kinesthetic, Musical-rhythmic, Interpersonal, Intrapersonal, Naturalistic, and Existential. The Myers-Briggs model is known as the Type Indicator Myers-Briggs (MBTI for short). The MBTI identifies four scales in which conforms to all subjects. The scales are extraversion / introversion, sensitive / intuitive, thinking / feeling, and judgment / perception. Of all the combinations of the scales, we obtain 16 types of personalities. Each of

the combinations of types is described with four letters (one for each scale). The Felder-Silverman model was proposed by Richard Felder and Linda Silverman in 1988 [6]. The model includes four dimensions or categories, two of which replicate features found in the models of Myers-Briggs and Kolb. The four dimensions in the model are related to perception (sensory / intuitive), processing (active / reflective), input presentation (visual / verbal) and understanding (sequential / global). Learning styles are obtained by the combination of all categories. Thus, it is possible to have 16 different learning styles.

<p>Gregorc Bartlett Betts Dunn Dunn Gordon Marks Paivio Richardson Sheehan Torrance</p>	<p>Gardner Broverman Cooper Guilford Hulzman y Hudson Hunt Kagan Kogan Messick Pettigrew</p>	<p>Myers-Briggs Apter Epstein y Meier Harrison-Branson Jackson Miller</p>	<p>Felder-Silverman Herrmann Kolb Allison y Hayes Honey y Mumford Kaufmann Kirtan McCarthy</p>
--	---	--	---

Fig. 1. Main Learning Style Models

3 First Approach: Modeling Learning Styles with a NFN

The first approach to identify students' learning styles was using a neuro-fuzzy network. In this section, we present **MLTutor**, an authoring tool which can be used to build personalized or intelligent tutoring courses to be used in both learning settings: distance and mobile learning. The tool can be employed for developing learning material using SCORM [7] Learning Objects or other standard file formats. The output of the authoring tool will be either learning material for mobile devices or SCORM learning objects for e-learning environments. The learning material for mobile tools uses a neuro-fuzzy mechanism to identify and to predict learning styles in order to provide an adapted learning scheme. The learning styles are based on Gardner's Pedagogical Model of Multiple Intelligences [8].

3.1 MLTutor Architecture

Figure 2 presents the architecture of **MLTutor**. As shown in Figure 2, the tool has two main editors: the **content editor** and the **fuzzy set editor**. An author creates a tutoring system by first building a course structure using the content editor. This structure consists of a number of units where a unit can be linked with learning material and some assessments. A course is created by importing already prepared learning material in different standard formats like **html**, **pdf**, **doc** or **SCORM** learning objects from any type of source. The author can also introduce learning material by using a simple editor included in the tool. Other important learning materials the authors insert into each one of the units are quizzes. These can be in every part of each

section. The quiz is essential for the dynamic courseware generation because from the test results, the neural networks classify learning styles.

When the author introduces the learning material he/she creates four different instances corresponding to four different student learning styles (types) according to Gardner's Pedagogical Model of Multiple Intelligences: Logical/Mathematical, Verbal/Linguistic, Visual/Spatial and Musical/Rhythmic. There is a special interface for helping the author when building this material.

The fuzzy set editor helps the user to define **Fuzzy Membership Functions**. For **Fuzzy Inputs** there are seven linguistic variables defined for a user. They are: *answer selection order*, *correct answers*, *quiz spent time*, *topic spent time*, *number of tries until correct answer*, *number of visits to a question*, and *number of visits to a topic*. Each of the seven linguistic variables allows three different values: low, average, and high. A Fuzzy inference process is performed mapping input linguistic values to output multiple-intelligence styles. The output of the fuzzy set editor can be m-learning material in **XML format** along with a predictive engine that employs a **neuro-fuzzy inference algorithm** [9]. It operates online using present and former information for each individual learner. At the start of each learning unit, predictions are made as to what the learners preferred learning material is, based on former information built in the fuzzy set edition phase. Afterward, the neuro-fuzzy algorithm will learn from present information and will make adjustments, if necessary, to another best suited learning style. Another option to the output of the fuzzy editor is to export the learning material to SCORM format. The benefit of this format is the availability of the material in any distance learning environment. When a mobile course is exported to a mobile device, a **XML interpreter** is added to the course. This interpreter has the job of displaying the material of the course into the mobile device, according to some chosen learning style.

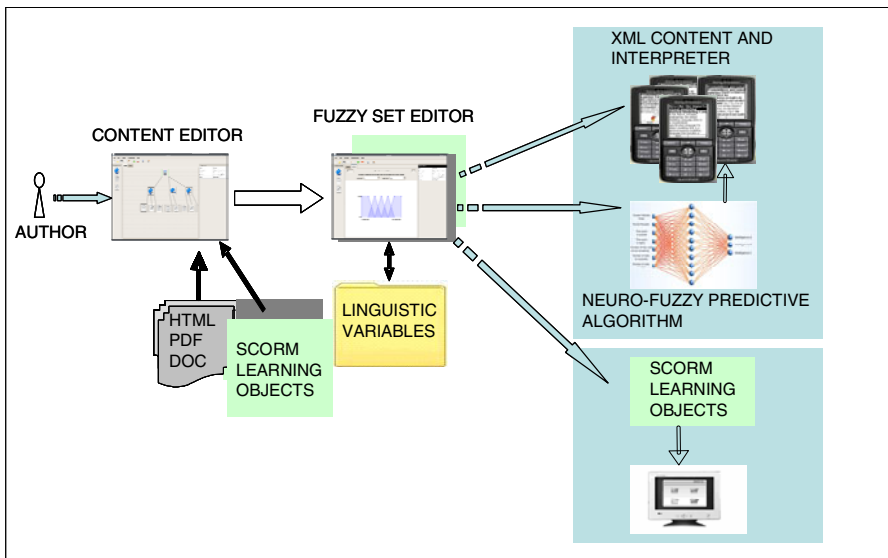


Fig. 2. MLTutor Architecture

3.2 Neuro-fuzzy Predictive Model

Figure 3 shows part of the MLTutor Neuro-Fuzzy system (just two linguistic variables are shown) implemented in order to represent knowledge (seven linguistic variables, four multiple intelligences, and inference rules), to learn from former and current data, and to make adjustment to new learning styles.

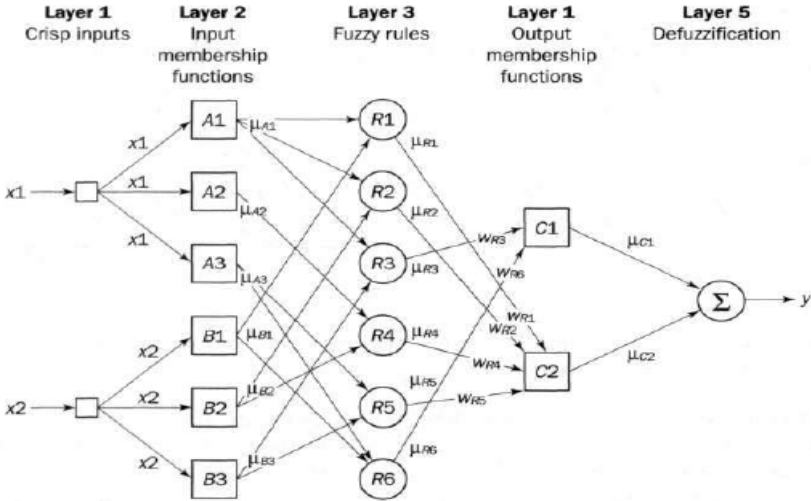


Fig. 3. Part of the HLTutor Neuro-Fuzzy System

As we can observe in Figure 3, the system configuration consists of input and output layers, and three hidden layers that represent membership functions and fuzzy rules. The complete input layer has seven neurons representing our seven linguistic variables. Every linguistic variable has three fuzzy sets (*low, average, or high sets*). The input layer sends out external crisp values directly to the next layer. The output of this layer is as follows:

$$y_i = x_i ,$$

where x_i and y_i are input and output respectively.

Each neuron of the input layer is connected to three neurons on layer 2 (fuzzyfication layer). In this layer crisp values from input layer are transformed to appropriate linguistic fuzzy sets (*low, average, or high sets*). Every neuron of layer 2 represents a fuzzy set for each one of the seven linguistic variables (see table 1). The output of the layer is the degrees of membership of each input value to each fuzzy set. Every neuron of Layer 3 or fuzzy rule layer represents a fuzzy rule (R1, R2, R3, etc.). Every fuzzy rule neuron takes inputs from layer 2 neurons. The rule is evaluated by the fuzzy intersection or t-norm, which in this case is the product operation. The output of each neuron is described as:

$$y_i = x_{1i} \times x_{2i} \times \dots \times x_{ki}$$

The output of layer 3 represents the weights of each one of the rules. The weights connecting layer 3 and layer 4 are changed or adjusted by training the neural network. The weight values are normalized on many adjustments, by dividing each weight into the greatest weight found on each one of the adjustments or iterations and is represented by

$$w_{ni}(p + 1) = w_i(p) / w_{\max}(p),$$

where w_{ni} is the normalized weight, p is the last iteration, w_i is the weight of connection i , and w_{\max} is the greatest weight of the iteration.

Table 1. Fuzzy sets for two Linguistic Variables

TopicsSpentTime	CorrectAnswers
Range=[0 1000]	Range=[0 10]
NumMFs=5	NumMFs=5
MF1='Low':trapmf,[-225-25 100 300]	MF1='Low':trapmf,[-2.25 -0.25 1 3]
MF2='Medium-Low':trimf,[100 300 500]	MF2='Medium-Low':trimf,[1 3 5]
MF3='Medium':trimf,[300 500 700]	MF3='Medium':trimf,[3 5 7]
MF4='Medium-High':trimf,[500 700 900]	MF4='Medium-High':trimf,[5 7 9]
MF5='High':trapmf,[700 900 1025 1225]	MF5='High':trapmf,[7 9 10.2 12.2]

Layer 4 or output membership layer takes inputs from the fuzzy rule neurons and merges them by using fuzzy **union**, which in our case is the algebraic sum and is defined as

$$x_{1i} \oplus x_{2i} \oplus \dots \oplus x_{ji}$$

The outputs of layer 4 are the different learning styles (Logical/Mathematical, Verbal/Linguistic, Visual/Spatial and Musical/Rhythmic) produced in the fuzzy rule layer.

Layer 5 or defuzzification layer is the output of our system. In this layer the **sum-product composition** method is used [10]. The output of this last layer is a recommended learning style for the student. The learning algorithm we used is **back-propagation**. Our algorithm takes a desired output (a default learning style), which is computed at the beginning of the iterations. Next, the algorithm computes the actual output and compares it with the desired output (which is dynamically adjusted according with the student test results). If there is a difference between the actual and the desired output, the error is calculated and propagated backward through the network, updating or adjusting the weights between connected neurons. The neural network was trained too using Matlab (version 7.1). Then, the trained neural network was implemented using Java along with the XML interpreter.

3.3 Tests and Results

We conducted a test of our tool by working with a group of different kinds of users (authors), mainly university professors, and college students. They produced personalized or intelligent tutoring courses for English Language (Elementary

school), Object-Oriented Analysis and Design, and a Basic Math Course. The courses dynamically adapt the content or style of instruction presented to the learner. Figure 4 shows some interfaces (snapshots) of the authoring tool when building the Basic Math course. As we can observe, one image shows the course tree structure (left-top); two images show the contents of the course under the Verbal/Linguistic type of Intelligence (right-top and left-bottom); another one illustrates the fuzzy set edition (right-bottom); and the last one (front image) illustrates a quiz edition.

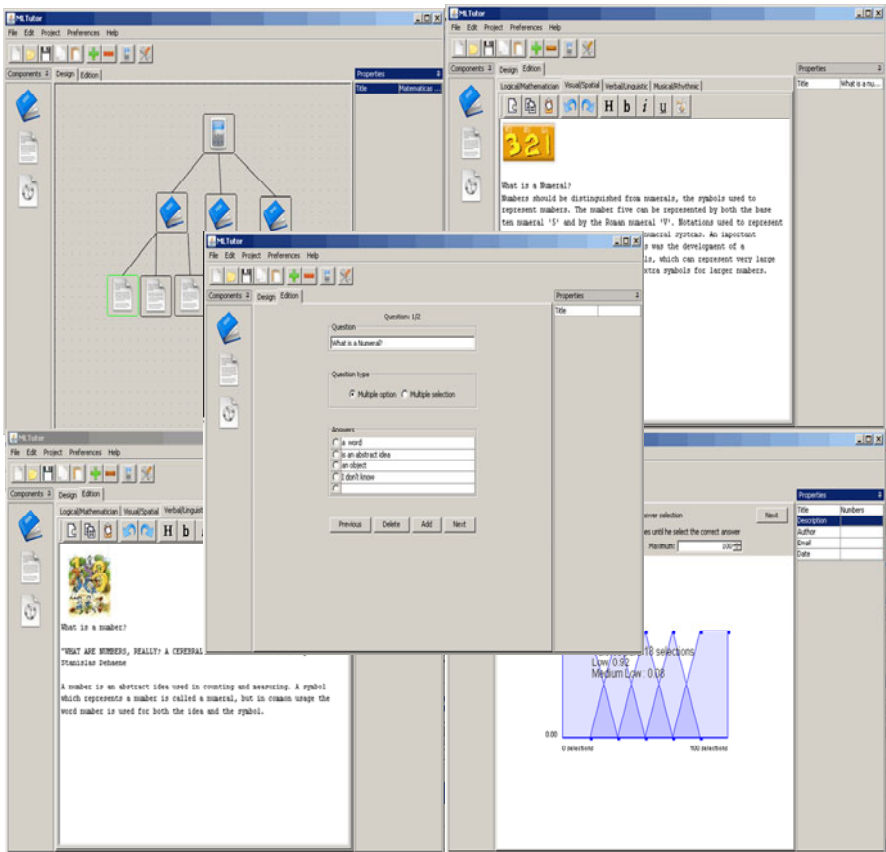


Fig. 4. Interfaces of MLTutor and a Math Course

Figure 5 presents some instances of the Math course running in a cell Phone (NHAL Win32 Emulator). The four mobile instances correspond (left-to-right, top-to-bottom) to *Three-Chapter menu*, *Chapter-1 menu (contents and quiz)*, *chapter-1 contents*, and *chapter-1 quiz*.



Fig. 5. The Math Course in a Cell Phone

4 Second Approach: A NFN and a Genetic Algorithm

The second approach to identify students' learning styles was using a neuro-fuzzy network together with a Genetic Algorithm. In this section, we present **EDUCA**, a Web 2.0 software tool to allow a community of authors and learners to create, share, and view learning materials and web resources in an adaptive environment which combines collaborative, mobile and e-learning methods. **EDUCA** applies different artificial intelligence techniques like a neural network and a genetic algorithm for selecting the best learning style or a recommendation-web mining system for adding and searching new learning resources.

4.1 EDUCA Architecture

Figure 6 illustrates the overall architecture of EDUCA. As we can observe, there are two authors: the main **tutor** (a teacher or instructor) and the **community of learners**. The student or learner is an important author of the course and

participate actively adding learning resources to the courses. The learner has a user profile with information like the GPA, particular learning style, or recommended resources to the course. When the authors add learning material, they first create four different instances corresponding to four different learning styles according to Felder-Silverman Learner Style Model [6].

We implemented a **fuzzy-neural network** using the fuzzy input values previously defined. The output of the network is the learning style for each student using a course. We also implemented a **genetic algorithm** for the optimization of the weights used in the network. The network was trained for 800 generations using a population of 150 chromosomes. In order to train the network, we created three set of courses for high school students. Each course was presented in four different teaching styles according to the Felder-Silverman model. When a mobile course is exported to a mobile device, a XML **interpreter** is added to the course. A SCORM file for the course can also be exported.

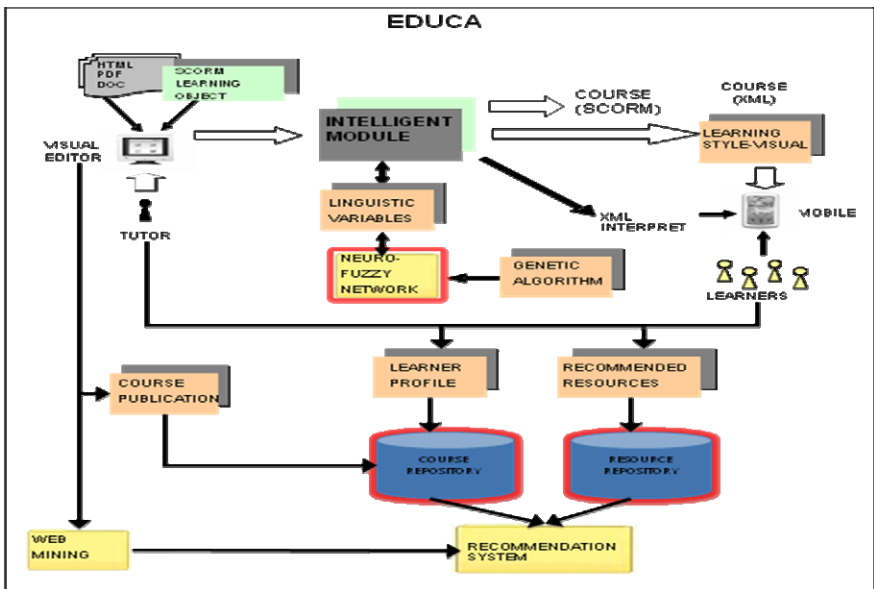


Fig. 6. EDUCA Architecture

Once a course is created, a Course Publication Module saves it into a Course Repository. Whenever a learner accesses a course, a recommender system implemented in EDUCA presents links or Web sites with learning material related to the current topic. Such material is stored in a resource repository of EDUCA, which was searched previously by using Web mining techniques implemented also in EDUCA.

4.2 The Intelligent Module

The Intelligent Module takes as an input the learning material for four different learning styles. Then, it creates a NFN used to classify the learning style of the

user/student and produces as an output, an adaptive course (a special type of Intelligent Tutoring System). An adaptive course is structured with two components: A XML file (it contains the learning material), and the XML Interpreter. The Interpreter uses the NFN as a dynamic classifier to show the learning material according to the best learning style of the user.

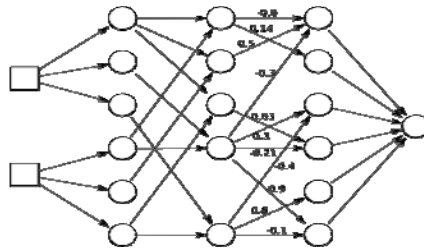
The first layer of the NFN has 7 neurons (see figure 3) representing the 7 linguistic variables used in the classification of the learning style. Every neuron of layer 1 is connected to 3 neurons of layer 2 (*fuzzyfication layer*). Due to the fact that we are using triangular membership functions, the activation function for the layer 2 neurons is as follow:

$$\begin{aligned} y_i^{(1)} &= 0, \text{ if } x_i^{(1)} \leq a - b / 2 \\ y_i^{(1)} &= 1 - 2 |x_i^{(1)} - a| / b, \text{ if } a - b / 2 < x_i^{(1)} < a + b / 2 \\ y_i^{(1)} &= 0, \text{ if } x_i^{(1)} \geq a + b / 2 \end{aligned}$$

where a and b are the centre and width of the triangle, $x_i^{(1)}$ and $y_i^{(1)}$ are input and output of neuron i respectively.

The output of layer 3 represents the strength of each one of the fuzzy rules. The best weight values between layer 3 and layer 4 are calculated using a genetic algorithm. Layer 5 is the output of the NFN. We applied a *Centroid* technique to make *defuzzyfication*. The value of the output is the learning style for the current student of the course.

Training the NFN with a Genetic Algorithm



At the beginning, we create a chromosome population with random values. Each decoded element of the population, represent the weights in the NFN (see figure 7).

-0.6	0.5	-0.3	0.14	0.3	-0.4	0.63	-0.21	0.8	0.9	-0.1
------	-----	------	------	-----	------	------	-------	-----	-----	------

Fig. 7. Decoding the weights in the NFN into a chromosome

For encoding the chromosome, the weights of the NFN are sorted using a Bucket Sort Algorithm, ordering first by layer, and then by neurons. The output of this algorithm is a chromosome. Each gene of the chromosome represents a weighted link in the NFN.

To evaluate the chromosome's performance, we assign each weight (gene) contained in the chromosome to the links of a respective NFN. Then we test the

network with a set of training values. Last, the sum of squared errors is evaluated, which will be used as the fitness value of the tested chromosome.

In order to train the network, we create three set of courses for high school students. Each course was presented in eight different styles for Felder-Silverman model. The chosen courses were *Teaching Digital Photography*, *Eolic Energy*, and *Introduction to Computers*. In order to get their best learning style we applied *Felder-Silverman Index of Learning Style Questionnaire* to every student. We also designed an exam to evaluate the course taken by the student. The input data to the network was the performance of the student under each course and the learning style of the taken course (randomly chosen). The Desired Output was the learning style calculated from Felder-Silverman Questionnaire. We made tests with two groups of 40 students.

4.3 Using the NFN on Cell Phones

Whenever a course is created and exported to a cell phone or PDA (using XML format), the NFN is also exported to those devices, along with a XML parser or interpreter. This program run any course stored in the mobile by reading the XML file where the course is stored. At the beginning, a learning style is assigned to the student or user of the course. Then, depending of the results from questions prepared to the students inside the learning objects, the learning material is adapted to the best learning style (Visual, Verbal, Sequential, etc.) of the student (the NFN is consulted in order to find the best appropriated learning style to the student).



Fig. 8. Dynamic sequencing of Learning Material

An important feature is the possibility of tracing the different learning styles that a user or student follow during lesson learning. The interpreter of the course optionally displays the current student learning styles plus the values of fuzzy variables. This information is relevant for doing different types of analysis with

respect to behaviors of the students and the way they learn. Figure 8 shows two snapshots of a handheld device (a cell phone) showing values of each learning style in a range between -100 to +100 (left device) and a dynamic sequence of the style behavior during the execution of the course (right device).

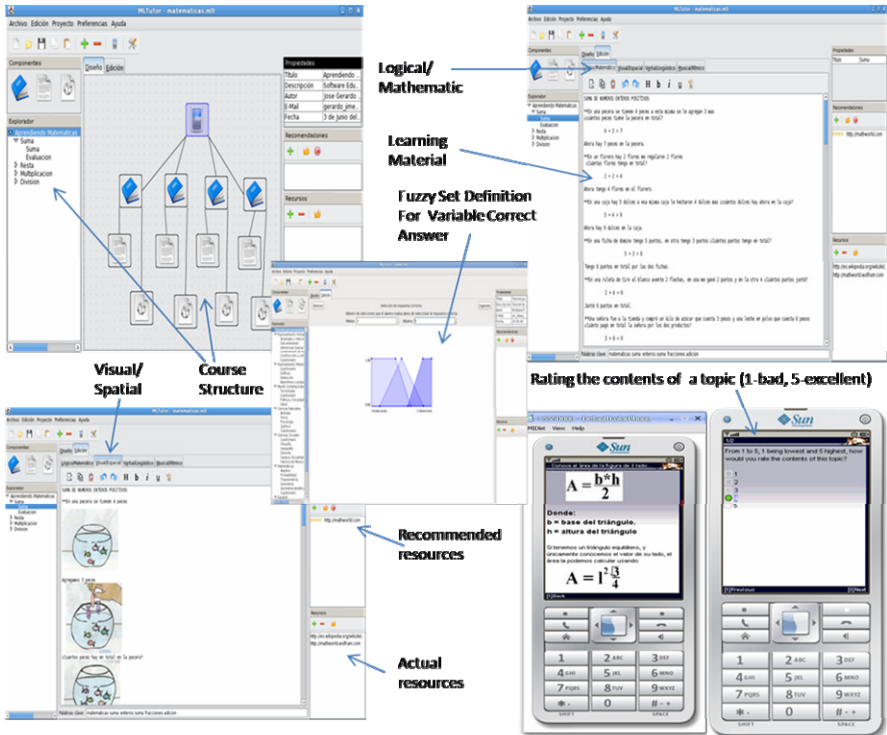


Fig. 9. Creation of Learning Material for a Basic Math course

4.4 Test and Results

We tested the tool with 15 professors/teachers and their respective students of different teaching levels. They developed different kinds of courses like a GNU/Linux course, a Basic Math Operation course, and learning material for preparation to the University Admission test EXANI-II. Each one of the courses had from 4 to 10 units of learning. They included evaluations on each unit. The students participated by reading, evaluating and adding material (Web resources) to the courses. We applied a questionnaire or survey to instructors and students in order to evaluate the effectiveness of EDUCA. More than 90% of them agree that they would like to use the tool for teaching their courses. This point is very important because the tool was developed to be used by instructors of any level of teaching (from elementary to college level).

Next, we present an example of how an author creates/updates learning material for a **Basic Math course** (figure 9). We first create the structure of the course (left-top). Then, we add learning material for each learning style (right-top and left-bottom). In this stage, we also assign fuzzy set values to each linguistic variable, and use recommended and actual resources for inclusion in the course. Last, we export and display the course (right-bottom).

5 Third Approach: A Self-Organizing Map

In this section, we present a new version of the authoring tool **EDUCA** using a self-organizing map or Kohonen network [11] to identify students' learning styles. Advantages of Kohonen networks include implementation simplicity, execution speed and a shorter training process; however maybe the most important advantage of these unsupervised neural networks is that they do not require an external teacher for presenting a training set. During a training session, the SOM receives a number of different input patterns (the student learning style obtained from the ILSQ, the course learning style, and the student's grade in the course), discovers significant features in these patterns (Felder-Silverman learning styles) and learns how to classify input.

5.1 New EDUCA Architecture

Figure 10 presents the new architecture of **EDUCA**. The most important modification in this new version is the SOM which substitutes the NFN used in the first version. The functionalities in this new authoring tool are very similar for the user with only a new editor for the creation of learning material. After the author creates a course or intelligent tutoring system, she/he can save it and export it to a Mobile Learning format used to display tutoring systems on mobile devices. The saved/exported file will enclose three elements: a XML file corresponding to the learning style model, a predictive engine for navigation purposes, and the Kohonen Neural Network for learning style classification.

Once a Tutoring System has been created, the module **Published Course** stores it in a Course Repository. We know that learners usually read tutoring systems stored in some kind of repository, but they also consult other learning resources in different web sites. For example a student who needs to implement a LR parser in a compiler course, could use the Tutoring System stored in the course repository, but she/he could also consult extra resources in the Web. This material found by the student would be rated and recommended to be added to the regular compiler course. **EDUCA** uses a Hybrid Recommendation System [12] which stores new resources into a Resource Repository. Last, **EDUCA** uses a Data or Text Mining Subsystem used to search resources in the Web.

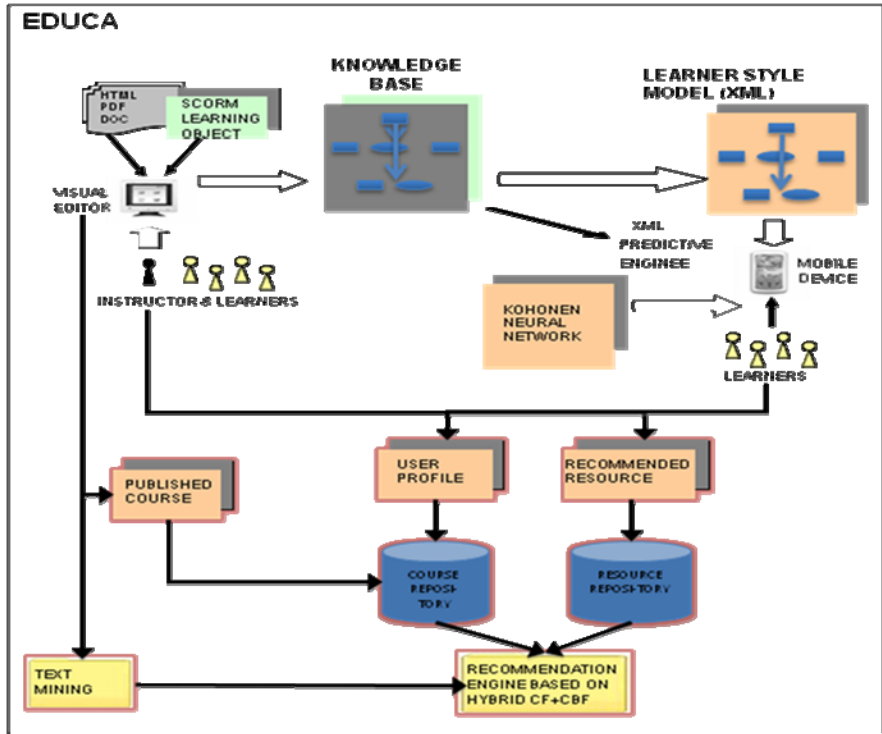


Fig. 10. New Architecture of EDITA

5.2 Training the SOM

An input vector in our neural network is described as:

$$D = [d_{fs} \ d_a \ p]$$

where D is the input vector, and it is formed by three elements: d_{fs} which is the student learning style identified by applying the Felder-Silverman ILSQ questionnaire; d_a which is the learning style used in the learning material read by the student; and p is the student grade obtained in the test when the student has learning style d_{fs} and the course was offered using learning style d_a .

Vectors d_{fs} and d_a are also composed as:

$$d_a = d_{fs} = [c_1 \ c_2 \ c_3]$$

where $c_1 \ c_2 \ c_3$ represents three scores for Perception, Input, and Understanding Dimension in the ILSQ questionnaire.

Figure 11 shows part of the Kohonen network architecture. We can see that the input layer has seven neurons, corresponding to three-dimensional vectors d_{fs} and d_a ,

and the student grade p . Vector's input data vary between -11 to +11, which is massaged to the range -1 to +1 with the equation:

$$\text{Massaged value}_{i1} = v_i / |v_{\text{imax}}|$$

where v_i is a ILSQ score, and v_{imax} is the maximum ILSQ score value (+11). As we observe in figure 11, -3, an ILSQ score for visual/verbal learning style, is mapped to -0.272. On the other hand, the student grade, which varies between 0 and 100, is also massaged to the range -1 to +1, with equation:

$$v_{nj} = v_j \times 2 / v_{j\text{max}} - 1$$

where v_j is the student grade or score, and $v_{j\text{max}}$ is the maximum student grade (+100). For example, 70 is mapped to 0.4.

The Kohonen layer has 1600 neurons arranged in a structure of a 40x40 hexagonal grid. The output of the network consists of three signals or values ranging between -1 to +1. These data are then decoded to the ILSQ range (-11 to +11); they represent the learning style of the student.

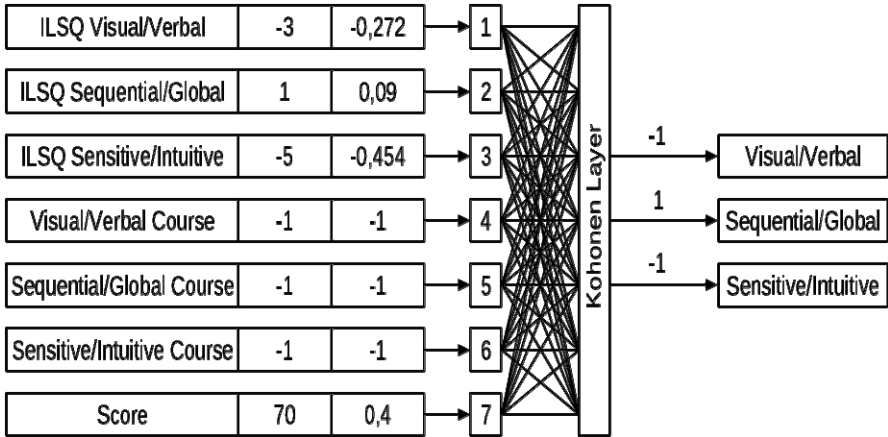


Fig. 11. The Kohonen Network for Learning Styles

The procedure to train the neural network consisted in three steps. The sample for this experiment was 140 high-school students. During the first step we applied the ILSQ questionnaire in order to identify the learning style of each student. In the second step we created three different introductory courses: Computer Science, Photography, and Eolithic Energy. For each course we produced eight versions, one for each learning style (visual/verbal, sequential/global, sensing/intuitive, and active/reflective). Randomly, the students received each of the three courses, in only one learning style. We recorded the assigned learning styles of each student in the three courses. In the third step and after the students took the courses, we applied tests in order to evaluate the performance of the students in each course.

With these three elements (student learning styles, course learning styles, and student evaluation or results) we created 140 input vectors for training the network. We trained the network with 6000 iterations, an initial neighbourhood ratio of 50% of the lattice size and a learning rate of 0.1. The network has 1600 neurons arranged in the form of a two-dimensional lattice with 40 rows and 40 columns, and every input vector has seven elements, as we established previously.

5.3 Using the Kohonen Network

As we explained before the network is applied to identify learning styles of students. The network is then exported together with the ITS and the predictive engine to the mobile (see figure 1). When the student is displaying an ITS in a time t , the learning material shown to the students is using a learning style g . Whenever the student answers a quiz as part of the learning contents, he receives a grade k .

Therefore, the network r takes as input the learning style of the displayed contents, and performs a search to find the winner-takes-all (best-matching) neuron j_x at iteration p , using the minimum-distance Euclidean criterion:

$$j_x(p) = \min \|X - W_j(p)\| = [\sum(x_i - w_{ij})^2]^{1/2}$$

where n is the number of neurons in the input layer, m is the number of neurons in the Kohonen layer, X is the input pattern vector and W is the weight vector.

After, finding neuron j_x , the output of the network is the student learning style which is defined as

$$g(t+1) = r(g(t), k)$$

This process of identifying the student learning style is performed every time the student answers a quiz on the ITS.

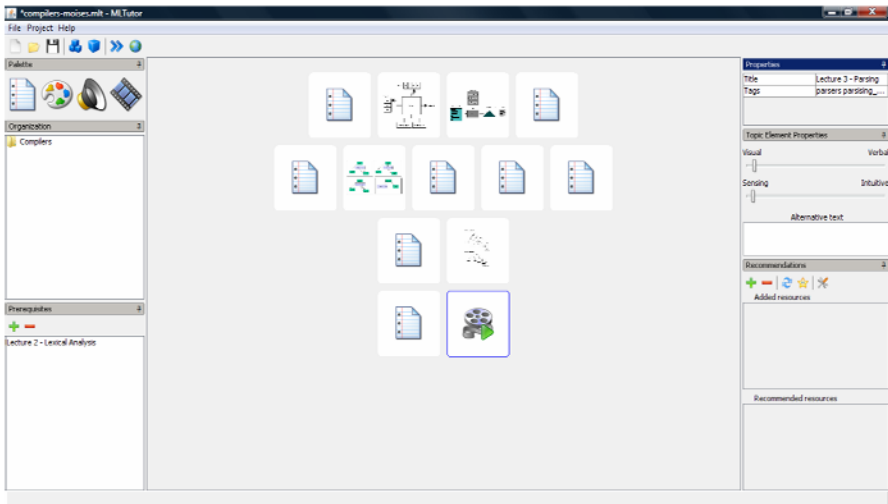


Fig. 12. EDUCA Main Interface

5.4 Tests and Results

The tool was used by a group of different users (authors), which consisted mainly of university professors/students. They produced different intelligent tutoring systems like Compiler Constructions, Introduction to Computer Science, teaching the Maya Language, and an introduction to Elementary mathematics. The intelligent systems were initially created by a professor and later the students were using and adding new learning resources with the recommendation system. Figure 12 shows the main interface of EDUCA for lecture 3 "Parsing" in a Compiler course. As we can observe, there are different kinds of learning objects, depending of the learning style, added to the topic. In our example, the arrow is pointing to a learning object which was added to be displayed only for "visual/sensitive/sequential/reflective" learners. The other objects are also displayed for different learning style combinations.

Figure 13 presents three snapshots of the Compiler course running in a mobile phone (Sun Java Wireless Toolkit 2.5.2). The first mobile phone (left) shows learning material about the parsing topic. The second and third mobile phone gives us a trace of the students' learning styles (in three stages) along the course. This information is valuable in that it shows how students' learning styles are varying from the one is obtained with the ILSQ to the correct one which is obtained with help of the Kohonen Network.

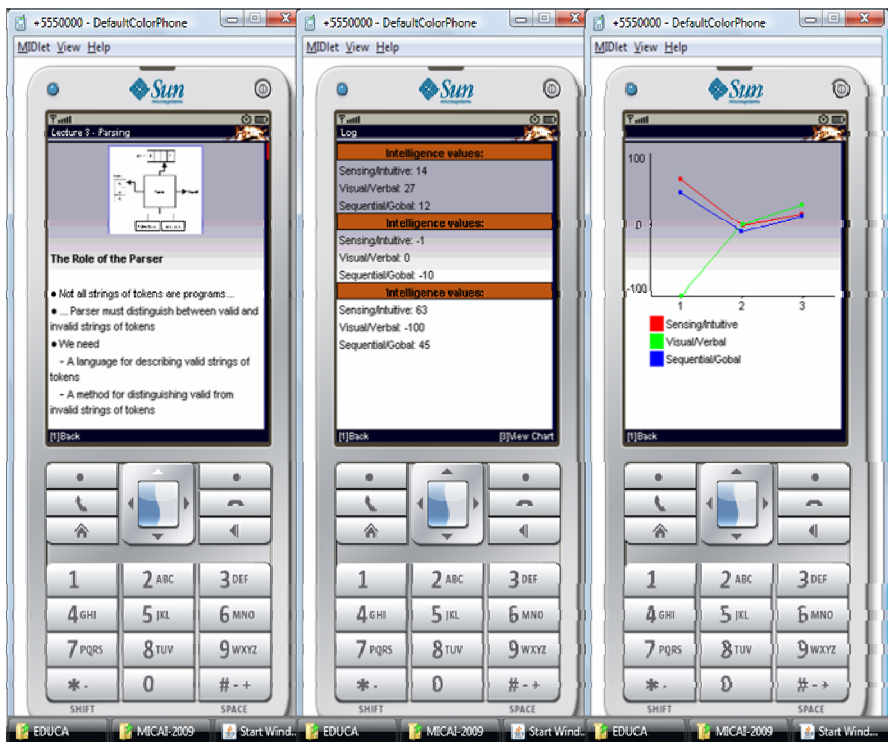


Fig. 13. The compiler course in a Cell Phone

6 Related Work

Many approaches and implementations have been developed in recent years in order to model students' learning styles [13, 14, 15, 16]. Most of these approaches use Bayesian Networks, Linear Temporal Logic, or Neuro-fuzzy Networks. Learning styles are calculated based only on the ILSQ questionnaire and none of these works are authoring tools.

Since the beginning of research in the field of ITS, more than twenty authoring systems for ITS have been developed [17]. Those authoring tools can be classified according to the type of tutoring system they produce; for example the author tool SIMQUEST [18] produces "simulation-based learning" systems, IRIS [19] creates "multiple knowledge types" systems, and InterBook [20] generates Intelligent/adaptive Hypermedia. One shared feature of all of those tools is the separate role of authors and learners. For instance, in SIMQUEST one author creates a simulation model, learner interface, instructional design, and environment. In this case, the learner is never involved. With IRIS, the author produces an ITS by following two main phases: Specifying the general requirements of the tutoring system, and filling the learning contents of the ITS. Building adaptive educational material is a similar procedure: the developer has to create content objects and specify the links between them. Authoring tools for developing adaptive educational/tutoring systems like InterBook, SIGUE [21], NetCoach [22], and Ale [23] differentiate only by the interface tools the authors operate. They use special markup languages or GUI interfaces. The author creates the learning environments and the learners read the learning contents.

There are several authoring tools used to create mobile applications like MyLearning [24], Test Editor [25], or mediaBoard [26]. Some of them are more PocketPC's oriented and some are focused to quiz editing or game-based learning. None of these author tools however, have the ability to adapt to the user's learning style, nor have portability across different computer and operating system platforms.

7 Conclusions and Future Work

This paper presents different approaches for modeling Students' Learning Styles for two different models and different platforms of learning. The approaches were implemented under three different authoring tools that allow the production of personalized learning material (Intelligent Tutoring Systems) to be used under web-based and mobile learning environments. The software tools were implemented with Java version 1.6 and XML. The courses or Tutoring Systems produced with the tools are platform independent and have been tested with different software emulators (NHAL Win32 Emulator, J2ME Wireless Toolkit and Sony Ericsson SDK), cell and smart phones (Sony Ericsson and Nokia). We present tests with some of those emulators. Currently empirical studies are taking place to examine the students' reaction to the learning material produced using **EDUCA**. Future work should focus on continuing testing with official and complete material for course programs in public and private schools in Mexico.

Acknowledgments

The work described in this paper is fully supported by a grant from the DGEST (Dirección General de Educación Superior Tecnológica) in México under the program “support and development of academic bodies” [Academic Body: Research in Software Engineering].

References

1. Nichols, M.: E-Learning in Context. Laidlaw College, Auckland, New Zealand. E-Primer Series (2008)
2. Seibel, C.: An Examination of Teaching and Learning Online. University of Saskatchewan, Canada. Occasional papers in Educational Technology (2008)
3. Ally, M.: Foundations of educational theory for online learning. In: The theory and practice of online learning, vol. 1(1), pp. 15–44. AU Press, Canada (2008)
4. Felder, R.: Matters of style. Department of Chemical Engineering North Carolina State University Raleigh. ASEE Prism 6(4), 18–23 (1996)
5. Coffield, F., Moseley, D., Hall, E., Ecclestone, K. (eds.): Learning Styles and Pedagogy in post-16 Learning: A Systematic and Critical Review. Learning and Skills Research Centre, Wiltshire (2004)
6. Felder, R., Silverman, L.: Learning and Teaching Styles in Engineering Education. North Carolina State University and Institute for the Study of Advanced Development. Engineering Education 78(7), 674–681 (1988)
7. SCORM, The Sharable Content Object Reference Model. April 22 (2010), <http://www.adlnet.org>
8. Gardner, H.: Frames of Mind: The theory of multiple intelligences. Basic Books, New York (1983)
9. Negnevitsky, M.: Artificial Intelligence: A guide to Intelligent Systems, 2nd edn. Addison-Wesley, Reading (2005)
10. Jang, J.: ANFIS: Adaptive Network-based Fuzzy Inference Systems. IEEE Transactions on Systems, Man and Cybernetics 23(3), 665–685 (1993)
11. Kohonen, T.: Self-Organization and Associative memory, 3rd edn. Springer, Heidelberg (1989)
12. Burke, R.: Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interactions 12(4), 331–370 (2002)
13. Carmona, C., Castillo, G., Millán, E.: Designing a Bayesian Network for Modeling Student's Learning Styles. In: Díaz, P., Kinshuk, A.I., Mora, E. (eds.) ICALT 2008, pp. 346–350. IEEE Computer Society, Los Alamitos (2008)
14. Graf, S., Kinshuk, A.I., Liu, T.: Identifying Learning Styles in Learning Management Systems by Using Indications from Students' behavior. In: Díaz, P., Kinshuk, A.I., Mora, E. (eds.) ICALT 2008, pp. 482–486. IEEE Computer Society, Los Alamitos (2008)
15. Limongelli, C., Sciarrone, F., Vaste, J.: LS-PLAN: An Effective Combination of Dynamic Courseware Generation and Learning Styles in Web-based Education. In: Nejdl, W., Kay, J., Pu, P., Herder, E. (eds.) AH 2008. LNCS, vol. 5149, pp. 133–142. Springer, Heidelberg (2008)

16. Zatarain-Cabada, R., Barrón-Estrada, M.L., Sandoval, G., Osorio, M., Urías, E., Reyes-García, C.A.: Authoring Neuro-fuzzy Tutoring Systems for M and E-Learning. In: Gelbukh, A., Morales, E.F. (eds.) MICAI 2008. LNCS (LNAI), vol. 5317, pp. 789–796. Springer, Heidelberg (2008)
17. Murray, T., Blessing, S., Ainsworth, S.: Authoring Tools for Advanced Technology Learning Environments. Kluwer Academic Publishers, Dordrecht (2003)
18. Jong, T., de, L.R., Gellelvi, M., Kuyper, M., Pieters, J., Joolingen, W.R.: Cognitive tools to support the instructional design of simulation-based discovery learning environment: the SIMQUEST authoring system. In: Plomp, T., van den Akker, J., Nieveen, N., Gustafson, K. (eds.), pp. 215–224. Kluwer Academic Publishers, The Netherlands (1999)
19. Arruarte, A., Fernández, I., Ferrero, B., Greer, J.: The IRIS Shell: How to build ITSs from Pedagogical and Design Requisites. *International Journal of Artificial Intelligence in Education* 8, 341–381 (1997)
20. Brusilovsky, P., Schwarz, E.: Web-based education for all: A tool for developing adaptive courseware. *Computer Networks and ISDN Systems* 30(1-7), 291–300 (1998)
21. Carmona, C., Bueno, D., Guzmán, E., Conejo, R.: SIGUE: Making Web Courses Adaptive. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds.) AH 2002. LNCS, vol. 2347, pp. 376–379. Springer, Heidelberg (2002)
22. Weber, G., Kuhl, H.-C., Weibelzahl, S.: Developing adaptive internet based courses with the authoring system NetCoach,
<http://www.wis.win.tue.nl/ah2001/papers/GWeber-UM01.pdf>
23. Specht, M., Kravcik, M., Klemke, R., Pesin, L., Hüttenhain, R.: Adaptive Learning Environment (ALE) for Teaching and Learning in WINDS. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds.) AH 2002. LNCS, vol. 2347, pp. 572–581. Springer, Heidelberg (2002)
24. Attewell, J.: Mobile technologies and learning: A technology update and mlearning project summary. *Learning and Skills Development*,
<http://www.m-learning.org/reports.shtml>
25. Romero, C., Ventura, S., Hervás, C., De Bra, P.: An Authoring Tool for Building Both Mobile Adaptable Tests and Web-Based Adaptive or Classic Tests. In: Wade, V.P., Ashman, H., Smyth, B. (eds.) AH 2006. LNCS, vol. 4018, pp. 203–212. Springer, Heidelberg (2006)
26. Attewell, J.: From Research and Development to Mobile Learning: Tools for Education and Training Providers and their Learners,
<http://www.mlearn.org.za/CD/papers/Attewell.pdf>

Towards a Fuzzy Model for RAMSET: Role Assignment Methodology for Software Engineering Teams

Luis G. Martínez, Juan R. Castro, Guillermo Licea, Antonio Rodríguez-Díaz, and Carlos Álvarez

Universidad Autónoma de Baja California, Calzada Tecnológico 14418,
Mesa de Otay, CP 22390 Tijuana BC, México
{luisgmo, jrcastro, glicea, ardiaz, fco_alvarez}@uabc.mx

Abstract. Current studies related to performance and integration of Software Engineering teams, focus on efficiency boarding different factors such as team abilities, management, tools, methodologies, personality and roles. The present work is a follow up of RAMSET (Role Assignment Methodology for Software Engineering Teams) methodology that relates personality, abilities and software roles for integration of Software Engineering Teams, applying sociometric and psychometric techniques under a Fuzzy Approach due measurements' subjective appreciation. A Fuzzy Model System is proposed to identify the best Role to be performed by a Software Engineer in Software Development Project Teams.

Keywords: Fuzzy Logic, Software Engineering, Psychometrics, Sociometrics.

1 Introduction

Software development is an impressively complex socio-technical activity. It requires people to interact with each other and with technical methods and computing technologies they use to perform their work [1]. We must integrate a group of persons taking into account abilities and capacities for software development process. Each member contributes in a team as part of the total necessary capacities for team success, which is why each individual must do activities where they are most experienced in performing a specific role in the work team.

In a working team it is not only important to know capacity, skills, techniques and experience of each member, it is also fundamental to know the sociopsychological characteristics and personality of each one to form the team [28].

Achieving and maintaining excellence in modern corporations requires an effort of combining talents from different individuals (strengths and weaknesses) to accomplish structure, integration and consolidation of working teams.

This paper introduces a technique that relates personality, abilities and team roles in a software developing team. RAMSET (Role Assignment Methodology for

Software Engineering Teams) is based in personality tests with a Fuzzy Model System approach in role assignment for software engineering teams. Personality tests are based on interpretation; therefore to tackle uncertainty a fuzzy model can bring us a more appropriate decision for recommending and assigning a software role.

The paper is organized as follows: section 2 introduces fundamental aspects of personality and software engineering roles in working teams with background related with the projective personality Tree Test. Section 3 describes RAMSET steps, section 4 a Fuzzy Model System is proposed for the personality Tree Test in particular. Section 5 shows the results of our work and relationships between linguistic variables of the proposed model. Finally section 6 concludes with observations and experiences obtained.

2 Background

2.1 Personality and Role Assignment Relationship

Human Psychology studies are old, before the 80's personality aspects were considered of low value in personnel selection. Personnel selection methodologically chooses individuals for the right job. The first tests [24] used for a programmer's performance prediction and personnel selection in software development projects where PAT test (Programmer Aptitude Test), WPT test (Wonderlic Personnel Test) and PMA test (Primary Mental Abilities). In personality analysis, different and popular tests exist, like Jung's, Keirsey and Myers-Briggs Type Indicator (MBTI) tests. Rutherford [29] manages Personality Inventories in his Software Engineering classes to integrate heterogeneous teams, obtaining good results in teams' performance and individuals' growth. Karn and Cowling [17] also implemented personality tests in Sheffield University England, documenting personality effects in Software Engineering Teams performance, mentioning that is a vast area and many subjects to consider relating software engineer's personality and their teams.

Empirical studies investigating integration and building of Software Engineering Teams focus in different factors that influence their performance like team abilities, team administration, efficiency, development methods, diversity, size, genre, personality [11] and roles [7]. Pieterse and Kourie [27] reflect on the performance and diversity of Software Engineering Teams from the University of Pretoria where they applied personality tests, establishing metrics to build the teams. These metrics were ability, diversity and performance, concluding that not only diversity must be taken into account but also effective leadership, communication and cohesion. Hogan and Thomas [13] emphasize about team work in their Software Engineering Courses where they have helped students to be better professionals teaching them work team abilities. Beranek [3] has been defining Functional Team Roles examining an informal distribution of team roles between students, whereas they measure their abilities and attitudes, working styles and behavior, based on Belbin [4] Role Tests, complementing them with students opinion's about their preferences and working attitudes. Feldt et al. [8] corroborate that empirical studies should collect psychometrics focusing in correlating personality and attitudes to software engineering processes and tools.

Personality type was measured with MBTI (Myers-Briggs Type Indicator) using Keirsey Temperament Sorter by Gorla and Lam [11], their survey study shows a relationship between Software Engineering Roles and MBTI dimensions, primarily analyzing team leaders, analysts and programmers. These experiences resemble the findings by Capretz [5] in surveying people in the US working in software engineering.

Table 1. Personality and Role Relationship by Gorla and Lam

Criterion	Personality dimension	Team performance
Team leader	Information gathering	Intuitive > Sensing
	Decision making	Feeling > Thinking
System analyst	Decision making	Thinking > Feeling
Programmer	Interaction with the world	Extrovert > Introvert

Shen [30] and colleagues have applied Myers-Briggs Type Indicator (MBTI) and Keirsey Temperament Sorter to form engineering design teams based on Wilde's [35] own method of selection with this tests. They recommend a Sensing-Intuitive (SN) type as the most important preferred type linked to creativity for selection of engineers in a leader role for design process. Also Sodiya et al. [31] from Nigeria have developed a tool integrating personality traits proposing a Cognito-Personality Assessment Model for Software Engineering (CPAMSE) relating the Big Five Traits with roles and activities of engineers in software engineering. Fuzzy based approaches have been considered like Lather's [20] fuzzy model to evaluate suitability of Software Developers, also Ghasem-Aghaee and Oren's [25][26] use of fuzzy logic to represent personality for human behavior simulation. Consequently everyone encouraging engineering educators to make greater use of type theory when selecting and forming engineering design teams and delegating team roles, in benefit of achieving productivity and efficiency in team performance, themes open to investigation.

2.2 Graphology Tree Test

In psychology the Tree Test is a projective test or graphical test. Emil Jucker [18] initiated the legacy of this psychodiagnostic test. Later Karl Koch [19] contributes Tree Test formulations, interpretations related in graphology as quality strokes, zone and sheet placement. Every graphical product instates the psychic life of an individual. Jung's tree symbology makes us consider it as a life symbol, of growth and development. He states "the growth from bottom up and viceversa, the maternal aspect: protection, shadow, cover, fruits, and fountain of life" [14]. Carl G. Jung also stated "In the unconscious leaving context, the trees are like fish in the water" [15]. Thus trees are symbolically associated with man's inner person and can describe their growth and emotional stability.

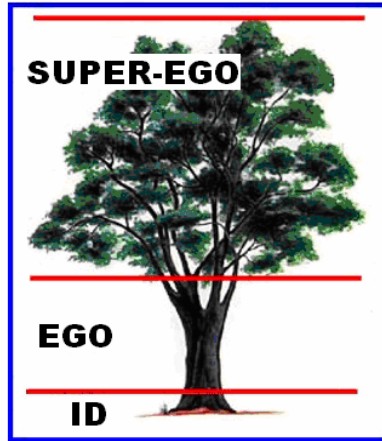


Fig. 1. EGO relationships expressed in Tree

The tree expresses the relationship between Id, Ego and Super-Ego. The Id comprises the unorganized part of personality structure that contains the basic drives, everything that is inherited, that is present at birth, therefore, instincts which originate from the somatic organization, and which find a first psychical expression here (in the id) in forms unknown to us [9]. The Ego comprises the organized part of personality structure that includes defensive, perceptual, intellectual-cognitive, and executive functions. The Ego is that part of the Id which has been modified by direct influence of the external world. The Ego mediates among the Id, the Super-Ego and the external world. Its task is to find a balance between primitive drives and reality while satisfying the id and super-ego. Its main concern is with the individual's safety and allows some of the id's desires to be expressed, but only when consequences of these actions are marginal. The Super-ego aims for perfection. It comprises that organized part of the personality structure, mainly but not entirely unconscious, that includes the individual's ego ideals, spiritual goals, and psychic agency (commonly called 'conscience') that criticizes and prohibits his or her drives, fantasies, feelings, and actions.

Perfect equilibrium of this personality instances assures a psychic stability while their disproportion suggests a pathology appearance. The Tree Test presents an observation type evaluation of the Id strength level, emotional stability level and autoevaluation degree. Id strength level permits the subject reality assertion, persevere in his objectives and goals, to overcome and resist pressures and frustrations of his surroundings. Emotional stability's level is the presence and degree of conflicts associated to susceptibility, vulnerability, sensibility, rigidity, adaptability, and so on. Autoevaluation degree refers to how the subject finds itself in that moment, his reality criteria, strength, and ability to control impulses and emotions.

The tree's crown represents the subject's fantasies, mental activities, his thoughts, spirituality and reality conception, it covers foliage and branches. The

unconscious world or instincts are symbolized in the root. Renate Griffiths [12] states “the root tree is the origin of it, is the symbol of its fountain of life”. Finding the roots on the third or inferior zone, it tells us of the material, physical, earthly life, sexuality, and basis for his reality criteria.

A personality Tree Test does not define us a total projection or whole image of the personality, but sheds valuable information of the sketcher. Relevance and merit of applying this test is a combination with other tests, which is why is just one of many tests used in RAMSET to reveal personalities of working team members and assign them the best advisable performing role.

2.3 Fuzzy Model System

Fuzzy sets and fuzzy logic are effective techniques for handling fuzzy uncertainties with well-developed mathematical properties. Imprecise concepts are attributes of which we have a cognitive perception, yet are impossible to define precisely like in a defined model. That is why fuzzy logic provides an excellent way to represent and process linguistic variables.

A non-fuzzy method weakness is dealing with imprecision and uncertainty in handling linguistic terms. Fuzzy set theory provides a natural method for dealing with the linguistic terms by which an expert will describe a domain [16]. It is an established fact that fuzzy model is the best choice for managing ambiguous, doubtful, contradicting and diverging opinions. All the way it's a better choice for very high complexity and nonlinearities. Fuzzy systems are intuitive and numeric systems, mapping input and output data.

Fuzzy Inference Systems (FISs) also called Fuzzy Models are fuzzy production systems used for modeling input-output relationships. From this input-output view, Babuska [2] describes these systems as “flexible mathematical functions which can approximate other functions or just data (measurements) with a desired accuracy”. Fuzzy Production Rules define the relationship between input and output variables. Input variables are defined in the antecedent part of the rule and the consequent part defines the output variable.

Figure 2 represents these FIS that are basically composed of five modules:

- 1) Rule Base: the set of fuzzy production rules.
- 2) Database: where the membership functions are defined.
- 3) Fuzzy Inference Engine: executes the fuzzy inference operations.
- 4) Fuzzifier: transforms the inputs-numerical data into linguistic values.
- 5) Defuzzifier: transforms the fuzzy results into numerical data.

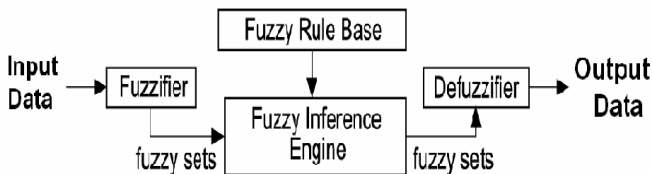


Fig. 2. Fuzzy Model System Structure

A fuzzy system is associated with a set of rules with meaningful linguistic variables, such as (1)

$$R^l : \text{if } x_1 \text{ is } F_1^l \text{ and } x_2 \text{ is } F_2^l \text{ and } \dots x_n \text{ is } F_n^l \text{ then } w \text{ is } G^l \quad (1)$$

Actions are combined with rules in antecedent/consequent format, and then aggregated according to approximate reasoning theory, to produce a nonlinear mapping from input space $U = U_1 \times U_2 \times U_3 \times \dots \times U_n$ to the output space W where $F_k^l, k = 1, 2, \dots, n$ are the antecedent membership functions, and G^l is the consequent membership function. Input linguistic variables are denoted by $u_k, k = 1, 2, \dots, n$, and the output linguistic variable is denoted by w .

The most resorted fuzzy inference models are those of Mamdani and Takagi-Sugeno [32]. Mamdani is direct and simple in describing empirical knowledge, has clarity in significance of linguistic variables and design parameters. Takagi-Sugeno enhances a simpler process using first degree equations in most of its applications, at a cost of less clarity in linguistic variables significance. Mamdani fuzzy rules take the form (2), where x and y are activated variables of the membership function, z is the consequent fuzzy variable and the connective AND the conjunction operation with the antecedent.

$$\text{IF } x \text{ is } X_o \text{ AND } y \text{ is } Y_o \text{ THEN } z \text{ is } Z_o \quad (2)$$

It is in this context we present here a fuzzy based approach that provides an integrated quantity measure for abilities of software development personnel which incorporates all aspects of personality traits involved for role assignment.

3 RAMSET Methodology

In Software Engineering Courses development of projects by work teams is usually implemented applying the traditional Software Life Cycle. Classic software engineering courses consisting of lecture notes and Waterfall Model application for project development is being substituted by more practical methods emphasizing in helpful tools and agile processes, as divulged by North Carolina State University [21].

At the University of Baja California, Tijuana Mexico teaching of Software Engineering in our Computer Engineering Program is being conducted with development of real software projects applying RAMSET: Role Assignment Methodology for Software Engineering Teams based on personality. What is unique about RAMSET is a combination of Sociometric techniques, Psychometrics and Role Theory in Software Engineering Development Projects, this methodology consists of the next steps:

- a) Survey for abilities and skills.
- b) Implementation of Personality Tests.
- c) Execute Personal Interviews.
- d) Implementation of Sociometric Technique.
- e) Assignment of Team Roles.
- f) Follow up of Team Role fulfillment

specifications; developer-programmer, responsible for implementation and code design; tester, responsible for tests and evaluation of the system; document specialist, responsible for compiling every document for evidence and defining documentation standards; and image and presentation role as a representative in charge of selling and promoting the product.

We can choose from a set of different Personality Tests all revolving on the dimension types or personality traits that display the behavior of the individual member of the team. Combining these tests we can acquire the most valuable information for decision making in assignment of roles.

4 Tree Fuzzy Model System

Personality's Tree Test throws subjective information, based on the point of view and perception of the evaluator. That is why we propose a Fuzzy Model System that can shed numerical values that solve the output uncertainty. Fuzzy Inference Systems are based on Fuzzy Set Theory [36] permitting the incorporation of an uncertainty component that makes them more effective for reality approximation. Linguistic variables are used to manipulate imprecise qualitative and quantitative information; the linguistic variable is a variable whose values are not numbers but words or sentences in a natural or artificial language [6].

A linguistic variable is characterized by a quintuple $(x, T(x), U, G, M)$, in which x stands for the name of the variable, $T(x)$ denotes the set of x of fuzzy variable values, ranging over a universe of discourse U . G is a syntactic rule for generating names of x , and M is a semantic rule for associating each x to its meaning being a subset of U .

For our Tree Test FIS we selected three linguistic variables: (R) Root, (T) Trunk and (F) Foliage. These input variables enter the FIS model and obtain an output variable result of the Role (Q). Each linguistic variable is introduced to a filter of sketch recognition identifying the variable's input value for the FIS's Role. Therefore the model has three input and one output variables as seen in figure 4.

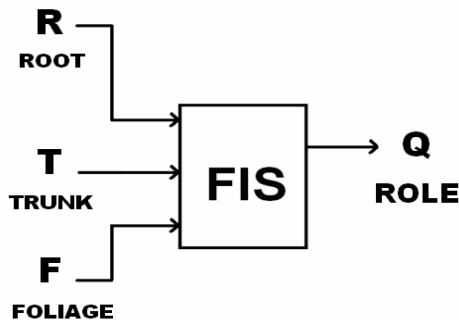


Fig. 4. Tree Test FIS Model

Psychodiagnostics analyze specific characteristics from the drawing. For Root we can select sketching type and size, the root represents the past and reflects person's dependency. For Trunk we can select form, area, height, sketch intensity and curvature, the trunk depicts the present and reflects person's affectivity. For Foliage we can select form, size and extra features, it symbolizes achievements or goals reached. As a first model proposal the Tree Fuzzy Sets were defined as follows:

The Set of Linguistic Variables for Root is: $R(x) = \{\text{null, none, with}\}$. When there is no sketch of any root Null is defined as the attribute, if the root is hidden the attribute is None, and any sketch of roots the attribute has been defined as With.

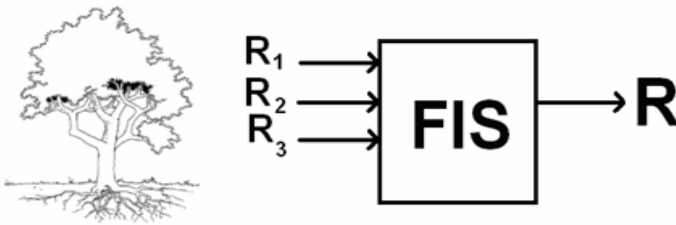


Fig. 5. Root Recognition FIS

The Set of Linguistic Variables for Trunk is: $T(x) = \{\text{straight, wave, trapeze}\}$. When the sketch of the trunk is two parallel lines the attribute is defined as Straight, if one or two of the trunk lines are curved the attribute is Wave, and two straight or curved lines with a wider bottom than the top the attribute is Trapeze.

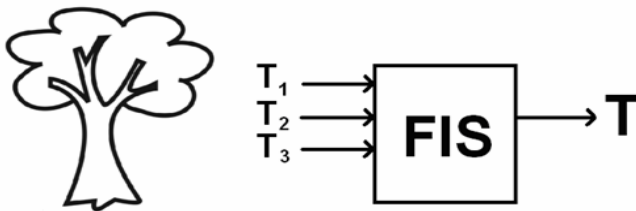


Fig. 6. Trunk Recognition FIS

The Set of Linguistic Variables for Foliage is: $F(x) = \{\text{circular, cloud, fruit, null}\}$. Just a round sketch of foliage the attribute is defined as Circular, if it has wavy contour with or without faint sketches inside the attribute is Cloud, if it has any fruit the attribute is Fruit, and any sketch of only branches or leaves the attribute is Null.

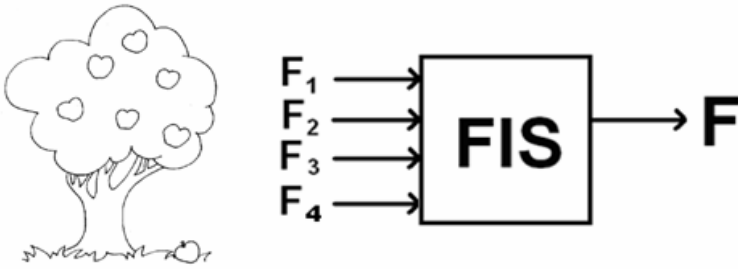


Fig. 7. Foliage Recognition FIS

The Set of Linguistic Variables for the output Role is: $Q(x) = \{ \text{Analyst, Architect, Developer-Programmer, Documenter, Tester, Image and Presenter} \}$.

Labels were assigned to each attribute of later sets, and consecutive values starting on one were also assigned. Figure 8 illustrates attribute's membership functions of linguistic variables Root (R), Trunk (T), Foliage (F) and Role (Q), displaying intervals for each label.

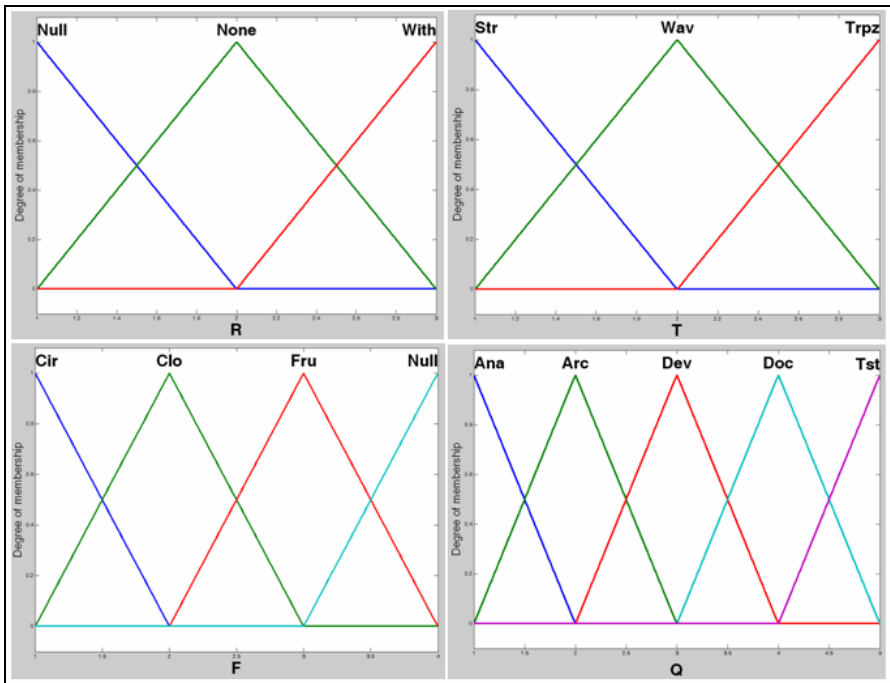


Fig. 8. Input and Output Membership Functions

With these intervals in combination with the Set of Rules obtained from conclusive results {3} we define RAMSET's Tree Test Fuzzy Model application.

5 Results

Since 2007-2 until 2009-1, 56 software engineers have worked in real projects, assigning 88 roles in software development teams. Of these 15 have been assigned as Analysts, 15 Architects, 20 Developers/Programmers, 14 Documenters, 15 Tester and validator, 9 Image and presentation.

Table 2 shows Roles and MBTI types member relationship's of the teams. ENFJ, ENFP, INTP, INFJ, INFP types have not come up in the working samples. We can observe that personality type traits with higher percentage are (E) extroverted 70%, (S) sensing 74%, (T) thinking 66% and (J) judging 81%, traits to be considered for role assignment without a question. Traits as (I) introverted, (N) intuitive, (F) feeling or (P) perceptive are of lower value, only in combination with other traits can be considered of importance for role assignment.

Table 2. MBTI types and Roles Frequency

	J1*	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	TOT
Analyst	7	2	4	1	0	1	0	0	0	0	0	15
Architect	7	4	0	1	0	1	1	1	0	0	0	15
Developer Programmer	2	2	1	5	3	1	0	1	2	2	1	20
Documenter	5	2	1	1	1	1	1	1	1	0	0	14
Tester	5	3	4	0	1	1	1	0	0	0	0	15
Image and Presenter	2	3	2	0	1	0	1	0	0	0	0	9
Totals	28	16	12	8	6	5	4	3	3	2	1	88
*J1=ESTJ, J2=ENTJ, J3=ESFJ, J4=ISTP, J5=INTJ, J6=ISTJ, J7=ISFJ, J8=ESFP, J9=ISFP, J10=ESTP, J11=ENTP												

By implementing a fuzzy model the 16 jungian types are considered as antecedents relating them with the consequent resulting role. Using weighted mean method a specific weighted factor was assigned in order of importance, by this the data instead of contributing equally to the final average, some data points contribute more than others.

Taking the Analyst role as an example factors J1, J3, J2, J4, J6, J5, J7, J8, J9, J10 and J11 were put in decrementing order giving the first a value of 11 as seen in Table 3. The value (V) was multiplied by its frequency factor (F). This result was then divided by the sum total obtaining the final weight of each factor.

Table 3. Analyst Role weights

(V)alue	11	10	9	8	7	6	5	4	3	2	1	TOT
(F)requency	7	4	2	1	1	0	0	0	0	0	0	15
V * F	77	40	18	8	7	0	0	0	0	0	0	150
Final Weight	.513	.267	.12	.053	.047	0	0	0	0	0	0	1
Attribute	J1	J3	J2	J4	J6	J5	J7	J8	J9	J10	J11	

Applying the same method to attributes' frequencies from table 1 viewing MBTI types and Software Engineering Roles performed Table 4 is obtained.

We observe that ESTJ type is a highly qualified individual to perform different roles; our stats recommend the Analyst or Architect role. For an ISTP individual Developer or Programmer is best fitted, an ESTJ for Tests although others can perform this role. And for our presenter and image we recommend an ENTJ, it can also be ESTJ or ESFJ. Viewed by those not recommended, types ESFP, ISFP, ESTP, ENTP should not be assigned as analysts, architects or testers and ENFP, INTP, INFJ, INFP haven't been considered for any role.

Table 4. Software Engineer Roles weights

	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11
ANA	.513	.12	.267	.053	0	.047	0	0	0	0	0
ARC	.524	.272	0	.061	0	.054	.048	.041	0	0	0
DEV	.113	.101	.031	.346	.189	.025	0	.019	.088	.075	.013
DOC	.47	.171	.077	.068	.051	.06	.043	.034	.026	0	0
TST	.385	.189	.28	0	.056	.049	.042	0	0	0	0
PRS	.233	.384	.209	0	.093	0	.081	0	0	0	0

Analyzing by attributes the ESTJ is an analyst and architect being (E) extroverted and with good (J) judgment to relate with other people and take important decisions. The developer or programmer is an ISTP commonly (I) introverted and (T) thoughtful to his work with logic for problem solving.

We analyzed the Tree Test in the same periods of time, having 74 drawings of trees obtaining weights for the attributes sketched using same mean weights method. Table 5, 6 and 7 are the universe of weights for Root, Trunk and Foliage respectively.

Table 5. Root's weights

R(x)	ANA	ARC	DEV	DOC	TST	PRS
R1	0.103	0.182	0.441	0.030	0.067	0.050
R2	0.276	0.636	0.441	0.727	0.333	0.200
R3	0.621	0.182	0.118	0.242	0.600	0.750

Table 6. Trunk's weights

T(x)	ANA	ARC	DEV	DOC	TST	PRS
T1	0.174	0.174	0.25	0.091	0.097	0.316
T2	0.652	0.174	0.656	0.455	0.452	0.632
T3	0.174	0.652	0.094	0.455	0.452	0.053

Table 7. Foliage's weights

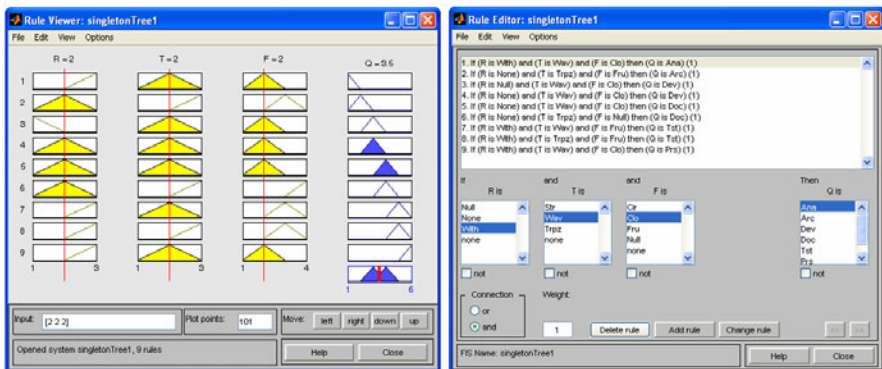
F(x)	ANA	ARC	DEV	DOC	TST	PRS
F1	0.225	0.153	0.340	0.243	0.243	0.130
F2	0.6	0.307	0.545	0.540	0.162	0.695
F3	0.15	0.512	0.090	0.162	0.540	0.087
F4	0.025	0.025	0.022	0.054	0.054	0.087

From this data we can obtain the Set of Fuzzy Rules annotated below:

Set of Rules (3)

IF R is R3 AND T is T2 AND F is F2 THEN Q is Q1
 IF R is R2 AND T is T3 AND F is F3 THEN Q is Q2
 IF R is R1 AND T is T2 AND F is F2 THEN Q is Q3
 IF R is R2 AND T is T2 AND F is F2 THEN Q is Q3
 IF R is R2 AND T is T2 AND F is F2 THEN Q is Q4
 IF R is R2 AND T is T3 AND F is F4 THEN Q is Q4
 IF R is R3 AND T is T2 AND F is F3 THEN Q is Q5
 IF R is R3 AND T is T3 AND F is F3 THEN Q is Q5
 IF R is R3 AND T is T2 AND F is F2 THEN Q is Q6

These set of rules are implemented in MatLab's commercial Fuzzy Logic Toolbox [10] as seen in figure 9 to simulate our case studies and have a first approach to automate role assignment with RAMSET in software engineering projects.

**Fig. 9.** RAMSET's Rules Edited in Matlab's Fuzzy Logic Toolbox

If we analyze a Root-Trunk relationship shown on figure 10, we can find 3 zones, between 1 and 2.4 weigh for (R) and (T) we have an average 3 to 3.5 weigh thus recommending Developer-Programmer or Documenter. Above 2.4 for (R) and (T) it recommends a Tester. (T) above 2.4 and (T) below 2 it recommends an Architect.

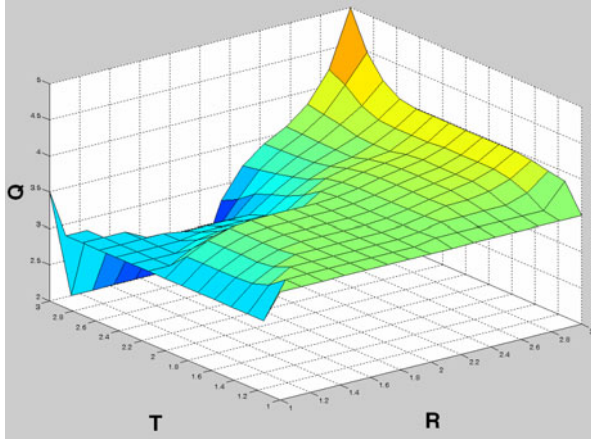


Fig. 10. Root – Trunk Relationship

If we analyze the Root-Foliage relationship as in figure 11, we can find 3 data zones, between 1 and 2.5 weigh for (R) and (F) we have an average 3.5 weigh thus recommending Developer-Programmer or Documenter. Above 2.4 for (R) and (F) it recommends a Tester or Presenter. (T) below 1.5 and (F) below 3 it recommends a Developer-Programmer.

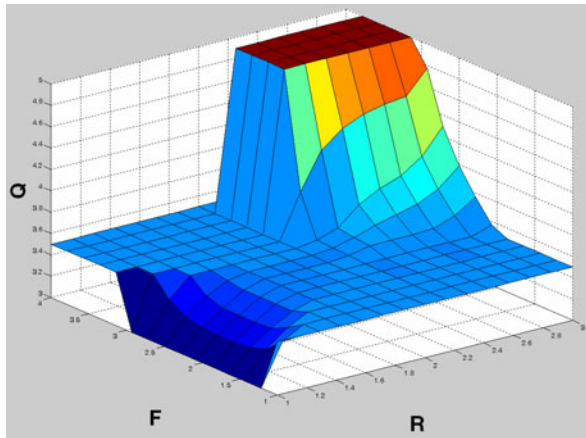


Fig. 11. Root-Foliage Relationship

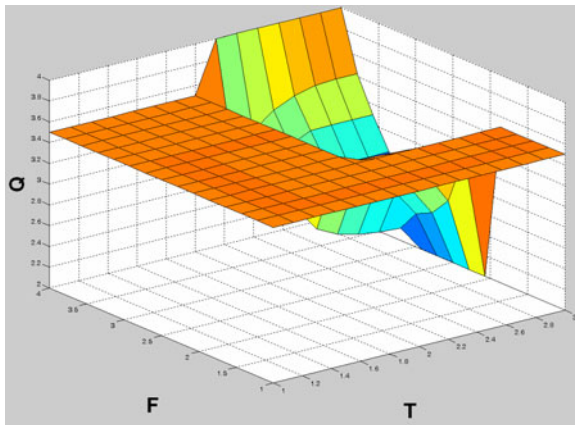


Fig. 12. Trunk-Foliage Relationship

If we analyze the Foliage-Trunk relationship as in figure 12, we can find 4 data zones, for any (F) and (T) below 2 we have an average 3.5 weigh thus recommending Developer-Programmer or Documenter. Above 2 for (T) and between 2 and 3.5 for (F) it recommends an Architect or Programmer. Above 2 for (T) and above 3.5 for (F) it recommends a Tester or Programmer.

From the sets of linguistic variables we can analyze each attribute highlighting for example, when Root (R) is null (R1) the most probable role is Developer-Programmer (Q3). Without visible root (R2) we can assign Architect (Q2) or Documenter (Q4). Any sketch of root (R3) we are talking about an Analyst (Q1) or Tester (Q5), even Image and Presenter (Q6). The Image and Presenter (Q6) role consists in selling, distribution and image design. The individual's quality performing this role has been related with his own personal image, and a high percentage present the attribute (R3), drawing roots even highlighting thick roots, as we analyze this individual we can see he wants to draw more attention, wants to be noticed and depends of what other people say.

Analyzing the Trunk (T) there are less differences between roles, wavy trunks (T2) are Analysts (Q1), Developer-Programmers (Q3), Testers (Q5) or Presenters (Q6). What it is sure in this attribute, we can distinguish an Architect (Q2) from the others because he draws the trunk in a trapeze shape (T3). The Foliage (F) distinguishes an Architect (Q2) and a Tester (Q5) from other roles as they draw trees with Fruits (F3), others draw cloudy (F2) type most of the times.

With the set of rules defined {3} we can distinguish two roles from others. Architect (Q2) has the only combination of without root (R2), trapeze (T3) and fruits (F3); and the Tester (Q5) is the only one with root (R3), trapeze or wavy (T3 o T2) and fruits (F3). Drawing fruits means this individual has a clear view of what he wants to do, he has achieved personal goals in life, giving him serenity to take charge of any project and achieve goals set and obtain the final product, qualities of a leader and architect.

There's a similarity between developer-programmer (Q3) and documenter (Q4) and between analyst (Q1) and presenter (Q6). Some cases are differentiable between programmer with {R1, T2, F2} and the documenter with {R2, T3, F4}, although combination {R2, T2, F2} pops up more frequently. Also the combination {R3, T2, F2} does not distinguish between analyst and presenter, this results give us no significant difference in these cases, thus applying and increasing more cases will give us a more significant result.

The results obtained for the FIS rules are from 2007-2 thru 2009-1, in the semester 2009-2 we applied the methodology in a group of 16 engineers, after the sociogram and personality tests; we used the tree test results and simulated them in the Fuzzy Logic Toolbox of Matlab [10] to prove RAMSET in assigning roles with our FIS model (2009-2). If we work only with Analyst, Architect and Developer-Programmer roles, our fuzzy model can help us 100 percent in distinguishing each role. For larger teams that perform with more roles it helps us but we cannot base role assignment in only the Tree Test, which is why we are proposing the use of other personality tests to complement each other for best role assignment of team members.

The Tree Test is only one test of many personality tests applied and proposed, later analysis is a first instance of observations of tree drawings, a more detailed analysis and collaborative judgment for interpretation and classification of sketches will confirm selection of them and we can propose a broader range of attributes to help us distinguish roles better.

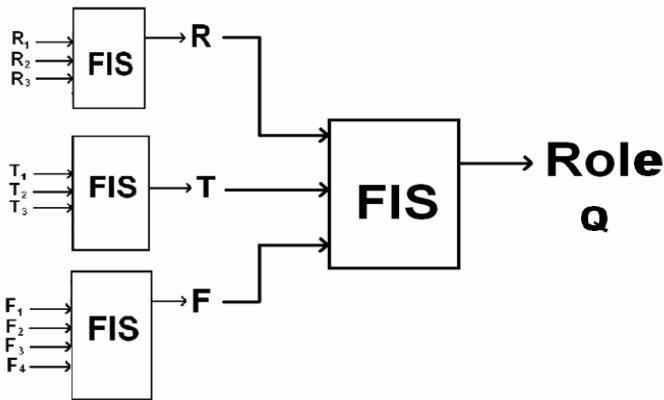


Fig. 13. Tree Test FIS Model Architecture

6 Conclusions

Through the years in Software Engineering Courses, usual integration of work teams for software development projects has form all kind of teams, conflictive teams, good working teams, mediocre teams, excellent teams. Teacher's intuitive experience is the main tool for integrating the best teams. Using the sociogram

technique and personality tests has helped us in forming working teams less conflictive, those which by having more affinity between their members they form natural groups that work more fluidly and perform better.

The objective of using RAMSET is identifying the individual's qualities to perform the most suitable role in a working team. Some personalities and typologies have been identified to perform a type of role; we need more evidence in other type of teams to prove our results applied in software engineering courses established until now.

The FIS model of personality Tree Test proposes linguistic variables, weights and membership functions applied and proven in our Software Engineering Courses, continuity of this work can improve the Data Base Rules arisen from this study and implement it as a computer aided software tool making RAMSET an automated tool for role assignment.

Future work is designing and implementing direct image recognition of Trees drawn by individuals, and applying uncertainty in sketch analysis to determine input linguistic variables making RAMSET's automated tool more efficient and effective.

We come to the conclusion that integrating work teams in Software Engineering Courses is a highly potential area of exploration, whereas the influence of personality in team role assignment is a task to be clearly defined. We know that personality is an important factor to performance of the team, thus is latent the difficulty to assign the adequate role to each member so the team can perform with success. Automation of RAMSET will help us with this task. Follow up of RAMSET not only in Software Engineering Courses but in other courses of the same program can give us a comparative study about role assignment in work teams based in personality, confirming RAMSET as a methodology for integrating teams and an instrument to select personnel for Software Engineering Teams.

References

1. Brooks, F.: The mythical man-month. *Datamation*, 44–52 (1974)
2. Babuska, R.: *Fuzzy Modeling for Control*. Kluwer Academic Publishers, Norwell (1998)
3. Beranek, G., Zuser, W., Grechenig, T.: Functional Group Roles in Software Engineering Teams. In: *HSSE 2005*, St. Louis Missouri USA, pp. 1–5 (2005)
4. Belbin, M.: *Team Roles at Work*. Butterworth-Heinemann, Butterworths (1993)
5. Capretz, L.F.: Personality types in software engineering. *International Journal of Human-Computer Studies*, 207–214 (2002)
6. Cox, E.: *The Fuzzy Systems Handbook*. Academic Press, London (1994)
7. Dubinsky, Y., Hazzan, O.: Using a Roles Scheme to Derive Software Project Metrics. *Journal of Systems Architecture* 52(11), 693–699 (2006)
8. Feldt, R., Torkar, R., Angelis, L., Samuelsson, M.: Towards Individualized Software Engineering: Empirical Studies Should Collect Psychometrics. In: *CHASE 2008*, Leipzig Germany, May 13 (2008)
9. Freud, S.: *An Outline of Psycho-analysis* (1940)

10. Fuzzy Logic Toolbox: User's Guide of Matlab, p. 343. The Mathworks, Inc.. (1995-2009)
11. Gorla, N., Lam, Y.W.: Who Should Work With Whom? Building Effective Software Project Teams. *Communications of the ACM* 47(6), 79–82 (2004)
12. Griffiths, R.: *Graphology: early recollections and the tree test*. The Author West Vancouver British Columbia (1988)
13. Hogan, J., Thomas, R.: Developing the Software Engineering Team. In: *ACM ICPS 7th ACCE 42 Newcastle New South Wales Australia*, vol. 106, pp. 203–210 (2005)
14. Jung, C.G.: *The philosophical Tree (1945/1954)*
15. Jung, C.G.: *Aspects of the Masculine*. Routledge, New York (2003)
16. Kandel, A., Hall, L.O.: The evaluation from Expert Systems to Fuzzy Expert Systems. In: *Fuzzy Expert Systems*, p. 19. CRC Press, Boca Raton (1991)
17. Karn, J., Cowling, T.: A follow up study on the effect of personality on the performance of software engineering teams. In: *ISESE 2006 Rio de Janeiro, Brazil*, pp. 232–241 (2006)
18. Koch, K.: *Le test de l'arbre*, Lyon Vitte, 9 (1958)
19. Koch, K.: *El test del árbol: el dibujo del árbol como medio psicodiagnóstico auxiliar*. Kapelus, Buenos Aires (1986)
20. Lather, A., Kumar, S., Singh, Y.: Suitability Assessment of Software Developers: A Fuzzy Approach. In: *ACM SIGSOFT Software Engineering Notes*, vol. 25(3), pp. 30–31 (May 2000)
21. Layman, L., Comwell, T., Williams, L.: Personality Types, Learning Styles, and an Agile Approach to Software Engineering Education. In: *SIGCSE 2006*, pp. 428–432. ACM, Houston Texas (March 2006)
22. Martínez, L.G., Rodríguez, A., Licea, G., Castro, J.R.: Utilización de Test Sociométrico y Test de Personalidad en la Integración de Equipos de Trabajo en Cursos de Ingeniería de Software. In: *XVI Congreso Internacional de Educación Superior en Computación CIESC 2008 en el marco CLEI 2008*, Santa Fe Argentina (2008)
23. Martínez, L.G., Rodríguez, A., Licea, G., Castro, J.R.: Application of Personality and Sociometric Tests to Integrate Work Teams in Software Engineering Courses in Educational Programs. In: *9no. Encuentro Internacional de Computación ENC 2008*, Mexicali B.C., México (2008)
24. Mayer, D.B., Stalaker, A.W.: Selection and evaluation of computer personel – the research history of SIG/CPR. In: *Proceedings of the 1968 23rd ACM National Conference*, pp. 657–670 (1968)
25. Oren, T.I., Ghasem-Aghae, N.: Personality Representation Processable in Fuzzy Logic for Human Behavior Simulation. In: *SCSC 2003, Montreal PQ, Canada, July 20-24*, vol. 18, pp. 11–18 (2003)
26. Oren, T.I., Ghasem-Aghae, N.: Towards Fuzzy Agents with Dynamic Personality for Human Behavior Simulation. In: *SCSC 2003, Montreal PQ, Canada, July 20-24*, pp. 3–10 (2003)
27. Pieterse, V., Kourie, D., Sonnekus, I.: Software Engineering Team Diversity and Performance. In: *SAICSIT 2006, Somerset West South Africa*, vol. 204, pp. 180–186 (2006)
28. Rodríguez, J.: *Formación de grupos de desarrollo de software*. Ediciones Yoltéotl, Guadalajara México (2004)
29. Rutherford, Rebecca, H.: Using Personality Inventories to Help Form Teams for Software Engineering Class Projects. In: *CITiCSE 2001, Canterbury UK*, vol. 33(3), pp. 73–76. ACM, New York (2001)

30. Shen, S., Prior, S.D., White, A.S., Karamanoglu, M.: Using Personality Type Differences to Form Engineering Design Teams. *Engineering Education* 2(2), 54–66 (2007)
31. Sodiya, A.S., Longe, H., Onashoga, S.A., Awodele, O.: An Improved Assessment of Personality Traits in Software Engineering. *Interdisciplinary Journal of Information, Knowledge, and Management* 2, 163–177 (2007)
32. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. *IEEE TSMC* 15, 116–132 (1985)
33. Tomayko, J.E.: Teaching a Project-Intensive Introduction to Software Engineering. SEI Carnegie Mellon University Tech. Rep., Pittsburgh Pennsylvania (1996)
34. U.S. Department of the Interior,
[http://permanent.access.gpo.gov/websites/doigov/
www.doi.gov/octc/typescar.html](http://permanent.access.gpo.gov/websites/doigov/www.doi.gov/octc/typescar.html)
35. Wilde, D.J.: Creative teams, individual development and personality classification. ME310 Course Notes. Mechanical Engineering, Stanford University (2003)
36. Zadeh, L.A.: Fuzzy Sets. *Information and Control* 8, 338–353 (1965)

Academic Timetabling Design Using Hyper-Heuristics

Soria-Alcaraz Jorge A.¹, Carpio-Valadez J. Martín², and Terashima-Marin Hugo¹

¹ Instituto Tecnológico de León, Maestría en Ciencias en Ciencias de la Computación, León Guanajuato, México

soajorgea@gmail.com; jmcarpio61@hotmail.com

² Tecnológico de Monterrey – Center for Intelligent Systems, Monterrey, NL, 64849, México

terashima@itesm.mx

Abstract. The Educational timetabling problem is a common and hard problem inside every educative institution, this problem tries to coordinate Students, Teachers, Classrooms and Timeslots under certain constrains that dependent in many cases the policies of each educational institution. The idea behind hyper-heuristics is to discover some combination of straightforward heuristics to solve a wide range of problems. This paper presents a GA-based method that produces general hyper-heuristics for the educational timetabling design problem using API-Carpio methodology. The GA uses static-length representation; witch involves the complete encoding of a solution algorithm capable to solve schedule design instances. this hyper-heuristic is achieved by learning and testing phases using real instances from Intituto Tecnologico de León producing encouraging results for most of the instances. Finally we analyze the quality of our hyper-heuristic in the context of real Academic timetabling process.

Keywords: Genetic Algorithm, Timetabling, Educational Timetabling, Heuristics, Meta-heuristics, Hyper-heuristics.

1 Introduction

The timetabling problem is one of the most difficult and common problems inside a company, where it must done some activities such time assignment to workers and machinery, this activities are examples of timetabling.

A correct timetabling is fundamental for a good performance of every company, to assign and utilize efficiently the resources that owned. The same problem can be seen into an academic environment where classrooms, teachers and students have a important role in the schedule design process.

An automatic educational timetabling generation provides great benefits not only decreasing the time required to produce it and by, optimizing human and physical resources, but also by improving the precision and quality of schedules eliminating the human factor, in this way, a lot of many errors and conflicts like two or more classes assigned at the same time to a student can be avoided.

The proposal is the creation of a computational module which allows permits academic timetabling design process automatically, saving time and resources for the academic institution where it applies.

The aim of this paper is to explore a novel alternative on the usage of evolutionary several approaches to generate hyper-heuristics for the schedule design process. A hyper-heuristic is used to define a high-level heuristic that controls low-level heuristics [3]. The hyper-heuristic should decide when and where to apply each single low-level heuristic, depending on the given problem state. The choice of low-level heuristic may depend on the features of the problem state, such number of conflicts, quantity of classrooms, numbers of events at the day, etc.

The investigation in this article, presents a method to generate a general hyper-heuristic intended to provide a way to design schedules into an academic institution, The procedure learns the hyper-heuristic by going through a training phase using several real instances of academic schedules of the Instituto Tecnológico de León, México, The general method is based on a static-length genetic algorithm, where the chromosome is conformed of a blocks of heuristics, these blocks are directed by an auxiliary structure called Vector Direction Column (*CDV*) that contains the semester where the heuristics may be applied. also a novel component the *Supervisor* defined as a guiding component that ensures the applicability of the hyper-heuristic in real world problems.

The paper is organized as follows. Section 2 presents the Timetabling design problem, the solution method proposed and its justification. Section 3 contains the experimental setup, the results, their analysis and discussion. Finally Section 4 we include our conclusions and some ideas for future work.

2 Solution Approach

In the literature, one can see that Evolutionary Computation has been used in scheduling and timetabling problems [2], [14], [15],[21]. Recently a novel methodology of schedule design has emerged [1], [4], this API-Carpio methodology has shown based on experience several great results in the educational timetabling design for regular and non-regular students, this methodology is capable to find good solutions minimizing the conflict for *regular students* defined as students who only have in their schedules subjects of the same semester or period, and *non-regular students* defined as students who have subjects from different semesters or periods in their schedules. [1], [4] unfortunately, there still no exist an automated computational module that can perform this method. Also Terashima et.al [26]. used a combination between low-level heuristics and a Genetic Algorithm for Constraint satisfaction strategies in examination timetabling, but did not incorporate the concept of hyper-heuristic.

Evolutionary Computation usually includes several types of evolutionary algorithms [29] one common example is Genetic Algorithms [8],[9],[10],[11], In this research we use a Hyper-heuristic approach to resolve the schedule design problem using the API-carpio methodology.

2.1 The Academic Timetabling Design Problem through API-Carpio Metodology

The solution of academic timetabling (TTP) produces a table of times [25], in general, the TTP problem can be seen as a set of events e , a set of time spaces (slots) s and a set of constrains c between events and slots. Every timetabling process tries to assign every event e into a slot s without violating the constrains c . [25].

So basically our problem consist into achieved a series of elements e assigned into several slots s , how allow us to cover several academic necessities. Each element e have some properties:

- A) One student group is assigned to an event e .
- B) One teacher is assigned to an event e .
- C) One subject is assigned to an event e .
- D) Each event e is assigned to a Slot s .
- E) One classroom is assigned to an event e .

Commonly usually we have more Events e than timeslots s , so our timetabling process needs to assign several Events e to a single timeslot, then our Timetabling primary objective is assign these events to timeslot tending to several constrains like:

- A) Normally the same assignment of events will be repeated each week.
- B) There may be special schedules per Teacher.
- C) We need to offer special subjects per semester.
- D) There are limited number of classrooms and laboratories.
- E) Each teacher can only teach some types of subjects.
- F) Each group has a different schedule.
- G) There are a minimum and maximum number of hours that a teacher need to achieved in a week.

The Api-Carpio methodology[5] presents a mathematical description of this behavior, we have:

$$f(x) = FA(x) + FP(x) + FI(x) \quad (1)$$

Where:

$FA(x)$ =number of students conflicts inside the timetablig x .

$FP(x)$ =number of teacher conflicts inside the timetablig x .

$FI(x)$ =number of classrooms conflicts inside the timetablig x .

With:

$$FA(x) = \sum_{i=1}^k FA_{s_i} \quad (2)$$

$$FP(x) = \sum_{i=1}^k FP_{s_i} \quad (3)$$

$$FI(x) = \sum_{i=1}^k FI_{s_i} \quad (4)$$

Where:

FA_{s_i} = Number of student conflicts into Slot i of our current timetabling.

FP_{s_i} = Number of teacher conflicts into Slot i of our current timetabling.

FI_{s_i} = Number of classroom conflicts into Slot i of our current timetabling.

K = Number of timeslots available for our timetabling.

Then:

$$FA_{s_i} = \sum_{g=1}^{M_{s_i}} \sum_{h=1}^{M_{s_i}-g} (A_{i,g} \wedge A_{i,g+1}) \quad (5)$$

$$FP_{s_i} = \sum_{g=1}^{M_{s_i}} \sum_{h=1}^{M_{s_i}-g} (P_{i,g} \wedge P_{i,g+1}) \quad (6)$$

$$FI_{s_i} = \sum_{g=1}^{M_{s_i}} \sum_{h=1}^{M_{s_i}-g} (I_{i,g} \wedge I_{i,g+1}) \quad (7)$$

Where:

$A_{i,g} \wedge A_{i,g+1}$ = number of students who demand the concurrent enrollment of subjects ($M_{i,g}$ and $M_{i,g+1}$).

$P_{i,g} \wedge P_{i,g+1}$ = number of teachers who demand the concurrent teach of subjects ($M_{i,g}$ and $M_{i,g+1}$).

$I_{i,g} \wedge I_{i,g+1}$ = number of classrooms who serves concurrent the subjects ($M_{i,g}$ and $M_{i,g+1}$).

M_{s_i} = Number of subjects assigned to our Slot i , inside of our timetabling.

$M_{i,g}$ = Subjects assigned to our slot i , inside of our timetabling.

As it can be seen, the educational timetabling is not an easy task, so in this paper we focused into the first part of the methodology, the design of a timetabling only considering the first variable of API-Capio strategy, Students.

Now our problem is to choose a way of combining subjects of an academic career into several timeslots in order to have the less possible student conflicts, a student conflict is defined as a student who have two or more subjects of the same semester into the same timeslot.[5]

2.2 API-Carpio Educational Timetabling Student Model

This model considers timeslots as vectors monitoring students, teachers, classrooms and laboratories simultaneously. This method allows the solution construction through simple heuristics handled by the genetic algorithm, as stated previously we only perform the first step into the API-Carpio methodology, the minimization of Student conflicts into a timetabling. Because we don't perform a complete timetabling we consider our work as a design of a timetabling with a great characteristic, this timetabling will be offer the less student conflict so it is a excellent start point to the timetabling complete process.[5]

Table 1 shows the general structure of a academic career from the point of view API-Carpio methodology.

Table 1. General structure of an academic career

Relative subjects position						
1	ℓ_1	ℓ_1+1	$\ell_1+\ell_2$	$\left(\sum_{j=1}^{n-1} \ell_j\right)+1$	$\left(\sum_{j=1}^{n-1} \ell_j\right)+2$	L
<i>Semester 1</i>		<i>Semester 2</i>		<i>Semester n</i>		
m_{11}	$m_{1\ell_1}$	m_{21}	$m_{2\ell_2}$	m_{n1}	m_{n2}	$m_{n\ell_n}$

Where:

n = number of semesters or periods of the career.

ℓ_i = number of different subjects for semester i , $i=1,2,3,\dots,n$

L = total of subjects of a specific career.

$$L = \sum_{i=1}^n \ell_i = \ell_1 + \ell_2 + \dots + \ell_n \tag{8}$$

M_{ij} = Subject from a i semester and a j position inside the same semester

This vectorization is performed by assigning subjects to different vectors (timeslots). a hard constrain in API-Carpio methodology says : **Never assign two or more subjects from a same semester to a same vector.**[5] This rule ensures that there are no subjects of the same semester at the same timeslot, so a regular student, student who have in her schedule only subjects of the same semester or period, never will have a two o more subjects assigned at the same time. These vectors have a number which are directly related to the number of timeslots offered by the academic institution. An example of this timeslots can be seen in Table 2.

Table 2. Timeslots ID offered by Instituto Tecnológico de León

Hour	Monday & Wednesday	Tuesday & Thursday
7:00-08:40	1	2
8:45-10:25	3	4
10:30-12:10	5	6
12:15-13:55	7	8

API-Carpio Hard constrain can be achieved in the following way: in the first step all the subjects are arranged by semester or period as show in Table 1, then to each subject from first semester is assigned randomly different vector number between l and k , where k is determined by the maximum number of timeslot offered by the institution, for our example we use number from l to 8 , after we proceed similarly with subjects of second, third and N-semester.

Once assigned this number we perform an ordering of all the subjects by this vector number, leaving a partition of the career in independent vectors, where each vector contains subjects of different semesters or periods. An example of how is constructed this Vector shown in Table 3.

Table 3. Vector Constrtuctions

No	SUBJECT	SEMESTER	VECTOR
1	ACM0403	1	1
2	SCM0414	1	6
3	SCB0421	1	7
4	SCE0418	1	5
5	ACH0408	1	4
6	ACM0401	1	2
7	TUT0001	1	3

Now we can read our schedule design on the following way, if one subject of the first semester or period was assigned the number 5 as vector number, this subject will be offered by the institution at 10:30-12:10 on Monday and Wednesday. all the subjects have now assigned a specific timeslot or vector.

2.3 Hyper-Heuristic Approach

As noted, until now we have seen how to model the schedule design by API-carpio methodology, this methodology can be used by an Hyper-heuristic approach, first is necessary to identify the hyper-heuristic concept enunciated by Burke[3], which defines the hyper-heuristic approach as an algorithm that manages the choice of low-level heuristics that must be applied to our problem state, depending on our current space region, This implies that the algorithm must choose which heuristic ingredient will selected to modify the actual state of our problem. A scheme of this approach can be seen on Figure 1.

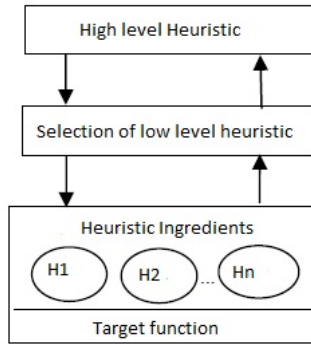


Fig. 1. Common Hyper-heuristic Approach

2.4 The Set of Heuristic Used

As show on the figure 1 is essential in first instance have a set of low-level heuristics, as well a high-level heuristic that choose the low-level ingredients to change the current state of the problem, in our approach we use as high-level heuristic the Genetic Algorithm (GA) because the adaptation of this technique to combinatorial problems, then we need to define a set of low-level heuristics that can been chosen by our GA, this set was constructed by the experience and the experimentation of the authors, we define our low-level heuristics as follows.

H1: Selection of subjects from semesters or periods separated, this heuristic receives a subject and returns another subject from a semester or period with a distance of +2 o -2, for example, this heuristic receives an subject for 6th semester then, the heuristic could chose a subject from 8th semester or 4th semester to return, this check obey the number of subject currently assigned to the schedule.

H2: Selection of a subject that minimizes the sum of current conflicts into the vector, this heuristic works over a vector, looking what subject minimizes the number on conflicts in the current vector, once selected makes the change assigned the subject to the vector.

H3: Selection of a subject that minimizes the product of current conflicts into a vector, this heuristic works over a vector, looking for a subject that minimizes the product of the current conflicts (commonly a zero conflict subject) then perform the assignation of this subject to the vector.

H4: Selection of a subject with the greatest conflict, this heuristic receives a subject then performs a check looking for the worse conflict. finally this worse case subject is assigned to another vector.

This four heuristic are encoded as shows in table 4 for its use by the GA.

Table 4. Codification of each heuristic into the chromosome

H	H1	H2	H3	H4
codification	1	2	3	4

2.5 Combining Heuristics with the Proposed GA Using API-Carpio Methodology

The concept of hyper-heuristics is motivated by the objective to provide more general procedure for optimization [3] Meta-heuristics ,methods usually solve problems by operating directly on the problem [11] [24], Hyper-heuristics deal with the process to choose the right heuristic for solving the problem at hand [29], [3] The aim is to discover a combination of simple heuristics that can perform well on a whole range of problems of the same kind [29] For real applications, exhaustive methods are not a practical approach [3] The search space might be too large, or number and types of constraints may generate a complex space of feasible solutions [9].

It is common to sacrifice quality solutions by using quick and simple heuristics to solve problems. Many heuristics have been developed for specific problems. But, is there a single heuristic for a problem that solves all instances well? , the answer is no [29], [3]. Certain problems may contain features that would make a specific heuristic work well, but those features may not be suitable for another heuristics. The idea with hyper-heuristics is to combine heuristics in such a way that a heuristic's strengths make up for the drawbacks of another [3].

The solution model used in this investigation carries features from previous methodology by Aguayo-Rios [1] and Carpio-Valadez [4] in witch the main objective is perform an optimal academic schedule design and work by Terashima et. al [29].

The basic concept is that, given a problem state P , we perform an algorithmic step to change the state from P to P' . The purpose is to solve a problem by constructing the answer, deciding on the heuristic to apply at each step, then finally determine the quality of the algorithm of solution, that is to say our hyper-heuristic.

A chromosome in our GA represents the sequence of algorithmic steps necessary to constructs a solution, this chromosome is helped by an auxiliary structure, called *CDV*. This *CDV* defines the direction of the heuristic that must be applied to construct a valid solution. moreover another software component called *Supervisor* this supervisor can change arbitrarily the content of the chromosome or its *CDV* in order to hold the solution into the real world constraint. The GA is searching for the best chromosome and *CDV* (representing the hyper-heuristic) which contains the best algorithm that resolves the Schedule design problem using the Api-Carpio Methodology.

Our instances are divided into two groups *training* and *testing* sets, this instances are a real historical ones from the Intituto Tecnológico de León from 2003 to 2006. The two groups have instances from a same career, the first group had 4 instances each one between 400- 450 students and 52 subjects, the test group consist into 2 instances from the same career with 430 and 440 students respectively. The general procedure consist in solve simultaneously all training instances with Each *CDV* and Singles Heuristics, then the GA must search for chromosomes (hyper-heuristics) that brings better results in term of conflicts than the single Heuristics. Finally we use our best found hyper-heuristic to solve a test instance an see how good solves the educational timetabling design problem, it is important to

say that our match point will be the expert because there is no another computational module that uses API-Carpio Methodology to solve timetabling design problem.

Representation. Each chromosome is composed of a series of *blocks*. the number of this series of blocks is $L - v$, where L is the total of subjects on a specific career and v is the number of *vectors or timeslots* offered by the institution, our problem have 52 subjects and 8 timeslots so our chromosome will have 44 blocks, this subtraction is performed because all our heuristic need to work with one subject to make a comparison, so in the beginning we need to initialize each vector or timeslot with one subject, this initialization is done by separating the worse conflicts.

Each block have a number between 1 and 4 this number is the code of a heuristic as seen on Table 4 additionally for each chromosome we have another number array with the same length, this array is called *CDV* that constrains the number of vector or timeslot which will be applied the heuristic coded into the chromosome, a proper example of this representation is shown in Table 5.

Table 5. Representation of chromosome with CDV

Position	1	2	...	$l-v$
Heuristic	3	2	...	4
CDV	8	5	...	3

With the addition of *CDV* we can see our chromosome like an algorithm that can solve the academic timetabling design problem using the API-Carpio methodology, this algorithm can perform it if we move upward the chromosome and apply the step coded, following our example in the table 2, in the first step we apply the heuristic 3 to the *vector or timeslot* 8, Each heuristic assigns one subject to a specific timeslot, in the next step we apply heuristic 2 to timeslot 5 and so on until reach the $L-v$ position. finally we will have all of our subject assigned into our timeslots.

The Fitness Function. The most common criterion to measure the efficiency of a search algorithm used to determine the quality of a timetabling design using API-Carpio methodology is a conflict check this verification counts the number of conflicts into timetabling, A conflict is defined by the API-Carpio methodology as a student who have two or more subjects assigned at the same time. despite the fact that this methodology guarantees zero conflicts with *regular* students, *non-regular* students could have conflicts, so our high-level heuristic need to search the lowest conflict algorithm. To calculate the relative quality of an individual there are 3 steps that must be done:

A)Every timetabling design instance is solved using 4 predefined chromosomes each chromosome have only one type of heuristic, for example the first predefined chromosome only have in each block the value "1" and so on, and the CDV of our current individual to measure.

B)This results are saved into a matrix M , this matrix have a size of pxq , where p is the number of our test instances and q is the number of our predefined chromosomes, in our case is 4.

C) Now we proceed to solve each instance with our current chromosome and *CDV*, this measure is saved into a numeric array *C* of the same size of *p*. The fitness function *FF_i* is computed as follows .

$$FF_i = \sum_{i=1}^p \sum_{j=1}^q (C_i - M_{ij}) \quad (9)$$

Where:

p = Number of our test instances.

q= Number of our predefined chromosomes.

C_i= Matrix with the values of our conflicts checks for each our chromosomes applied to test instances.

M_{ij} = Matrix with the values of our conflicts checks for each our predefined chromosomes applied to test instances.

This evaluation guarantees that, when the general hyper-heuristic produces better results, the fitness function will return lower values so essentially we search for the minimum possible value.

3 Experiments and Results

This section presents the experiments carried out during the investigation and the results obtained. Our training set is composed for several real instances from the career of Engineering in computer system at Instituto Tecnológico de León under the API-Carpio methodology from 2003 to 2008, this instances are composed by a matrix called MMA, this matrix have in its rows and columns subjects from a specific career, its cells contains the number of conflicts that the assignation of both subjects into a same timeslot could do. an example of this matrix is presented in Fig 2.

	ACM0403	SCM0414	SCB0421	SCE0418	ACH0408	ACM0401	ACM0404	SCM0426	ACU0402	SCV0407	SMC0409	SCC0428	ACM0405	ACM0406	SCM0435	SCC0408
ACM0403	4	1	1													
SCM0414	1	10	3	3			6		5	5	8	3	1			
SCB0421	1	3	3	2			2		3	3	3					
SCE0418		3	2	3			3		3	3	3	1				
ACH0408																
ACM0401						4	1	2		1	1	2	2		1	
ACM0404		6	2	3		1	86	44	6	22	37	24			35	35
SCM0426	3					2	44	69	3	20	34	26	13	2	1	
ACU0402	1	5	3	3			6	3	9	8	7	3	1		1	
SCV0407	1	5	3	3		1	22	20	8	36	20	13	5	1	12	8
SMC0409	3	8	3	3		1	37	34	7	20	57	26	4	1	13	10
SCC0428	3	3		1		2	24	26	3	13	26	48	8	2	14	9
ACM0405		1				2		13	1	5	4	8	84	4	52	27
ACM0406								2		1	1	2	4	30	14	5
SCM0435						1	35	1	1	12	13	14	52	14	124	71

Fig. 2. Example of MMA Matrix

The training set was used to generate through our high-level heuristic the best hyper-heuristic that can solve the training set simultaneously, then we use that hyper heuristic to solve another different instance from 2009.

3.1 Experiment Type 1

We perform several tests using different initial population sizes with the aim of taking the time that our AG uses to run 100 generations. The main idea is to analyze the resolution time through hyper-heuristics, our results are shown in Table 6.

Table 6. Time needed to achieve 100 generations

Population	2	4	8	16	32	64	128	256
Time (s)	3.16	53.15	96.10	170.50	257.20	539.15	640.3	1035.41

In general the algorithm performs a reasonable resolution time, considering that it resolves 4 instances simultaneously.

3.2 Experiment Type 2

In this experiment we search for the best hyper-heuristic using the table 6 with changes in some AG features like elitism, selection method, initial population, percentage of mutation and crossover, the best hyper-heuristic was found with the features presented in Table 7 and Fig 3. The idea is to obtain a good hyper-heuristic with a good time and fitness.

Table 7. Best Fitness Achieved

Feature	Value
Initial Population	20
Selection Method	Roulette
Elitism	20%
Crossover	95%
Mutation	1%
Iterations	100
Fitness Achieved	-4750
Time (s)	188.56

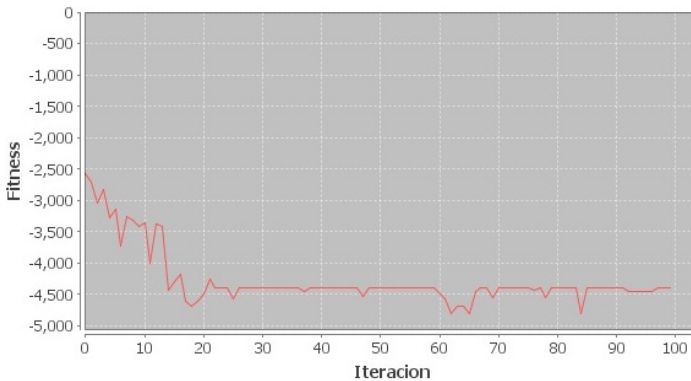


Fig. 3. Performance of our High-level GA

A closer analysis of the components of this hyper-heuristic can be done if we consider their heuristic occurrence frequency as seen on figure 4.

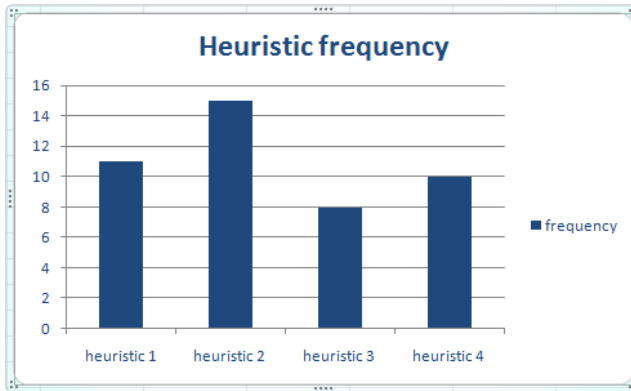


Fig. 4. Heuristic frequency inside our hyper-heuristic

The most important heuristic in this result is heuristic 2, this heuristic search for the minimum sum conflict inside a vector or slot, in the another hand the less important heuristic is number 3, heuristic 1 and 2 have a similar frequency, this behavior show us the importance of each heuristic step in the generation of educational timetabling using API-Carpio methodology.

3.3 Analysis of Our Best Achieved Hyper-Heuristic

After we have our best general hyper-heuristic, we use it with a new instance of schedule design problem, our final conflicts number was 162, this improves the quality of solution against the human expert who takes 1 week in order to achieve a schedule design solution without a proper conflict check . this 162 conflicts means 162 *non-regular* students in the whole career who could have a conflict in theirs schedule, in a deeper study this career have 9 semester so it is possible to find the exact number of conflicts per semester, this number is $162/9 = 18$ and each semester have near 6 subjects so the number of conflicts per subject is $18/6=3$, finally we have 3 students affected per subject but this conflict could be easily corrected if we consider 2 or 3 classes per subjects. Also we have a final hyper-heuristic that is applicable to any instance of computer systems engineering under API-Carpio Methodology without the use of the high-level GA again.

4 Conclusions and Future Work

This documents has described experimental results in schedule design problem under API-Carpio methodology to a real academic institution by an Hyper-heuristic approach which involves several low-level heuristic defined by experience and experimentation. Overall the scheme identifies efficiently general

hyper-heuristics after going through a learning procedure with training and testing phases. When applied to unseen examples those hyper-heuristics solve many of the problems efficiently, in some cases better than the best single heuristic for each instance. Ideas for future work involve the search and development of new low-level heuristics, as also, the application of new techniques inside the high-level GA to perform a better search for the hyper-heuristics.

References

1. Aguayo, R., Carpio-Valadez, J. M. : Optimization and Automatization in task assignation applied to Academic Area. Master Degree Thesis, Instituto Tecnológico de León, México (2009)
2. Aparecido, L.: The school timetabling problem: a focus on elimination of open periods and isolated classes. In: Proceedings 6th International conference on hybrid intelligent systems (HIS 2006). IEEE, Los Alamitos (2006)
3. Burke, E., Hart, E., Kendall, G., Newall, J., Ross, P., Schulenburg, S.: Hyperheuristics: An emerging direction in modern research technology. In: Handbook of Metaheuristics, pp. 457–474. Kluwer Academic Publishers, Dordrecht (2003)
4. Carpio-Valadez, J.M.: Integral Model for the optimal academic task assignation using a heuristic algorithm. In: Investigation in electrical engineering, Mexico (2006)
5. Carpio-Valadez, J.M.: Integral Model for optimal assignation of academic tasks, encuentro de investigacion en ingenieria electrica. ENVIE, Zacatecas, 78–83 (2006)
6. Cowling, P., Chakhlevitch, K.: Using a large set of low level heuristics in hyperheuristics approach to personal scheduling. Studies in computational Intelligence (SCI), pp. 3–29 (2008)
7. Chakhlevitch, K., Cowling, P.: Hyperheuristics Recent Developments. In: Adaptive and multilevel metaheuristics, University of London, pp. 3–29 (2008)
8. Fogel, D.B., Owens, L.A., Walsh, M.: Artificial Intelligence through Simulated evolution. Wiley, New York (1966)
9. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, Reading (1989)
10. Goldberg, D., Korb, B., Deb, K.: Messy genetic algorithms: Motivation, analysis and first results. Complex Systems, 93–130 (1989)
11. Golberg, D.: Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley, New York (1989)
12. Holland, J.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
13. Kendall, G.: A tabu-search hyper-heuristics approach to the examination time-tabling problem at the MARA University of Technology. In: Burke, E.K., Trick, M.A. (eds.) PATAT 2004. LNCS, vol. 3616, pp. 270–293. Springer, Heidelberg (2005)
14. Lewis, R.: A survey of metaheuristics-based techniques for University time-tabling problem. Spinger Online Publications (2007)
15. Limin, H., Kendall, G.: An investigation of Tabu Assisted Hyper-heuristic Genetic Algorithm Automated Scheduling Optimization and planning research group, University of Nottingham (2006)
16. López, B., Jonhston, J.: Academic Task Assignment model using Genetic Algorithms, Intituto Tecnológico de Nuevo laredo, Mexico (2006)

17. Zhipeng, L., Hao, j.-l.: Solving the course timetabling problem with a hybrid heuristic algorithm. In: Dochev, D., Pistore, M., Traverso, P. (eds.) AIMSA 2008. LNCS (LNAI), vol. 5253, pp. 262–273. Springer, Heidelberg (2008)
18. Milena, K.: Solving Timetabling Problems Using Genetic Algorithms. In: Proceedings 27th spring seminar on electronics technology, University of Varna (2004)
19. Minton, S., Johnston, M.D., Phillips, A., Laird, P.: Minimizing conflicts: A heuristic repair method for csp and scheduling problems. *Artificial Intelligence* 58, 161–205 (1992)
20. Minton, S., Phillips, A., Laird, P.: Solving large-scale csp and scheduling problems using a heuristic repair method. In: Proceedings of 8th AAAI Conference, pp. 17–24 (1990)
21. Ozcan, E., Bilgen, B.: Hill climbers and mutational heuristics in hyperheuristics. Yeditep University, Stambul, pp. 202–211 (2006)
22. Padilla, F., Coello, C.: Generation of schedules with Genetic algorithms. In: Proceedings 2nd Congress on Evolutionary Computation, Mexico, pp. 159–163 (2005)
23. Pillay, N., Banzhaf, W.: A genetic Programming Approach to the generation of hyperheuristics for the incapacitated examination timetabling problem. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS (LNAI), vol. 4874, pp. 223–234. Springer, Heidelberg (2007)
24. Rattadilok, P., Gaw, A.: Distributed choice function hyper-heuristics for time-tabling and scheduling. In: Burke, E.K., Trick, M.A. (eds.) PATAT 2004. LNCS, vol. 3616, pp. 51–67. Springer, Heidelberg (2005)
25. Martinez, R., Aguilera, Q.: Educational Timetabling generation with genetic algorithms, *Memorias Segundo congreso de computación evolutiva*. In: COMCEV, Aguascalientes México, pp. 159–163 (2005)
26. Ross, P., Hart, E.: Some observations about GA-based exam Timetabling. University of Edinburg, United Kindom (2005)
27. Russell, S., Norving, P.: *Artificial Intelligence A Modern Approach*, 2nd edn. Prentice Hall, Englewood Cliffs (2007)
28. Terashima-Marín, H., Ross, P.: Evolution of constrain satisfaction strategies in examination timetabling. In: Proceedings GECCO 1999, pp. 635–642 (1999)
29. Terashima-Marín, H., Calleja-Manzanedo, R., Valenzuela-Rendon, M.: Genetic Algorithms for Dynamic Variable Ordering in Constrain Satisfaction Problems. *Advances in Artificial Intelligence Theory* 16, 35–44 (2005)
30. Terashima-Marín, H., Farías-Zárate, C.J., Ross, P., Valenzuela-Rendon, M.: A GA Based Method to Produce Generalized Hyper-heuristics for the 2D-Regular Cutting Stock Problem. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation, Seattle, Washington, USA, pp. 591–598 (2006)
31. Terashima-Marín, H., Ortiz-Bayliss, J.C., Ross, P., Valenzuela-Rend'on, M.: Using Hyper-heuristics for the Dynamic Variable Ordering in Binary Constraint Satisfaction Problems. In: Gelbukh, A., Morales, E.F. (eds.) MICAI 2008. LNCS (LNAI), vol. 5317, pp. 407–417. Springer, Heidelberg (2008)
32. Vazquez, J., Salhi, A.: A Robust Meta-Hyper-Heuristic approach to hybrid flowshop scheduling. *SCI*, vol. 49, pp. 125–142 (2007)

Logistics Optimization Service Improved with Artificial Intelligence

Alberto Ochoa^{1,2}, Yazmani Garcia², and Javier Yañez²

¹ Juarez City University, Mexico

² CIATEC, Mexico

alberto.ochoa@uacj.mx

Abstract. Today the issue of logistics is a very important within companies to the extent that some have departments devoted exclusively to it. This has evolved over time and today is a fundamental aspect in the fight business seeking to consolidate or remain leaders in their field. With the above we know that logistics can be divided into different classes, however, in this regard, our study is based on the timely distribution to the customer with a lower cost, higher sales and better utilization of space resulting in excellent service. Finally, prepare a comparative analysis of the results with respect to another method of optimization solution space.

Keywords: Logistics, Data Mining, Cultural Algorithms, Population Space, Space of Beliefs, Protocol, Bin Packing, Simplex Method.

1 Introduction

Within the area of distribution of purified water, there isn't methodology to what the Service Logistics and space optimization in delivery vehicles, so that the service within the "La Noria" become at the logic or the need. But not have defined a pattern of optimal service logistics.

The optimization problems have been attacked widely in the area of evolutionary computation; this has been due largely to the kindness they have shown to solve such problems [2] to name a few.

This paper addresses the solution of Logistics Service Based on Data Mining in combination with other techniques such as evolutionary algorithms, as is cultural algorithm, once the tool is implemented together with the Bin Packing is to optimize for the space within distribution vehicles purified water.

At present there are many optimization techniques, such is the case of the Simplex Method, and Simplex Method is an iterative process that progressively allows an optimal solution to linear programming problems [15] to name a few, the main feature of this method is that it attacks the problem by restricting its maximum capacity through the vertices of the same [10] to name a few.

2 Methodology

Data mining is a process that uses several data analysis tools to discover patterns and relationships in data that can be used to make valid predictions [16] to name a few. The foundations of data mining is in the artificial intelligence and the statistical and using the models extracted using data mining techniques which addresses the solution to problems of prediction, classification and segmentation [14] to name a few.

The process for conducting the data mining are:

- Selecting the dataset.
- Analysis of the properties of the data.
- Transforming the input data set.
- Select and apply the technique of data mining.
- Evaluate the results. (See Figure 1).

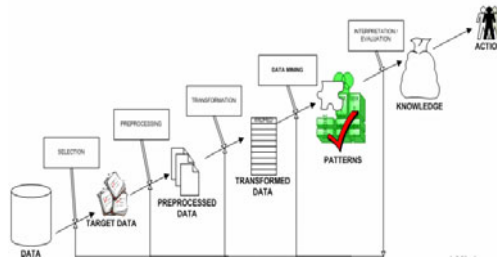


Fig. 1. Diagram of Data Mining

Data mining aims to generate information similar to that which could generate a human analyst: patterns, associations, changes, anomalies and significant structures.

The Cultural algorithms (CA's) are a class of computational models derived from observing the process of cultural evolution in nature as proposed by Reynolds in [11] to name a few. They consist of a population and a belief space as shown in Figure (2). The selected individuals from the population space contribute to cultural knowledge through the role of acceptance [8] to name a few. Cultural knowledge resides in the space of beliefs where it is stored and updated based on individual experiences of either success or failure. The knowledge in the belief space can also be used to influence their individual memories [9] to name a few. The (AC's) in addition to space and space population of beliefs, have a third component of importance: communication protocol, describes how knowledge is exchanged between the first two components [13] to name a few. The population space can support any population based on a computational model, such as Genetic Algorithms and Evolutionary Programming (See Figure 2).

Cultural Algorithms are a dual system of inheritance that characterizes the evolution of human culture in the macro-evolutionary level, which occurs within the

space of beliefs, and micro-evolutionary level, which occurs in the area of population [4] to name a few. The knowledge produced in the population that the space in the micro-evolutionary level is accepted or to be passed to the belief space and used selectively to adjust the knowledge structures there [12] to name a few. This knowledge can then be used to influence the changes made by the population in the next age.

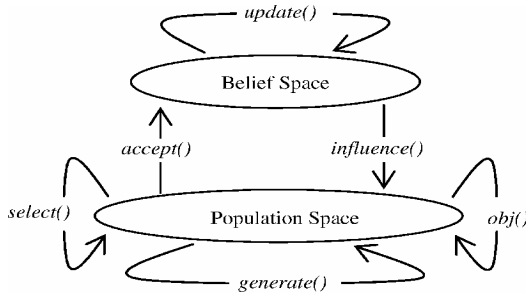


Fig. 2. Diagram Concept of Cultural Algorithms

Cultural algorithms using five kinds of basic knowledge to generate an adequate solution in the search space where the problem is solved. The sources of expertise include regulatory knowledge (ranges of acceptable behavior), situational awareness (or copies of reports of successful or unsuccessful solutions, etc.), domain knowledge (knowledge of objects in the domain of relations they and their interactions), historical knowledge (temporal patterns of behavior), and topographical knowledge (spatial patterns of behavior) [5] to name a few. To make our programming is relatively simple, as shown in the pseudocode in (See Fig 3).

```

Begin
   $t = 0$ ;
  initialize  $B^t, P^t$ 
  repeat
    evaluate  $P^t \{obj()\}$ 
    update ( $B^t, accept(P^t)$ )
    generate ( $P^t, influence(B^t)$ )
     $t = t + 1$ ;
    select  $P^t$  from  $P^{t-1}$ 
  until (termination condition achieved)
End

```

Fig. 3. Cultural Algorithm Pseudocode

The main difference between Cultural Algorithms and Evolutionary Computation other techniques lies in the belief space utilization as well as the cultural influence of the same, as are those that guide us in obtaining optimum best [3] to name a few.

3 Tools Developed

The prototype (see Figure 4) is an intelligent hybrid system developed with the Java programming language that combines data mining techniques to cultural algorithms. He got a map of Fresnillo and divided into 4 zones. To get the coordinates of each colony, then we started building the data warehouse, organized in the following fields: area, district, year, month, day, time and sales.

It was necessary the creation and implementation of algorithms capable of finding information in n dimensions, as well as a data clustering algorithm called K-Means [1] to name a few, which generates data pooling, without predefined classes, based on a function of similarity of the values that have different attributes, done in unsupervised [7] to name a few (i.e., discover patterns or trends in the data). K-Means is a partitioning clustering method (i.e. we start altogether the particular), where partitioning is performed a database of n objects in a set of k groups, seeking to optimize the chosen partitioning criterion. In K-Means each cluster of data is represented by a centroid. K-Means is trying to form k groups with k predetermined before the start of the process. The goal is to minimize the within-group variance [6] to name a few.

With the use of these tools Data Mining, the prototype is able to determine (given a zone and a particular date) in which colonies are more likely to occur at a given time sale. Once these colonies is generated a population of n agents (an agent is the computer simulation of a person), which form a society based on cultural algorithms, which are responsible for determining, over the ages, the course optimal performance.

At the time zero (when initializing the program, and the agents beliefs have an empty space), all agents obtain the information generated from each colony, each propose a route, it will lead to negotiations between the agents to select the best route proposed at a certain time, the belief space will be updated only when the proposed route is better than the previously stored in the belief space, beginning a cycle of improvements that will be interrupted when they occur many times m (iterations in the behavior of agents) without improvements to the paths or when a stop condition is performed.

Once the data mining software, proceeded to the development of software for Bin Packing, the prototype is a hybrid intelligent system developed in Java, using the technique of Cultural Algorithms.

It began with the taking of measures of distribution vehicle on which the study was conducted to determine the space with which states in m^3 and measuring the various presentations and their respective volume, for in doing so raise the problem and their respective restrictions.

Also taken into account the demand of different presentations, to thus more accurately determine a product's usefulness.

Table 1. Product Description

PRODUCT	CAPACITY	% DEMAND	VOLUME	UTILITY
1	20 lt	45	36500cm ³	\$ 10.00
2	1.5 lt	15	26731cm ³	\$ 26.50
3	1 lt	15	18435cm ³	\$ 21.60
4	500 lt	25	18177cm ³	\$ 38.50

Once the measurements were made, it was necessary to create an algorithm capable of finding the right mix in terms of cargo is concerned, so as to optimize the gain of the pickup and hence of the company, the algorithm uses a population basis, and initializes the other as an area of belief at that time its value is unknown.

It makes adjusting the demands for this way fine-tune the value, which based on the percentage of sales of products. Initial population is evaluated based on the problem and the same restriction as shown in the table 1.

$$\begin{aligned}
 \text{Max } z = & \quad r_1 m_1 + r_2 m_2 + \dots + r_n m_n \\
 \text{Subject to} & \quad v_1 m_1 + v_2 m_2 + \dots + v_n m_n \leq V \quad (1) \\
 & \quad m_1, m_2 \dots m_n \geq 0 \text{ Integer} \\
 & \quad V = 1138425\text{cm}^3
 \end{aligned}$$

Where:

- r: Profit per unit.
- v: Volume of each unit.
- m: Are the units of each product type.
- V: Maximum volume capacity.

Any condition which violates the restrictions will be penalized so that only the best combinations are obtained and thus we get an average by which we can have the best individuals and thus influence the next generation based on the mean individuals.

The result (Epochs) is the proposed solution will stop condition from which is repeated 7 times without change, i.e., be = > to previous.

4 Results

The prototype used a database of sales generated at random, with it launched the system functions: information classification, clusters, the generation of routes (see Figure 4).

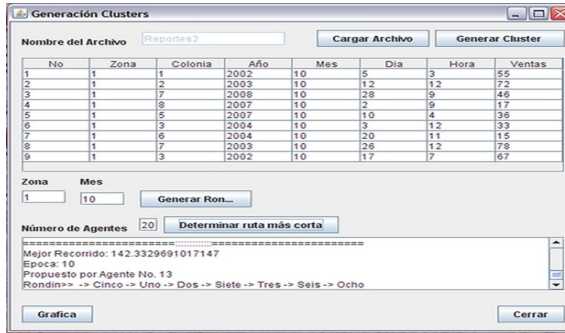


Fig. 4. Generating optimal routes

The system will determine the colonies in which sales have been registered within the specified date range. Based on the number of colonies and vehicle using the K-Means algorithm clusters to be generated immediately after being delivered to a society formed by artificial intelligent agents (representatives of a group of individuals), which will determine the most optimal route for each cluster (see Fig 5).

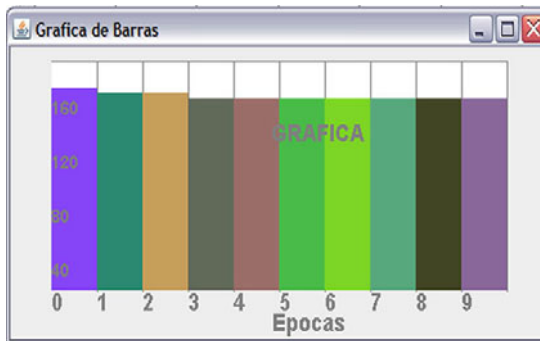


Fig. 5. Graphic Convergence Times represented in épocas

Moreover, the software for the Bin Packing use an initial population of 100 individuals and an area of beliefs with the same number of individuals, and that they could launch the system where the initial population is initialized, the Space of Beliefs, evaluated the results thus able to apply variation operators under the influence of belief space and get different times to achieve the status of unemployment or the convergence of results.

The program determines the best combination of load so as to maximize his profit, that based on the volumes handled for each presentation and the capacity of trucks in m^3 and the demand for each presentation (see figure 6).

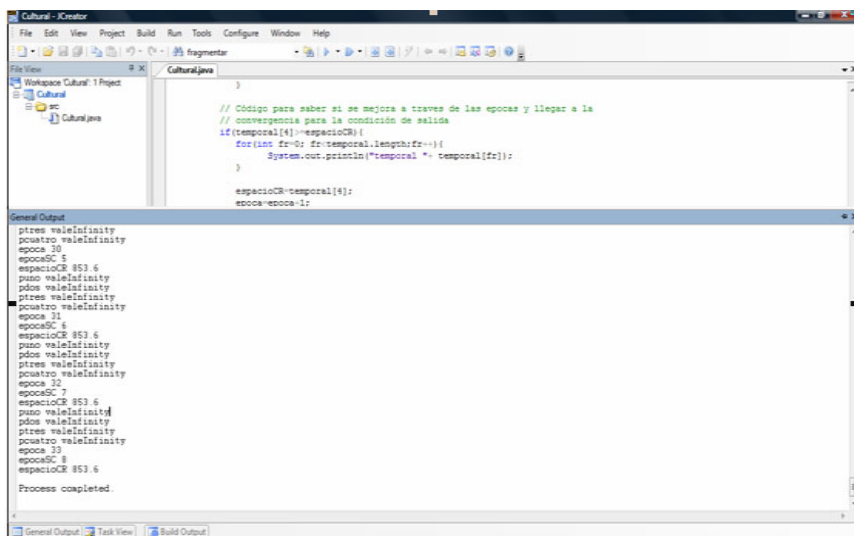


Fig. 6. Generation of Bin Packing

At the time 33 was found the best result after 8 times without change, that the following responses:

Table 2. Results

PRESENTATION	20lt	1.5lt	1lt	500 lt
QUANTITY	16	6	6	9
NET PROFIT \$ 853.60				

However, making a comparison with what is the Simplex Method, we can see that this method gives us less effective results, as we see in Table 3.

Table 3. Result of the Simplex Method

14:37:33		Friday	April	17	2009			
Decision Variable	Solution Value	Unit Cost or Profit c[]	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c[]	Allowable Max. c[]	
1	X1	0.0225	10.0000	0.2250	0	basic	0	M
2	X2	0.1000	26.5000	2.6500	0	basic	0	M
3	X3	0.1500	21.6000	3.2400	0	basic	0	M
4	X4	0.5000	38.5000	19.2500	0	basic	0	M
Objective Function (Max.) =			25.3650					
Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS	
1	C1	0.4500	<=	0.4500	0	0.5000	0	615.8346
2	C2	0.1500	<=	0.1500	0	17.6667	0	63.1710
3	C3	0.1500	<=	0.1500	0	21.6000	0	61.0709
4	C4	0.2500	<=	0.2500	0	77.0000	0	31.1428
5	C5	15,348.1000	<=	1,138,425.0000	1,123,077.0000	0	15,348.1300	M

As we can see from the table, the maximum utility proposed by the Simplex Method is \$ 615.8346, i.e. less than the \$ 237.76 proposed by our program.

5 Conclusions

This research is being used within the purified "La Noria" trying to demonstrate that data mining can be used to increase sales and have a better logistics services, this software has a high value added for the generation and analysis logistics coupled with cultural algorithms responsible for the creation of routes, founded as a tool for decision making, based on the data generated daily by mobile sales. This is intended to provide the product to the larger population that requires service.

Similarly, using Bin Packing algorithm optimizes the space within the distribution units and thus provide the company a way to optimize new and creative through the use of this heuristic.

It concludes with the work that has been done for this kind of logistics in terms of logistics service and space optimization of delivery vehicles is satisfactory and allows the tool to see implemented are good choices. These tools have a high added value because it had not previously been used for this purpose, this further if we consider that until recently there was no practical implementation of these algorithms.

References

1. Ajith, A., et al. (eds.): *Swarm Intelligence in Data Mining*. SCI, vol. 34, p. 270. Springer, Berlin (September 2006) ISBN: 3-540-34955-3
2. Lourdes, A., Carlos, C.: *Algoritmos Evolutivos: un enfoque practico*. In: Grupo, A. (ed.) *Primera Edicion*, México (2007)
3. Alan, C.M.E., Jaime, M.A., Manuel, A.R.J., Calleros, G., Rafael, R., Antonio, D.C.: *Acceso a Repositorios de Objetos de Aprendizaje a Travès de un Sistema de Gestión de Contenidos*. In: *Conferencia Conjunta Iberoamericana sobre Tecnologías de Aprendizaje (CCITA 2009)*, July 6-10, Mèrida Yucatán, México (2009)
4. Cowan George, S., Reynolds Robert, G.: *Acquisition of Software engineering knowledge*, vol. 14. World Scientific, Singapore
5. Dasarathy/Belur, V.: *Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, Orlando (Aprill 2001)
6. Gill, S., et al.: *Data Warehousing. La integracion de la informacilon para la mejor toma de decisions*. Prentice Hall, Mexico (1996)
7. Landa-Becerra, R.: *Uso de Información del Dominio para Mejorar el Desempeño de un Algoritmo Evolutivo*. CINVESTV PhD Thesis (2007)
8. Muñoz, A., Hernandez, A., Villa, E.: *Constrained optimization via particle evolutionary swarm optimization algorithm (PESO)*. In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2005*, pp. 209–216. Association for Computing Machinery, Newyork (2005)
9. Jaime, M.A., Rene, S.S., John Squires, A.R.-g., Ricardo, A., Ricardo, M.G.: *Aprendizaje Multiculturales*. In: *Topicos selectos de Tecnologia Educativa*, Universidad de Colima, Compilador, Acosta Ricardo (April 2010)

10. Ochoa, A., Gonzalez, S.: Simulación Social de una Sociedad Artificial basada en Algoritmos Culturales. *International Journal of South American Archeology (IJSA)* 2011-0626 (2009)
11. Juan, P.: *Méodos y Modelos de Investigación de Operaciones: Modelos Determinísticos*, vol. 1. Limusa
12. Reynolds, G.R., Sverdlink, W.: Problem Solving Using Cultural Algorithms. In: *International Conference on Evolutionary Computation* (1994)
13. Reynolds, G.R., Peng, B., Whallon, R.: *Emergent Social Structures in Cultural Algorithms* (2008)
14. Carlos, R.L.J.: *Uso de la minería de datos con fines predictorios de la infraestructura de seguridad de redes Monterrey, N. L* (2004)
15. Taha, H.: *Investigación de Operaciones. Séptima edición*, México D.F., pp. 71–90. Prentice Hall, Englewood Cliffs (2004)
16. Ian, W.H., Frank, E.: *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd edn. Elsevier, Amsterdam

Accelerometer-Based Hand Gesture Recognition Using Artificial Neural Networks

Francisco Arce and José Mario García Valdez

Tijuana Institute of Technology, Tijuana México
Mario@tectijuana.edu.mx

Abstract. In this paper a hand gesture recognition method using Artificial Neural Networks (ANN) is presented, to evaluate this approach the three-axis accelerometer found in the *Wimote* controller was used to generate a dataset of hand gestures of certain geometric shapes and letters. The gesture recognition process and its evaluation are discussed.

1 Introduction

A simple gesture like pointing is an effective method for most people to communicate with each other, even in the presence of language barriers [2]. This natural way of communication can also be an effective alternative to users who cannot or don't want to use traditional input devices like the keyboard, mouse or controllers for computer interaction. A gesture as defined by Kendon [5] is a form of non-verbal communication in which visible bodily actions communicate particular messages, either in place of speech or together and in parallel with spoken words. Gestures include movement of the hands, face, or other parts of the body. Gesture recognition research focuses on interpreting human gestures. Face and hand gesture recognition have received more attention, but also posture, gait, proxemics, and other human behaviors are also of interest [12].

In this paper we present an ANN approach to hand gesture recognition. The hand gestures were acquired using the controller of the Nintendo *Wii* [9] video game console, this is a common approach as discussed in [8]. As an experiment several users were asked to perform certain hand gestures using the controller and samples of the outputs of accelerometer were recorded for each gesture. This data was used to train an artificial neural network to recognize each gesture. This work is presented as follows: Section 2 contains a brief explanation of the device used, in section 3 related works are discussed, in section 4 the description of the problem and the approach is presented, section 5 presents an evaluation and in Section 6 conclusions drawn and future work are presented.

2 Wii Remote

The Wii Remote, sometimes unofficially nicknamed *Wiimote*, is the primary controller for Nintendo's Wii console [9]. A main feature of the Wii Remote is its motion sensing capability, which allows the user to interact with and manipulate items on screen via pointing and gesture recognition, through the use infrared (IR) tracking technology and a three-axial accelerometer. The Wii Remote has the ability to sense acceleration along three axes through the use of an ADXL330 accelerometer [19]. The readings received are the positive or negative acceleration in three axes a_x , a_y , a_z , with a minimum full-scale range of ± 3 g and also *Roll*, *Pitch* and *Yaw* angular orientation in degrees, a schematic of these readings is shown in Figure 1.

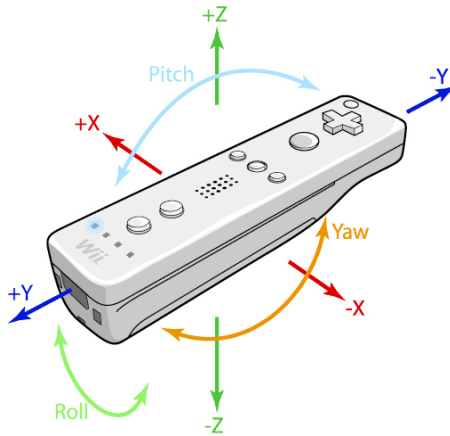


Fig. 1. The *Wiimote* acceleration and orientation readings [8]

These readings can be sent through a Bluetooth [1] connection and there are several libraries to acquire data from this device [17][18]. For this work the C library *wiiuse*[6] was selected, this library has support for motion sensing, IR tracking, and also other Nintendo controllers. The accelerometer is very sensitive and can produce significant noise, to alleviate this; the *wiiuse* library generates a reading (an event) only if a certain threshold is reached. IR tracking can be used to estimate the position of the controller, but has the disadvantage of requiring additional devices and users need to point the controller in a certain direction. Using just the accelerometer readings to estimate the position of the controller is a difficult task, in this approach the hand gesture is not interpreted as the geometric shape, instead the pattern of acceleration changes sampled as the user is drawing the gesture are recognized.

3 Related Work

Gesture recognition can be conducted using different techniques and devices from computer vision to image processing, next some works on sensor based gesture recognition are presented. William Freeman and Craig Weissman [4] have worked on virtual controllers for systems “the act of finding or acquiring a physical controller could require too much time, gestures can be used as an alternative control mechanism”. Also research in robot assisted patient rehabilitation is carried out using different sensors (accelerometers and gyros) worn on the body of a patient [2,16,11,15]. Also C Dinh, D Tantinger, M Struck [3] have worked on a wearable sensor based fall detection system for the elder. Tong Zhang, Jue Wang, Ping Liu and Jing Hou [20], use a cell phone to detect falls. Parera, Cecilio Angulo [10] proposed the use of accelerometers to identify daily activities.

WiiRemote based hand gesture recognition recently has gained attention, Jozef Mlích [8] presents an approach using Hidden Markov Models, Jun Ki Lee [7] have worked on a “Gesture Recognition on Avatar Control System: A Puppeteering System for the Huggable”. Thomas Schlömer and Benjamin Poppinga [14], aim at recognizing gestures to interact with an application in their work they present the design and evaluation of sensor-based gesture recognition using Hidden Markov Model and a Bayesian approach; in their experiments they used 7 users making 15 gestures for each of their figures, their results have an average of 90% of accuracy using a leave-one-out test.

4 Problem Description and Strategy

As an experiment five gestures where defined: *Circle*, *Square*, *Triangle*, and the letters *S* and *Z*. Although these are two-dimensional shapes, they are expressed by users as a hand gesture in a three dimensional space, some of these gestures are also used in work done by Thomas Schlömer [14] and Jozef Mlích [8]. Gestures that involve orientation readings (*Roll*, *Pitch*, *Yaw*) where not considered.

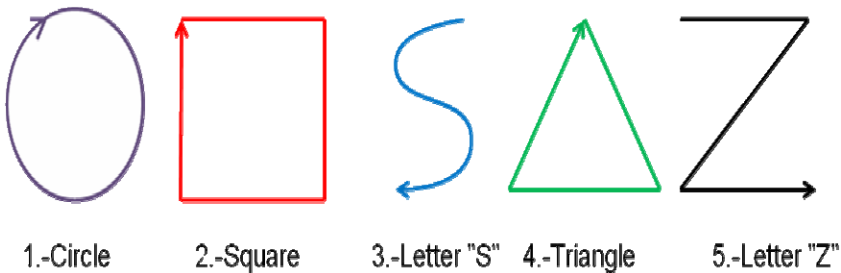


Fig. 2. Gestures made by people to generate the dataset

4.1 Dataset

To read each gesture a Python script program was implemented using the *wiiuse* library. Only the three-axial acceleration was considered for a feature vector of the

gestures, Figure 3 shows an example a plot of the samples for the *Circle* gesture. The plot consists of 50 samples of the acceleration reported by the accelerometer; each sample is defined as a triplet (a_x, a_y, a_z) with acceleration in axes X, Y and Z. Samples were gathered by constantly polling the *wiiuse* library for events, an event is triggered when a significant change in acceleration in any of the axes is encountered. Because of this sampling method, the number of samples for each gesture is not constant, not even for the same gesture and the same user. Also although samples are ordered in time, they cannot be taken in constant intervals of time. Time was not a variable considered in these experiments. Each gesture is recorded as a variable-length list of triplets (a_x, a_y, a_z) .

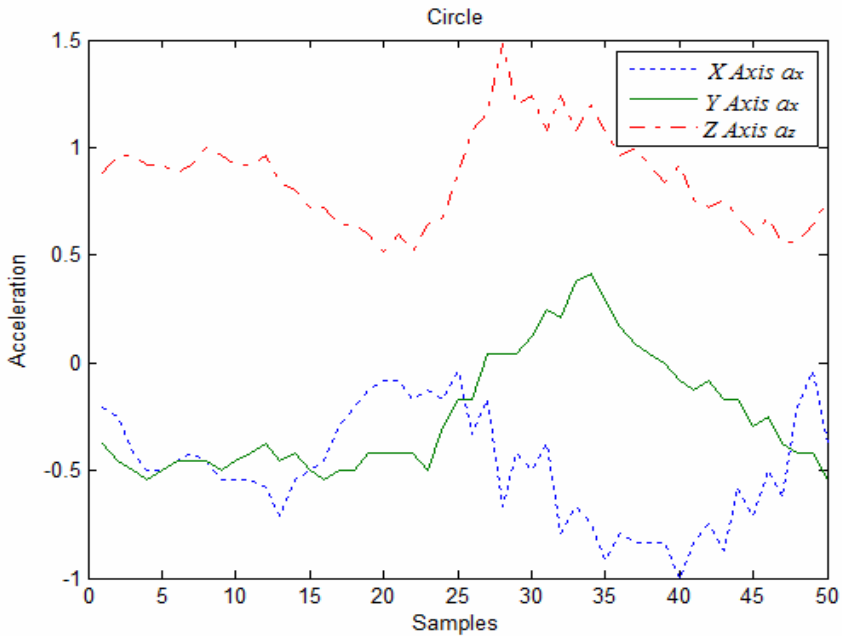


Fig. 3. Acceleration samples for the *Circle* gesture

To generate the dataset, three users were asked to draw in the air the gestures shown in Figure 2, starting at the position where the arrow begins and finishing at the same point. To indicate when each gesture started and ended, the users kept a button pressed as they draw the hand gesture. All users were right-handed, with no disabilities and had used the *WiiRemote* before. Each gesture was repeated ten times by each user. The average number of samples triggered by each user is shown in Figure 4. For recognition purposes a single feature vector was generated for each gesture. Observing the average samples generated by users, a vector with a length of 100 samples was defined. If a gesture is captured with fewer samples, as is usually the case, the remaining samples are filled with zeros, and if the gesture generates more samples, those samples are ignored. The length of the vector

depends on the type of gestures captured, for the simple gestures defined for this experiment, 100 samples were adequate. Figure 5 shows the acceleration samples for 4 hand gestures, made by the same user, it can be observed that there are differences in the pattern and number of samples.

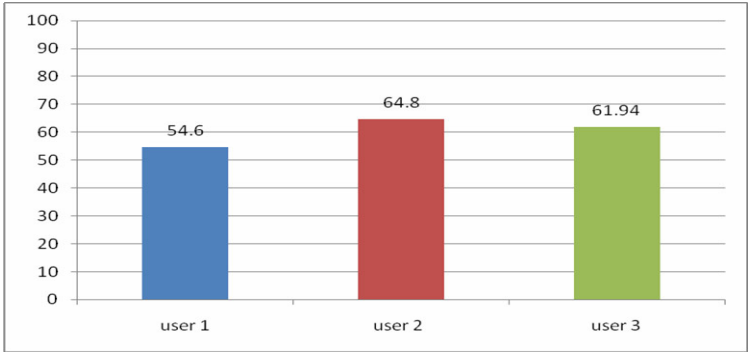


Fig. 4. Average number of samples in gestures made by each user

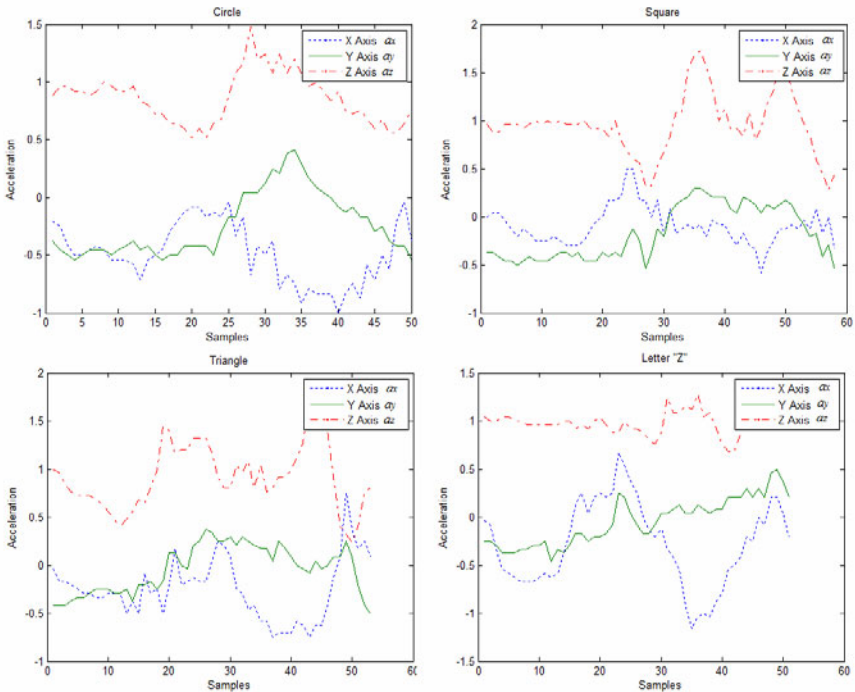


Fig. 5. Examples of the samples of 4 of the gestures used

The samples of each axis were concatenated resulting in a feature vector of 300 samples, an example is shown in Figure 6. The first 100 samples are from a_x , the next 100 from a_y and the rest from a_z . The final dataset consists of 150 feature vectors, ten for each gesture for three users.

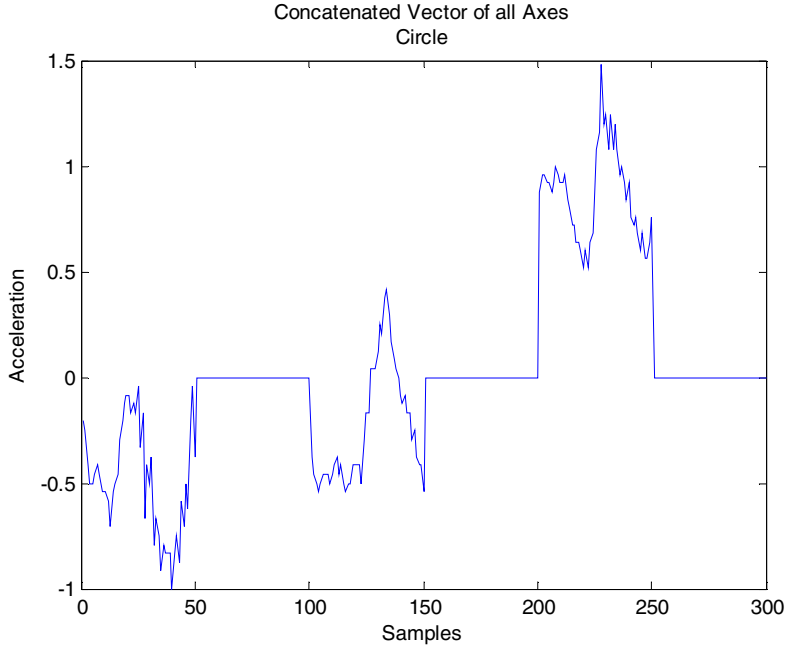


Fig. 6. Example of the resulting feature vector after the concatenation of the samples taken from each axis

Figure 7 shows the feature vectors of the *Circle* gesture made by each user. Although is the same gesture, differences between users are noticeable, but also gestures made by the same user are similar.

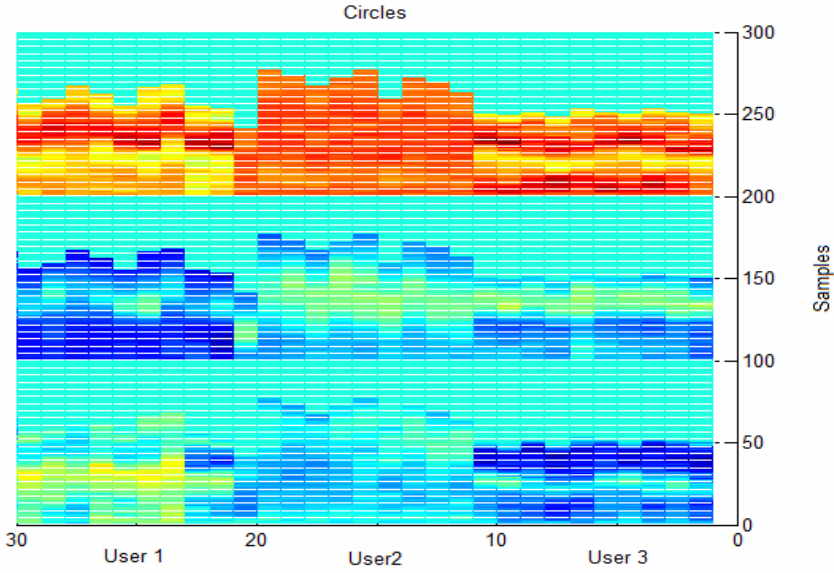


Fig. 7a. Feature vectors for the *Circle* gesture, 10 by each user. (Up view)

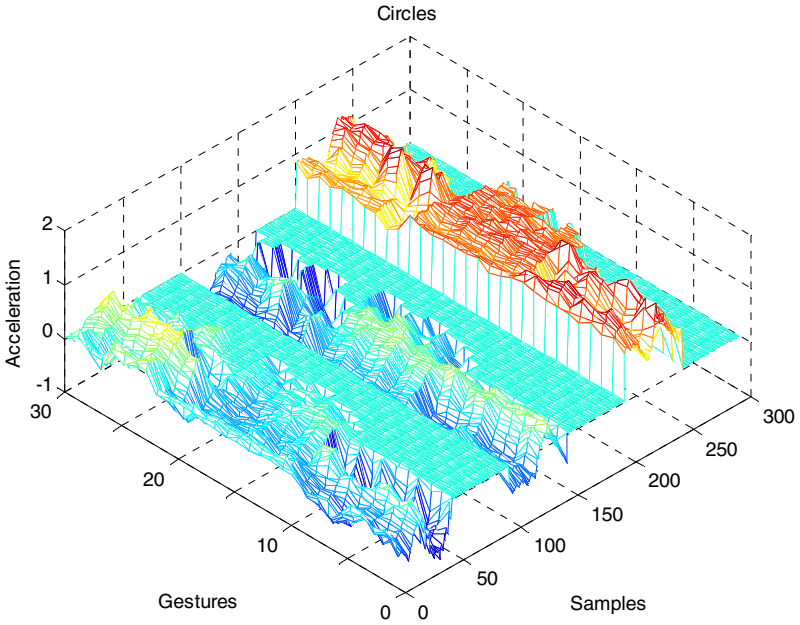


Fig. 7b. Feature vectors for the *Circle* gesture, 10 by each user. (3D view)

4.2 ANN Architecture

Neural networks have been used successfully in other studies of pattern recognition like biometric measurements or predictions of time series, to recognize each pattern may have different optimal architecture for this case is proposed an architecture shown in Figure 8.

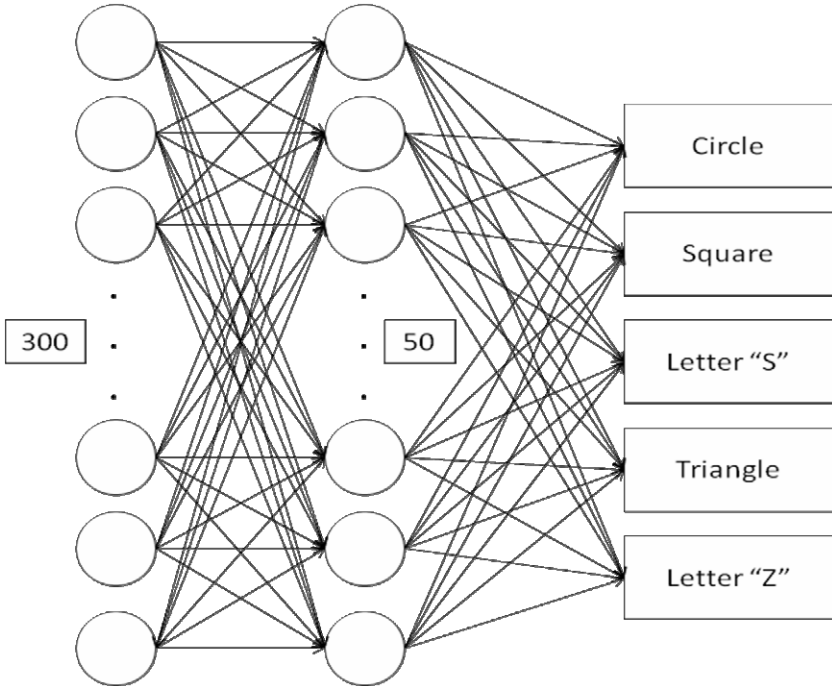


Fig. 8. Proposed Feed Forward Neural Network

This architecture has 300 neurons entries, 50 neurons in the hidden layer and 5 outputs but can be change for including or using fewer gestures.

4.3 Implementation

This neural network is implemented in Python using the PyBrain library [13]. The neural network used is a backpropagation classifier with 0.001 of learning rate, 0.3 of momentum and trained by 5000 epochs.

The recognition process can be summarized in these steps:

1. Data is acquired from the Wiimote via Bluetooth, by a Python script.
2. Variable length vectors were exported to MatLab to create and concatenate the feature vectors of each gesture. Several plots can be generated from the feature vectors.
3. Then a matrix from MatLab is exported to a textfile.

4. A Python script reads the textfile and the Neural Network is trained and evaluated.

5 Evaluation

5.1 First Experiment

In this experiment, the entire feature vectors was used, 24 gestures of each figure for training and the remaining 6 gestures for testing. The network had a hidden layer of 50 neurons and was trained for 5000 epochs with a learning rate of 0.001. The experiment was repeated 10 times with an average of 83.33% Accuracy. In Table 1 the confusion matrix of this experiment is shown.

Table 1. Confusion matrix of all gestures

	Circle	Square	“S”	Triangle	“Z”
Circle	4	0	2	0	0
Square	0	6	0	0	0
“S”	0	0	6	0	0
Triangle	2	0	0	4	0
“Z”	0	0	0	0	6

5.2 Second Experiment

In this experiment, the two gestures with false results in experiment 1 were used, 24 gestures of each figure for training and 6 for testing. The network was trained with a hidden layer of 20 neurons with 5000 epochs and 0.001 of learning rate, with an average of 91.6% accuracy in ten runs. In Table 2 the confusion matrix of this experiment is shown.

Table 2. Confusion matrix of Circle and Triangle Gestures

	Circle	Triangle
Circle	5	1
Triangle	0	6

6 Conclusions

In this paper a present a hand gesture recognition method using Artificial Neural Networks (ANN) is presented. Some preliminary results showed that neural networks are viable tool for hand gesture recognition.

As future work, the dataset of the gestures needs to be extended, with more users and more figures in order to produce more valid experiments and results. Also more features can be proposed to achieve higher accuracy.

References

- [1] Bluetooth, <http://www.bluetooth.com/> (12-04-2009)
- [2] Robert, C.: Recognizing gestures: Interface design beyond point-and-click. Gesture interfaces are evolving in complexity and capability, adding new dimensions to the control of electronic devices from game systems to mobile phones to industrial systems. Technical Editor – EDN (8/16/2007)
- [3] Dinh, C., Tantinger, D., Struck, M.: Automatic Emergency Detection Using Commercial Accelerometers and Knowledge-Based Methods. Institute of Biomedical Engineering and Informatics, Ilmenau University of Technology, Germany Fachhochschule Kärnten, Carinthia University of Applied Sciences, Klagenfurt, Austria Fraunhofer Institute for Integrated Circuits, Erlangen, Germany
- [4] William, F., Craig, W.: Television control by hand gestures, Mitsubishi Electric Research Lab. (1995)
- [5] Kendon, A.: *Gesture: Visible Action as Utterance*. Cambridge University Press, Cambridge (2004) ISBN 0-521-83525-9
- [6] Michael, L.: Library written in C that's connects with several Nintendo wii remotes, <http://www.wiiuse.net/docs/> (12-04-2009)
- [7] Ki, L.J.: Affordable Gesture Recognition Based Avatar Control System: A Puppeteering System for the Huggable. Massachusetts Institute of Technology (October 26, 2007)
- [8] Jozef, M.: Wiimote gesture recognition, Department of Computer Graphics and Multimedia Faculty of Information Technology, Brno University of Technology Božetěchova 2, 612 66 Brno, Czech Republic (2009)
- [9] Nintendo, <http://wii.nintendo.com>
- [10] Jordi, P., Cecilio, A.: Accelerometer signals analysis using svm and decision tree in daily activity identification. Higher Polytechnic School of Engineering of Vila-nova Spain
- [11] Porta, M.: Vision-based user interfaces: methods and applications. *International Journal of Humann Computer Studies* 57(11), 27–73 (2002)
- [12] Matthias, R., Nikolaus, B., Elisabeth, A.: Wave Like an Egyptian - Accelerometer Based Gesture Recognition for Culture Specific Interactions. *British Computer Society* (2007)
- [13] Tom, S., Justin, B., Daan, W., Sun, Y., Martin, F., Frank, S., Thomas, R., Jürgen, S.: PyBrain, Dalle Molle Institute for Artificial Intelligence in Switzerland and the Technische Universität München in Germany
- [14] Thomas, S., Benjamin, P., Niels, H., Susanne, B.: Gesture recognition with a wii controller. In: *TEI 2008: Proceedings of the 2nd international conference on Tangible and embedded interaction*, New York, NY, USA, pp. 11–14 (2008)
- [15] Afshin, S., Yaser, Y., Davis Larry, S.: Employing the Hand as an Interface Device. *Journal of Multimedia* 1(2), 18–29
- [16] Matthew, T., Mathias, K.: *Perceptual Interfaces*, University of California, Santa Barbara UCSB Technical Report 2003-33 (2003)

- [17] Wiiyourself! library (2008), <http://wiiyourself.gl.tter.org/>
- [18] Wiili.org (2009), <http://www.wiili.org/>
- [19] Wisniowski, H.: Analog Devices And Nintendo Collaboration Drives Video Game Innovation With iMEMS Motion Signal Processing Technology. Analog Devices, Inc. (09-05-2006)
http://www.analog.com/en/pressrelease/May_09_2006_ADI_Nintendo_Collaboration/press.html (retrieved 31-01-2009)
- [20] Tong, Z., Jue, W., Ping, L., Jing, H.: Fall Detection by Embedding an Accelerometer in Cellphone and Using KFD Algorithm, Key Laboratory of Biomedical Information Engineering of Education Ministry, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China

Direct and Indirect Neural Identification and Control of a Continuous Bioprocess via Marquardt Learning

Ieroham Baruch¹, Carlos-Roman Mariaca-Gaspar¹, Josefina Barrera-Cortes², and Oscar Castillo³

¹ Department of Automatic Control, CINVESTAV-IPN, Av. IPN No 2508, 07360 Mexico City, Mexico
{baruch, cmariaca}@ctrl.cinvestav.mx

² Department of Biotechnology and Bioengineering, CINVESTAV-IPN, Av. IPN No 2508, 07360 Mexico City, Mexico
jbarrera@cinvestav.mx

³ Graduate Division, Computer Science, Institute of Technology, Tijuana, B.C., Mexico
ocastillo@tectijuana.mx

Abstract. This paper applied a new Kalman Filter Recurrent Neural Network (KFRNN) topology and a recursive Levenberg-Marquardt (L-M) learning algorithm capable to estimate parameters and states of highly nonlinear unknown plant in noisy environment. The proposed KFRNN identifier, learned by the Backpropagation and L-M learning algorithm, was incorporated in a direct and indirect adaptive neural control schemes. The proposed control schemes were applied for real-time recurrent neural identification and control of a continuous stirred tank bioreactor model, where fast convergence, noise filtering and low mean squared error of reference tracking were achieved.

Keywords: Backpropagation learning, continuous stirred tank bioreactor model, direct and indirect adaptive neural control, Kalman filter recurrent neural identification model, Levenberg-Marquardt recursive learning, recurrent trainable neural network controller, sliding mode control.

1 Introduction

The universal approximation abilities of the artificial neural networks to approximate complex non-linear relationships without prior knowledge of the model structure, makes them a very attractive alternative to the classical modeling and control techniques [1], [2], [3]. This property has been proved by the universal approximation theorem, [3]. Among several possible network architectures the ones most widely used are the Feedforward (FFNN) and the Recurrent Neural Networks (RNN). In a feed-forward neural network the signals are transmitted only in

one direction, starting from the input layer, subsequently through the hidden layers to the output layer, which requires applying a tap delayed global feedbacks and a tap delayed inputs to achieve a nonlinear autoregressive moving average neural dynamic plant model. A recurrent neural network has local feedback connections to some of the previous layers. Such a structure is suitable alternative to the first one when the task is to model dynamic systems, and the universal approximation theorem has been proved for the recurrent neural networks too. Several RNN models applications has been described in [4]-[10]. The preferences given to recurrent neural network identification with respect to the classical methods of process identification are clearly demonstrated in the solution of the “bias-variance dilemma”, [3]. Furthermore, the derivation of an analytical plant model, the parameterization of that model and the Least Square solution for the unknown parameters have the following disadvantages: (a) the analytical model did not include all factors having influence to the process behavior; (b) the analytical model is derived taking into account some simplifying suppositions which not ever match; (c) the analytical model did not described all plant nonlinearities, time lags and time delays belonging to the process in hand; (d) the analytical model did not include all process and measurement noises which are sensor and actuator dependent. In (Sage, [11]) the method of invariant imbedding has been described. This method seemed to be a universal tool for simultaneous state and parameter estimation of nonlinear plants but it suffer for the same drawbacks because a complete nonlinear plant model description is needed. Furthermore, the managing of noisy input/output plant data is required to augment the filtering capabilities of the identification RNNs, [12]. Driven by these limitations, a new Kalman Filter Recurrent Neural Network (KFRNN) topology and the recursive Backpropagation (BP) learning algorithm in vector-matrix form has been derived [13] and its convergence has been studied [13], [14]. But the recursive BP algorithm, applied for KFRNN learning, is a gradient descent first order learning algorithm which does not allow to augment the precision and accelerate the learning [12], [14]. Therefore, the aim of this paper was to use a second order learning algorithm for the KFRNN, as the Levenberg-Marquardt (L-M) algorithm is, [15]. The KFRNN with L-M learning was applied for Continuous Stirred Tank Reactor (CSTR) model identification [16], [17]. The application of KFRNNs together with the recursive L-M could prevent all the problems caused by the use of the FFNN, thus improving the learning and the precision of the plant state and parameter estimation in presence of noise. Here, the parameters and states, obtained from the KFRNN identifier will be used to design a Direct and Indirect Adaptive Neural Control (DANC and IANC) of CSTR bioprocess plant model.

2 Kalman Filter RNN

This section is dedicated to the KFRNN topology, the recursive Backpropagation and the recursive Levenberg-Marquardt algorithms for the KFRNN learning. The KFRNN is applied as a state and parameter estimator of nonlinear plants.

2.1 Topology of the KFRNN

Let us consider the linearized plant model (1), (2), represented in a state-space form:

$$X_d(k+1) = A_d(k) X_d(k) + B_d(k) U(k) + \Theta_1(k) \quad (1)$$

$$Y_d(k) = C_d(k) X_d(k) + \Theta_2(k) \quad (2)$$

Where: $E[.]$ means mathematical expectation; the process and measurement noises $\Theta_1(.)$, $\Theta_2(.)$ are white, with $\Theta_1(k)$, $\Theta_2(s)$ and the initial state $X_d(k_0)$ independent and zero mean for all k , s , with known variances $E[X_d(k) X_d^T(k)] = P_0$, $E[\Theta_1(k) \Theta_1^T(k)] = Q(k) \delta(k-\tau)$, $E[\Theta_2(k) \Theta_2^T(k)] = R(k) \delta(k-\tau)$, where $\delta(k-\tau)=1$ if $k= \tau$, and 0 otherwise. The optimal Kalman filter theory is completely described in [4], and we would not repeat it here. For us the Kalman Filter (KF) is a full rank optimal state estimator capable to estimate the system states, to filter the process and measurement noises, taking in hand all plant information available like: input/output plant data, all parameters of the plant model (1), (2), and the given up noise and initial state statistics (mean and variance). The basic Kalman filter equations for the estimated state and output variables are given by:

$$X_e(k+1) = A_e(k) X_e(k) + K_e(k) Y_d(k) + B_d(k) U(k) \quad (3)$$

$$A_e(k) = A_d(k) - K_e(k) C_d(k) \quad (4)$$

$$Y_e(k) = C_d(k) X_e(k) \quad (5)$$

Where: $X_e(k)$ is the estimated state vector with dimension N_e ; $A_e(k)$ is a $(N_e \times N_e)$ closed-loop KF state matrix; $Y_e(k)$ is the estimated plant output vector variable with dimension L ; $K_e(k)$ is the optimal Kalman filter gain matrix with dimension $(N_e \times L)$. This gain matrix is computed applying the optimal Kalman filtering methodology given in [11]. So, the KF performed noise filtration by means of an optimal closed-loop feedback which has the drawback that the feedback amplified the noise components of the error, especially when the feedback gain is high. The second drawback is that the KF design needs complete plant parameter and noise information, which means that if the plant data are incomplete the process noise level is augmented. To overcome this we need to take special measures like to augment the filtering capabilities of the KF. The third drawback is that the KF could not estimate parameters and states in the same time processing noisy measurements with unknown noise statistics, and it will be our task. To resolve this task we need to derive the topology and the BP learning algorithm of a new recurrent KF-like neural network subject to learning and capable to estimate parameters and states in the same time. First of all we could rewrite the equation (3) defining a new extended input vector, containing all available input/output information issued by the plant, and second – we could modify the output equation (5) so to convert it to an output noise filter. After that we obtain:

$$X_e(k+1) = A_d(k) X_e(k) - K_e(k) Y_e(k) + B_2(k) U_e(k) \quad (6)$$

$$B_2 = [B_d ; K_e]; U_e^T = [U ; Y_d] \quad (7)$$

$$Z(k) = C_d(k) X_e(k) \quad (8)$$

$$Y_e(k+1) = A_2 Y_e(k) + Z(k) \quad (9)$$

The obtained new KF RNN topology is given on Fig. 1.

The first layer of the KFRNN represented the plant model (equations (10)-(13)) and the second layer - represented the output noise filtering model (equations (14)-(18)). The KF RNN topology is described by the following equations:

$$X(k+1) = A_1 X(k) + BU(k) - DY(k) \quad (10)$$

$$B = [B_1 ; B_0]; U^T = [U_1 ; U_2] \quad (11)$$

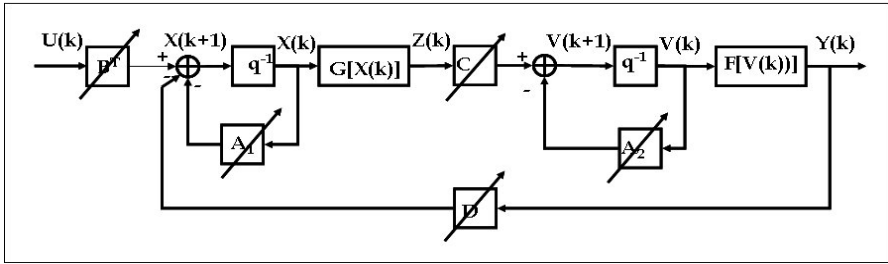


Fig. 1. Block - diagram of the KFRNN topology

$$A_1 = \text{block-diag}(A_{1,i}), |A_{1,i}| < 1 \quad (12)$$

$$Z_1(k) = G[X(k)] \quad (13)$$

$$C = [C_1 ; C_0]; Z^T = [Z_1 ; Z_2] \quad (14)$$

$$V_1(k) = CZ(k) \quad (15)$$

$$V(k+1) = V_1(k) + A_2 V(k) \quad (16)$$

$$A_2 = \text{block-diag}(A_{2,i}), |A_{2,i}| < 1 \quad (17)$$

$$Y(k) = F[V(k)] \quad (18)$$

Where: X , Y , U are vectors of state, output, and augmented input with dimensions N , L , $(M+1)$, respectively, Z is an $(L+1)$ -dimensional input of the feedforward output layer, where Z_1 and U_1 are the $(N \times 1)$ output and $(M \times 1)$ input of the hidden layer; the constant scalar threshold entries are $Z_2 = -1$, $U_2 = -1$, respectively; V is a $(L \times 1)$ pre-synaptic activity of the output layer; the super-index T means vector transpose; A_1 , A_2 are $(N \times N)$ and $(L \times L)$ block-diagonal weight matrices; B and C are $[N \times (M+1)]$ and $[L \times (N+1)]$ - augmented weight matrices; B_0 and C_0 are $(N \times 1)$

and $(L \times 1)$ threshold weights of the hidden and output layers; $F[\cdot]$, $G[\cdot]$ are vector-valued $\tanh(\cdot)$ or $\text{sigmoid}(\cdot)$ -activation functions with corresponding dimensions. Here the input vector U and the input matrix B of the KF RNN are augmented so to fulfill the specifications (7) and the matrix D corresponded to the feedback gain matrix of the KF. The dimension of the state vector of the KF RNN is chosen using the simple rule of thumb, which is: $N=L+M$.

2.2 BP Learning of the KFRNN

So the KF RNN topology corresponded functionally to the KF definition (6)-(9) and ought to be learnt applying the BP learning algorithm derived using the adjoint KF RNN (see Fig.2) based on KF RNN topology applying the diagrammatic method, [18]. The BP learning algorithm, expressed in vector - matricial form is as follows:

$$W(k+1) = W(k) + \eta \Delta W(k) + \alpha \Delta W(k-1); |W_{ij}| < W_0 \quad (19)$$

$$E(k) = Y_d(k) - Y(k); E_1(k) = F'[Y(k)] E(k) \quad (20)$$

$$F'[Y(k)] = [1 - Y^2(k)] \quad (21)$$

$$\Delta C(k) = E_1(k) Z^T(k) \quad (22)$$

$$\Delta A_2(k) = E_1(k) V^T(k) \quad (23)$$

$$\text{Vec}(\Delta A_2(k)) = E_1(k) \circ X(k) \quad (24)$$

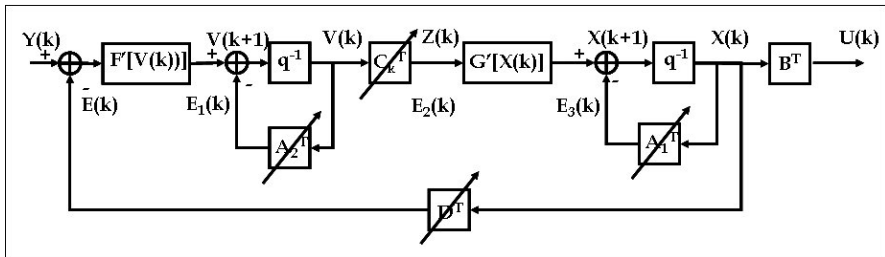


Fig. 2. Block - diagram of the adjoint KFRNN topology

$$E_3(k) = G'[Z(k)] E_2(k); E_2(k) = C^T(k) E_1(k) \quad (25)$$

$$G'[Z(k)] = [1 - Z^2(k)] \quad (26)$$

$$\Delta B(k) = E_3(k) U^T(k) \quad (27)$$

$$\Delta D(k) = E_3(k) Y^T(k) \quad (28)$$

$$\Delta A_1(k) = E_3(k) X^T(k) \quad (29)$$

$$\text{Vec}(\Delta A_1(k)) = E_3(k) \circ X(k) \quad (30)$$

Where: $F'[\cdot]$, $G'[\cdot]$ are derivatives of the $\tanh(\cdot)$ activation functions; W is a general weight, denoting each weight matrix (C , A_1 , A_2 , B , D) in the KF RNN model, to be updated; ΔW (ΔC , ΔA_1 , ΔA_2 , ΔB , ΔD), is the weight correction of W ; Y_d is an L -dimensional output of the approximated plant taken as a reference for KF RNN learning; η , α are learning rate parameters; ΔC is an weight correction of C ; ΔB is an weight correction of B ; ΔD is an weight correction of D , ΔA_1 is the weight correction of A_1 , ΔA_2 is the weight correction of A_2 ; the diagonals of the matrices A_1 , A_2 are denoted by $\text{Vec}(A_1(k))$, $\text{Vec}(A_2(k))$, respectively, where (24), (30) represented their learning as an element-by-element vector products; E , E_1 , E_2 , E_3 , are error vectors (see Fig. 2), predicted by the adjoint KF RNN model. So, the KF RNN is capable to issue parameter and state estimations for control purposes, thanks to the optimization capabilities of the BP learning algorithm, applying the “correction for error” delta rule of learning (see Haykin, [3]). The stability of the KF RNN model is assured by the activation functions $[-1, 1]$ bounds and by the local stability weight bound conditions given by (12), (17). The stability of the KF RNN movement around the optimal weight point has been proved by one theorem and the rate of convergence lemma, (see the Ph.D. thesis of Mariaca [14]). It is stated below.

Theorem of stability of the BP KF RNN applied as a plant identifier, [14]. Let the KF RNN topology is given by equations (10)-(18) (see Fig.1) and the nonlinear plant model, is as follows:

$$X_d(k+1) = G[X_d(k), U(k)] \quad (31)$$

$$Y_d(k) = F[X_d(k)] \quad (32)$$

Where: $\{Y_d(\cdot), X_d(\cdot), U(\cdot)\}$ are output, state and input variables with dimensions L , N_d , M , respectively; $G(\cdot)$, $F(\cdot)$ are vector valued nonlinear functions with respective dimensions.

Under the assumption of KF RNN identifiability made, the application of the BP learning algorithm for C , A_1 , A_2 , B , D , in general vector-matrix form, described by equation (19)-(30), and the learning rates $\eta(k)$, $\alpha(k)$ (here they are considered as time-dependent and normalized with respect to the error) are derived using the following Lyapunov function:

$$L(k) = L_1(k) + L_2(k) \quad (33)$$

Where: $L_1(k)$ and $L_2(k)$ are given by:

$$L_1(k) = \frac{1}{2} e^2(k)$$

$$L_2(k) = \text{tr}(\tilde{W}_{A1(k)} \tilde{W}_{A1(k)}^T) + \text{tr}(\tilde{W}_{A2(k)} \tilde{W}_{A2(k)}^T) + \text{tr}(\tilde{W}_{B(k)} \tilde{W}_{B(k)}^T) + \text{tr}(\tilde{W}_{C(k)} \tilde{W}_{C(k)}^T) + \text{tr}(\tilde{W}_{D(k)} \tilde{W}_{D(k)}^T)$$

Where:

$$\tilde{W}_{A1(k)} = \hat{A}_{1(k)} - A_1^*, \tilde{W}_{A2(k)} = \hat{A}_{2(k)} - A_2^*, \tilde{W}_{B(k)} = \hat{B}(k) - B^*, \tilde{W}_{C(k)} = \hat{C}(k) - C^*, \tilde{W}_{D(k)} = \hat{D}(k) - D^*$$

are vectors of the estimation error and $(A_1^*, A_2^*, B^*, C^*, D^*)$ and $(\hat{A}_{1(k)}, \hat{A}_{2(k)}, \hat{B}(k), \hat{C}(k), \hat{D}(k))$ denote the ideal neural weight and the estimate of neural weight at the k-th step, respectively, for each case.

Then the identification error is bounded, i.e.:

$$L(k+1) = L_1(k+1) + L_2(k+1) < 0 \tag{34}$$

$$\Delta L(k+1) = L(k+1) - L(k) \tag{35}$$

Where the condition for $L_1(k+1) < 0$ is that:

$$\frac{\left(1 - \frac{1}{\sqrt{2}}\right)}{\Psi_{\max}} < \eta_{\max} < \frac{\left(1 + \frac{1}{\sqrt{2}}\right)}{\Psi_{\max}}$$

and for $L_2(k+1) < 0$ we have:

$$\Delta L_2(k+1) < -\eta_{\max} |e(k+1)|^2 - \alpha_{\max} |e(k)|^2 + d(k+1) \tag{36}$$

Note that η_{\max} changes adaptively during the learning process of the network and:

$$\eta_{\max} = \max_{i=1}^5 \{\eta_i\}$$

Where all: the unmodeled dynamics, the approximation errors and the perturbations, are represented by the d-term, and the complete proof of that theorem is given in [14].

2.3 Recursive Levenberg-Marquardt Learning of the KFRNN

The general recursive L-M algorithm of learning, [12], [14], [15] is given by the following equations:

$$W(k+1) = W(k) + P(k) \nabla Y[W(k)] E[W(k)] \tag{37}$$

$$Y[W(k)] = g[W(k), U(k)] \tag{38}$$

$$E^2[W(k)] = \{Y_p(k) - g[W(k), U(k)]\}^2 \tag{39}$$

$$DY[W(k)] = \left. \frac{\partial}{\partial W} g[W, U(k)] \right|_{W=W(k)} \tag{40}$$

Where: W is a general weight matrix ($A_1, A_2, B, C,$ and D) under modification; P is the covariance matrix of the estimated weights updated; $DY[.]$ is an n-dimensional gradient vector; Y is the KFRNN output vector which depends of the updated weights and the input; E is an error vector; Y_p is the plant output vector, which is in fact the target vector. Using the same KFRNN adjoint block diagram (see Fig.2), it was possible to obtain the values of the gradients $DY[.]$ for each updated weight, propagating the value $D(k) = I$ through it. Following the block diagram of Fig. 2, equation (37) was applied for each element of the weight

matrices (A_1 , A_2 , B , C , D) in order to be updated. The corresponding gradient components are as follows:

$$DY[C_{ij}(k)] = D_{1,i}(k)Z_j(k) \quad (41)$$

$$DY[A_{2ij}(k)] = D_{1,i}(k)V_j(k) \quad (42)$$

$$D_{1,i}(k) = F'_i[Y_i(k)] \quad (43)$$

$$DY[A_{ij}(k)] = D_{2,i}(k)X_j(k) \quad (44)$$

$$DY[B_{ij}(k)] = D_{2,i}(k)U_j(k) \quad (45)$$

$$DY[D_{ij}(k)] = D_{2,i}(k)Y_j(k) \quad (46)$$

$$D_{2,i}(k) = G'_i[Z_i(k)]C_iD_{1,i}(k) \quad (47)$$

Therefore, the Jacobean matrix could be formed as:

$$DY[W(k)] = [DY(C_{ij}(k)), DY(A_{2ij}(k)), DY(B_{ij}(k)), DY(A_{ij}(k)), DY(D_{ij}(k))] \quad (48)$$

The $P(k)$ matrix was computed recursively by the equation:

$$P(k) = \alpha^{-1}(k)\{P(k-1) - P(k-1)\Omega^T[W(k)]S^{-1}[W(k)]\Omega^T[W(k)]P(k-1)\} \quad (49)$$

Where the $S(\cdot)$, and $\Omega(\cdot)$ matrices were given as follows:

$$S[W(k)] = \alpha(k)\Lambda(k) + \Omega^T[W(k)]P(k-1)\Omega[W(k)] \quad (50)$$

$$0.97 \leq \alpha(k) \leq 1; 10^3 \leq P(0) \leq 10^6$$

$$\Omega^T[W(k)] = \begin{bmatrix} \nabla Y^T[W(k)] & & & & \\ 0 & \dots & 1 & \dots & 0 \end{bmatrix}; \quad (51)$$

$$\Lambda(k)^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & \rho \end{bmatrix}; 10^{-4} \leq \rho \leq 10^{-6}$$

The matrix $\Omega(\cdot)$ had a dimension (nwx2), whereas the second row had only one unity element (the others were zero). The position of that element was computed by:

$$i = k \bmod(nw) + 1; k > nw \quad (52)$$

After this, the given up topology and learning were applied for the CSTR system identification.

3 Recurrent Trainable NN

This section is dedicated to the topology, the BP and the L-M algorithms of RTNN learning. The RTNN could be obtained from the KFRNN removing the output local and global feedbacks. The RTNN was used as a feedback/feedforward controller.

3.1 Topology of the RTNN

The RTNN model and its learning algorithm of dynamic BP-type, together with the explanatory figures and stability proofs, are described in [13], [14], so only a short description will be given here. The RTNN topology, derived in vector-matrix form, was given by the following equations:

$$X(k+1) = AX(k) + BU(k) \quad (53)$$

$$B = [B_1 ; B_0]; U^T = [U_1^T ; U_2^T] \quad (54)$$

$$A = \text{block-diag} (A_i), |A_i| < 1 \quad (55)$$

$$Z_1(k) = G[X(k)] \quad (56)$$

$$C = [C_1 ; C_0]; Z^T = [Z_1^T ; Z_2^T] \quad (57)$$

$$V(k) = CZ(k) \quad (58)$$

$$Y(k) = F[V(k)] \quad (59)$$

Where: X, Y, U are vectors of state, output, and augmented input with dimensions N, L, (M+1), respectively, Z is an (L+1) –dimensional input of the feedforward output layer, where Z₁ and U₁ are the (Nx1) output and (Mx1) input of the hidden layer; the constant scalar threshold entries are Z₂ = -1, U₂ = -1, respectively; V is a (Lx1) pre-synaptic activity of the output layer; the super-index T means vector transpose; A is (NxN) block-diagonal weight matrix; B and C are [Nx(M+1)] and [Lx(N+1)]- augmented weight matrices; B₀ and C₀ are (Nx1) and (Lx1) threshold weights of the hidden and output layers; F[.], G[.] are vector-valued tanh(.) or sigmoid(.) -activation functions with corresponding dimensions. Equation (55) represents the local stability condition imposed on all blocks of A. The dimension of the state vector X of the RTNN is chosen using the simple rule of thumb which is: N=L+M.

3.2 BP Learning of the RTNN

The same general BP learning rule (19) was used here. Following the same procedure as for the KFRNN, it was possible to derive the following updates for the RTNN weight matrices:

$$E(k) = Y_d(k) - Y(k); E_1(k) = F'[Y(k)] E(k) \quad (60)$$

$$\Delta C(k) = E_1(k) Z^T(k) \quad (61)$$

$$E_3(k) = G'[Z(k)] E_2(k); E_2(k) = C^T(k) E_1(k) \quad (62)$$

$$\Delta B(k) = E_3(k) U^T(k) \quad (63)$$

$$\Delta A(k) = E_3(k) X^T(k) \quad (64)$$

$$\text{Vec}(\Delta A(k)) = E_3(k) \circ X(k) \quad (65)$$

Where ΔA , ΔB , ΔC are weight corrections of the of the learned matrices A , B , and C , respectively; E , E_1 , E_2 , and E_3 are error vectors; X is a state vector; $F'(\cdot)$ and $G'(\cdot)$ are diagonal Jacobean matrices, whose elements are derivatives of the tanh activation functions (see equations (21) and (26)) . Equation (64) represents the learning of the full feedback weight matrix of the hidden layer. Equation (65) gives the learning solution when this matrix is diagonal vA, which is the present case. The initial values of the weight matrices during the learning are chosen as arbitrary numbers inside a small range. The stability of the RTNN model used as a direct controller is assured by the activation functions $[-1, 1]$ bounds and by the local stability weight bound condition given by (55). The stability of the RTNN movement around the optimal weight point has been proved by one theorem (see thesis of Mariaca [14]).

Theorem of stability of the BP RTNN applied as a direct system controller, [14]. Let the RTNN with Jordan Canonical Structure is given by equations (53)-(59) and the nonlinear plant model is given by (31), (32). Under the assumption of RTNN identifiability made, the application of the BP learning algorithm for $A(\cdot)$, $B(\cdot)$, $C(\cdot)$, in general matricial form, described by equation (19), (63)-(65) without momentum term, and the learning rate $\eta(k)$ (here it is considered as time-dependent and normalized with respect to the error) are derived using the following Lyapunov function:

$$L(k) = L_1(k) + L_2(k) \quad (66)$$

Where: $L_1(k)$ and $L_2(k)$ are given by:

$$L_1(k) = \frac{1}{2} e^2(k)$$

$$L_2(k) = \text{tr}(\tilde{W}_{A(k)} \tilde{W}_{A(k)}^T) + \text{tr}(\tilde{W}_{B(k)} \tilde{W}_{B(k)}^T) + \text{tr}(\tilde{W}_{C(k)} \tilde{W}_{C(k)}^T)$$

Where

$$\tilde{W}_{A(k)} = \hat{A}(k) - A^*, \tilde{W}_{B(k)} = \hat{B}(k) - B^*, \tilde{W}_{C(k)} = \hat{C}(k) - C^*$$

are vectors of the estimation error and (A^*, B^*, C^*) and $(\hat{A}_{(k)}, \hat{B}_{(k)}, \hat{C}_{(k)})$ denote the ideal neural weight and the estimate of neural weight at the k -th step, respectively, for each case.

Let us define: $\psi_{\max} = \max_k \|\psi(k)\|$, and $\vartheta_{\max} = \max_k \|\vartheta(k)\|$, where $\psi(k) = \partial o(k) / \partial W(k)$, and $\vartheta(k) = \partial y(k) / \partial u(k)$, where W is a vector composed by all weights of the RTNN, used as a system controller, and $\|\cdot\|$ is an Euclidean norm in \mathfrak{R}^n .

Then the identification error is bounded, i.e.:

$$L(k+1) = L_1(k+1) + L_2(k+1) < 0 \tag{67}$$

$$\Delta L(k+1) = L(k+1) - L(k) \tag{68}$$

Where the condition for $L_1(k+1) < 0$ is that:

$$0 < \eta_{\max} < \frac{2}{\vartheta_{\max}^2 \psi_{\max}^2}$$

and for $L_2(k+1) < 0$, we have:

$$\Delta L_2(k+1) < -\eta_{\max} |e(k+1)|^2 + \beta(k+1) \tag{69}$$

Note that η_{\max} changes adaptively during the learning process of the network and

$$\eta_{\max} = \max_{i=1}^3 \{\eta_i\}$$

Where all: the unmodelled dynamics, the approximation errors and the perturbations, are represented by the β -term, and the complete proof of that theorem and the rate of convergence lemma, are given in [14].

3.3 Recursive Levenberg-Marquardt Learning of the RTNN

The general recursive L-M algorithm of learning [12], [14], [15] is given by equations (37)-(40), where W is the general weight matrix (A, B, C) under modification; Y is the RTNN output vector; E is an error vector; Y_p is the plant output vector. Using the RTNN adjoint block - diagram [12], it was possible to obtain the values of DY [.] for each updated weight propagating $D=I$. Applying equation (40) for each element of the weight matrices (A, B, C) , the corresponding gradient components are obtained as:

$$DY[C_{ij}(k)] = D_{1,i}(k) Z_j(k) \tag{70}$$

$$D_{1,i}(k) = F_i'[Y_i(k)] \tag{71}$$

$$DY[A_{ij}(k)] = D_{2,i}(k) X_j(k) \quad (72)$$

$$DY[B_{ij}(k)] = D_{2,i}(k) U_j(k) \quad (73)$$

$$D_{2,i}(k) = G'_i[Z_i(k)]C_i D_{1,i}(k) \quad (74)$$

Therefore the Jacobean matrix could be formed as:

$$DY[W(k)] = [DY[C_{ij}(k)], DY[A_{ij}(k)], DY[B_{ij}(k)]] \quad (75)$$

The P (k) matrix was computed recursively by equations (49)-(52). Next, the given up RTNN topology and learning were applied for CSTR system control.

4 Adaptive Control Systems Design

This section is dedicated to the design of direct and indirect (sliding mode) adaptive control system using the KF RNN as a nonlinear plant identifier. The RTNN was used as a feedback/feedforward controller in the case of direct adaptive neural control.

4.1 Direct Adaptive Neural Control Scheme

This section describes the direct adaptive control using KFRNN as plant identifier and RTNN as a plant controller (feedback / feedforward). The block-diagram of the control system is given in Fig. 3. The following study describes the linearized model of that closed-loop control system.

Let us present the following z-transfer function representations of the plant, the state estimation part of the KFRNN, and the feedback and feedforward parts of the RTNN controller:

$$W_p(z) = C_p (zI - A_p)^{-1} B_p \quad (76)$$

$$P_i(z) = (zI - A_i)^{-1} B_i \quad (77)$$

$$Q_1(z) = C_c (zI - A_c)^{-1} B_{1c} \quad (78)$$

$$Q_2(z) = C_c (zI - A_c)^{-1} B_{2c} \quad (79)$$

The control systems z-transfer functions (76)-(79) are connected by the following equation, which is derived from Fig. 3, and is given in z-operational form:

$$Y_p(z) = W_p(z) [I + Q_1(z)P_i(z)]^{-1} Q_2(z)R(z) + \theta(z) \quad (80)$$

$$\theta(z) = W_p(z)\theta_1(z) + \theta_2(z) \quad (81)$$

Where: $\theta(z)$ represents a generalized noise term. The RTNN and the KFRNN topologies were controllable and observable, and the BP algorithm of learning was convergent, [12], [14], so the identification and control errors tended to zero:

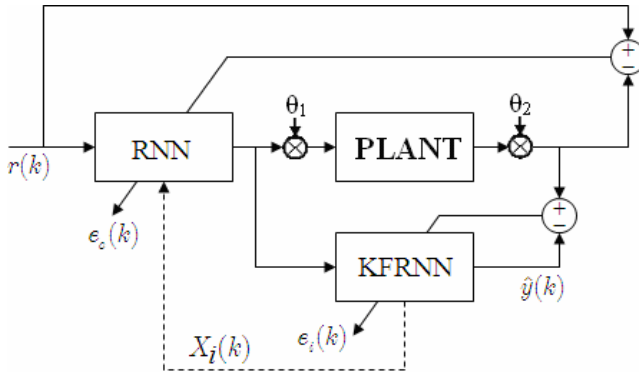


Fig. 3. Block - diagram of the closed-loop neural control system

$$E_i(k) = Y_p(k) - Y(k) \rightarrow 0; k \rightarrow \infty \quad (82)$$

$$E_c(k) = R(k) - Y_p(k) \rightarrow 0; k \rightarrow \infty \quad (83)$$

This means that each transfer function given by equations (76)-(79) was stable with minimum phase. The closed-loop system was stable and the feedback dynamical part of the RTNN controller compensated the plant dynamics. The feed-forward dynamical part of the RTNN controller was an inverse dynamics of the closed-loop system one, which assured a precise reference tracking in spite of the presence of process and measurement noises.

4.2 Indirect Adaptive Control Scheme (Sliding Mode Control)

The indirect adaptive control using the RTNN as plant identifier has been described in [12]. Later the proposed indirect control has been derived as a Sliding Mode Control (SMC) and applied for control of unknown hydrocarbon biodegradation processes, [13], using the KF RNN identifier with BP learning. Here we applied the KF RNN identifier with L-M learning. The block diagram of the indirect adaptive control scheme is shown on Fig. 4. It contains identification and state estimation KF RNN and a sliding mode controller. The stable nonlinear plant is identified by a KF RNN model with topology, given by equations (10)-(18) learned by the stable BP-learning algorithm, given by equations (19)-(30), or using the second order LM-learning algorithm, given by equations (37)-(52). The simplification and linearization of the neural identifier equations (10)-(18), omitting the $DY(\cdot)$ term, leads to the next local linear plant model, extracted from the complete KF RNN model:

$$X(k+1) = A_1 X(k) + BU(k) \quad (84)$$

$$Z(k) = HX(k); H = CG'(Z) \quad (85)$$

Where $G'(\cdot)$ is the derivative of the activation function and $L = M$, is supposed.

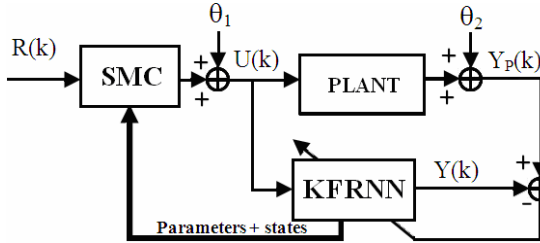


Fig. 4. Block - diagram of the closed-loop system containing KF RNN identifier and a SMC

In [19], the sliding surface is defined with respect to the state variables, and the SMC objective is to move the states from an arbitrary space position to the sliding surface in finite time.

In [20], the sliding surface is also defined with respect to the states but the states of the SISO systems are obtained from the plant outputs by differentiation. In [21], the sliding surface definition and the control objectives are the same. The equivalent control systems design is done with respect to the plant output, but the reachability of the stable output control depended on the plant structure.

In [13], the sliding surface is derived directly with respect to the plant outputs which facilitated the equivalent SMC systems design. Let us define the following sliding surface equation as an output tracking error function:

$$S(k+1) = E(k+1) + \sum_{i=1}^p \gamma_i E(k-i+1); |\gamma| < 1 \quad (86)$$

Where: $S(\cdot)$ is the Sliding Surface Error Function (SSEF) defined with respect to the plant output; $E(\cdot)$ is the systems output tracking error; γ_i are parameters of the desired stable SSEF; p is the order of the SSEF. The tracking error in two consecutive moments of time is defined as:

$$E(k) = R(k) - Z(k); E(k+1) = R(k+1) - Z(k+1) \quad (87)$$

Where: $R(k)$, $Z(k)$ there are L -dimensional reference and output vectors of the local linear plant model. The objective of the sliding mode control systems design is to find a control action which maintains the systems error on the sliding surface which assure that the output tracking error reaches zero in P steps, where $P < N$. So, the control objective is fulfilled if:

$$S(k+1) = 0 \quad (88)$$

Now, let us to iterate (85) and to substitute (84) in it so to obtain the input/output local linear plant model, which yields:

$$Z(k+1) = FX(k+1) = F[AX(k) + BU(k)] \quad (89)$$

From (86)-(87), and (89) it is easy to obtain:

$$R(k+1) - Z(k+1) + \sum_{i=1}^P \gamma_i E(k-i+1) = 0 \quad (90)$$

The substitution of (89) in (90) gives:

$$R(k+1) - FAX(k) - FB U(k) + \sum_{i=1}^P \gamma_i E(k-i+1) = 0 \quad (91)$$

As the local approximation plant model (84), (85), is controllable, observable and stable (see [13], [14]), the matrix A_1 is diagonal, and $L = M$, then the matrix product (HB) , representing the plant model static gain, is nonsingular, and the plant states $X(k)$ are smooth non-increasing functions. Now, from (91) it is easy to obtain the equivalent control capable to lead the system to the sliding surface which yields:

$$U_{eq}(k) = (FB)^{-1} \left[-FAX(k) + R(k+1) + \sum_{i=1}^P \gamma_i E(k-i+1) \right] \quad (92)$$

Following [19], the SMC avoiding chattering is taken using a saturation function instead of sign one. So the SMC takes the form:

$$U^*(k) = \begin{cases} U_{eq}(k) & \text{if } \|U_{eq}(k)\| < U_0 \\ -U_0 U_{eq}(k) / \|U_{eq}(k)\| & \text{if } \|U_{eq}(k)\| \geq U_0 \end{cases} \quad (93)$$

The SMC substituted the multi-input multi-output coupled high order dynamics of the linearized plant with desired decoupled low order one.

5 Description of the CSTR Bioprocess Plant

The CSTR model given in [16], [17] was chosen as an example of RNN applications in system identification and control of biotechnological plants. Numerical values for the parameters and nominal operating conditions of this model are given in Table 1.

The CSTR is described by the following continuous time nonlinear system of ordinary differential equations:

$$\begin{aligned} \frac{dC_A(t)}{dt} = & \frac{Q}{V} (C_{Af} - C_A(t)) - \\ & -k_0 C_A(t) \exp\left(-\frac{E}{RT(t)}\right) \end{aligned} \quad (94)$$

Table 1. Parameters and operating conditions of the CSTR

Parameters	Parameters	Parameters
$Q = 100$ (L / min)	$E / R = 9.95 \times 10^3$ (K)	$Q_{c0} = 103.41$ (L / min)
$C_{Af} = 1.0$ (mol / L)	$-\Delta H = 2 \times 10^5$ (cal / mol)	$hA = 7 \times 10^5$ (cal / min K)
$T_f = T_{jc} = 350$ (K)	$\rho \cdot \rho_c = 1000$ (g / L)	$T_0 = 440.2$ (K)
$V = 100$ (L)	$C_p C_{pc} = 1$ (cal / gK)	$k_0 = 7.2 \times 10^{10}$ (1 / min)

$$\begin{aligned} \frac{dT(t)}{dt} = & \frac{Q}{V} (T_f - T(t)) + \frac{(-\Delta H)C_A(t)}{\rho C_p} \exp\left(\frac{E}{RT(t)}\right) \\ & + \frac{\rho_c C_{pc}}{\rho C_p V} Q_c(t) \left[1 - \exp\left(\frac{-hA}{Q_c(t)\rho_c C_{pc}}\right) \right] (t_{ef} - T(t)) \end{aligned} \quad (95)$$

In this model it is enough to know that within the CSTR, two chemicals are mixed and that they react in order to produce a product compound A at a concentration $C_A(t)$, and that the temperature of the mixture is $T(t)$. The reaction is exothermic and it produces heat which slows down the reaction. By introducing a coolant flow-rate $Q_c(t)$, the temperature can be varied and hence the product concentration can be controlled. Here C_{Af} is the inlet feed concentration; Q is the process flow-rate; T_f and T_{ef} are the inlet feed and coolant temperatures, respectively. All this variables are assumed constant at nominal values. Likewise, k_0 , E/R , V , ΔH , ρ , C_{pc} , C_p , and ρ_c are thermodynamic and chemical constants related to this particular problem. The quantities Q_{c0} , T_0 , and C_{A0} , shown in Table 1, are steady values for a steady operating point in the CSTR. The objective was to control the product compound A by manipulating $Q_c(t)$. The operating values were taken from [16] and [17], where the performance of a NN control system is reported.

6 Simulation Results

Some simulation results of the CSTR biotechnological plant neural identification and control are summarized in this part.

6.1 Simulation Results of Bioprocess Plant Neural Identification

Results of detailed comparative graphical simulation of CSTR KFRNN plant identification by means of the BP and the L-M learning are given in Fig.5 and Fig.6. A 10% white noise with different variance (SEED parameter of MATLAB) for each run was added to the plant inputs and outputs and the behavior of the plant identification was studied accumulating some statistics of the final MSE% (ξ_{sav}) for KFRNN BP and L-M learning. The results for 20 runs are given in Tables 3 and 4.

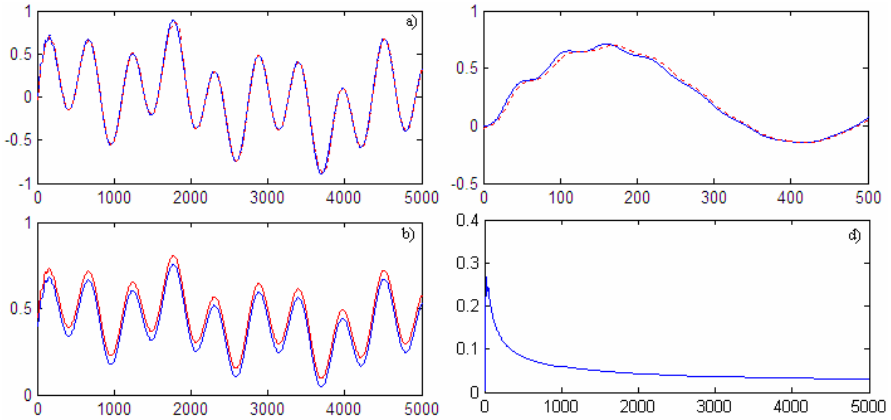


Fig. 5. Graphical results of identification using BP KFRNN learning. a) Comparison of the plant output (continuous line) and KFRNN output (pointed line); b) state variables; c) comparison of the plant output (continuous line) and KFRNN output (pointed line) in the first instants; d) MSE% of identification

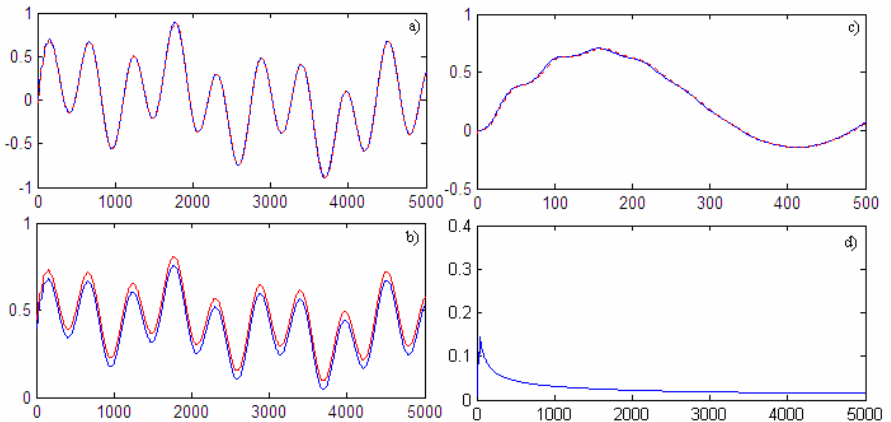


Fig. 6. Graphical results of identification using L-M KFRNN learning. a) Comparison of the plant output (continuous line) and KFRNN output (pointed line); b) state variables; c) comparison of the plant output and KFRNN output in the first instants; d) MSE% of identification

The mean average cost for all runs (ε) of KFRNN plant identification, the standard deviation (σ) with respect to the mean value, and the deviation (Δ) are presented in Table 2 for the BP and L-M algorithms. They were computed by the formulas:

$$\varepsilon = \frac{1}{n} \sum_{k=1}^n \xi_{av_k}, \quad \sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n \Delta_i^2}, \quad \Delta = \xi_{av} - \varepsilon \tag{96}$$

Table 2. Standard deviations and mean average values of identification validation using the BP and L-M algorithms of KF RNN learning

BP algorithm	L-M algorithm
$\varepsilon = 0.9457$	$\varepsilon = 0.8264$
$\sigma = 0.0416$	$\sigma = 0.0188$

Table 3. MSE% of 20 runs of the identification program using the KFRNN BP algorithm

No	1	2	3	4	5
MSE%	0.9559	0.9654	0.8821	0.9614	0.8798
No	6	7	8	9	10
MSE%	0.9444	0.9591	0.9700	0.9685	1.0034
No	11	12	13	14	15
MSE%	0.8523	0.8105	0.9863	0.9038	1.0122
No	16	17	18	19	20
MSE%	0.9688	0.8630	0.8624	0.8521	0.8898

Table 4. MSE% of 20 runs of the identification program using the KFRNN L-M algorithm

No	1	2	3	4	5
MSE%	0.8123	0.8001	0.8553	0.8360	0.8149
No	6	7	8	9	10
MSE%	0.8072	0.8072	0.8285	0.8236	0.8037
No	11	12	13	14	15
MSE%	0.8659	0.8105	0.8269	0.8218	0.8118
No	16	17	18	19	20
MSE%	0.8628	0.8226	0.8514	0.8288	0.8280

The numerical results given in Tables 2, 3, and 4 are illustrated by the bar-graphics in Figures 7 a, b.

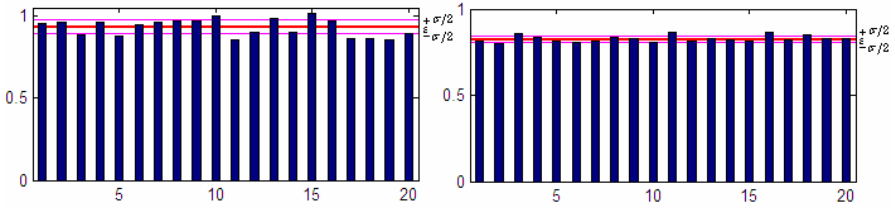


Fig. 7. Comparison between the final MSE% for 20 runs of the identification program: a) using BP algorithm of learning, b) using L-M algorithm of learning

The comparative results showed inferior MSE%, ε , and σ for the L-M algorithm with respect to the BP one.

6.2 Simulation Results of Bioprocess Plant Adaptive Neural Control

The graphical simulation results of DANC using the L-M algorithm of learning are presented in Fig.8 where the final MSE% was 0.854% for the L-M algorithm of learning. Similar results are given on Fig.9 for the indirect SMC control. The final value of the MSE% obtained for the indirect SMC using the L-M algorithm of learning for the KFRNN identifier is of 0.434%. The graphical results and the obtained final MSE% showed that the indirect SMC control is about twice times more precise than the DANC due to the utilization of the estimated states and parameters in

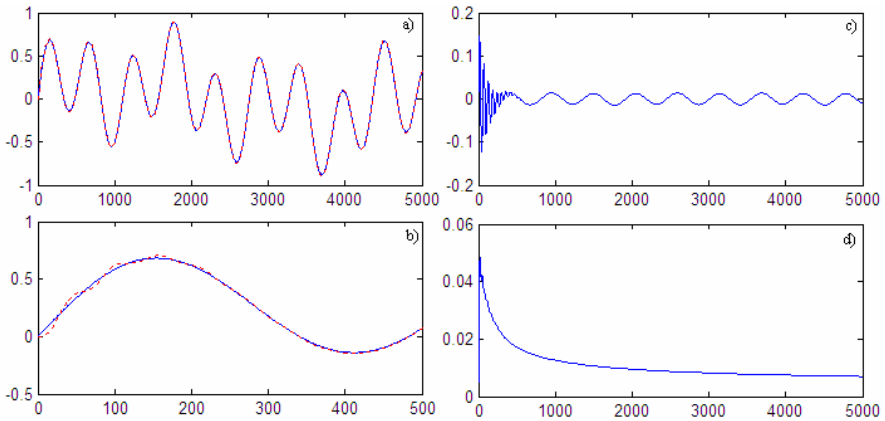


Fig. 8. Detailed graphical simulation results of CSTR plant DANC using L-M learning. a) comparison between the plant output and the reference signal; b) comparison between the plant output and the reference signal in the first instants; c) control signal; d) MSE% of control

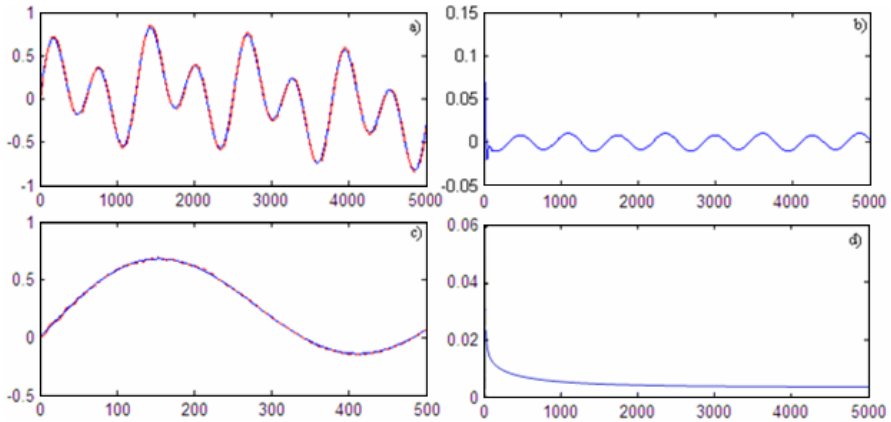


Fig. 9. Detailed graphical simulation results of CSTR plant Sliding Mode Indirect Control using L-M KFRTNN learning. a) comparison between the plant output and the reference signal; b) comparison between the plant output and the reference signal in the first instants; c) control signal; d) MSE% of control

that case, and also due to the SMC algorithm of control which substitute the plant dynamics by a decoupled lower order one.

7 Conclusions

The paper proposed a new KFRNN model for system identification and state estimation of nonlinear plants. The KFRNN is learnt by the first order BP and by the second order L-M recursive learning algorithms. The validating results of system identification reported here gave priority of the L-M algorithm of learning over the BP one which is paid by augmented complexity. The estimated states and parameters of the plant, obtained by this Kalman filter recurrent neural network model are used for direct and indirect adaptive trajectory tracking control system design. The applicability of the proposed neural control system, learnt by the BP and L-M algorithms, was confirmed by simulation results with a CSTR plant. The results showed good convergence of the two algorithms applied. The graphical and numerical validation identification results showed that the L-M algorithm of learning is more precise but more complex then the BP one. The control results of DANC and IANC (SMC) showed a great precision of reference tracking (the final MSE% is 0.854% for the DANC and 0.434% for the indirect SMC). The better results obtained with the indirect SMC are due to the utilization of the estimated states and parameters in that case, and also due to the SMC algorithm of control which substitute the plant dynamics by a decoupled lower order one.

Acknowledgments. The graduated Ph.D. student Carlos-Roman Mariaca-Gaspar is thankful to CONACYT for the scholarship received during his studies at the Department of Automatic Control, CINVESTAV-IPN, MEXICO CITY, MEXICO.

References

1. Narendra, K.S., Parthasarathy, K.: Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Transactions on Neural Networks* 1(1), 4–27 (1990)
2. Hunt, K.J., Sbarbaro, D., Zbikowski, R., Gawthrop, P.J.: Neural Network for Control Systems (A survey). *Automatica* 28, 1083–1112 (1992)
3. Haykin, S.: *Neural Networks, a Comprehensive Foundation*, 2nd edn., Section 2.13, pp. 84–89, Section 4.13, pp. 208–213. Prentice-Hall, Upper Saddle River (1999)
4. Chen, S., Billings, S.A.: Neural Networks for Nonlinear Dynamics System Modeling and Identification. *International Journal of Control* 56, 319–346 (1992)
5. Boskovic, J.D., Narendra, K.S.: Comparison of Linear, Nonlinear and Neural – Network - Based Adaptive Controllers for a Class of Fed - Batch Fermentation Processes. *Automatica* 31, 817–840 (1995)
6. Ku, C.C., Lee, K.Y.: Diagonal Recurrent Neural Networks for Dynamic Systems Control. *IEEE Transactions on Neural Networks* 6(1), 144–156 (1995)
7. Jin, L., Gupta, M.: Stable Dynamic Backpropagation Learning in Recurrent Neural Networks. *IEEE Transactions on Neural Networks* 10, 1321–1334 (1999)
8. Melin, P., Castillo, O.: *Hybrid Intelligent Systems for Pattern Recognition*. Springer, Germany (2005)
9. Mastorocostas, P.A., Theocharis, J.B.: A Stable Learning Algorithm for Block - Diagonal Recurrent Neural Networks: Application to the Analysis of Lung Sounds. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics* 36(2), 242–254 (2006)
10. Kazemy, A., Hosseini, S.A., Farrokhi, M.: Second Order Diagonal Recurrent Neural Network. In: *Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE, Vigo, Spain, June 2007*, pp. 251–256. IEEE Inc., New York (2007)
11. Sage, A.P.: *Optimum Systems Control*. In: *Library of Congress Catalog Number*, pp. 68–20862. Prentice-Hall Inc., Englewood Cliffs (1968)
12. Baruch, I.S., Mariaca-Gaspar, C.R.: A Levenberg-Marquardt Learning Applied for Recurrent Neural Identification and Control of a Wastewater Treatment Bioprocess. *International Journal of Intelligent Systems* 24, 1094–1114 (2009)
13. Baruch, I.S., Mariaca-Gaspar, C.R., Barrera-Cortes, J.: Recurrent Neural Network Identification and Adaptive Neural Control of Hydrocarbon Biodegradation Processes. In: Hu, X., Balasubramaniam, P. (eds.) *Recurrent Neural Networks*, Ch. 4, pp. 61–88. I-Tech Education and Publishing KG, Vienna (2008) ISBN 978-953-7619-08-4
14. Mariaca Gaspar, C.R.: *Topologies, Learning and Stability of Hybrid Neural Networks, Applied for Nonlinear Biotechnological Processes*, Ph. D. Thesis (in spanish), Baruch, I.S., Martínez-García, J.C (thesis directors), Department of Automatic Control, CINVESTAV-IPN, Mexico City, 3 (July 2009)
15. Ngia, L.S., Sjöberg, J.: Efficient Training of Neural Nets for Nonlinear Adaptive Filtering Using a Recursive Levenberg Marquardt Algorithm. *IEEE Trans. on Signal Processing* 48, 1915–1927 (2000)
16. Zhang, T., Guay, M.: Adaptive Nonlinear Control of Continuously Stirred Tank Reactor Systems. In: *Proceedings of the American Control Conference, Arlington, June 25–27*, pp. 1274–1279 (2001)
17. Lightbody, G., Irwin, G.W.: Nonlinear Control Structures Based on Embedded Neural System Models. *IEEE Trans. on Neural Networks* 8, 553–557 (1997)

18. Wan, E., Beaufays, F.: Diagrammatic Method for Deriving and Relating Temporal Neural Network Algorithms. *Neural Computations* 8, 182–201 (1996)
19. Young, K.D., Utkin, V.I., Ozguner, U.: A Control Engineer's Guide to Sliding Mode Control. *IEEE Transactions on Control Systems Technology* 7(3), 328–342 (1999)
20. Levent, A.: Higher Order Sliding Modes, Differentiation and Output Feedback Control. Fridman, L.M.(guest ed.) *International Journal of Control*, Special Issue Dedicated to Vadim Utkin on the Occasion of his 65th Birthday 9/10 (June 15-July 10, 2003) ISSN 0020-7179
21. Eduards, C., Spurgeon, S.K., Hebden, R.G.: On the Design of Sliding Mode Output Feedback Controllers. Fridman, L.M.(guest ed.) *International Journal of Control*, Special Issue Dedicated to Vadim Utkin on the Occasion of his 65th Birthday 76(9/10), 893–905 (June 15-July10, 2003) ISSN 0020-7179

Simple Tuning of Type-2 Fuzzy Controllers

Eduardo Gómez-Ramírez, Patricia Melin, and Oscar Castillo

Laboratory of Advance Technology Research and Development, LIDETEA,
La Salle University, Benjamín Franklin 47 Col. Condesa, 06140, México,
D.F. México, Division of Graduate Studies and Research, Tijuana Institute of Technology,
Calzada Tecnológico s/n, Fracc. Tomas Aquino, Tijuana, Mexico

Abstract. The number of applications in the industry using PID controllers is bigger than fuzzy controllers. One reason is the problem of the tuning, because it implies the handling of a great quantity of variables like: the shape, number and ranges of the membership functions, the percentage of overlap among them and the design of the rule base. The problem is more complicated when it is necessary to control multivariable systems due that the number of parameters. The importance of the tuning problem implies to obtain fuzzy system that decrease the settling time of the processes in which it is applied, or in some cases, the settling time must be fixed to some specific value. In this paper a very simple algorithm is presented for the tuning of a type-2 fuzzy controller using only one variable to adjust the performance of the system. The results are based on the relation that exists between the shape of the membership functions and the settling time. Some simulations are presented to illustrate the proposed algorithm.

1 Introduction

The implementation of a fuzzy controller is not so complex. The knowledge of the dynamics of the system can be used to find a very good estimation of the rules and the sets of membership functions. But to find an optimal and well tuned fuzzy controller is another problem. Normally the methodology for tuning fuzzy controller is heuristic work and for every problem it is necessary to consider some elements like: the bandwidth, the error in steady state, or the settling time. In some cases it is possible to use this information to find the optimal parameters of the controller. In the case of a PID controller it is necessary to find three parameters (proportional gain, derivative time, and integral parameters). In the case of fuzzy controllers, there are many parameters to compute like, number of membership functions used, the ranges of every function, the rules, the shape, the percentage of overlap, etc. [1] [2] [3]. Many researchers prefer to use the very well known PID controller instead of a fuzzy controller due the tuning complexity. Many times this is a very important reason to avoid the use in the industry of this type of “intelligent” controller.

The tuning of any type of controller implies the adjustment of the parameters to obtain a desired behavior or a good approach with a minimal error of the desired

response. The different methods published in the area for the problem of fuzzy controllers' tuning use methodologies like evolutionary computation [1] [2] [3] [4] and artificial neural networks [5] [6] [7] [8] [9]. These methods search the solution according to objective functions, parameter estimation, gradient error, etc., but in many cases these alternatives have serious convergence problems and a very complex mathematical representation. The computation time is big, or it is possible that the solution computed is only a local minimum of the general solution. The problem is even more complex for type-2 fuzzy logic controllers because there are more design parameters to find.

In this paper a very simple method for tuning type-2 fuzzy controllers is presented using only two parameters. In this case, the paper is based in the relation between the stabilization time and the range of the type-2 membership functions. This paper uses the results of a previous work [10] using only one parameter for tuning all the membership functions. The paper is structured in the following way: In section 2 the relationship that exists between the positions of the membership functions with the transfer characteristic is presented. In section 3 the non-linear system used is described with the controller's description for the tuning. The section 4 outlines an algorithm of parametric tuning that modifies the operation points that define the group of membership functions. In the section 5 the results of the simulations are shown for different values of the tuning factor and different graphics that show the behavior of the settling time in function of the tuning factors and finally the conclusions of the work.

2 Type-2 Fuzzy Logic Systems

If for a type-1 membership function, as in Figure 1, we blur it to the left and to the right, as illustrated in Figure 2, then a type-2 membership function is obtained. In this case, for a specific value x' , the membership function (u'), takes on different values, which are not all weighted the same, so we can assign an amplitude distribution to all of those points.

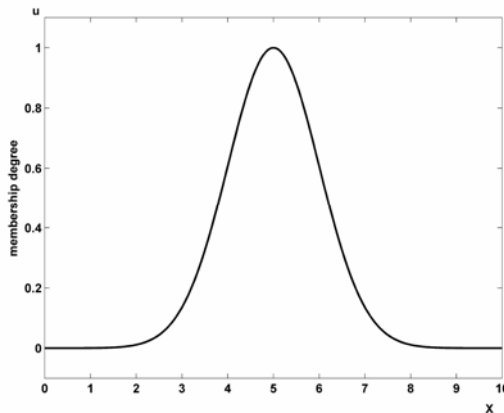


Fig. 1. Type-1 membership function

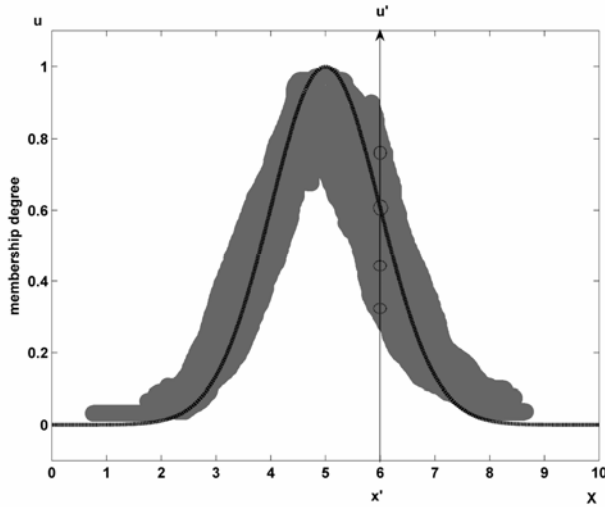


Fig. 2. Blurred type-1 membership function

Doing this for all $x \in X$, we create a three-dimensional membership function – a type-2 membership function – that characterizes a type-2 fuzzy set. A type-2 fuzzy set \tilde{A} , is characterized by the membership function:

$$\tilde{A} = \{(x,u), \mu_{\tilde{A}}(x,u) \mid \forall x \in X, \forall u \in J_x \subseteq [0,1]\} \tag{1}$$

in which $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$. Another expression for \tilde{A} is,

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u) / (x,u) \quad J_x \subseteq [0,1] \tag{2}$$

Where $\int \int$ denotes the union over all admissible input variables x and u . For discrete universes of discourse \int is replaced by \sum . In fact $J_x \subseteq [0,1]$ represents the primary membership of x , and $\mu_{\tilde{A}}(x,u)$ is a type-1 fuzzy set known as the secondary set. Hence, a type-2 membership grade can be any subset in $[0,1]$, the primary membership, and corresponding to each primary membership, there is a secondary membership (which can also be in $[0,1]$) that defines the possibilities for the primary membership. Uncertainty is represented by a region, which is called the footprint of uncertainty (FOU). When $\mu_{\tilde{A}}(x,u) = 1, \forall u \in J_x \subseteq [0,1]$ we have an interval type-2 membership function, as shown in Figure 3. The uniform shading for the FOU represents the entire interval type-2 fuzzy set and it can be described in terms of an upper membership function $\bar{\mu}_{\tilde{A}}(x)$ and a lower membership function $\underline{\mu}_{\tilde{A}}(x)$.

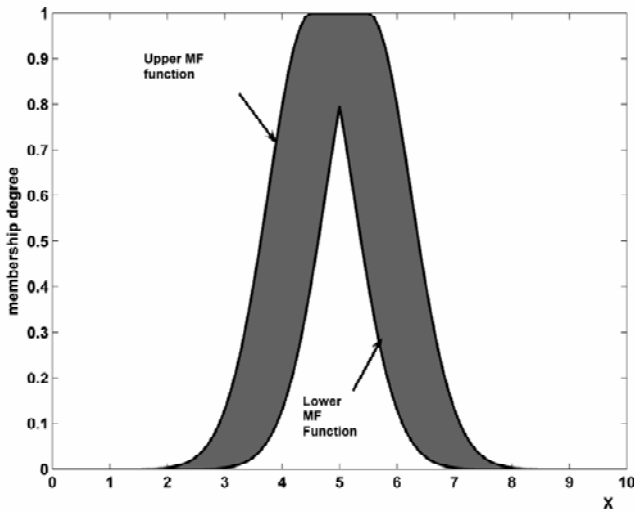


Fig. 3. Interval type-2 membership function

A FLS described using at least one type-2 fuzzy set is called a type-2 FLS. Type-1 FLSs are unable to directly handle rule uncertainties, because they use type-1 fuzzy sets that are certain. On the other hand, type-2 FLSs, are very useful in circumstances where it is difficult to determine an exact membership function, and there are measurement uncertainties.

It is known that type-2 fuzzy sets enable modeling and minimizing the effects of uncertainties in rule-based FLS. Unfortunately, type-2 fuzzy sets are more difficult to use and understand than type-1 fuzzy sets; hence, their use is not widespread yet. As a justification for the use of type-2 fuzzy sets are mentioned at least four sources of uncertainties not considered in type-1 FLSs:

1. The meanings of the words that are used in the antecedents and consequents of rules can be uncertain (words mean different things to different people).
2. Consequents may have histogram of values associated with them, especially when knowledge is extracted from a group of experts who do not all agree.
3. Measurements that activate a type-1 FLS may be noisy and therefore uncertain.
4. The data used to tune the parameters of a type-1 FLS may also be noisy.

All of these uncertainties translate into uncertainties about fuzzy set membership functions. Type-1 fuzzy sets are not able to directly model such uncertainties because their membership functions are totally crisp. On the other hand, type-2 fuzzy sets are able to model such uncertainties because their membership functions are themselves fuzzy. A type-1 fuzzy set is a special case of a type-2 fuzzy set; its secondary membership function is a subset with only one element, unity.

A type-2 FLS is again characterized by IF-THEN rules, but its antecedent or consequent sets are now of type-2. Type-2 FLSs, can be used when the circumstances are too uncertain to determine exact membership grades such as when the training data is corrupted by noise. Similar to a type-1 FLS, a type-2 FLS includes a fuzzifier, a rule base, fuzzy inference engine, and an output processor, as we can see in Fig. 4. The output processor includes type-reducer and defuzzifier; it generates a type-1 fuzzy set output (from the type-reducer) or a crisp number (from the defuzzifier). Now we will explain each of the blocks of Figure 4.

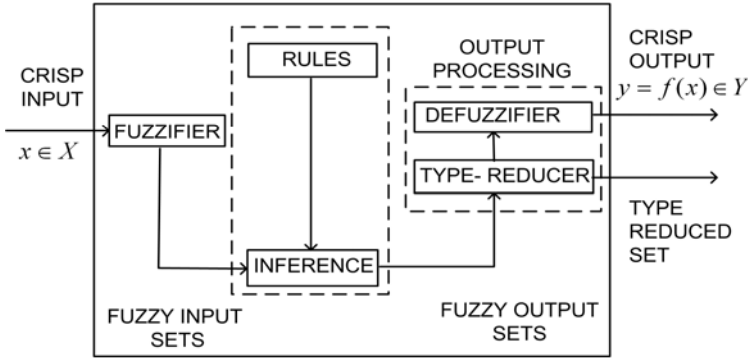


Fig. 4. Type-2 Fuzzy Logic System

2.1 Fuzzifier

The fuzzifier maps a crisp point $\mathbf{x}=(x_1, \dots, x_p)^T \in X_1 X_2 \dots X_p \equiv \mathbf{X}$ into a type-2 fuzzy set \tilde{A}_x in \mathbf{X} , interval type-2 fuzzy sets in this case. We will use type-2 singleton fuzzifier, in a singleton fuzzification, the input fuzzy set has only a single point on nonzero membership. \tilde{A}_x is a type-2 fuzzy singleton if $\mu_{\tilde{A}_x}(x) = 1/1$ for $\mathbf{x}=\mathbf{x}'$ and $\mu_{\tilde{A}_x}(x) = 1/0$ for all other $\mathbf{x} \neq \mathbf{x}'$.

2.2 Rules

The structure of rules in a type-1 FLS and a type-2 FLS is the same, but in the latter the antecedents and the consequents will be represented by type-2 fuzzy sets. So for a type-2 FLS with p inputs $x_1 \in X_1, \dots, x_p \in X_p$ and one output $y \in Y$, Multiple Input Single Output (MISO), if we assume there are M rules, the l th rule in the type-2 FLS can be written as follows:

$$R^l: \text{IF } x_1 \text{ is } \tilde{F}_1^l \text{ and } \dots \text{ and } x_p \text{ is } \tilde{F}_p^l, \text{ THEN } y \text{ is } \tilde{G}^l \tag{3}$$

$$l=1, \dots, M$$

2.3 Inference

In the type-2 FLS, the inference engine combines rules and gives a mapping from input type-2 fuzzy sets to output type-2 fuzzy sets. It is necessary to compute the join \sqcup , (unions) and the meet \sqcap (intersections), as well as extended sup-star compositions (sup star compositions) of type-2 relations. If $\tilde{F}^1 \times \dots \times \tilde{F}^p = \tilde{A}^l$, equation (3) can be re-written as

$$R^l : \tilde{F}^1 \times \dots \times \tilde{F}^p \rightarrow \tilde{G}^l = \tilde{A}^l \rightarrow \tilde{G}^l \quad l=1, \dots, M \quad (4)$$

R^l is described by the membership function $\mu_{R^l}(\mathbf{x}, y) = \mu_{R^l}(x_1, \dots, x_p, y)$, where

$$\mu_{R^l}(\mathbf{x}, y) = \mu_{\tilde{A}^l \rightarrow \tilde{G}^l}(\mathbf{x}, y) \quad (5)$$

can be written as:

$$\begin{aligned} \mu_{R^l}(\mathbf{x}, y) &= \mu_{\tilde{A}^l \rightarrow \tilde{G}^l}(\mathbf{x}, y) = \mu_{\tilde{F}_1^l}(x_1) \sqcap \dots \sqcap \mu_{\tilde{F}_p^l}(x_p) \sqcap \mu_{\tilde{G}^l}(y) \\ &= [\sqcap_{i=1}^p \mu_{\tilde{F}_i^l}(x_i)] \sqcap \mu_{\tilde{G}^l}(y) \end{aligned} \quad (6)$$

In general, the p -dimensional input to R^l is given by the type-2 fuzzy set \tilde{A}_x whose membership function is

$$\mu_{\tilde{A}_x}(\mathbf{x}) = \mu_{\tilde{x}_1}(x_1) \sqcap \dots \sqcap \mu_{\tilde{x}_p}(x_p) = \sqcap_{i=1}^p \mu_{\tilde{x}_i}(x_i) \quad (7)$$

where $\tilde{X}_i (i=1, \dots, p)$ are the labels of the fuzzy sets describing the inputs. Each rule R^l determines a type-2 fuzzy set $\tilde{B}^l = \tilde{A}_x \circ R^l$ such that:

$$\mu_{\tilde{B}^l}(y) = \mu_{\tilde{A}_x \circ R^l} = \sqcup_{x \in X} [\mu_{\tilde{A}_x}(\mathbf{x}) \sqcap \mu_{R^l}(\mathbf{x}, y)] \quad y \in Y \quad l=1, \dots, M \quad (8)$$

This equation is the input/output relation in Figure 4 between the type-2 fuzzy set that activates one rule in the inference engine and the type-2 fuzzy set at the output of that engine.

In the FLS we used interval type-2 fuzzy sets and meet under product t-norm, so the result of the input and antecedent operations, which are contained in the firing set $\sqcap_{i=1}^p \mu_{\tilde{F}_{ii}}(x_i \equiv F^l(\mathbf{x}'))$, is an interval type-1 set,

$$F^l(\mathbf{x}') = \left[f^l(\mathbf{x}'), f^{-l}(\mathbf{x}') \right] \equiv \left[f^l, f^{-l} \right] \quad (9)$$

Where

$$f^l(\mathbf{x}') = \mu_{-\tilde{F}_1^l}(x_1') * \dots * \mu_{-\tilde{F}_p^l}(x_p') \quad (10)$$

And

$$\bar{f}^l(\mathbf{x}') = \bar{\mu}_{\bar{F}_1^l}(x_1^i) * \dots * \bar{\mu}_{\bar{F}_p^l}(x_p^i) \quad (11)$$

where * is the product operation.

2.4 Type Reducer

The type-reducer generates a type-1 fuzzy set output, which is then converted in a crisp output through the defuzzifier. This type-1 fuzzy set is also an interval set, for the case of our FLS we used center of sets (cos) type reduction, Y_{cos} which is expressed as:

$$Y_{\text{cos}}(\mathbf{x}) = [y_l, y_r] = \int_{y_l^i \in [y_l^i, y_r^i]} \dots \int_{y_r^M \in [y_l^M, y_r^M]} \int_{f_l^i \in [f_l^i, f_l^i]} \dots \int_{f_r^M \in [f_r^M, f_r^i]} 1 / \frac{\sum_{i=1}^M f^i y^i}{\sum_{i=1}^M f^i} \quad (12)$$

this interval set is determined by its two end points, y_l and y_r , which corresponds to the centroid of the type-2 interval consequent set \tilde{G}^i ,

$$C_{\tilde{G}^i} = \int_{\theta_i \in J_{y_l}} \dots \int_{\theta_i \in J_{y_r}} 1 / \frac{\sum_{i=1}^N y_i \theta_i}{\sum_{i=1}^N \theta_i} = [y_l^i, y_r^i] \quad (13)$$

before the computation of $Y_{\text{cos}}(\mathbf{x})$, we must evaluate equation (13), and its two end points, y_l and y_r . If the values of f_i and y_i that are associated with y_l are denoted f_l^i and y_l^i , respectively, and the values of f_i and y_i that are associated with y_r are denoted f_r^i and y_r^i , respectively, from equation (13), we have

$$y_l = \frac{\sum_{i=1}^M f_l^i y_l^i}{\sum_{i=1}^M f_l^i} \quad (14)$$

$$y_r = \frac{\sum_{i=1}^M f_r^i y_r^i}{\sum_{i=1}^M f_r^i} \quad (15)$$

The values of y_l and y_r define the output interval of the type-2 fuzzy system, which can be used to verify if training or testing data are contained in the output of the fuzzy system. We will consider this measure of covering the data as one of the design criteria in finding an optimal interval type-2 FS. The other criteria, is that the length of this output interval should be as small as possible to be optimal.

2.5 Defuzzifier

From the type-reducer we obtain an interval set Y_{cos} , to defuzzify it we use the average of y_l and y_r , so the defuzzified output of an interval singleton type-2 FLS is

$$y(\mathbf{x}) = \frac{y_l + y_r}{2} \tag{16}$$

In this paper we will be more interested on the left and right bounds of the type-reduction as we will be checking if the output interval is covering the training or testing data. In this respect defuzzification is not as relevant.

3 Type-2 Fuzzy Controller Characterization

The fuzzy controller's behavior, considering its output speed, sensitivity and reaction under disturbances can be described using the transfer characteristic and the position of the operation points [10]. This is related with the choice of the fuzzy controller's gain dy/dx (where y is the output and x is the input of the system), in different regions of the domain x .

Case 1. For a flat slope in the middle of the domain x and increasing slopes toward increasing $|x|$ values, choose larger distances between operations points in the middle of domain (see figure 5). This means:

$$\text{For } |x_2| > |x_1| \Rightarrow |dy/dx|_{x_2} > |dy/dx|_{x_1} \tag{1}$$

Case 2. For a steep slope in the middle of the domain x and decreasing slopes toward increasing $|x|$ values, choose smaller distances between operations points in the middle of domain (see figure 6). This means:

$$\text{For } |x_2| > |x_1| \Rightarrow |dy/dx|_{x_2} < |dy/dx|_{x_1} \tag{2}$$

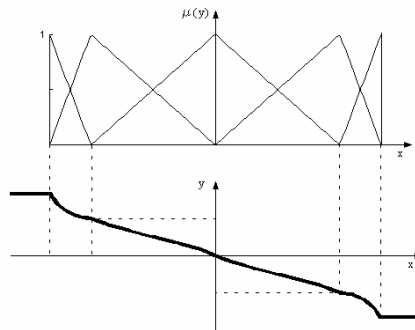


Fig. 5. Relationship between the location of the membership functions and the transfer characteristic for the case 1

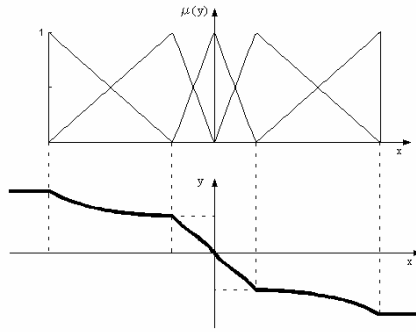


Fig. 6. Relationship between the location of the membership functions and the transfer characteristic for the case 2

Option 1 should be chosen if for small errors a slow reaction to disturbances of the system under control is required. Option 2 should be chosen if for small errors the system is supposed to be sensitive with respect to disturbances.

In the previous figures the values for the intervals of the membership function are important for the slopes and the speed of the controller response. If the membership functions “expand” (figure 5) then the response is slower than a compress group of membership functions (figure 6).

4 Dynamic System and Fuzzy Controller

For the analysis and simulations with the tuning algorithm a second order system has been considered:

$$\frac{1}{0.45s^2 + 2s + 1} \tag{3}$$

overdamped with a damping ratio $\xi = 1.4907$ and a natural frequency $\omega_n = 1.4907 \text{ rad/s}$.

The fuzzy controller designed for the control of the plant described previously is a system TISO (two inputs-one output) where the inputs are the error and the change of error while the output is the control action. Each one of the controller’s variables has been divided in 5 fuzzy regions. The fuzzy associative memory, integrated by 25 rules, it is shown in the Figure 8 and the surface in Figure 7.

Table 1. Controller Fuzzy variables

Input variables		Output variable
<i>error</i>	<i>change of error</i>	<i>control action</i>
GN: Big negative	GN: Big negative	DG: Big diminution
MN: Medium negative	MN: Medium negative	DP: Small diminution
Z: Zero	Z: Zero	M: Hold
MP: Medium positive	MP: Medium positive	AP: Small increase
GP: Big positive	GP: Big positive	AG: Big increase

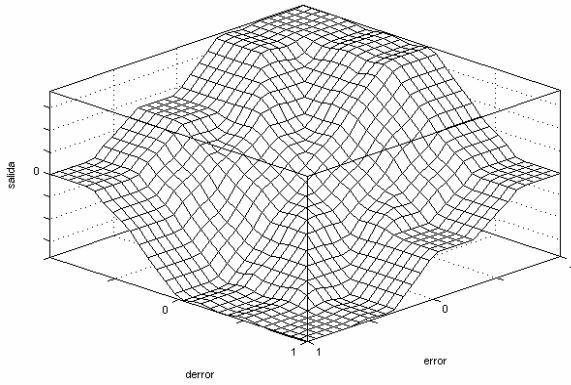


Fig. 7. Example of control surface of a fuzzy controller

e						
	e	GN	MN	Z	MP	GP
GN		AG	AG	AP	DP	DG
MN		AG	AP	M	M	DG
Z		AG	AP	M	DP	DG
MP		AG	M	M	DP	DG
GP		AG	AP	DP	DG	DG

Fig. 8. Fuzzy associative memory for the control system

The membership functions were defined in triangular shape for the middle and in a trapezoidal shape in the extremes; such that always have it overlap in the grade of membership $\mu(x) = 0.5$. These membership functions will be considered later as the initial conditions for the proposed algorithm. The control surface for the fuzzy controller under its initial conditions is similar to Figure 9.

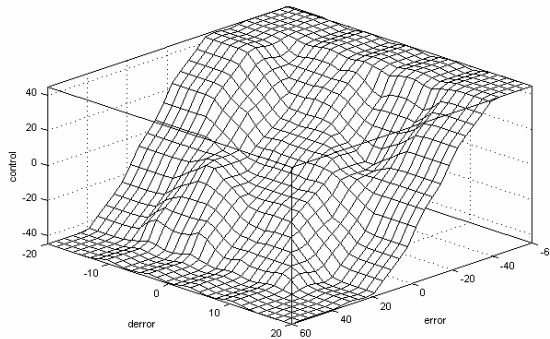


Fig. 9. Control surface for the fuzzy controller with the membership functions under their initial conditions

The output of the system, with a step input of amplitude 40, is shown in figure 10.

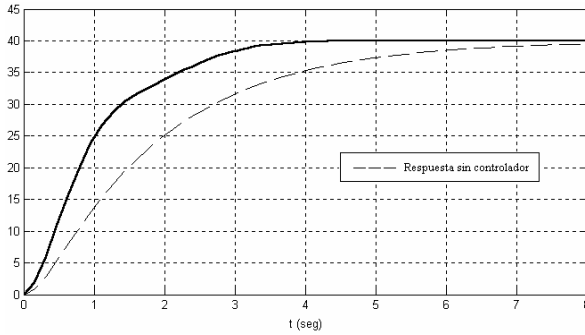


Fig. 10. Output of the system with the membership functions under their initial conditions

4.1 Tuning Algorithm

The objective of the tuning algorithm is to be able to manipulate, by means of a single variable and in a simple way, the settling time of the system, from the answer without controller until the response equivalent to 1/5 of the settling time of the answer without controller. The response must be fulfilled too with the constraints of small overshoots and without persistent oscillations, which means, a very smooth response.

This algorithm is based on the properties of the transfer characteristic or, in this case, of the control surface that it allows to modify the controller's behavior by means of modifications in the position and support of the membership functions maintaining fixed the fuzzy controller's structure. Obtaining a slower answer for configurations with wide or expanded membership functions in the center (see fig. 1) and reduced in the ends, and the other way, a faster answer for configurations with reduced or compressed membership functions in the center and wide in the ends (fig. 2).

The tuning algorithm only modifies the membership functions of the input variables and the membership functions of the output fuzzy variable remains constant since this disposition is only in function of a proportion of the range of the control action, in other words, they always remain uniformly spaced.

4.2 Tuning Factor Selection

The tuning factor is a number $k \in [0, 1]$ that determines the grade of tuning adjustment obtaining for $k = 0$ the biggest settling time (fig. 1) and for $k = 1$ the smallest settling time (fig. 2).

4.3 Normalization of the Ranges of the Fuzzy Controller's Variables

In this step the range of each input fuzzy variable is modified so that their upper and lower limits are equal to +1 and -1, respectively.

4.4 Tuning Factor Processing

When the range is normalized in the range between 0 and 1 the values can be computed using a power function, because to expand it is only necessary to use an exponent less than 1 and to compress an exponent more than 1 (see fig. 12). Recall the previous step that all the values belong to the interval $[-1,1]$. In an experimental way different values of exponents were tested (table 2) such that the new vector of operation points will be given by:

$$Vop_{final} = (Vop_{initial})^{r(k)} \quad (4)$$

Where $Vop_{initial}$ are the values normalized of the membership function in the x-axis and $r(k)$ is a polynomial.

Table 2. Important values of $r(k)$.

k	r
0	1/40
0.5	1
1	3

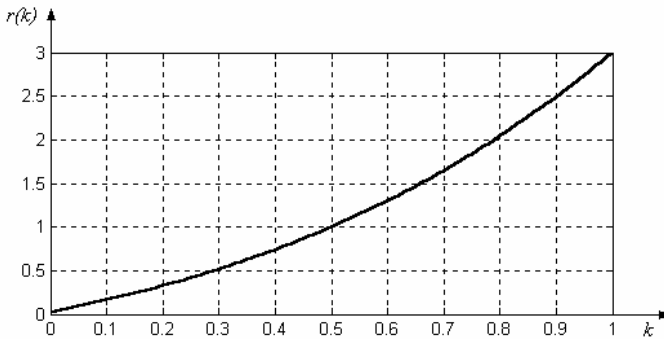


Fig. 11. Plot of function $r(k)$

The initial coefficients of the polynomial were obtained using mean square method. The values of k were defined in this way to be able to make an estimate over all their range $k \in [0, 1]$. The values of r , since it is an exponent, they were defined considering the increasing or decreasing of a number that is powered to the exponent r . Remember that the goal is to expand (slow response) for $k=0$ and to compress (fast response) $k=1$. For values below $r = 1/40$ the answer of the system was not satisfactory and in the same way, for values more than $r = 3$. With all these elements the polynomial obtained was:

$$r(k) = \frac{30k^3 + 37k^2 + 52k + 1}{40} \quad (5)$$

This $r(k)$ was found testing the optimal response for different dynamical systems (linear and non-linear) and finding the optimal parameters of the polynomial that fix the function for different values of k ($k = 0, 0.5, 1$) (figure 11).

To visualize the effect of this processing it is useful the graph of curves of adjustment for the vectors of operation points (figure 12). The values that can take an operation point (positive section) are in the horizontal axis and in the vertical axis, the values that takes this operation point once it has been powered to the exponent $r(k)$ where $k \in [0, 1]$.

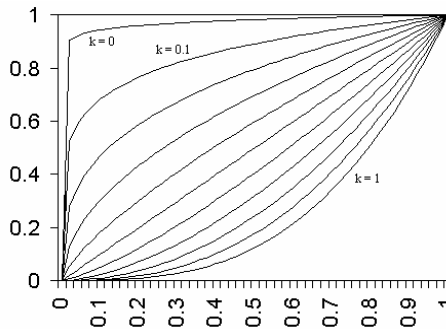


Fig. 12. Curves of adjustment for the operation points

4.5 Denormalization of the Ranges of the Fuzzy Variables

In this step it is necessary to convert the normalized range to the previous range of the system. This can be computed only multiplying the Vop vector by a constant factor.

5 Results of the Simulation: The Same Gain for Both Inputs

The cases will be analyzed for 3 different values of k , $k = 0, 0.5, 1$, showing the effect in the membership functions of the fuzzy variables, the control surface and the graph result of the simulation. For all the analyzed cases it will be used as input a step function with amplitude 40, the parameters that allow evaluating the quality of the tuning are the settling time (considered to 98% of the value of the answer in stationary state), the overshoots and the oscillations. Also for all the analyzed cases the controller's structure is fixed, that means, the fuzzy associative memory is the same in all the examples. In the simulations it is included (on-line dotted) the answer of the system without controller to compare with the response using different tunings of the controller. The controller's fuzzy variables and the

membership functions for the initial conditions are shown in the figures 6, 7 and 8: *error*, *change of error* and *control action* respectively.

5.1 Case 1: Adjusting the Membership Functions with a Tuning Factor $k = 0$

The function $r(k)$ takes the value $r(0) = 1/40$. With the tuning process the vectors of operation points for the fuzzy input variables are the following ones:

$$Vop\ error_{final} = [-59.3947, -58.3743, 0, 58.3473, 59.3947]$$

$$Vop\ d/dt(error)_{final} = [-19.7982, -19.4581, 0, 19.4581, 19.7982]$$

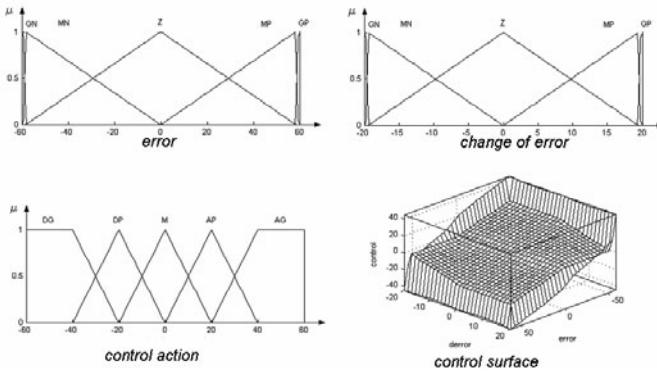


Fig. 13. Membership functions of the fuzzy variables and control surface for $k = 0$

Making the simulation with the controller's characteristics shown in the figure 13 the following answer was obtained:

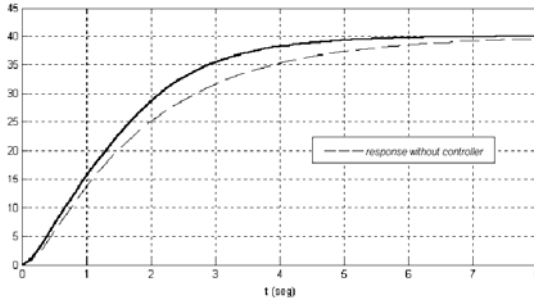


Fig. 14. Answer of the system for the case 1 with $k = 0$

In the figure 14 it is shown that with the tuning factor $k = 0$ the controller's effect on the answer of the system, due to the tuning, is small, approaching to the answer without controller. In this case the settling time is the biggest that can be obtained, $t_s = 4.96\ s$.

5.2 Case 2: Adjusting the Membership Functions with a Tuning Factor $k = 0.5$

The function $r(k)$ takes the value $r(0.5) = 1$. With the tuning process the vectors of operation points for the fuzzy input variables are the following ones:

$$Vop\ error_{final} = [-40, -20, 0, 20, 40]$$

$$Vop\ d/dt(error)_{final} = [-13.332, -6.666, 0, 6.666, 13.332]$$

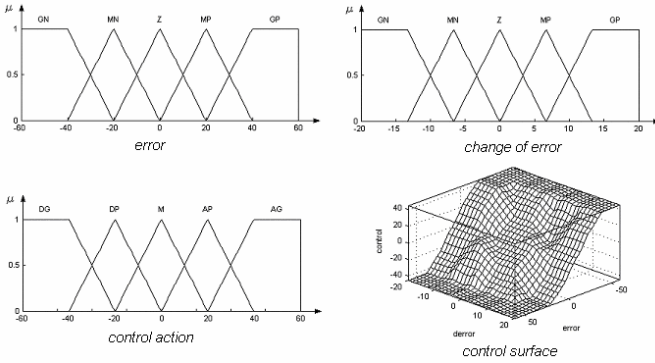


Fig. 15. Membership functions of the fuzzy variables and control surface for $k = 0.5$

Computing the simulation with the controller's characteristics shown in the figure 15 the following answer was obtained:

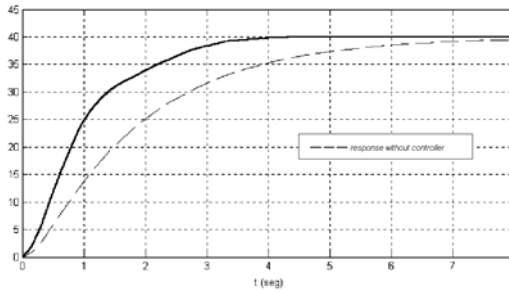


Fig. 16. Output of the system for the case 2 with $k = 0.5$

This case, with the tuning factor $k = 0.5$, is equal to operate with the initial conditions of the membership functions. The settling time is $t_s = 3.36\ s$.

5.3 Case 3: Adjusting the Membership Functions with a Tuning Factor $k = 1$

The function $r(k)$ takes the value $r(1) = 3$. With the tuning process the vectors of operation points for the fuzzy input variables are the following ones:

$$Vop\ error_{final} = [-17.7724, -2.2215, 0, 2.2215, 17.7724]$$

$$Vop\ d/dt(error)_{final} = [-5.9241, -0.7405, 0, 0.7405, 5.9241]$$

Computing the simulation with the controller's characteristics shown in the figure 17 the answer was obtained in figure 18, where the settling time is $t_s = 1.6\ s$ and it is the less value that can be obtained.

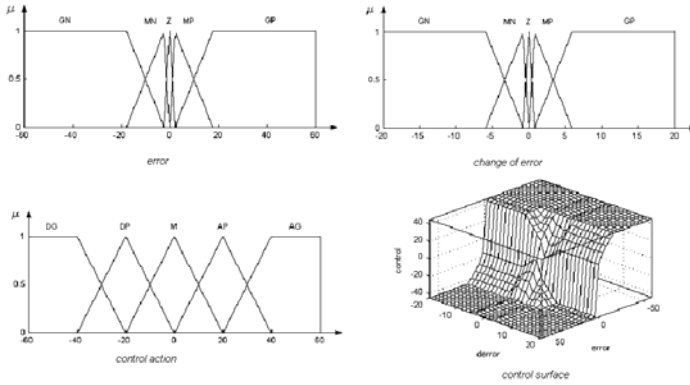


Fig. 17. Membership functions of the fuzzy variables and control surface for $k = 1$

The controller's effect on the answer of the system has begun to cause a small overshoot, due the bigger compression of the membership functions. If the value of $r(k)$ is increased, the settling time is not reduced and it only causes bigger overshoots and oscillations around the reference.

To visualize the effect of different values of the tuning factor k over the settling time of the answer of the system simulations with increments $\Delta k = 0.05$ in the interval $[0, 1]$ were computed. The result is shown in figure 18.

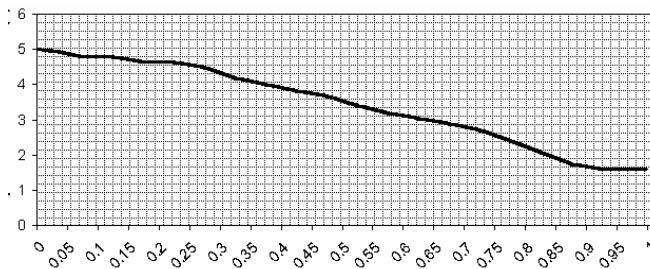


Fig. 18. Settling time versus tuning factor k

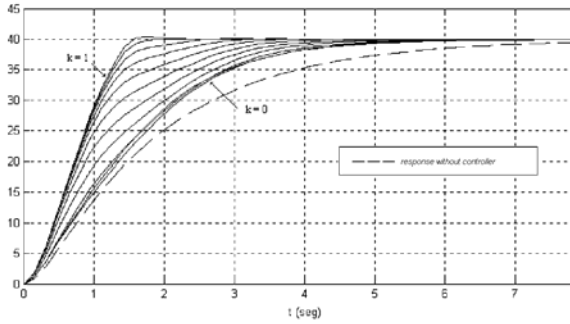


Fig. 19. Comparative graph of the system answers for different values of k

Figure 19 shows the behavior of the output of the system for increments $\Delta k = 0.1$. The next curve to the left is the corresponding for a tuning factor $k = 1$, with a settling time $t_s = 1.6$ s. For each increment, beginning in $k = 0$, a curve was plotted showing in the figure a smaller settling time.

Making use of the simulations, and the graphs in figures 19 and 20, it is possible to see that the optimal value of k for the tuning is $k = 0.9$. This value generates a settling time $t_s = 1.6$ s without a great overshoot and without oscillations (fig. 21).

Additionally, the fuzzy controller's performance was compared with a controller PID (Proportional-integral-derivative) whose parameters are the following ones $K_p = 25$, $T_i = 1.35$ and $T_d = 5$, and being that the differences are minimum as for time of establishment and general behavior (figures 20 and 21).

The disadvantage found in the controller PID is its inefficiency in comparison with the fuzzy controller since the control action generated by the PID can take very big values that are impossible to consider in a real implementation. On the other hand, the fuzzy controller uses real range of values.

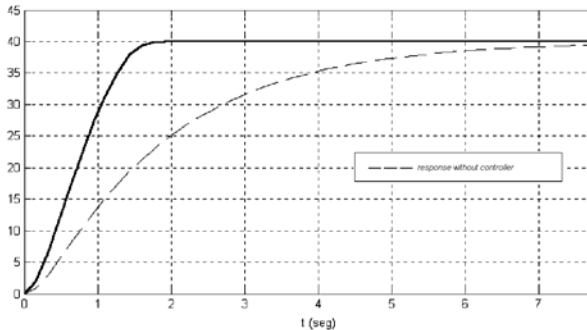


Fig. 20. Output of the system with $k = 0.9$

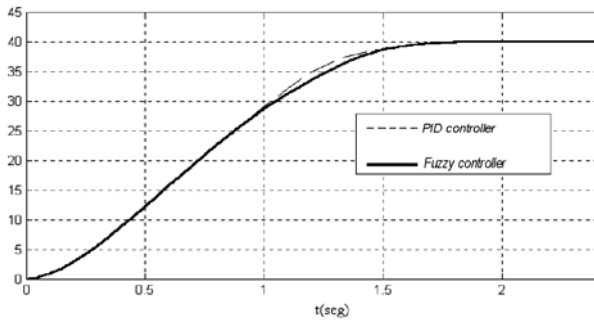


Fig. 21. Comparative graph of fuzzy controller answer versus PID

Considering that this is the fastest output that can be gotten with the controller PID, limited to the nature of the system, that is to say, limiting the range of the values that can take the control action to same values that those considered in the fuzzy controller's definition, it can be said that the tuning made on the fuzzy controller is satisfactory since it allows to vary the time of answer with very good behavior in the whole range of the tuning factor. Note that it is not evident to find three parameters of the PID for the optimal tuning and in the case of the fuzzy controllers it is necessary to increase or decrease the parameter k depending the settling time desired.

5.4 Simulation Results: Different Gains for Every Input

In this case a different gain k was considered for every input. The methodology used is exactly the same of the previous sections. The next figures show the stabilization time, the gain for the error k_1 and the gain used for the derivative of the error k_2 using a Δk of 0.01. For the axes k_1 and k_2 the value of 21 is equivalent to 1 in the figures 22, 24 and 25. Fig. 19 can be obtained for the case of $k_1=k_2$ in this surface (see fig. 23).

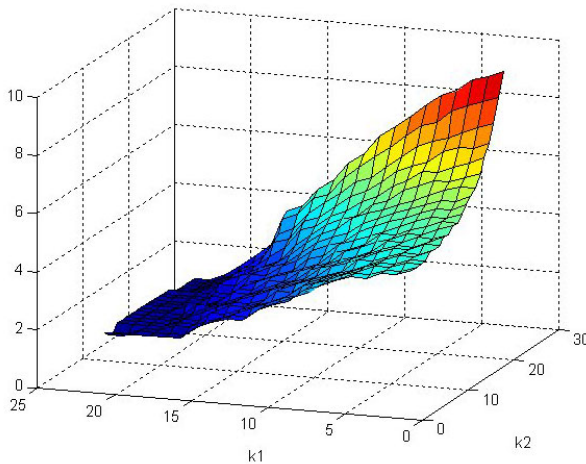


Fig. 22. Settling time versus tuning factor k_1 and k_2

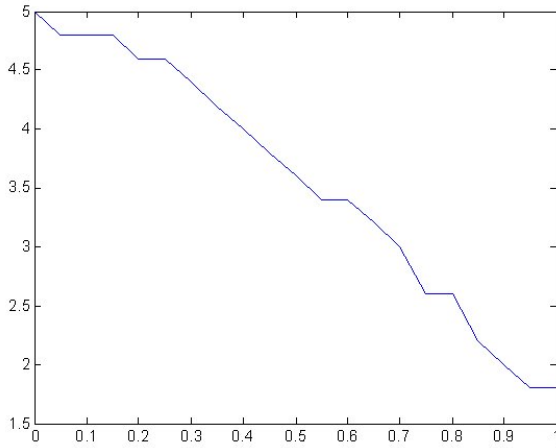


Fig. 23. Settling time versus tuning factor $k_1 = k_2$

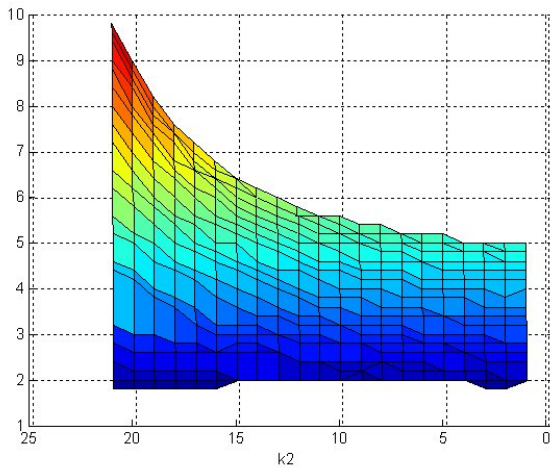


Fig. 24. View for tuning factor k_2

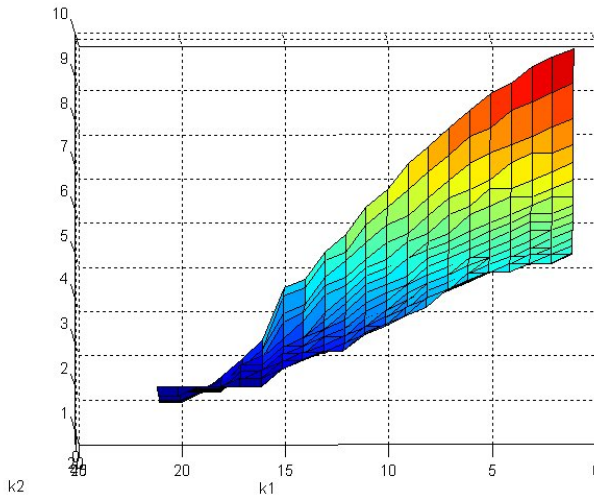


Fig. 25. View for tuning factor k_1

With this option the controller has other degree of freedom to control the response of the dynamic system. Fig. 25 shows how the effect of tuning factor k_2 , is less important than factor k_1 . Keeping the factor $k_1=1$, the response of the system is in the interval 1.8-2.0 secs.

6 Conclusions

The tuning methods of fuzzy controllers include the handling of a great quantity of variables that makes very difficult, and many times non satisfactory the search of structures and good parameters. The method proposed uses only one variable and operates considering the transfer characteristic, or in this case the control surface that is the fuzzy controller's property that defines their behavior allowing that the system can response with bigger or smaller speed and precision.

The function $r(k)$ can be generalized to any system that uses a fuzzy controller varying the values $r(0)$ and $r(1)$ as well as the coefficients of the function $r(k)$ depending on the desired behavior.

Another perspective is to create a self tuning algorithm that modifies by itself the factors k_1 and k_2 to find the desired response. In this point, the use of fuzzy controllers presents attractive aspects for its implementation in real systems.

References

- [1] Cox, E.: Fuzzy Fundamentals. IEEE Spectrum (October 1992)
- [2] Palm, R.: Fuzzy control approaches, General design schemes, Structure of a fuzzy controller, Handbook of Fuzzy Computation. Institute of Physics, Bristol. USA (1998)

- [3] Herrera, F., Lozanoy, M., Verdegay, J.L.: Tuning fuzzy logic control by genetic algorithms. *Int. J. Approx. Reasoning* 12, 295–307
- [4] Kinzel, J., Klawoony, F., Kruse, R.: Modifications of genetic algorithms for designing and optimizing fuzzy controllers. In: 1st IEEE Conf. Evol. Computations, ICEC 1994, Orlando, FL, USA (1994)
- [5] Leey, M.A., Tagaki, H.: Integrating design stages of fuzzy systems using genetic algorithms. In: 2nd IEEE Int. Conf. On Fuzzy Systems, Fuzz-IEEE 1993, San Francisco, CA, USA, pp. 612–617 (1993)
- [6] Bonissone, P., Kedhkary, P., Chen, Y.: Genetic algorithms for automated tuning of fuzzy controllers: a transportation application. In: 5th IEEE Conf. on Fuzzy Systems, Fuzz-IEEE 1996, New Orleans, LA, USA (1996)
- [7] Lee, S.C., Lee, E.T.: Fuzzy sets and neural network. *J. Cabernet* 4, 83–103 (1974)
- [8] Jang, J.: ANFIS: adaptive-network-based-fuzzy-inference-system. *IEEE Trans. Syst. Man Cybernet SMC-23*, 665–685 (1993)
- [9] Kawamura, A., Watanabe, N., Okada, H., Asakawa, K.: A prototype of neuro-fuzzy cooperation systems. In: 1st IEEE Int. Conf. On Fuzzy Systems, Fuzz-IEEE 1992, San Diego, CA, USA (1992)
- [10] Gómez-Ramírez, E., Chávez-Plascencia, A.: How to tune Fuzzy Controllers. In: FUZZ-IEEE 2004, Budapest, Hungary, July 25-29 (2004)

Increasing Energy Efficiency of a Preamble Sampling MAC Protocol for Wireless Sensor Networks Using a Fuzzy Logic Approach

Leocundo Aguilar¹, Oscar Castillo², J. Antonio García-Macías³,
and Guillermo Licea¹

¹ Autonomous University of Baja California, Faculty of Chemical Science and Engineering
Calz. Tecnológico 14418, Mesa de Otay, Tijuana Baja California, CP-22390. México
{Laguilar, glicea}@uabc.edu.mx

² Department of Computer Science, Tijuana Institute of Technology,
PO. Box 4207, Chula Vista, CA 91909, USA
ocastillo@tectijuana.mx

³ CICESE Research Center, Computer Science Department
Km. 107 Carretera Tijuana-Ensenada C.P. 22860, Ensenada, B.C., México
jagm@cicese.mx

Abstract. Wireless sensor networks (WSN) are an emerging technology based on the progress of electrical and mechanical engineering, as well as computer science in the last decade. In this work, a FIS (Fuzzy Inference System) approach is used to increase the energy efficiency of a preamble sampling MAC (Media Access Control) protocol for WSN. The FIS replaces a traditional average approach reducing the number of sampling points and the processing time required by the traditional approach. In addition, the FIS is designed to have a small memory footprint and fast response; this considering that the FIS must be implemented in WSN nodes that use microcontroller that are limited in memory and processing speed. The result of using the FIS approach shows up to 90% reduction (best case scenario) on the waste of energy consumed by overhearing having an increment of energy efficiency.

Keywords: Fuzzy approach, Wireless, Sensor, Network, microcontroller.

1 Introduction

Advances in digital electronics, micro-electro-mechanical systems and wireless communications have made possible the development of low power sensing devices with short range wireless communication capability at low cost. These sensing devices can autonomously collect, process and communicate information about their environmental conditions (i.e. temperature, vibration, sound and pressure). When a large number of sensor devices collaborate using their wireless links with a flow of information from each other they constitute a wireless sensor network [1].

In WSNs the sensing devices are called *sensor nodes* and they operate in a self-organized and decentralized manner that maintains the best connectivity as long as possible and communicate messages to other nodes via multi-hop forwarding (see Fig. 1).

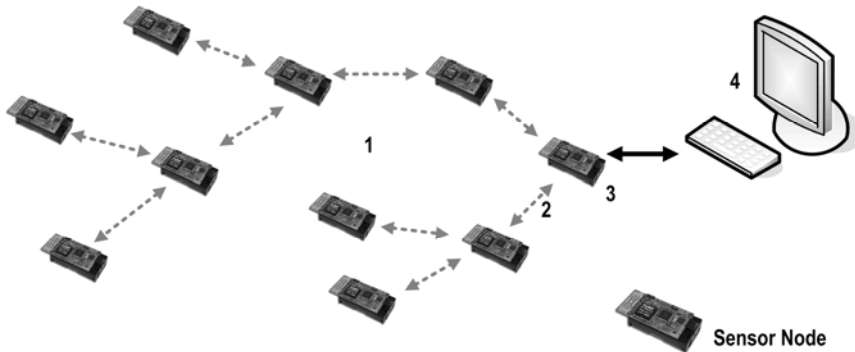


Fig. 1. Core Components of a WSN: set of distributed sensors (1), interconnection network (2), data sink (3) and resources to control, configure and manage the network (4)

Some early WSN applications were developed by the Defense Advanced Research Projects Agency (DARPA) in the U.S. for monitoring possible battle scenarios [2], but at present the scientific community has found an increased use in civilian applications [1]. WSNs have a wide range of applications due to their capabilities for detailed physical monitoring and manipulation offering enormous opportunities for almost every scientific discipline. Therefore, the research on this particular type of networks has increased in recent years, identifying opportunities and challenges for distributed signal processing in sensor devices and architectural challenges presented by these systems that are massively distributed, wirelessly connected and energy limited [1], [3], [4]. The state of most commercial applications is still immature, and there are a few companies that have platforms in the market at high cost [5] and [6]. However, given that WSNs are still in research stage they offer great opportunities for multidisciplinary science, technology, engineering, and mathematics education.

2 LiSANDRA: An Experimental Wireless Sensor Network

The University of Baja California (UABC), together with the Centre for Scientific Research and Higher Education of Ensenada (CICESE), have perceived as an opportunity the development, application, use, and diffusion of this technology. Therefore, projects in the research and development of this technology have given as results the designing and implementing of own WSN platform called LiSANDRA (*Lightweight Sensor and Actuator Network with Distance Range Auto-adjustable*) [7]. This network is an experimental WSN developed in order to

create the background and expertise and so assist the teaching-learning process for training students, providing them with the appropriate skills and knowledge on this technology.

2.1 Architecture

LiSANDRA has a multilevel hierarchical architecture, commonly called *tiered architecture* where the sensor nodes form a hierarchy in which a node at a given level performs a specific set of tasks on behalf of a subset of nodes in the level below. This in contrast to typically used WSNs called *flat architecture*, which has a single level where all sensor nodes are equal and are homogeneous in form and function. The reason to select hierarchical architecture is a cost-effectiveness issue due to this architecture reduces the cost of sensor nodes by allocating resources just where they can be most effectively utilized. These types of architecture and others are described widely in [8].

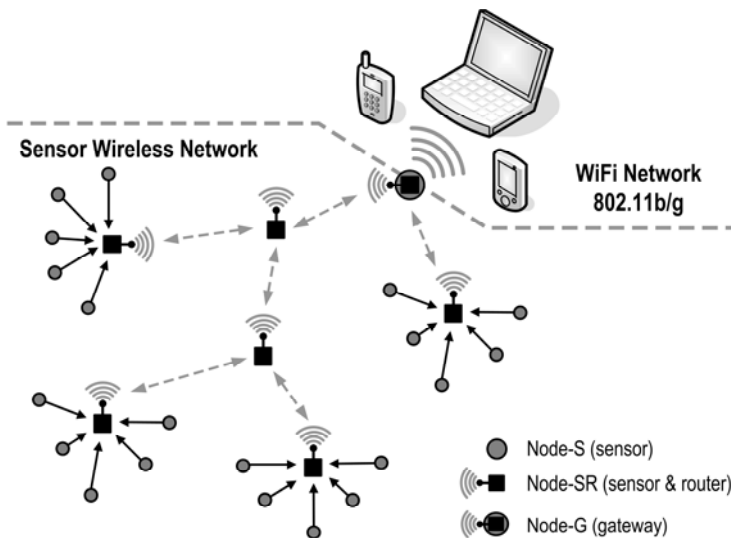


Fig. 2. Architecture used by LiSANDRA

As is shown in Fig. 2 the lowest layer in LISANDRA is basically composed by the node called sensor node (*node-S*) and they can only sense and transmit small data packets. These nodes can establish communication with just one sensor and router node (*node-SR*) which was previously associated to form a group of sensors called *sensor cluster*. The sensor and router nodes form the middle layer; this node type recognizes and interacts with the same type of nodes located in their neighborhood to form the primary mechanism for connectivity of the network (*backbone*). The network has another node at the highest layer of architecture

called *node-G* which operates as a *gateway* for connectivity to an 802.11b/g wireless network commonly called Wi-Fi network. This is a high cost node and for low cost applications it can set aside and make use of one node-SR as a gateway; this can be done by connecting its serial port to a personal computer (PC) that executes specific software for connectivity to the traditional wired or wireless networks.

2.2 Nodes Organization

Essentially the nodes are an embedded system composed of hardware and software sections. The hardware section is made up of four basic components which are: a sensing unit, a processing unit, a communication unit and a power unit. On other hand, the software section is embedded monolithic software called *firmware* composed of software subsystems that manage the hardware units. The three type nodes that compose LiSANDRA have similar organization, but this document is focused to the nodes-SR because these nodes are the most important due to they create the primary mechanism for connectivity of the network. For complete information about the all nodes see [9].

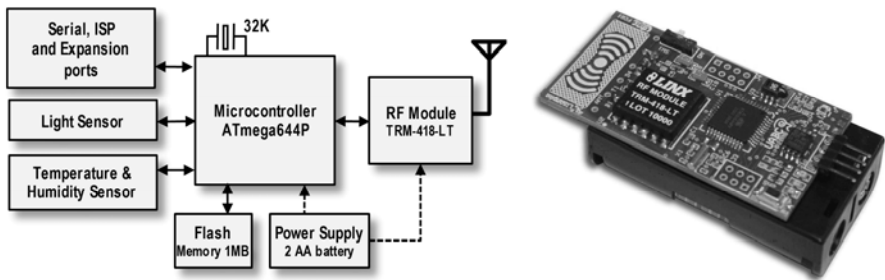


Fig. 3. Architecture blocks diagram and pictures of node-SR

2.2.1 Hardware

The picture and architecture block diagram of the node-SR is shown in Fig 3. This node is built on a 1.25" x 2.50" double side PCB and uses two rechargeable AA batteries as power unit. The processing unit uses the ATmega644P [10] microcontroller that has 64KB instruction flash memory and 4KB of data memory (RAM) and operates at 2 million instruction per second (MIPS) it means haft micro-second per instruction. In addition, this section uses an external 32 KHz watch crystal as the time base for the RTC (Real Time Clock) and 1MB external serial memory flash that works as data logging system.

The sensing unit is composed of a SHT11 temperature-humidity sensor [11] and a TSL2550T ambient light sensor [12]. However, the node hardware design considers microcontroller I/O lines as an expansion port that can be used to interface actuators and other types of sensors according to particular application requirements.

The communication unit includes the RF transceiver module TRM-418-LT [13] with an antenna embedded into the PCB. Also, there is a wired communication section based other the standard RS232 over TTL voltage, and it is use to communicate with the PC to download program and/or upload data stored in the data logging system.

2.2.2 Software

The main functions of node-SR are sense and routing data packets from/to other nodes of the same type. In addition this node has the ability to recognize up to 16 nodes-S that were previously associated to form a cluster sensor. These subsystems are the application, communication processing and drivers (communication and sensor); all of them shielded from the low-level functionality of the components and running by an executive main loop technique. Fig. 4a shows the firmware flow chart representing the big tasks activities that the node-SR is performing continuously.

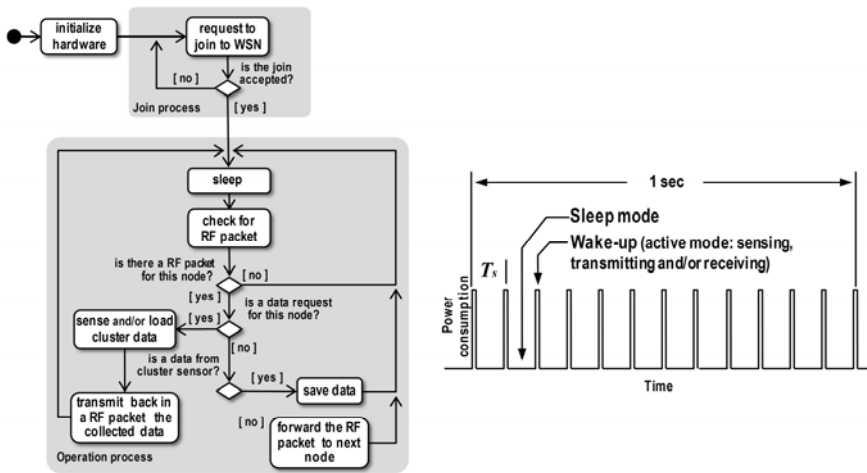


Fig. 4. Firmware flow charts of node-SR (a) and duty cycle sequence (b)

The firmware of node-SR is implemented in 90% C and 10% assembly language using AVR Studio [10] and WinAVR [14] as development tools. It is structured in 3 layers; the first one is the application layer where the user programs the access to the network using a simplified network abstraction similar to the Berkeley sockets [15] that is the “de facto” standard for network programming. A middle layer called network layer is used to route data packets to others node-SR of the network. Finally, a physical layer focused on the functionality of the RF receiving-transmitting process of raw data transceiver; this in order to execute the MAC using a *Carrier Sense Multiple Access* (CSMA) scheme; so before a transmission is attempted, a node that has a data packet to transmit first “listens” to the channel to determine if it is busy. If the RF channel is busy, the transmitting

node “backs-off” for a random period of time after which it senses the channel again. In other hand, for the reception process the node-SR periodically goes into the sleep mode during which the radio is turned off. Thus, each node sets a wake-up timer and goes to sleep for a short specific period of time. At the expiration of the timer, the node-SR wakes up and listens to determine if it needs to receive or retransmit data packets to other node-SR. The complete listen-and-sleep cycle is referred to as a frame or *duty cycle* and it is performed every T_s (125mS) that corresponds to 10 times per second (see Fig. 4b).

3 Preamble Sampling MAC Protocol

Nowadays, one of the most important requirements of the WSN applications is the low power consumption, so the nodes require lowering their energy in order to support an application for longer periods of time. The protocols type LPL (*Low-power-listening*) form a set of MAC protocols that reduce the *idle listening* concerns, a state of the node when its radio is turned on in receive mode, but not receiving any packets. In a LPL protocol, nodes verify the channel every specific period of time, and if they do not receive any data during this verification, then they return to sleep as was described before (section 2.2.2).

The protocols as Aloha with preamble sampling (PS) [16], WiseMAC [17], and B-MAC [18] were the first LPL proposed protocols. These protocols send data packets using very long preamble to guarantee that the intended receiver will stay on at the moment the channel is verified (see Fig 5a -- The preamble is commonly coded as ‘10101010’ for the purpose of leading the data slicer in the receiving end to a stable state). However, the protocols are not adapted to recent radios like the IEEE 802.15.4 [19] compliant Chipcon CC2420 [20] radio. Therefore, researchers introduced new compatible LPL protocols such as X-MAC [21], SpeckMac-D [22], and MX-MAC [23]. These new protocols are based on repeating either the data packet itself or a special advertisement packet, instead of the long preambles (see Fig. 5b).

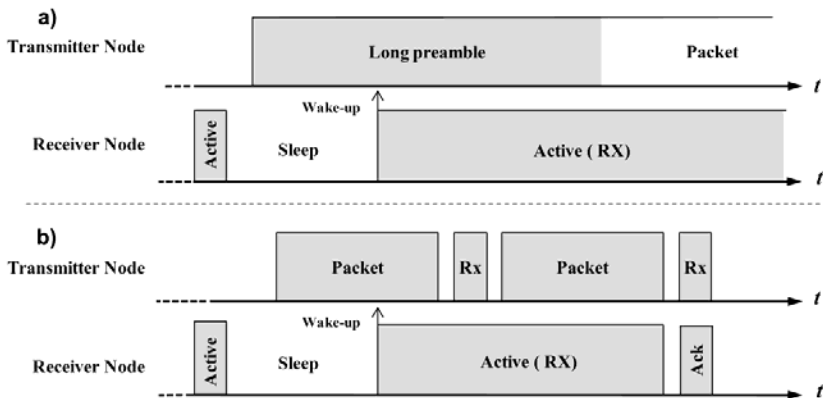


Fig. 5. MAC: using long preamble (a), and using a special advertisement packets (b)

These kinds of MAC protocols try to reduce the waste energy due to the *overhearing* produced by the use of the long preamble technic. The overhearing is an effect of always listening for incoming traffic, is that a node will receive all messages including those that concern its neighbors only. Overhearing these messages is simply a waste of energy, and becomes problematic in dense networks with many nodes inside the reception range of a node. In this scenario the nodes detect the long preamble and they stay wake-up until the packet is received, analyzed and detected that the packet is for some neighbor node. Therefore the nodes wasted energy during the hearing process of long preamble to be waiting for the *Start Of Packet* (SOP).

The nodes in LiSANDRA use the MP-MAC [7] (Modulated Preamble MAC) protocol type LPL (*Low-power-listening*) that uses the long preamble approach to avoid *idle listening* concerns. However, MP-MAC introduces a novel characteristic that is a mechanism to modulate the long preamble in order to minimize the overhearing problem. Thus, the transmitter node applies a modulation to the long preamble by changing the period length of the square wave; starting with long period length and reducing it proportionally until the SOP appears as is shown in Fig. 6. In consequence, the receiver nodes that wake-up are able to detect the long preamble and measure the period length of the square wave (w); and so to know the time in which the packet will start (SOP). Therefore, the receiver nodes knowing the period value (w) can make a decision to wait for the SOP if the period is short; or in other case return to sleep mode during the time given by the period value, and then to wake-up some time before the SOP appears.

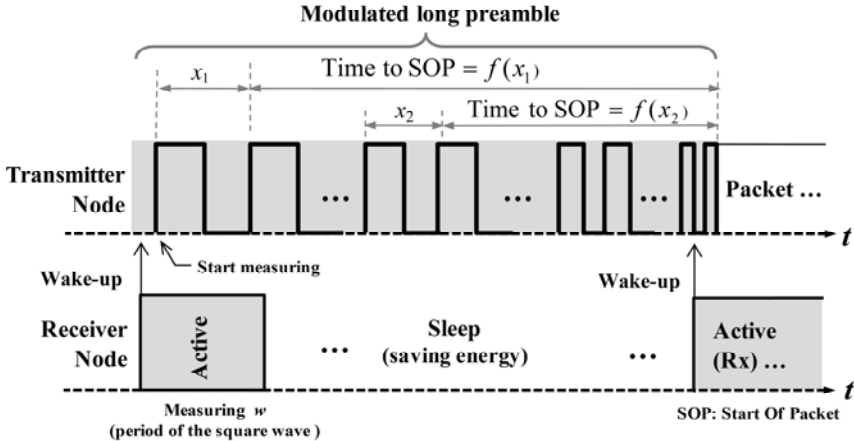


Fig. 6. Modulated preamble MAC protocol used in LiSANDRA

In order to achieve properly this method and get acceptable energy saving results is important to take the right measurements and quantity of readings on the square wave period length. This means that taking just one measure is not enough to make an acceptable decision due to inaccurate measure reading process. In other hand, taking high quantity reading and calculate the average will fallout in

overhearing problems again. Thus, the modulated preamble MAC protocol used in LiSANDRA requires a method that can be able to make a logical decision using very low quantity and imprecise data readings. However, the method should be supported with certain degree of reasoning based on facts, human knowledge and experience. This is the reason for use a fuzzy logic (FL) approach which is described in the next section.

4 Fuzzy Logic Approach

Since Lotfi Zadeh published his propose about “fuzzy set theory” [24], the progresses in number and variety of applications of the fuzzy logic have been increasing in the last two decades and supported by advances in the electronics and computing fields. The applications range from consumer products to industrial process control, instrumentation and decision-support systems. One of the successes of the fuzzy logic is due to the FL may be viewed as a methodology for computing with words rather than numbers, being the words more imprecise than numbers, and their use is much closer to human intuition and so including the tolerance for imprecision. In addition, the fuzzy logic has been used in inference systems to imitate the human behavior expertise on a particular field or scenario; and so to make decisions to accomplish a specific task inferring actions based on data, facts and rules. These kinds of systems are called Fuzzy Inference Systems (FIS), and they are the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made, or patterns discerned.

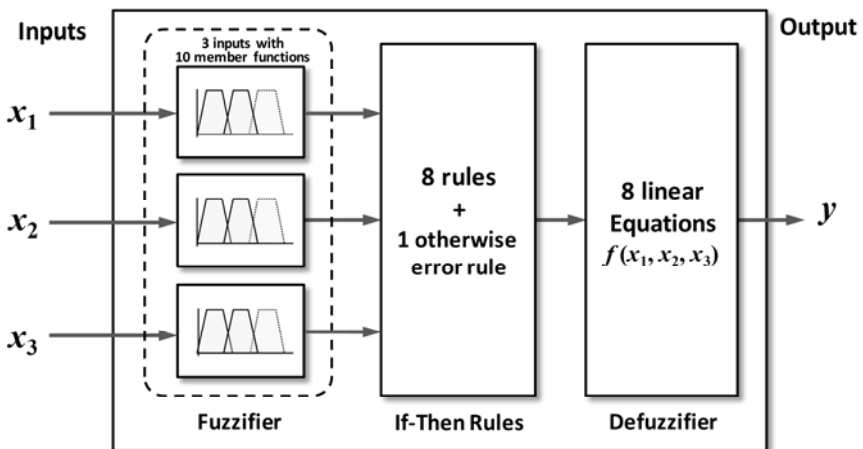


Fig. 7. Structure of the FIS used in the nodes.

There are two most common types of FIS, one of these is the Mamdani-type proposed in 1975 by Ebrahim Mamdani [25] as an attempt to control a steam

engine and boiler combination by synthesizing a set of linguistic control rules obtained from experienced human operators. In other hand is the Takagi-Sugeno-Kang [26] (most commonly called Sugeno) type introduced in 1985 and is similar to the Mamdani in some parts. The first two parts which are the fuzzifying the inputs and applying the fuzzy operator (rules) are exactly the same; and they differ in the way their outputs are determined. The main difference between Mamdani and Sugeno is that the Sugeno output membership functions (MF) are either linear or constant. These two types of fuzzy inference systems are describes widely in [25], [26] and [27].

4.1 Fuzzy Inference System Used in LiSANDRA

The FIS structure used in the nodes is shown in Fig. 7, and it is a Sugeno type FIS [26] with three inputs and one output. The inputs (x_1 , x_2 and x_3) are the values of three consecutive period length measures of preamble; and the output (y) is the time period during the node could sleep before the SOP appears. The total preamble has been divided in 10 equal length categories to create 10 MF which are used by the fuzzifier section of the FIS; so the fuzzifier converts crisp inputs data (x_1 , x_2 and x_3) into fuzzy data (membership value) that can then be used by the if-then rules section. The FIS has an if-then rule section composed by rigorously 1000 rules due to there are 3 input with 10 MF per input (rules = 10^3). However, just 9 rules are used at the time by the decision process, this according to the following logic facts.

- a) The period of the long preamble is modulated starting with a long period length and reducing it to a short length.
- b) Each category (member functions) of the modulated preamble is presented by the transmitter node during enough time to accomplish three period lengths measures by the receiver nodes.
- c) As effect of a) consecutive period length measures given as result that last measure must to be a length equal or shorter than previous ones.
- d) Based in b) and c), the three consecutive measures must be belonging to two adjacent member functions as maximum.

Table 1. If-then rules for twogeneric adjcent member functions A and B

Rule	IF	Inputs			THEN	Output
		x_1 is	x_2 is	x_3 is		
1		A	A	A		$y_1 = 1x_1 + 3x_2 + 4x_3$
2		A	A	B		$y_2 = 2x_1 + 4x_2 + 2x_3$
3		A	B	A		$y_3 = 2x_1 + 0x_2 + 6x_3$
4		A	B	B		$y_4 = 0x_1 + 2x_2 + 6x_3$
5		B	A	A		$y_5 = 0x_1 + 2x_2 + 6x_3$
6		B	A	B		$y_6 = 2x_1 + 0x_2 + 6x_3$
7		B	B	A		$y_7 = 2x_1 + 3x_2 + 3x_3$
8		B	B	B		$y_8 = 1x_1 + 3x_2 + 4x_3$
9	OTHERWISE				ERROR	

Therefore, the decision process uses the three inputs and two contiguous member functions of the total teen. This gives us a system with 3 inputs with 2 member function per input resulting in 8 rules. There is an extra rule that to represent the *otherwise* statement (see Table 1) which are the set of rules not fired, in other words the complement of fired rules set. The defuzzifier section is composed by a set of 8 linear functions (one per rule [y_1 to y_8]), where each function depends of the inputs measures (x_1 , x_2 and x_3). So each one affects proportionally each output function according to its significance in the rule (see Table 1 output column). Since each rule has a crisp output, the overall output is obtained via weighted sum operator instead of weighted average to reduce computational further in the process of defuzzification.

4.2 Implementation of FIS

As mentioned in section 2, the node-S has hardware limitations such as low performance and processing speed to name some ones. Thus, the limitations of the nodes are an important issue has to be considered in the implementation of FIS.

4.2.1 Member Functions

Considering that the microcontroller used in the node does not have a math coprocessor or float-point unit, the type MF to be selected should be the simplest ones which do not required a high processing algorithm and integer arithmetic is enough. This limits the selections to triangular and trapezoidal type [27]; however keeping in mind the fact b) described in last section: “*each category of the modulated preamble is presented by the transmitter node during enough time to accomplish three period lengths measures by the receiver nodes*”. The most appropriate MF in this case is a symmetrical trapezoid type which is shown in Fig. 8 and defined by the equation 1.

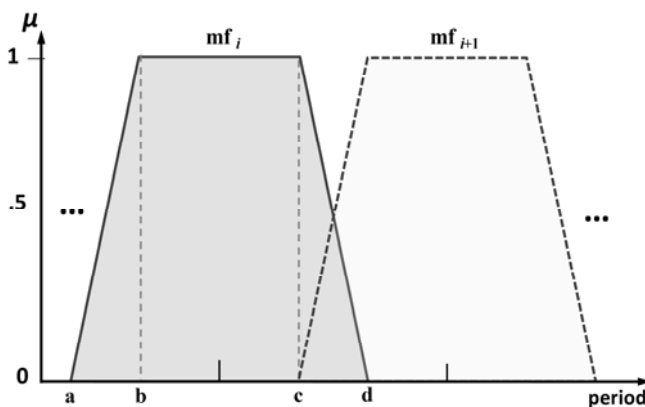


Fig. 8. Two adjacent MFs type trapezoidal

$$\text{trapezoid}(x; a, b, c, d) = \begin{cases} 0 & , \quad x < a \\ \frac{x-a}{b-a} & , \quad a \leq x \leq b \\ 1 & , \quad b \leq x \leq c \\ \frac{d-x}{d-c} & , \quad c \leq x \leq d \\ 0 & , \quad d \leq x \end{cases} \quad (1)$$

The microcontroller does have hardware 2-cylce multiplier instruction (MUL) and fractional multiplier instruction (FMUL) [10]; so to implanting or use standard math library introduce a computational overhead. In order to avoid that overhead and to achieve a compact and fast implementation, the trapezoid MF could be used scaling the functions to divisor (b-a) [or (d-c) because is a symmetrical] and so eliminating the division operation allowing to use a fast integer arithmetic operations; but keeping in mind the value is scaled.

Thus, using this approach the symmetrical trapezoidal MFs are scaled to 8 because is easy to remove it just using the shift-to-right (>>) instruction three times, it since this operation is equivalent to divide by two. In Fig. 9 are shown the first three MF that compose the 10 MF of the fuzzifier section.

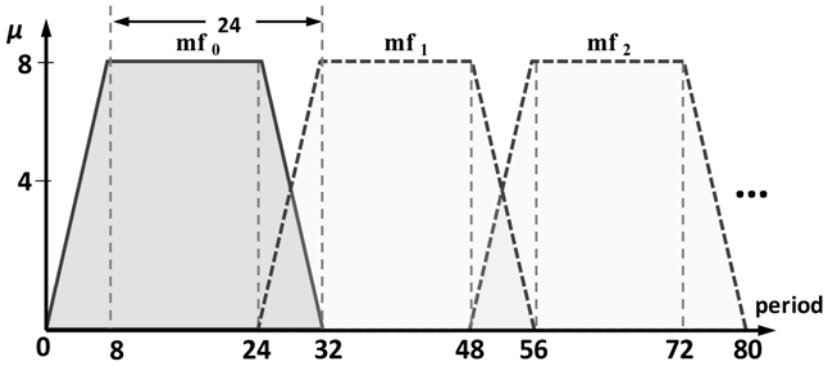


Fig. 9. Scaled trapezoidal member functions

The implementation to calculate the membership value for this MF is shown in the List 1 as a C language **trapMF** function. This function has a small size memory footprint (32 native instructions only) and is implemented to calculate the membership grade given a value (x) and the number (0...9) of the 10 MFs to be used (numFM). In best case scenario (returning by the first if instruction) the microcontroller spent 11uS to calculate and return the value and 18uS for worst case scenario (returning by the last else instruction).

List 1. Trapezoidal member function implemented in C language.

```

UINT8 trapMF( UINT8 x , UINT8 numMF ){

    UINT8 a=0,b=8,c=24,d=32,delta=24;

    delta *= numMF; /* delta according to # of MF */
    a += delta;      /* adjust parameter a,b,c and d */
    b += delta;      /* according to # of MF */
    c += delta;
    d += delta;

    if( x <= a ) return 0;
    if( x > a && x <= b ) return x-a;
    if( x > b && x <= c ) return 8;
    if( x > c && x <= d ) return d-x;
    else return 0;

}

```

4.2.2 FIS Processing

The Fuzzy Inference System used in this work uses the process algorithm showed in the List 2. This starts taking three period length measures values (x_1 , x_2 , and x_3) which are verified to check for error reading. The three values should be belonging to two consecutive MFs; if this is not complied with, the process starts again. In this verification the two adjacent MFs are identified (A and B) and now all data are ready to fuzzifier inputs and apply the If-then rule process. Thus the firing strength (w) of each rule is calculated using the *minimum* as AND operator. Then the linear function output of each rule is calculated using the values (x_1 , x_2 and x_3) and their corresponding parameters (p , q and r). Finally, the overall output (Y) is calculated using weighted sum; this is accomplished accumulating all results of multiply the linear function output of each rule by its corresponding firing strength.

List 2. FIS process algorithm.

1. Take 3 consecutive measures values (x1,x2,x3)
2. If error goto step 1 /*values aren't in 2 adjacent MF*/
3. Determine the two involved MF (A and B)
4. Apply rules to determinate firing strength:

$$w[0] = \min (mf_A(x1), mf_A(x2), mf_A(x3))$$

$$w[1] = \min (mf_A(x1), mf_A(x2), mf_B(x3))$$

$$w[2] = \min (mf_A(x1), mf_B(x2), mf_A(x3))$$

$$w[3] = \min (mf_A(x1), mf_B(x2), mf_B(x3))$$

$$w[4] = \min (mf_B(x1), mf_A(x2), mf_A(x3))$$

$$w[5] = \min (mf_B(x1), mf_A(x2), mf_B(x3))$$

$$w[6] = \min (mf_B(x1), mf_B(x2), mf_A(x3))$$

$$w[7] = \min (mf_B(x1), mf_B(x2), mf_B(x3))$$
5. Calculate linear functions outputs:

$$y[n] = p[n]*x1 + q[n]*x2 + r[n]*x3; n=[0..7]$$
6. Calculate overall Y output using weighted sum:

$$Y = \sum w[n]*y[n]; n=[0..7]$$

In order to achieve a small size memory program implementation and provide an acceptable computational performance; the algorithm (List 1) described before has been slightly modified and so to reduce its execution time and memory used. The modifications are described as follows:

- a) All membership values are calculated and stored in a small memory array (one per input) to be used at the moment if-then rules are applied and so avoid unnecessary recalculations.
- b) There is only one memory location for the firing strength (w) and other for linear function outputs (y) that are reused in each rule calculation.
- c) The weighted sum (Y) is calculated iteratively just after the firing strength (w) and linear function outputs (y) are calculated for each rule.

As result of these modifications the final code of the principal engine for the FIS is showed in List 3. This code uses just 128 native instructions that correspond to

256 bytes of program memory giving us a very small memory footprint as result. In addition, this code presents an acceptable performance having an execution time of around 700uS to complete the entire process having the input values (x_1 , x_2 , and x_3) to calculate the overall output (Y). It is important to remark that the w and the parameter p , q , s are scaled, so the linear function output (y) and the overall output (Y) are divided by 8 using the instruction shift-to-right three times.

List 3. FIS process implemented in C language.

```

n=0;          /* used as index                */
Y = 0;       /* overall FIS output                       */
B = A + 1;   /* member functions to be used in FIS      */

/* calculate the membership values of x1,x2 and x3 */
/* for each member function - A and B             */
mfx1[0]=trapMF(x1,A); mfx1[1]=trapMF(x1,B);
mfx2[0]=trapMF(x2,A); mfx2[1]=trapMF(x2,B);
mfx3[0]=trapMF(x3,A); mfx3[1]=trapMF(x3,B);

/* apply rules and calculate for each rule:      */
/* a) firing strength                             */
/* b) linear output function                       */
/* c) and accumulate it to the overall output    */
for( i=A; i<B; i++ )
{
  for( j=A; j<B; j++ )
  {
    for ( k=A; k<B; k++ )
    {
      /* AND operator to get firing strength      */
      w = min( mfx1[i], mfx2[j], mfx3[k] );

      /* linear output function using p q and r */
      y = ( p[n]*x1 + q[n]*x2 + r[n]*x3 ) >> 3;
    }
  }
}

```

```

    /* next index for parameters p q and r      */
    n++;

    /* overall output ==> weighted sum        */
    Y = Y + ( ( w*y ) >> 3 );
}
}
} /* output Y is ready */

```

5 Experimental Test and Results

In order to verify and validate the FIS performance some experimental test have been realized, however the most representative test is presented in this work. This test consists of one node-SR transmitting four broadcast type packets per second to 10 nodes. The 10 nodes are working with the MP-MAC protocol using the FIS and the average method. The simple average method is applied to the three inputs measures (x_1 , x_2 and x_3) to calculate the time-to-sleep. The plan to use both methods in each node is to make a comparison between them under same conditions, so determine the benefit of using the FIS versus a traditional approach. In addition, the uses of 10 nodes in the test give us a mechanism to detect abnormal behaviors on nodes; it because the same behavior is expected from all of them. A simple schema of the test configuration is shown in Fig.10 and a picture of the test bench used during the test period as well.

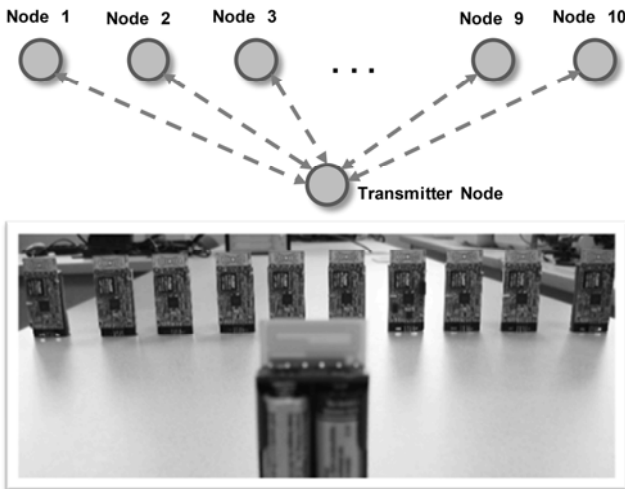


Fig. 10. Test configuration for nodes and test-bench

The test starts with the transmitter node that sends an initial broadcast packet to synchronize and notify to receiver nodes that the test has started. Then, the transmitter node sends 4 packets per second which are sequentially labeled during 5 minutes having a total of 1200 packets. The nodes do not use the calculated time-to-sleep value to sleep and save energy, instead they stay wake-up to receive the packets to avoid missing packets because both methods are in evaluations and they could carry out wrong calculations. The receiver nodes store the following important set of data for each received packet to be retrieved later for further analysis.

- 1) The three period length measures used in the methods (FIS or simple average).
- 2) The calculated time-to-sleep determinate by each method.
- 3) Number of tries used to get acceptable value (average approach is one)
- 4) The label of received packet in order to determinate missing packets.

As test result, all nodes presented highly similar stored information without any abnormal behavior on them. Thus, this gives us a valid and acceptable data to be analyzed. Fig. 11 shows part of data stored by a node and is presented as a comparative chart between the FIS and average methods.

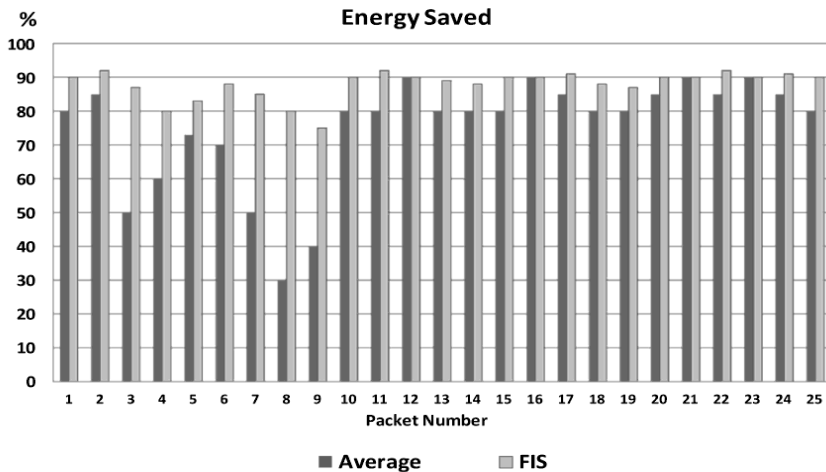


Fig. 11. Data stored by a receiver node

The following important interpretations are determinate based on the data analysis.

- 1) In a quiet scenario where there is not RF noise in the environment, the three period length measures are close and stable; in other words the measures belong to the same member function both method are compatible and produce same results. However, is important to remark that the simple

average are faster than FIS method due to last one requires more complex calculations.

- 2) If the three period length measures have discrepancies or they are dispersed, the simple average approach produces wrong output values. In other hand, the FIS could detect the problems and then take new measures to start again. This approach is able to detect that kind of problems because it has a certain human knowledge embedded into the rules section of the FIS.
- 3) The nodes using the FIS present better performance than the average approach having around 90% average in saving energy; this means that the nodes sleep during 90% of the long preamble reducing significantly the overhearing problem. This performance is related to the best case scenario where the nodes are synchronized to the duty cycle of the transmitter node. In a non- synchronized test scenario the FIS lower than 90% but still better than simple average approach, it is basically because the FIS has the capability to detect possible errors and starts again (interpretation 2).

6 Conclusions an Future Work

In this work, a fuzzy inference system has been proposed, implemented and tested in nodes of a WSN which use a novel modulated preamble MAC protocol to reduce the overhearing problem. The parameter used in the FIS were determined by using human experience and based in facts detected in the system. Experimental test were realized to verify and validate the FIS performance and so to determine the computational cost and benefit of use the FIS compared to a traditional approach (simple average). The test result presents a good and acceptable performance that in all cases results better than traditional approach. However, the FIS has an implementation that uses more memory footprint and computational execution time; but they are not presenting a big impact due to the memory used is just 125 native instructions (250 bytes of program memory) with an execution average time of 700 micro seconds to complete FIS calculation process. As propose for future work, the FIS should be optimized adjusting its parameter applying the back-propagation algorithm off-line and using the stored data obtained in the experiments realized.

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless Sensor Networks: a survey. *Computer Networks Journal* 38(4), 393–422 (2002)
2. Sohoraby, K., Minoli, D., Znati, T.: Wireless sensor networks. Technology, protocols and applications. John Wiley & Sons, New Jersey (2007)
3. Hac, A.: Wireless Sensor Network Designs. John Wiley & Sons, Honolulu (2003)
4. Callaway Jr., E., Callaway, E.: Wireless Sensor Networks: Architectures & Protocols. Auerbach Publications, Florida (2003)

5. Crossbow Technologies, Inc., <http://www.xbow.com>
6. Sentilla Corporation, <http://www.sentilla.com>
7. Aguilar, L.: LiSANDRA an Experimental Wireless Network, PhD thesis, Universidad Autónoma de Baja California (2009)
8. Ilyas, M., Mahgoub, I.: Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems. CRC Press, Florida (2004)
9. Aguilar, L., Licea, G., García-Macías, J.A.: An Experimental Wireless Network applied to engineering courses. Computer Applications in Engineering Education. Wiley-InterScience, Hoboken (2009)
10. Atmel Corporations, <http://www.atmel.com>
11. Sensirion, <http://www.sensirion.com>
12. Texas Advanced Optoelectronic Solutions, Inc., <http://www.taosinc.com>
13. Linx Technologies, Inc., <http://www.linxtechnologies.com>
14. WinAVR, <http://winavr.sourceforge.net>
15. Danahoo, M., Calvert, K.: TCP/IP Sockets in C: Practical Guide for Programmers. Morgan Kaufmann Publishers, San Francisco (2001)
16. El-Hoiydi, A.: Aloha with preamble sampling for sporadic traffic in ad hoc wireless sensor networks. In: Proc. IEEE Int. Conf. on Communications, ICC 2002 (April 2002)
17. El-Hoiydi, A., Decotignie, J.: WiseMAC: An ultra-low power MAC protocol for multi-hop wireless sensor networks. In: Proc. 1st Int. Workshop on Algorithmic Aspects of Wireless Sensor Networks (2004)
18. Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: Proc. SenSys 2004, pp. 95–107 (2004)
19. IEEE Computer Society LAN MAN Standards Committee: Wireless medium access control (MAC) and physical layer (PHY) specifications for low rate wireless personal area networks (LR-WPANs). IEEE Std. 802.15 (2004)
20. Chipcon, Products from Texas Instruments. CC2420 data sheet, 2.4 Ghz IEEE 802.15.4 / Zigbee-ready RF transceiver
21. Buettner, M., Yee, G.V., Anderson, E., Han, R.: X-MAC: a short preamble mac protocol for duty-cycled wireless sensor networks. In: Proc. 2nd ACM Conf. on Embedded Networked Sensor Systems, SenSys 2006, pp. 307–320 (2006)
22. Wong, K.-J., Arvind, D.: Speckmac: Low-power decentralized MAC protocol low data rate transmissions in specknets. In: Proc. 2nd IEEE Int. Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality, REALMAN 2006 (May 2006)
23. Merlin, C.J., Heinzelman, W.B.: Network-aware adaptation of MAC scheduling for wireless sensor networks. In: Aspnes, J., Scheideler, C., Arora, A., Madden, S. (eds.) DCOSS 2007 Poster Session. LNCS, vol. 4549. Springer, Heidelberg (2007)
24. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
25. Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies 7(1), 1–13 (1975)
26. Sugeno, M.: Industrial applications of fuzzy control. Elsevier Science Pub. Co., Amsterdam (1985)
27. Jang, J.S.R., Sun, C.T.: Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Prentice Hall, Englewood Cliffs (1997)

Multi-Agent System Based on Psychological Models for Mobile Robots

Arnulfo Alanis Garza, Oscar Castillo, and José Mario García Valdez

Division of Graduate Studies and Research, Instituto Tecnológico de Tijuana, Calzada Tecnológico, Tijuana, Mexico

alanis@tectijuana.edu.mx, ocastillo@tectijuana.mx

Abstract. We describe the design of Intelligent Agents and experimenting with learning environments in mobile robotics, using new technologies such as the Knowledge Query and Manipulation Language (KQML), and new features of the Jade programming language, for the creation of distributed applications such as Remote Method Invocation (RMI). A society of intelligent agents could solve these problems. These agents would be able to migrate from one node to another, the ability to communicate with other agents, using an expressive communication language, work together cooperatively to accomplish complex objectives for a user, act on their own initiative and use local information and knowledge to manage resources and requirements of other agents. Most existing software ignores the fact that generally the tasks performed by an individual are part of collective activities. There is individual work, is working with third-group contributions and consensus with others, is why the support systems to collaborative work are presented as a particular need for the development of future software and intelligent agents, as an essential component design and implementation.

1 Introduction

Making smarter robots is a problem that has captivated the scientific community for several years now, this being a big challenge to overcome even for man. The move from one place to another from an initial to a final point with only the information of what we want to reach at the end is a complicated task. The problem has been solved with soft computing methods, such as fuzzy logic, neural networks, genetic algorithms, hybrid systems, among others. Navigation of the robot has been achieved taking a completely controlled environment where we have total knowledge of the world [4,7, 11].

Human beings have the ability to react to unexpected situations and the ability to use their reasoning to react to situations that advise, an example is a situation of traffic management, to which we think and what is the best route and with this the rest of the trip is done reacting to what was observed. This could be, at a traffic

light, when it changes from green to yellow, the reaction will be learning, acceleration or deceleration, a situation that when you start, it has to be learned.

Knowing that we have this ability to react, we will use fuzzy logic to transfer the knowledge we use to carry out this process by establishing computer-type "if-then" fuzzy rules and exploiting the advantages offered to us by fuzzy logic, which is the use of linguistic variables.

A man driving a car as in many other activities derives its information by looking at distances, clearances and other identifying information that we collect through the light. In this research we use computer vision techniques to draw from that vital information for navigation control, supported with special sensors. An example is the ultrasonic sensors that are used to measure distance and the light sensors that we can use to measure the change in light intensity, all of these help to extract environmental information necessary for decision making.

2 Preliminary

2.1 Agents

Let's first deal with the notion of intelligent agents. These are generally defined as "software entities", which assist their users and act on their behalf. Agents make your life easier, save you time, and simplify the growing complexity of the world, acting like a personal secretary, assistant, or personal advisor, who learns what you like and can anticipate what you want or need. The principle of such intelligence is practically the same of human intelligence. Through a relation of collaboration-interaction with its user, the agent is able to learn from himself, from the external world and even from other agents, and consequently act autonomously from the user, adapt itself to the multiplicity of experiences and change its behavior according to them. The possibilities offered for humans, in a world whose complexity is growing exponentially, are enormous [12][13].

We need to be careful to distinguish between rationality and omniscience. An omniscient agent knows the actual outcome of its actions, and can act accordingly; but omniscience is impossible in reality. Crossing the street was rational because most of the time the crossing would be successful, and there was no way I could have foreseen the falling door. Note that another agent that was equipped with radar for detecting falling doors or a steel cage strong enough to repel them would be more successful, but it would not be any more rational [17].

2.2 Agent Based Simulations

As these non-linear, adaptive interactions are mostly too complex to be captured by analytical expressions, computer simulations are most often used. The basic idea of such simulations is to specify the rules of behavior of individual entities, as well as the rules of their interaction, to simulate a multitude of the individual entities using a computer model, and to explore the consequences of the specified individual-level rules on the level of population as a whole, using results of simulation runs. As the simulated entities are usually called agents, the

simulations of their behavior and interactions are known as agent-based simulations. The properties of individual agents describing their behavior and interactions are known as elementary properties, and the properties emerging on the higher, collective level are known as emergent properties [8].

2.3 Basic Properties of Agent-Based Models

What makes agent-based models particularly appealing and interesting is that consequences on the collective level are often neither obvious, nor expectable, even in many cases when the assumptions on individual agent properties are very simple. Namely, the capability of generating complex and intriguing emergent properties arises.

Not so much from the in-built rules of individual agent behavior, as from the complexity of the network of interactions among the agents. Precisely this multitude of agents, as well as the multitude and complexity of their interactions, are the main reasons why in most cases formal mathematical deduction of results of an agent-based model is not possible.

This is also the reason why the issues of complexity remained relatively underexplored until recently. Namely, as scientists regularly decide to pay attention to “problems defined by the conceptual and instrumental techniques already at hand” [4], “some facts [...] are pushed to the periphery of scientific investigation, either because they are thought not to be relevant, or because their study would demand unavailable techniques”. Accordingly, only after recent advances in the development of computational technology have enabled massive simulation experiments, the issues of emergent complexity came closer to the focus of scientific research.

Besides the above mentioned modeling of bottom-up effects, i. e. the effects originating at the individual level and influencing the collective one, more complex agent-based models are also capable of modeling top-down effects, arising at the collective level and influencing the level of individual agents[14].

3 Agent Based Models in Social Sciences

Although most well-established within the framework of natural sciences, the application of agent-based simulations within the field of social sciences since 199 is also significantly growing. It must be emphasized that the primary purpose of agent-based modeling and simulations in social sciences is not prediction. Namely, social processes are usually so complex that their maximally faithful replicas would hardly ever be possible. The consequence is that agent-based models mostly do not possess the level of “accuracy “needed for a model to be used for predictive purpose[14,8,15].

4 What Is Fuzzy Logic?

In this context, FL is a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data

acquisition and control systems. It can be implemented in hardware, software, or a combination of both. FL provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information. FL's approach to control problems mimics how a person would make decisions, only much faster [6].

FL incorporates a simple, rule-based IF X AND Y THEN Z approach to a solving control problem rather than attempting to model a system mathematically. The FL model is empirically-based, relying on an operator's experience rather than their technical understanding of the system. For example, rather than dealing with temperature control in terms such as "SP =500F", "T <1000F", or "210C <TEMP <220C", terms like "IF (process is too cool) AND (process is getting colder) THEN (add heat to the process)" or "IF (process is too hot) AND (process is heating rapidly) THEN (cool the process quickly)" are used. These terms are imprecise and yet very descriptive of what must actually happen. Consider what you do in the shower if the temperature is too cold: you will make the water comfortable very quickly with little trouble. FL is capable of mimicking this type of behavior but at very high rate [18].

5 Big Five Personality Factors

Why do we study personality?

The NEO that you have just completed looks at 5 personality traits, known as the Big Five. We will briefly look at what traits are, how these personality factors were determined, what the traits mean, what the Big Five predict about our behavior, and how these factors might relate to motivation.

What are traits?

Traits are consistent patterns of thoughts, feelings, or actions that distinguish people from one another. Traits are basis tendencies that remain stable across the life span, but characteristic behavior can change considerably through adaptive processes. A trait is an internal characteristic that corresponds to an extreme position on a behavioral dimension.

There have been different theoretical perspectives in the field of personality psychology over the years including human motivation, the whole person, and individual differences. The Big Five falls under the perspective of individual differences.

How were these personality factors determined?

The Big Five represents taxonomy (classification system) of traits that some personality psychologists suggest capture the essence of individual differences in personality. These traits were arrived at through factor analysis studies. Factor is a technique generally done with the use of computers to determine meaningful relationships and patterns in behavioral data. You begin with a large number of behavioral variables. The computer finds relationships or natural connections where variables are maximally correlated with one another and minimally

correlated with other variables and then groups the data accordingly. After this process has been done many times a pattern appears of relationships or certain factors that capture the essence of all of the data. Such a process was used to determine the Big Five Personality factors. Many researchers tested factors other than the Big Five and found the Big Five to be the only consistently reliable factors [6, 14, 16].

Strict trait personality psychologists go so far as to say our behavior is really determined by these internal traits, giving the situation a small role in determining behavior. In other words, these traits lead to an individual acting a certain way in a given situation.

Allport, Norman and Cattell were influential in formulating this taxonomy which was later refined. Allport compiled a list of 4500 traits. Cattell reduced this list to 35 traits. Others continued to analyze these factors and found congruence with self-ratings, ratings by peers and ratings by psychological staff that eventually became the Big Five factors.

The Big Five factors are:

- I – extraversion vs introversion
- II – agreeableness vs antagonism
- III – conscientiousness vs undirectedness
- IV – neuroticism vs emotional stability
- V – openness to experience vs not open to experience

There was a need for an integrative framework for measuring these factors. The NEO Personality Inventory was created by Costa and McCrae and originally measured only neuroticism, extraversion and openness. The other factors were added later. There are other measures of the Big Five, such as the BFI (Big Five Inventory) and the TDA (Traits Descriptive Adjectives). The NEO has the highest validity of the Big Five measurement devices[9].

What do the five traits mean?

Keep in mind that the traits fall on a continuum and this overhead shows characteristics associated with each of the traits. Looking at these characteristics we can formulate what each of the traits mean.

- **Openness** - (inventive / curious vs. cautious / conservative). Appreciation for art, emotion, adventure, unusual ideas, curiosity, and variety of experience.
- **Conscientiousness** - (efficient / organized vs. easy-going /careless). A tendency to show self-discipline, act dutifully, and aim for achievement; planned rather than spontaneous behavior.
- **Extroversion** - (outgoing / energetic vs. shy / withdrawn). Energy, positive emotions, urgency, and the tendency to seek stimulation in the company of others.
- **Agreeableness** - (friendly / compassionate vs. competitive / outspoken). A tendency to be compassionate and cooperative rather than suspicious and antagonistic towards others.

- **Neuroticism** - (sensitive / nervous vs. secure /confident). A tendency to experience unpleasant emotions easily, such as anger, anxiety, depression, or vulnerability [6, 14, 16].

6 Proposed Method

We present a sample application of ODD to a population based model of a football game, for reasons of limited space we have chosen a relatively simple model that describes many of the processes empirically by using probabilities, for example, “defend the doorman”.

Extraversion, Agreeableness

The purpose of the model is to understand how social Partre players' behavior, particularly territoriality.

Conscientiousness

The model comprises four hierarchical levels: individual territory of (Target) of the equipment and the environment. Individuals are characterized by state variables: identity number, age, sex, identity of the area where the individual and social life. The players have recently entered the process of additional state variable weight, which affects their behavior. The Complaint players who have not known his first game as minors, 1-game played, as initiates, and all others as completed. Apart from this, social rank is the main attribute which tells the difference between the complete key (experts) and subdominant (started).

Openness, Neuroticism

The product of the model in steps, the time of each match will be fingers or stages of processing modules in the following order: the individual, the expert, height, weight, changing players, changing positions. Within each module, the players and the field are processed in random order. A player in the life cycle is represented as a tree.

6.1 Design Concepts

Emergency: Dynamics of the population to leave the conduct of (the players), but the individual's life cycle and behavior are fully represented by empirical rules.

For this to work, each player creates an SMA (robot), which consists of three agents, an agent for the management of the sensors, an agent that is responsible for controlling the engine and the last agent to act as coordinating multi-agent system [1,2,3].

Summarizing the SMA is composed of the following agents:

Node Agent (AN), a set of sensors and personality factors.

Task Agent (AT), a set of servo motors.

System Agent (AS), coordinator of the system.

The Node Agent (NA) will be responsible for the sensors used by our robot that are:

- Ultrasonic Sensor
- Light Sensor left
- Light Sensor right
- Camera

The set of sensors has great features associated to the personality, such as openness, conscientiousness, extroversion Agreeableness, Neuroticism.

Node Agent through the sensors, receive their world and according to its environment and activity, has one of the five personality factors, which may change according to the perceptions of the world and the other elements.

These sensors are used to interact with the world to achieve our goal.

The Task Agent (TA) will be used to drive to move forward, rewind, or make some movement to escape.

The System Agent (SA) will be responsible for coordinating the other actors, the AN and AT, and will also have activities not only the coordination, which will be to recognize the object that is opposite, calculate the angle of the object, which will help us to know that will guide the robot. Depending on the recognition that was obtained, whether or not the object, which will tell us that there is no escape or the object based on this decision was to trigger the switch which will control the outcome to be used, reactive or path to better performance of the controllers, fig 1 [5].

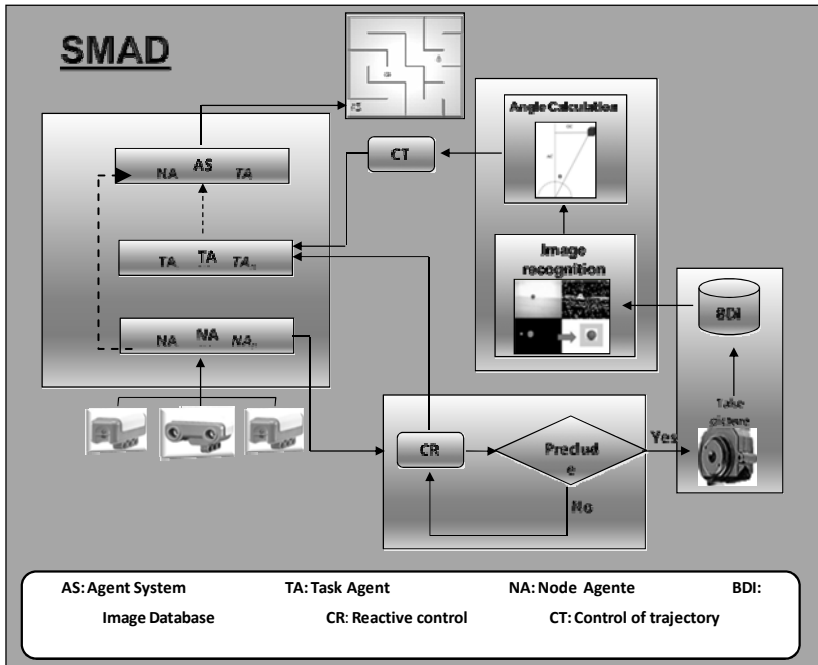


Fig. 1. Architecture of the complete system

We can describe the proposed method as follows:

In the operation of the model, we have 2 main blocks, which are responsible for knowledge and learning (paradigm of intelligent agents) and the vision and control (fuzzy logic).

These modulus are described below, the first module contains 3 agents, NA [Node Agent], TA [Task Agent] SA [System Agent] Agent System (AS), and will know every time the operation of the other agents.

To detect a change in the environment the Agent Node [that is in charge of the sensors (ultrasonic, camera and two light sensors)] with the ultrasonic sensor and two light sensors, starts its operation, and we start at the state called reactive control; this because you have to move and sense all the time until it finds an obstacle, this can happen once a photo is taken, which will be sent to the database of images (BDI), the image will be applied a pre-processing for recognition in order to know whether the object is found, this decision was taken on the angle (angle to take the decision to bypass the object), able to escape and move forward, trend data, they are caught in the trajectory control process, to observe the speed values. The Task Agent (which is responsible for the drive), once all the necessary parameters for the performance of the robot agent system (AS) who is the coordinator, and executes instructions in the robot.

For the development of the Multi-Agent System we used Gaia [19] and UML for the analysis and design of the agents. The completion of the Multi-Agent System (MAS) is based on the FIPA [20] standards for better utilization and greater possibility of extension [18].

6.2 Gaia

Below we show the diagrams in which we can observe the responsibilities, permissions, activities and protocols to be followed by each agent in the Multi-Agent System (MAS).

Agent node

Responsibilities:

Sense:

- Active
- Inactive
- And each of the Big Five Personality factors, and depending on which factor is the one needing the robot will have one or more factors, and this will give her unique personality (openness, conscientiousness, extroversion Agreeableness, Neuroticism).

Permissions:

Ultrasonic sensor: measures distances from 0 cm to 255 cm, with a delay of one millisecond signal.

Camera Sensor: take pictures in an estimated time of seconds, saves all images in a folder for processing. It has a degree of vision approximately of 50 degrees.

Light sensors: they measure the intensity with which an object reflects light. This makes a light emitting and measuring the portion of the return is received.

Activities:

The only activities are sensing and taking pictures of the world.

Protocols:

Ultrasound: This is responsible for sensing the distance to get to an obstacle. The initiative for this would be the agent system, which indicates the time of initiation.

Chamber is responsible for obtaining the images within the scene, and as for the ultrasonic sensor system depends for its initiation.

Light sensors: the role of these sensors is to measure the intensity with which an object reflects light (walls, diagrams). This makes a light emitting and measuring the portion of the return is received and handled the ranges are 0 to 1023 RAW.

Agent system

Responsibilities:

It is responsible for image processing and decision making, and sends a message to agent communication process completed?

Permissions:

This is the one that has full access, sensors, communication, engine and handling agents and NXT in general.

Activities:

Making decisions based on the behavior and implements what is needed to finish the job.

Protocols:

Full Access, sensors, communication, engine and handling agents and NXT in general, the entry for this information would be provided by the agents, getting a response from these so they generate an output and generate interaction among them.

Task agent

Responsibilities:

Motor movement

Permission:

Depends entirely on the agent system to perform its task.

Activities:

Sensors for the engine: they measure in degrees is given for each engine is 360 degrees for one revolution, a revolution for the tire of each motor, the motors can rotate independently, can also be used with constant speed motors.

6.3 Knowledge Base

The knowledge base with which the MAS is working, is shown below.

NA

$NA_i.SU.STATE = (X)$, where X can have 7 states (ON, OFF, OPENNESS, CONSCIENTIOUSNESS, EXTROVERSION AGREEABLENESS, NEUROTICISM)

$NA_i.SU.OBJ = (Y)$ where Y can have 2 states (DETECTS, NO DETECTS)

$NA_i.SC.STATE = (Z)$ where Z can have 7 states (ON, OFF, OPENNESS, CONSCIENTIOUSNESS, EXTROVERSION AGREEABLENESS, NEUROTICISM)

$NA_i.SC.TIMAGE$

$NA_i.SLI.STATE = (W)$ where W can have 2 states (ON, OFF)

$NA_i.SLD.STATE = (V)$ where V can have 2 states (ON, OFF)

7 Results

We show in Figure 2 the plant and the controller that we are used for achieving the simulation in Simulink of Matlab.

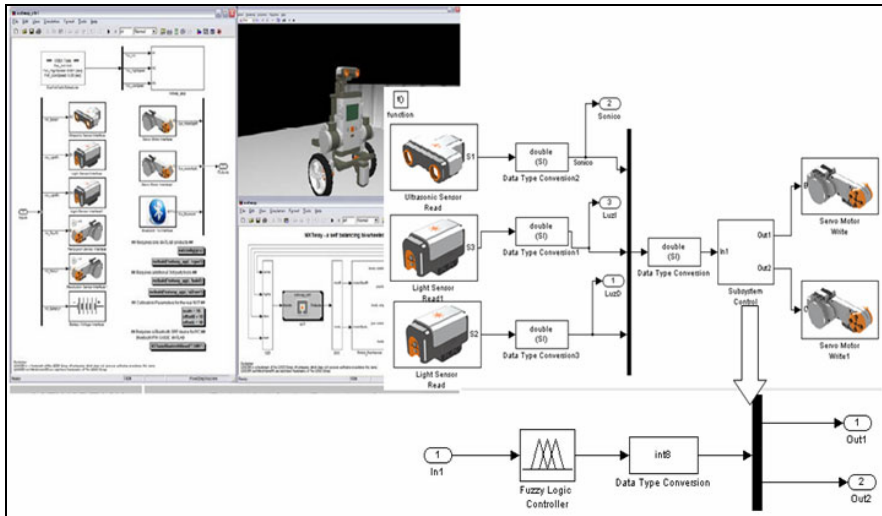


Fig. 2. Plant for simulation

Together with some fuzzy systems which were developed based on work taken as a reference [9,10] and that were the basis for the experimentation with different numbers of rules, parameters of membership functions, and types of membership functions was a good performance of our robot, achieving our goal.

The first input variable of the fuzzy system is the ultrasonic sensor which has three membership functions which are linguistic (close, near, far), as shown in Fig. 3.

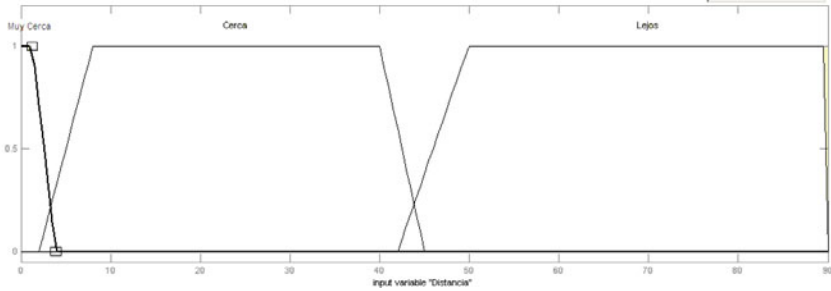


Fig. 3. The Ultrasonic sensor

The second variable of the fuzzy system is the light sensor that has three membership functions, which are low, middle and high, as shown in Fig. 4.

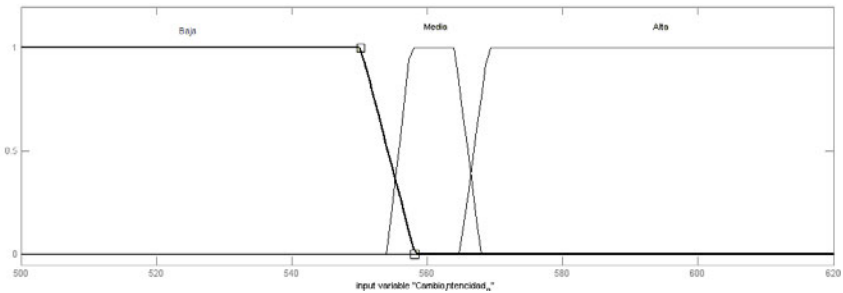


Fig. 4. The left light sensor

The third variable of the fuzzy system is the light sensor that has three membership functions, which are low, middle and high, as shown in Fig. 5.

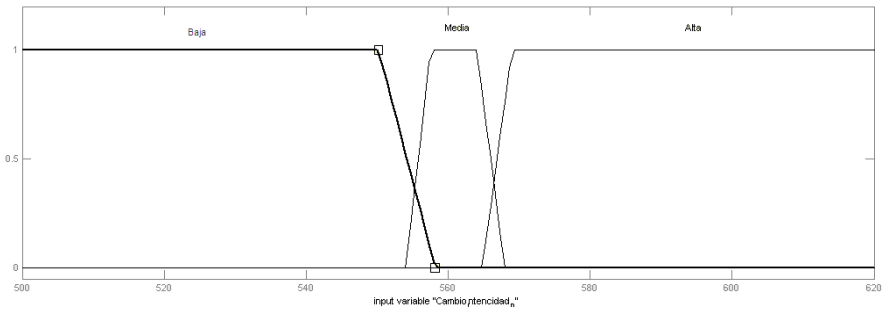


Fig. 5. The right light sensor

This is the first output variable is the speed of the left engine, has five membership functions as shown in Fig. 6.

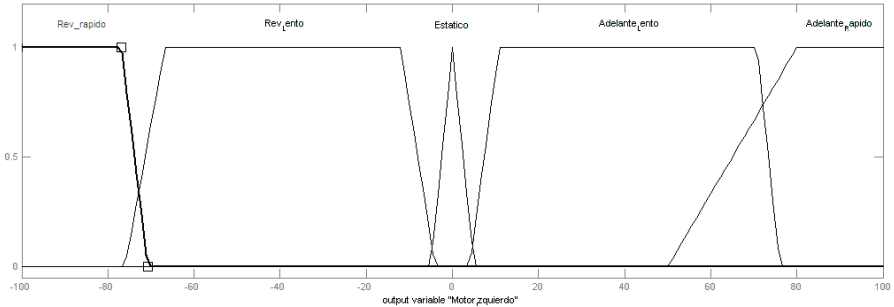


Fig. 6. Motor left

In Fig. 7 we show the rules that are used in our system, which are 10 fuzzy rules.

1. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Rev_Lento) (1)
2. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Rev_Lento) (1)
3. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Rev_Lento) (1)
4. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Rev_Lento) (1)
5. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
6. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Rev_Rapido) (1)
7. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Rev_Rapido)(Motor_Izquierdo is Adelante_Rapido) (1)
8. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Rev_Rapido)(Motor_Izquierdo is Adelante_Lento) (1)
9. If (Distancia is Muy_Cerca) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Rapido)(Motor_Izquierdo is Rev_Rapido) (1)
10. If (Distancia is Cerca) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
11. If (Distancia is Cerca) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
12. If (Distancia is Cerca) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Rev_Lento) (1)
13. If (Distancia is Cerca) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
14. If (Distancia is Cerca) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
15. If (Distancia is Cerca) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Rev_Lento) (1)
16. If (Distancia is Cerca) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
17. If (Distancia is Cerca) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
18. If (Distancia is Cerca) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
19. If (Distancia is Lejos) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
20. If (Distancia is Lejos) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
21. If (Distancia is Lejos) and (Cambio_Intencidad_D is Baja) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Rev_Lento) (1)
22. If (Distancia is Lejos) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
23. If (Distancia is Lejos) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
24. If (Distancia is Lejos) and (Cambio_Intencidad_D is Media) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Rev_Lento) (1)
25. If (Distancia is Lejos) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Baja) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
26. If (Distancia is Lejos) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Media) then (Motor_Derecho is Rev_Lento)(Motor_Izquierdo is Adelante_Lento) (1)
27. If (Distancia is Lejos) and (Cambio_Intencidad_D is Alta) and (Cambio_Intencidad_J is Alta) then (Motor_Derecho is Adelante_Lento)(Motor_Izquierdo is Adelante_Lento) (1)

Fig. 7. Rules of the fuzzy system (FIS)

An example of the results simulating this fuzzy system is shown in the figures 8, 9 and 10, in which we see that as our robot moves forward in the path of the maze to find out achieved or failing not find it.

Having modified and improved the fuzzy systems, we proceeded to a slight modification to our Multi-Agent System, which consists in adding a new agent which has the task of giving a second option in case our robot gets stuck somewhere in the world, for example corners in which our sensors cannot do some action, then come into action. The new agent called Support Agent enters into action as I mentioned in the case that the sensor values do not become active some rule of fuzzy systems that we throw the speed of the drive to get out of where we are stuck with only works with the task agent (TA) for which our multi-agent system would be as shown in Fig. 11.

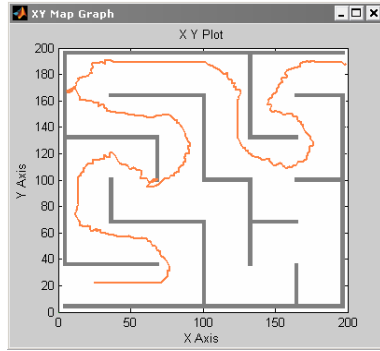


Fig. 8. Simulation-11 duration of 01:03 minutes

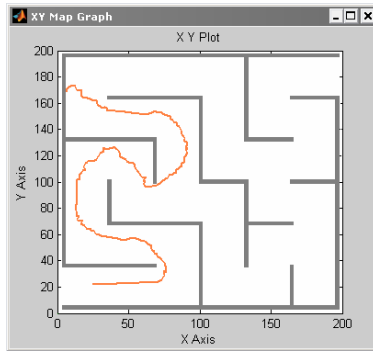


Fig. 9. Simulation-22 duration of 01:30 minutes

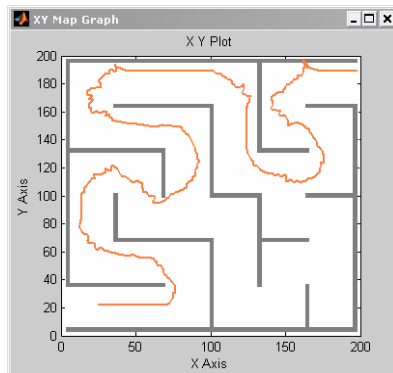


Fig. 10. Simulation-3 duration of 00:58 minutes

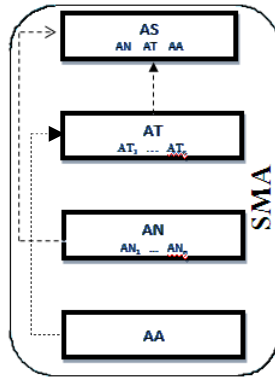


Fig. 11. New scheme of Multi-Agent System

Integrating it to our general scheme we would be as shown in Fig 12.

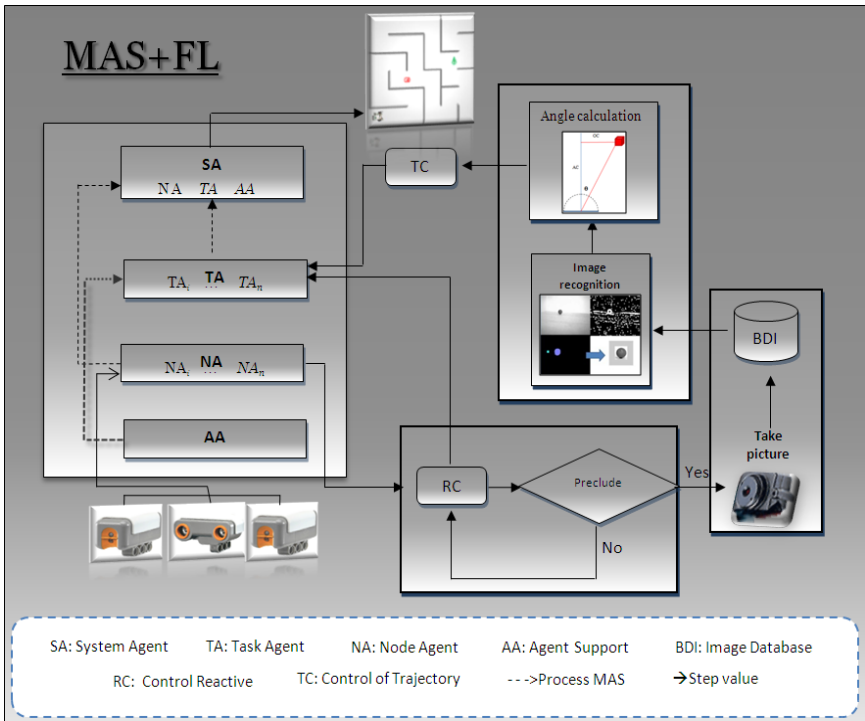


Fig. 12. New architecture of the complete system

Now with these changes we had better control of movements of the robot in the search for the exit of the maze, which can be seen in the graphs of the following experiments, fig. 13,14,15.

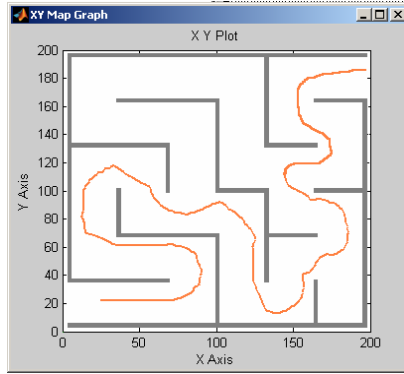


Fig. 13. Simulation-1 duration of 00:55 minutes

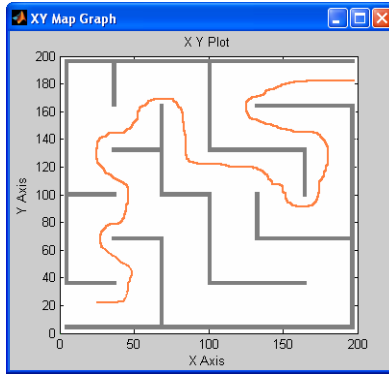


Fig. 14. Simulation-2 duration of 00:47 minutes

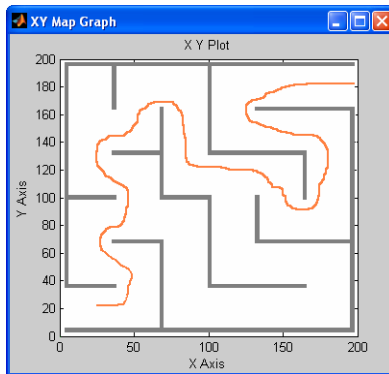


Fig. 15. Simulation-3 duration of 00:47 minutes

8 Conclusions

The development of intelligent applications is varied, in this particular case, the implementation of multi-agent systems is complex because it consists of several agents, while the incorporation of other techniques, making it even more complex.

One of the main objectives of this research initially was to open a new area of intelligent agents using fuzzy logic, which hitherto has not been given the approach that is intended here, which gives a degree of utilization of this paradigm of intelligent agents, but more research is needed in order to defend this idea as innovative.

Considering the non-optimized fuzzy systems, we have shown better results than were obtained in the work we take as a reference [14] for what we have achieved a good percentage of the overall project objectives and to improve not only the times but we have improved how it moves the robot by making our world more soft and therefore earns a bit of time helping to improve.

And we gave another level to our Multi-Agent System as it not only succeeded in getting out of the labyrinth with which we have been working but we show results where the maze was changed in different ways and still finds the exit and not in a very long time. Indeed the average that has been obtained in the work of reference and is even a bit less here, so we are happy with the results we have achieved over not satisfied because we still have work to resolve because we have not yet optimized the system, with the hope that optimization can further improve the results obtained previously and perhaps even lower computation times are achieved.

References

- [1] Alanis, A.: Contribution to the design of Fault Tolerant Distributed Systems for industrial control: proposal for a new paradigm based on Intelligent Agents, PhD, Universidad Politecnica de Valencia (November 2007)
- [2] Alanis, A., López, M., Parra, B., Serrano, M., Ayala, E., Solano, C.: Multi-agent system for search and object recognition using vision, in a Lego NXT robot 1, XXIX International Congress of Engineering in electronics, electro 2008 Instituto Tecnológico de Chihuahua, division of graduate studies and research, 29, 31, Chihuahua, Chih. Mexico (October 2008)
- [3] Alanis, A., López, M., Parra, B., Serrano, M., Ayala, E., Solano, C.: Multi-agent system for search and object recognition using vision, in a Lego NXT robot 1.8 °. In: National Congress of Electrical and Electronics Engineering of the Mayab conieem 2008 Instituto Tecnológico de Mérida, of Merida, Yucatan, Mexico April 21-25 (2008)
- [4] Azam, F.: Biologically Inspired Modular Neural Networks, Electrical and Computer Engineering, Blacksburg, Virginia (2000)
- [5] Hogan, R., Johnson, J., Briggs, S. (eds.): Handbook of personality psychology, vol. 9. Academic Press, California (1997)
- [6] <http://www.fipa.org/> FIPA is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies

- [7] David, H.: Japan starts Fuzzy Logic II, *Machine Design*, September 10, p. 16 (1992)
- [8] Jang, S.J., Mizutani, E.: “Neuro-Fuzzy” and soft computing: a computational approach to learning and machine intelligence. Prentice-Hall, Englewood Cliffs (1997)
- [9] Melendez, A., Castillo, O., Soria, J.: Reactive Control of a Mobile Robot in a Distributed Environment Using Fuzzy Logic. In: *NAFIS 2008* (2008)
- [10] Melendez, A.: Control and monitoring reagent for a mobile robot using fuzzy logic, thesis work, ITT (August. 2008)
- [11] Morgan, D.P., Scofield, C.L.: *Neural Networks and Speech Processing*. Kluwer Academic Publishers, Dordrecht (1991)
- [12] Norvig, P., Russell, S.: *Artificial intelligence a modern approach*. Prentice-Hall, Australia (1996)
- [13] Cohen, P.R.: An Open Agent Architecture. In: *Working Notes of the AAAI Spring symp.: Software Agent*, pp. 1–8. AAAI Press, Cambridge (1994)
- [14] Pervin, L., John, O. (eds.): *Handbook of personality: theory and research*, vol. 8. Guilford, New York (1999)
- [15] Pitt, W.C., Box, P.W., Knowlton, F.F.: An individual-based model of canid populations: modelling territoriality and social structure. *Ecol. Model.* 166, 11, 109–121 (2003)
- [16] Potkay, C., Allen, B.: *Personality: theory, research, and applications*. Brooks/Cole, 10, California (1986)
- [17] Russell, S., Norvig, P.: Chapter Intelligent Agent. In: *Artificial Intelligence to Modern Aproach*, Prentence artificial Hall series in intelligence, pp. 31–52 (1994)
- [18] Aragon, S.C.: Multi-agent system of fuzzy control for autonomous mobile robots, Masters Dissertation, Instituto Tecnológico de Tijuana, Mexico (September 2009)
- [19] University of Vigo, Department of Computer Science, Problems of search and optimization (October 2004)
- [20] Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G.: A standard protocol for describing individual-based and agent-based, vol. 198(1-2), pp. 115–126 (September 15, 2006)

Fuzzy Control for Dynamical Parameter Adaptation in a Parallel Evolutionary Method Combining Particle Swarm Optimization and Genetic Algorithms

Fevrier Valdez, Patricia Melin, and Oscar Castillo

Tijuana Institute of Tech. Tijuana BC. México
pmelin@tectijuana.mx

Abstract. We describe in this paper an approach for mathematical function optimization using parallel computing combining Particle Swarm Optimization (PSO) and Genetic Algorithms (GAs) and Fuzzy Logic for parameter adaptation and integrate the results. The parallel evolutionary method combines the advantages of parallel computing, PSO and GA to give us an improved parallel FPSO+FGA hybrid method. Fuzzy Logic is used to combine the results of the PSO and GA in the best way possible. The parallel hybrid FPSO+FGA method was developed using a computer with processor Intel Core 2 Quad of 64 bits that works to a frequency of clock of 2.5 GHz, 6 GB of RAM Memory and Ubuntu Linux Operating System.

Keywords: Parallel FPSO, FGA, GA.

1 Introduction

We describe in this paper a new parallel evolutionary method combining PSO and GA, to give us an improved FPSO+FGA hybrid method. We apply the parallel hybrid method to mathematical function optimization to validate the new approach using the parallel mode of the Matlab programming language. The application of a Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) separately using parallel computing for the optimization of mathematical functions is shown in [5]. In this case, we are using a set of mathematical benchmark functions [4] [13] to compare the optimization results between a GA, PSO and FPSO+FGA.

The main motivation of this method is to combine the characteristics of a GA and PSO in parallel mode using fuzzy logic. We are using several fuzzy systems to perform dynamical parameter adaptation. For decision making between the methods depending on the results that we are generating we are using one fuzzy system. The fuzzy system is used to decide and combine the outputs of both the GA and PSO, in this way obtaining the best solution to the optimization problem. Also, there are another two fuzzy systems to parameter adaptation in GA and PSO. The main goal of the fuzzy system is to evaluate the outputs of the GA and

PSO in each generation and change if is necessary some important parameters. The criterion for stopping the method is the maximum number of generations.

The paper is organized as follows: in section 2 a description about Genetic Algorithms for optimization problems is presented, in section 3 the Particle Swarm Optimization is presented, in section 4 the proposed method parallel FPSO+FGA and the fuzzy systems are described, in section 5 we can appreciate the mathematical functions that were used for this research, in section 6 the experimental results are described, in section 7 the conclusions obtained after the study of the proposed method are presented.

2 Genetic Algorithms for Optimization

Holland, from the University of Michigan initiated his work on genetic algorithms at the beginning of the 1960s. His first achievement was the publication of *Adaptation in Natural and Artificial System* [7] in 1975.

He had two goals in mind: to improve the understanding of natural adaptation process, and to design artificial systems having properties similar to natural systems [8].

The basic idea is as follows: the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution.

Holland's method is especially effective because it not only considers the role of mutation, but it also uses genetic recombination, (crossover) [9]. The crossover of partial solutions greatly improves the capability of the algorithm to approach, and eventually find, the optimal solution.

The essence of the GA in both theoretical and practical domains has been well demonstrated [1]. The concept of applying a GA to solve engineering problems is feasible and sound. However, despite the distinct advantages of a GA for solving complicated, constrained and multi-objective functions where other techniques may have failed, the full power of the GA in application is yet to be exploited [12] [14].

In figure 1 we show the reproduction cycle of the Genetic Algorithm.

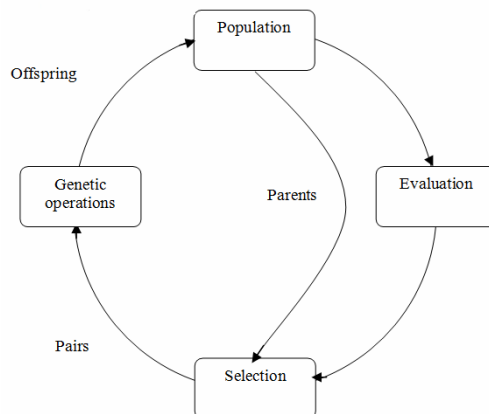


Fig. 1. The Reproduction cycle

The Simple Genetic Algorithm can be expressed in pseudo code with the following cycle:

1. *Generate the initial population of individuals aleatorily $P(0)$.*
2. *While (number _ generations <= maximum _ numbers _ generations)*
 - Do:*
 - {*
 - Evaluation;*
 - Selection;*
 - Reproduction;*
 - Generation ++;*
 - }*
3. *Show results*
4. *End*

3 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by the social behavior of bird flocking or fish schooling [3].

PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA) [6]. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike the GA, the PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles [10].

Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far (The fitness value is also stored). This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest*. When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest*.

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its *pbest* and *lbest* locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *lbest* locations [2].

In the past several years, PSO has been successfully applied in many research and application areas. It is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods [11].

Another reason that PSO is attractive is that there are few parameters to adjust. One version, with slight variations, works well in a wide variety of applications. Particle swarm optimization has been used for approaches that can be used across a wide range of applications, as well as for specific applications focused on a specific requirement.

The pseudo code of the PSO is as follows

For each particle

Initialize particle

End

Do

For each particle

Calculate fitness value

*If the fitness value is better than the best fitness value (pBest) in history
set current value as the new pBest*

End

Choose the particle with the best fitness value of all the particles as the gBest

For each particle

Calculate particle velocity

Update particle position

End

While maximum iterations or minimum error criteria is not attained

4 Full Model of the Parallel FPSO+FGA

The general approach of the proposed method FPSO+FGA can be seen in figure 2 with a sequential approach. In figure 3 it can be seen the parallel method. The parallel approach of the method can be described as follows:

1. It receives a mathematical function to be optimized
2. Four cores (called core1, core2, core3 and core4) are working in parallel mode to run the method.
3. Each core has the work of run the FPSO+FGA, beginning at same time and sending the results to matlab client (view figure 3).
4. The process in each core can be described as follow:
 - a. It evaluates the role of both GA and PSO.
 - b. A main fuzzy system is responsible for receiving values resulting from step 2.
 - c. The main fuzzy system decides which method to take (GA or PSO)
 - d. After, another fuzzy system receives the Error and DError as inputs to evaluates if is necessary change the parameters in GA or PSO.
 - e. There are 3 fuzzy systems. One is for decision making (is called main fuzzy), the second one is for changing parameters of the GA (is called fuzzyga) in this case change the value of crossover (k1) and mutation (k2) and the third fuzzy system is used to change parameters of the PSO (is called fuzzyposo) in this case change the value of social acceleration (c1) and cognitive acceleration (c2).
5. The main fuzzy system decides in the final step the optimum value for the function introduced in step 1.

6. Repeat the above steps until the termination criterion of the algorithm is met.
7. After, the results obtained by each core are sent to the matlab client.
8. Finally the comparison results between all cores are made in the matlab client.

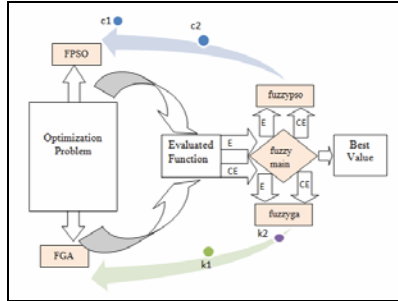


Fig. 2. The FPSO+FGA scheme

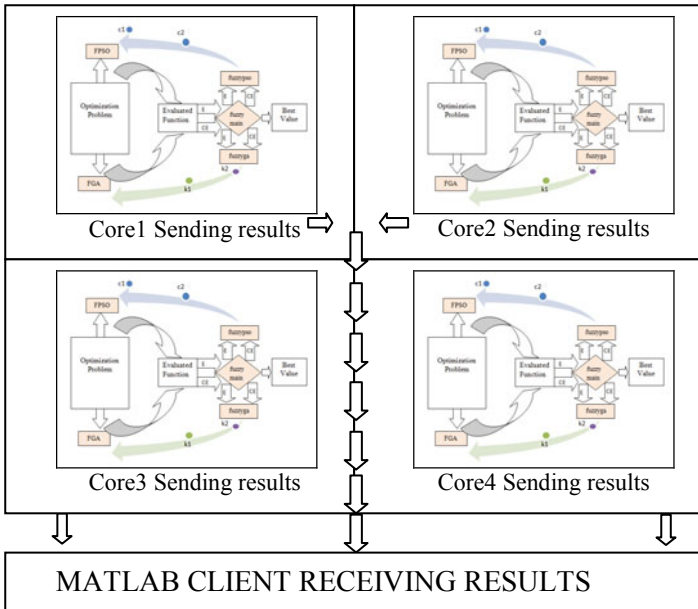


Fig. 3. The parallel FPSO+FGA scheme

The basic idea of the FPSO+FGA scheme is to combine the advantage of the individual methods using a fuzzy system for decision making and the others two fuzzy systems to improve the parameters of the GA and PSO when is necessary and parallel computing to obtain more results in a little time.

As can be seen in the proposed hybrid FPSO+FGA method, it is the internal fuzzy system structure, which has the primary function of receiving as inputs (Error and DError) the results of the FGA and FPSO outputs. The fuzzy system is responsible for integrating and decides which are the best results being generated at run time of the FPSO+FGA. It is also responsible for selecting and sending the problem to the “fuzzypso” fuzzy system when the FPSO is activated or to the “fuzzyga” fuzzy system when FGA is activated. Also activating or temporarily stopping depending on the results being generated. Figure 4 shows the membership functions of the main fuzzy system that is implemented in this method. The fuzzy system is of Mamdani type because it is more common in this type of fuzzy control and the defuzzification method is the centroid. In this case, we are using this type of defuzzification because in other papers we have achieved good results [4]. The membership functions are of triangular form in the inputs and outputs as is shown in figure 4. Also, the membership functions were chosen of triangular form based on past experiences in this type of fuzzy control. The fuzzy system consists of 9 rules. For example, one rule is if error is P and DError is P then best value is P (view figure 5). Figure 6 shows the fuzzy system rule viewer. Figure 7 shows the surface corresponding to this fuzzy system. The other two fuzzy systems are similar to the main fuzzy system.

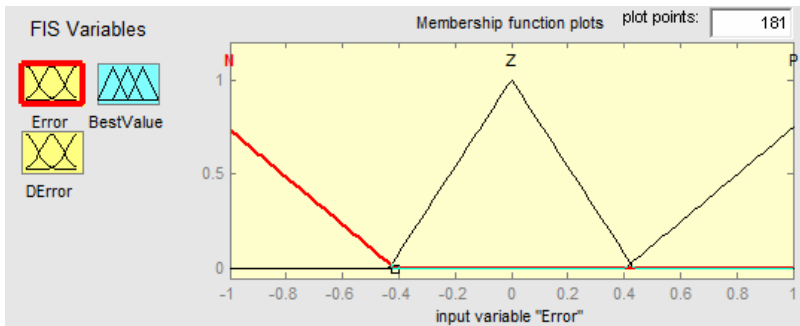


Fig. 4. Fuzzy system membership functions

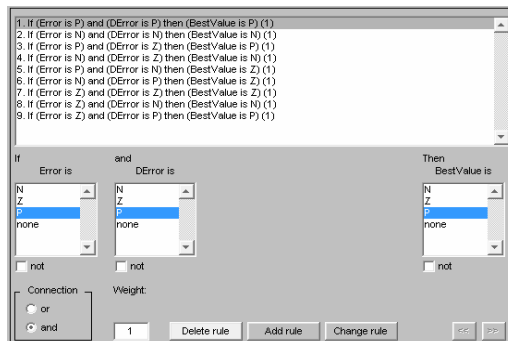


Fig. 5. Fuzzy system rules

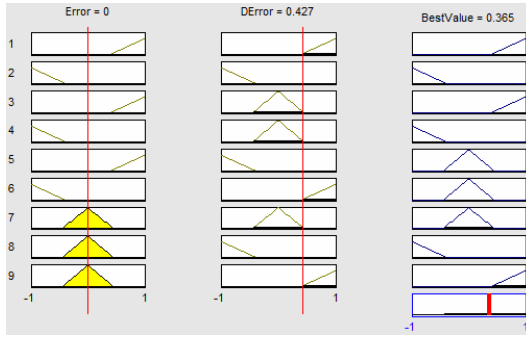


Fig. 6. Fuzzy system rules viewer

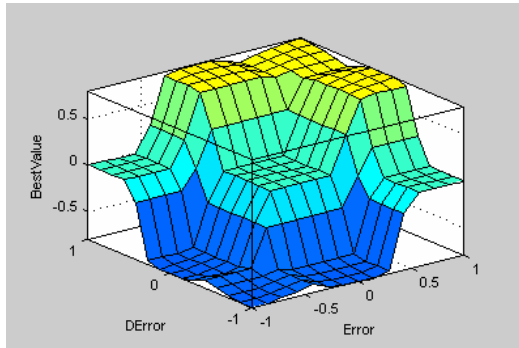


Fig. 7. Surface of fuzzy system

4.1 FPSO (Fuzzy Particle Swarm Optimization)

This section presents a detailed description of the FPSO model. The classical representation scheme for GAs is binary vectors of fixed length. In the case of an n_x – dimensional search space, each individual consists of n_x variables with each variable encoded as a binary string.

The swarm is typically modeled by particles in multidimensional space that have a position and a velocity. These particles fly through hyperspace (i.e., R^n) and have two essential reasoning capabilities: their memory of their own best position and knowledge of the global or their neighborhood's best. In a minimization optimization problem, "best" simply meaning the position with the smallest objective value. Members of a swarm communicate good positions to each other and adjust their own position and velocity based on these good positions. So a particle has the following information to make a suitable change in its position and velocity:

A global best that is known to all and immediately updated when a new best position is found by any particle in the swarm.

The neighborhood best that the particle obtains by communicating with a subset of the swarm.

The local best, which is the best solution that the particle has seen.

In this case, the social information is the best position found by the swarm, referred as $\hat{y}(t)$. For gbest FPSO, the velocity of particle i is calculated as

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \quad (1)$$

Where $v_{ij}(t)$ is the velocity of particle i in dimension $j = 1, \dots, n_x$ at time step t , $x_{ij}(t)$ is the position of particle i in dimension j at time step t , C_1 and C_2 represents the cognitive and social acceleration. In this case, these values are fuzzy because they are changing dynamically when the FPSO is running, and $r_{1j}(t)$, $r_{2j} \sim U(0,1)$ are random values in the range $[0,1]$.

4.2 FGA (Fuzzy Genetic Algorithm)

This section presents a detailed description of the FGA. Several crossover operators have been developed for GAs, depending on the format in which individuals are represented. For binary representations, uniform crossover, one point crossover and two points cross over are the most popular. In this case we are using two points crossover with fuzzy crossover rate because we are adding a fuzzy system called ‘fuzzyga’ that is able of change the crossover and mutation rate.

4.3 Definition of the Fuzzy Systems Used in FPSO+FGA

‘fuzzypso’: In this case we are using a fuzzy system called ‘fuzzypso’, and the structure of this fuzzy system is as follow:

Number of Inputs: 2

Number of Outputs: 2

Number of membership functions: 3

Type of the membership functions: Triangular

Number of rules: 9

Defuzzification: Centroid

The main function of the fuzzy system called ‘fuzzypso’ is to adjust the parameters of the PSO. In this case, we are adjusting the following parameters: ‘ c_1 ’ and ‘ c_2 ’; where:

‘ c_1 ’ = Cognitive Acceleration

‘ c_2 ’ = Social Acceleration

We are changing these parameters to test the proposed method. In this case, with ‘fuzzypso’ is possible to adjust in real time the 2 parameters that belong to the PSO.

‘fuzzyga’: In this case we are using a fuzzy system called ‘fuzzyga’, the structure of this fuzzy system is as follows:

Number of Inputs: 2

Number of Outputs: 2

Number of membership functions: 3

Type of membership functions: Triangular

Number of rules: 9

Defuzzification: Centroid

The main function of the fuzzy system called ‘fuzzygaso’ is to adjust the parameters of the GA. In this case, we are adjusting the following parameters:

‘ k_1 ’, ‘ k_2 ’; where:

‘ k_1 ’ = mutation

‘ k_2 ’ = crossover

‘fuzzygaso’: In this case, we are using a fuzzy system called ‘fuzzygaso’. The structure of this fuzzy system is as follows:

Number of Inputs: 2

Number of Outputs: 1

Number of membership functions: 3

Type of membership functions: Triangular

Number of rules: 9

Defuzzification: Centroid

The main function of the fuzzy system, called ‘fuzzygaso’ is to decide on the best way for solving the problem, in other words if it is more reliable to use the FPSO or FGA. This fuzzy system is able to receive two inputs, called error and derror, it is to evaluate the results that are generated by FPSO and FGA in the last step of the algorithm.

5 Benchmark Mathematical Functions

To validate our method we used a set of 5 benchmark mathematical functions; all functions were evaluated in parallel mode with different numbers of variables, in this case, the experimental results were obtained with 16, 32, 64 and 128 variables.

5.1 De Jong's Function 1

The simplest test function is De Jong's function 1. It is also known as sphere model, and is defined by this equation:

$$f_1 = \sum_{n=1}^N x_n^2 \quad \text{minimum} = 0, \text{ for } -\infty \leq x \leq \infty \quad (2)$$

The visualization of this function it can be seen in figure 8.

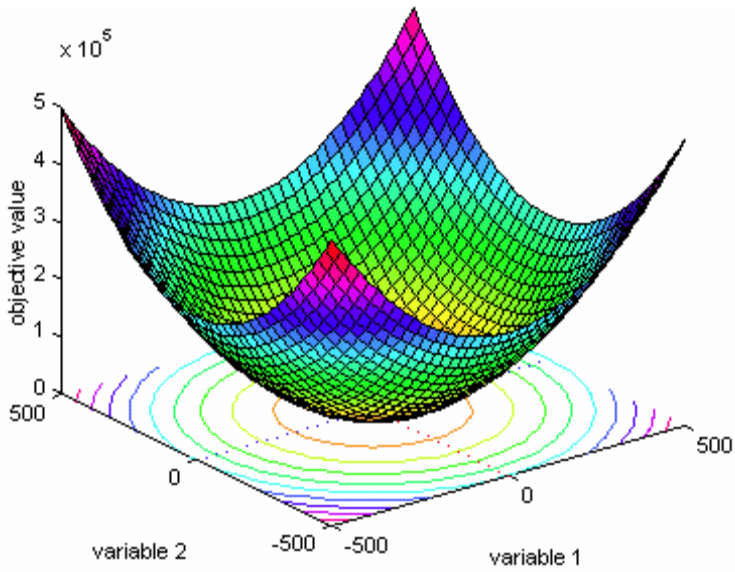


Fig. 8. De Jong's function

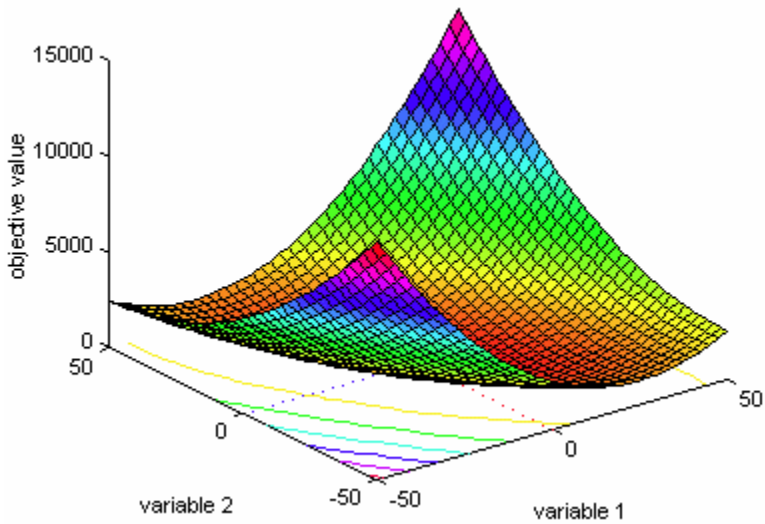


Fig. 9. Rotated hyper-ellipsoid function

5.2 Rotated Hyper-ellipsoid Function

The visualization of Rotated hyper-ellipsoid function; surf/mesh plot of the first two variables in an area from -50 to 50 it can be seen in figure 9, and is defined by this equation:

$$f(x) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2 \quad \text{minimum} = 0, \text{ for } -65.536 \leq x_i \leq 65.536 \quad (3)$$

5.3 Rosenbrock's Valley (De Jong's Function 2)

The visualization of Rosenbrock's function can be seen in figure 10. Rosenbrock's valley is a classic optimization problem, also known as Banana function. The global optimum is inside a long, narrow, parabolic shaped flat valley. The convergence to the global optimum is difficult and hence this problem has been repeatedly used in assess the performance of optimization algorithms. The equation is defined by:

$$f(x) = \sum_{i=1}^{n-1} 100 \cdot (x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \quad (4)$$

Minimum = 0, for $-2.048 \leq x_i \leq 2.048$

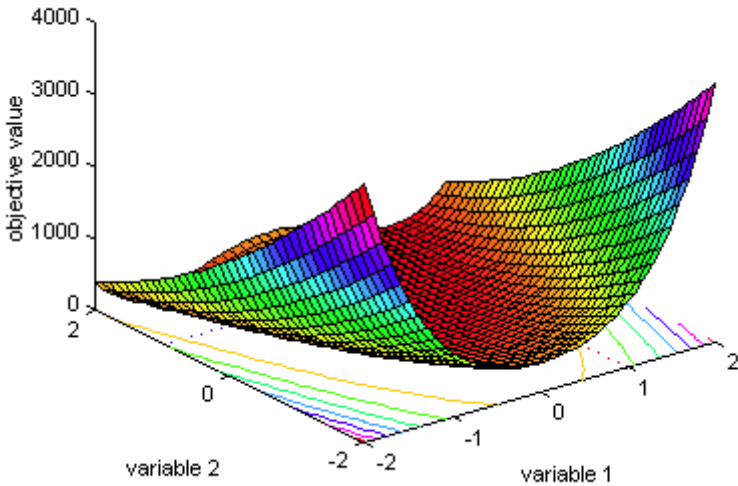


Fig. 10. Rosenbrock's valley function

5.4 Rastrigin's Function

Rastrigin's function is based on function 1 with the addition of cosine modulation to produce many local minima. Thus, the test function is highly multimodal. The equation is defined by:

$$f(x) = 10.n + \sum_{i=1}^n (x_i^2 - 10.\cos(2\pi x_i)) \quad (5)$$

minimum = 0, for $-\infty \leq x \leq \infty$

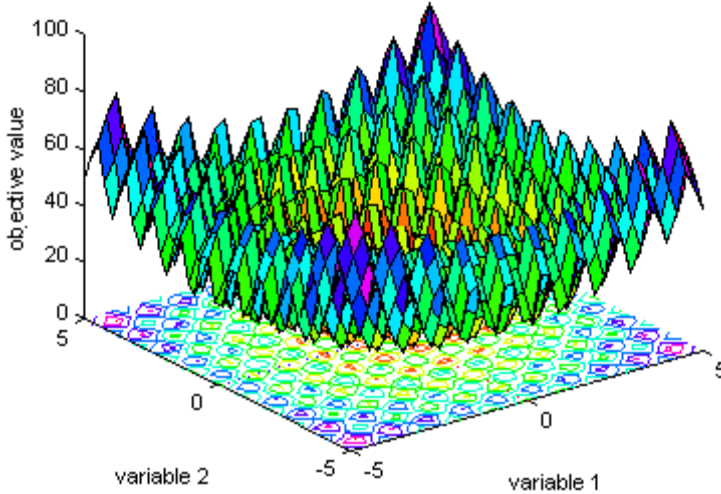


Fig. 11. Rastrigin's function

5.5 Griewanks Function

Griewank's function is similar to Rastrigin's function. In figure 12 we can see the visualization for this function. It has many widespread local minima. However, the location of the minima are regularly distributed. The equation is defined by this equation:

$$f(x) = \sum_{i=1}^n \frac{x_i^2}{4000} - \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (6)$$

minimum = 0, for $-\infty \leq x \leq \infty$

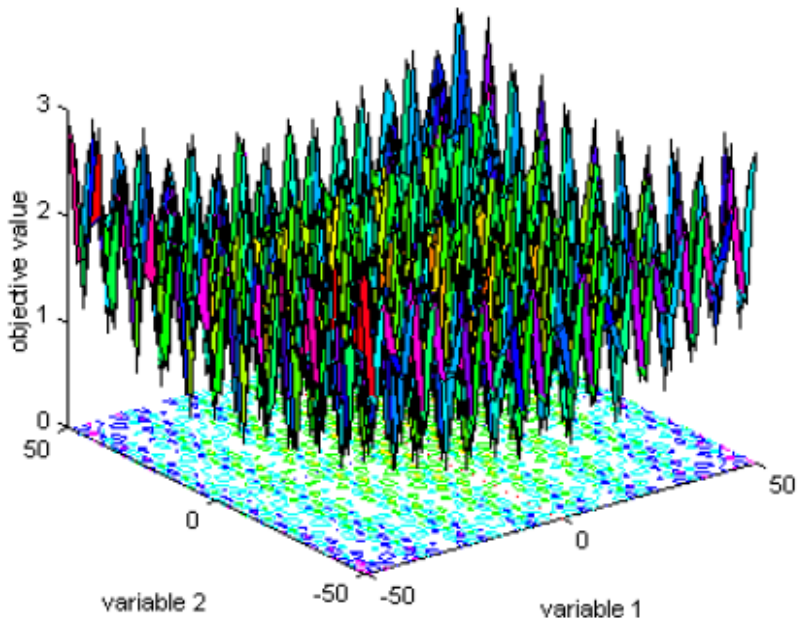


Fig. 12. Griewank's functions

6 Experimental Results

Several tests of the FPSO+FGA algorithms were made with an implementation in the Matlab programming language.

The implementation was developed using a computer with processor Intel Core 2 Quad of 64 bits that works to a frequency of clock of 2.5 GHz, 6 GB of RAM Memory and Ubuntu Linux Operating System. The results obtained after applying the GA to the mathematical functions are shown in tables 1 to 10:

The parameters used in the Tables are:

V= Number of variables used to evaluation.

BEST VALUE= The best result obtained.

AVERAGE= The average of 50 times in each core.

WORST VALUE= The worst result obtained.

The parameters used initially on all tests with the GA were: population size = 100 individuals, crossover (k_1) = 80%, mutation (k_2) = 5%, selection= roulette. On PSO the parameters were: swarm size= 100 particles, cognitive acceleration (c_1) = 1, social acceleration (c_2) = 0.5, value of velocity at the beginning = 0.95, constriction factor = 1. On execution time in FPSO+FGA the parameters were changing for k_1 , k_2 , c_1 and c_2 , this four parameters were fuzzy parameters, because were obtained dynamically when the method is running, the fuzzy systems can be obtaining this four parameters the best way for achieve good results.

6.1 Experimental Results with FPSO+FGA

The results obtained after applying the proposed method FPSO+FGA to the mathematical functions are shown in tables 1 to 10. One simulation results is shown in figure 13 in parallel mode. The use of CPU is shown in figure 14, we can observe when the method is running the four CPUs are working the maximum.

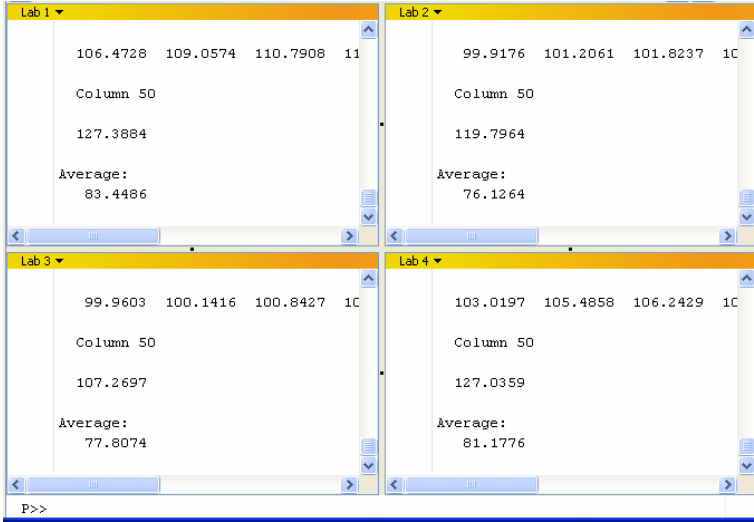


Fig. 13. Simulations results for one test function in parallel mode

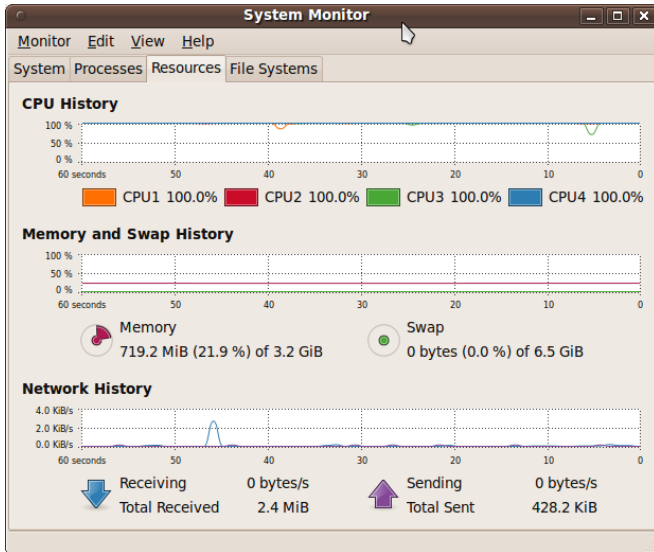


Fig. 14. CPU usage

Table 1 shows the experimental results for the De Jong's function. The table shows the number of variables for the different evaluations; it can be seen the best and worst value obtained, the average of 50 times after executing the method by each core. The time of execution by each core is shown in table 2. The minimum value for this function is in 0, however, we can see in the table 3, the best values obtained with the proposed method; we can observe when the number of variables is increased, is more complex by the method to obtain good results when the parameters are statics, therefore, the adjust of parameters is necessary in this cases and the proposed method achieved better results because the parameters are changed in execution time.

Table 1. Experimental results with FPSO+FGA for the De Jong's function

V	BEST VALUE	AVERAGE				WORST VALUE
		Core1	Core2	Core3	Core4	
16	0.00001 (Core1)	0.0002	0.1874	0.0001	0.0034	0.9205 (Core2)
32	0.0299 (Core3)	0.0148	0.0808	0.0458	0.0111	1.5926 (Core2)
64	0.0123 (Core2)	0.2529	0.3346	0.9649	0.3856	1.79 (Core3)
128	0.1681 (Core3)	0.6030	0.6703	0.2060	0.2954	2.089 (Core2)

Table 2. Time in minutes (De Jong's function)

V	TIME IN MINUTES			
	Core1	Core2	Core3	Core4
16	0.8	0.8	0.7	0.8
32	1.11	1.11	1.10	1.11
64	2.01	2.01	2.01	2.011
128	3.05	3.06	3.07	3.07

Table 3. Experimental results with FPSO+FGA for the hyper-ellipsoid function

V	BEST VALUE	AVERAGE				WORST VALUE
		Core1	Core2	Core3	Core4	
16	0.0271 (Core1)	0.3774	0.7951	0.2905	0.2784	1.4401 (Core1)
32	0.0690 (Core2)	0.9970	1.3363	1.3798	1.0722	8.4104 (Core3)
64	2.1667 (Core1)	4.0698	5.8161	4.0255	4.9087	42.872 (Core4)
128	3.0999 (Core2)	7.7459	6.3345	4.9908	8.9876	78.09 (Core2)

Table 4. Time in minutes (hyper-ellipsoid function)

V	TIME IN MINUTES			
	Core1	Core2	Core3	Core4
16	0.36	0.38	0.38	0.38
32	1.16	1.20	1.22	1.16
64	1.41	1.39	1.40	1.42
128	3.01	3.03	3.04	3.04

In table 5 it can be seen the simulations results for the Rosenbrock's valley function. Table 6 show the time of execution for this test function in parallel mode

Table 5. Experimental results with FPSO+FGA for the Rosenbrock's function

V	BEST VALUE	AVERAGE				WORST VALUE
		Core1	Core2	Core3	Core4	
16	0.0110 (Core3)	0.0456	0.0234	0.0768	0.0925	2.3001 (Core2)
32	0.3422 (Core1)	1.0023	1.5631	1.6505	1.0994	9.8790 (Core4)
64	2.534 (Core2)	3.0568	3.4456	4.5673	5.9807	7.7765 (Core3)
128	2.9909 (Core2)	8.3456	7.8956	6.0676	6.7878	9.0456 (Core2)

Table 6. Time in minutes(Rosenbrock's function)

V	TIME IN MINUTES			
	Core1	Core2	Core3	Core4
16	1.02	1.05	1.04	1.07
32	1.20	1.20	1.23	1.25
64	1.55	1.55	1.57	1.59
128	3.50	3.49	3.47	3.50

In table 7, we can appreciate the simulation results in parallel mode for the Rastrigin's function. In Table 8, we show the time of execution of the method for this test function.

Table 7. Experimental results with FPSO+FGA for the Rastrigin's function

V	BEST VALUE	AVERAGE				WORST VALUE
		Core1	Core2	Core3	Core4	
16	0.0110 (Core1)	0.0065	0.0098	0.0076	0.0045	2.3001 (Core3)
32	0.0012 (Core1)	1.0023	0.9988	0.9760	0.7890	3.3345 (Core4)
64	0.9766 (Core1)	2.6777	2.3455	2.3443	2.3465	5.6666 (Core4)
128	1.01 (Core1)	3.4560	3.4433	3.6678	4.0001	10.098 (Core3)

Table 8. Time in minutes (Rastrigin's function)

V	TIME IN MINUTES			
	Core1	Core2	Core3	Core4
16	1.01	1.01	1.02	1.05
32	1.30	1.31	1.23	1.26
64	1.58	1.59	2.00	2.01
128	3.30	3.31	3.32	3.30

In table 9, we can appreciate the simulation results in parallel mode for the Griewank's function. In Table 10, we show the time of execution of the method for this test function.

Table 9. Experimental results with FPSO+FGA for the Griewank's function

V	BEST VALUE	AVERAGE				WORST VALUE
		Core1	Core2	Core3	Core4	
16	0.0345 (Core1)	0.3333	0.3345	0.7768	0.9980	1.0989 (Core3)
32	0.1045 (Core1)	1.9800	0.9761	0.8801	0.8809	4.5678 (Core4)
64	0.9981 (Core1)	2.0983	2.0987	2.0967	2.2345	6.4561 (Core4)
128	1.5567 (Core1)	4.3245	5.7891	4.8907	5.6789	12.980 (Core3)

Table 10. Time in minutes (Griewank's function)

V	TIME IN MINUTES			
	Core1	Core2	Core3	Core4
16	1.01	1.01	1.02	1.05
32	1.30	1.31	1.23	1.26
64	1.58	1.59	2.00	2.01
128	3.30	3.31	3.32	3.30

7 Conclusions

The analysis of the experimental results of the evolutionary method considered in this paper, parallel FPSO+FGA lead us to the conclusion that for the optimization of this benchmark mathematical function with this method is a good alternative because it is easier and very fast to optimize achieve good results than to try it with PSO or GA separately [5]. This is, because the combination PSO and GA with fuzzy rules gives a parallel hybrid method FPSO+FGA. Also, we can observe the method is very fast for the execution, with architecture of four cores, we can generate four results in the same time.

Acknowledgment

We would like to express our gratitude to the CONACYT, Institute of Technology for the facilities and resources granted for the development of this research.

References

- [1] Man, K.F., Tang, K.S., Kwong, S.: *Genetic Algorithms: Concepts and Designs*. Springer, Heidelberg (1999)
- [2] Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, Nagoya, Japan, pp. 39–43 (1995); Lu, J.-G.: Title of paper with only the first word capitalized. *J. Name Stand. Abbrev.* (in press)
- [3] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: *Proceedings of IEEE International Conference on Neural Networks*, Piscataway, NJ, pp. 1942–1948 (1995)
- [4] Holland, J.H.: *Adaptation in natural and artificial system*. The University of Michigan Press, Ann Arbor (1975)
- [5] Valdez, F., Melin, P.: Parallel Evolutionary Computing using a cluster for Mathematical Function Optimization, *Nafips*. San Diego CA, USA, 598–602 (June 2007)
- [6] Castillo, O., Melin, P.: Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic, and fractal theory. *IEEE Transactions on Neural Networks* 13(6), 1395–1408 (2002)
- [7] Fogel, D.B.: An introduction to simulated evolutionary optimization. *IEEE transactions on neural networks* 5(1), 3–14 (1994)
- [8] Goldberg, D.: *Genetic Algorithms*. Addison Wesley, Reading (1988)
- [9] Emmeche, C.: *Garden in the Machine. The Emerging Science of Artificial Life*, p. 114. Princeton University Press, Princeton (1994)
- [10] Angeline, P.J.: Using Selection to Improve Particle Swarm Optimization. In: *Proceedings 1998 IEEE World Congress on Computational Intelligence*, Anchorage, Alaska, pp. 84–89. IEEE, Los Alamitos (1998)
- [11] Angeline, P.J.: Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences. In: Porto, V.W., Waagen, D. (eds.) *EP 1998*. LNCS, vol. 1447, pp. 601–610. Springer, Heidelberg (1998)
- [12] Back, T., Fogel, D.B., Michalewicz, Z. (eds.): *Handbook of Evolutionary Computation*. Oxford University Press, Oxford (1997)
- [13] Montiel, O., Castillo, O., Melin, P., Rodriguez, A., Sepulveda, R.: Human evolutionary model: A new approach to optimization. *Inf. Sci.* 177(10), 2075–2098 (2007)
- [14] Castillo, O., Valdez, F., Melin, P.: Hierarchical Genetic Algorithms for topology optimization in fuzzy control systems. *International Journal of General Systems* 36(5), 575–591 (2007)

Optimization of Type-2 Fuzzy Logic Controllers Using PSO Applied to Linear Plants

Ricardo Martinez¹, Oscar Castillo², Luis T. Aguilar³, and Antonio Rodriguez¹

¹ Universidad Autónoma de Baja California Tijuana, México

mc.ricardo.martinez@hotmail

² Tijuana Institute of Technology, Tijuana México

ocastillo@tectijuana.mx

³ Instituto Politécnico Nacional, Centro de Investigación y Desarrollo de Tecnología Digital, Ave. Del Parque 1310, Mesa de Otay, Tijuana B.C. 22510

luis.aguilar@ieee.org

Abstract. We use the Particle Swarm Optimization (PSO) method to find the parameters of the membership functions of a type-2 fuzzy logic controller (Type-2 FLC) in order to minimize the state error for linear systems. PSO is used to find the optimal Type-2 FLC to achieve regulation of the output and stability of the closed-loop system. For this purpose, we change the values of the cognitive, social and inertia variables in the PSO. Simulation results, with the optimal FLC implemented in Simulink, show the feasibility of the proposed approach.

Keywords: PSO, Fuzzy Logic Optimizations.

1 Introduction

Optimization algorithms are search methods, where the goal is to find a solution to an optimization problem, such that a given quantity is optimized, possibly subject to a set of constraints [6]. Some optimization methods are based on populations of solutions. Unlike the classic methods of improvement for trajectory tracking, in this case each iteration of the algorithm has a set of solutions. These methods are based on generating, selecting, combining and replacing a set of solutions. Since they maintain and they manipulate a set, instead of a unique solution throughout the entire search process, they used more computer time than other metaheuristic methods. This fact can be aggravated because the “convergence” of the population requires a great number of iterations. For this reason a concerted effort has been dedicated to obtaining methods that are more aggressive and manage to obtain solutions of quality in a nearer horizon. This paper is concerned with bio-inspired optimization methods like particle swarm optimization (PSO) to design optimized

fuzzy logic controllers (FLC) for linear problems. This method is used to find the parameters of the membership functions obtaining the optimal FLC for plant control.

This paper is organized as follows: Section 2 presents the theoretical basis and problem statement. Section 3 introduces the controller design where a PSO is used to select the parameters. Robustness properties of the closed-loop system are achieved with a type-2 fuzzy logic control system using a Takagi-Sugeno model where the error and the change of error, are considered the linguistic variables. Section 4 provides a simulation study of the plant using the controller described in Section 3. Finally, Section 5 presents the conclusion.

2 Theoretical Basis and Problem Statement

Particle Swarm Optimization (PSO). PSO is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling [5]. PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA) [7].

The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike the GA, the PSO has no evolution operators such as crossover and mutation. In the PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles [2]. Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far (The fitness value is also stored). This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest*. When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest* [10].

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its *pbest* and *lbest* locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *lbest* locations [9]. In the past several years, PSO has been successfully applied in many research and application areas. It is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods [1], [9]. Another reason that PSO is attractive is that there are few parameters to adjust. One version, with slight variations, works well in a wide variety of applications. Particle swarm optimization has been used for approaches that can be used across a wide range of applications, as well as for specific applications focused on a specific requirement [9].

The basic algorithm of the PSO has the next nomenclature:

\mathbf{x}_k^i -Particle position

\mathbf{v}_k^i -Particle velocity

\mathbf{p}_k^i -Best “remembered” individual particle position

\mathbf{p}_k^g -Best “remembered” swarm position

$\mathbf{c}_1, \mathbf{c}_2$ -Cognitive and Social parameters

r_1, r_2 -Random numbers between 0 and 1

The equation for calculate the velocity is:

$$\mathbf{v}_{k+1}^i = \mathbf{v}_k^i + \mathbf{c}_1 r_1 (\mathbf{p}_k^i - \mathbf{x}_k^i) + \mathbf{c}_2 r_2 (\mathbf{p}_k^g - \mathbf{x}_k^i) \quad (1)$$

and the position of individuals particles is updated as follows:

$$\mathbf{x}_{k+1}^i = \mathbf{x}_k^i + \mathbf{v}_{k+1}^i \quad (2)$$

The basic PSO algorithm flow diagram as follows:

1) Initialize

a) Set constants $k_{\max}, \mathbf{c}_1, \mathbf{c}_2$

b) Randomly initialize particle position $\mathbf{x}_0^i \in D$ in R^n for $i = 1, \dots, p$

c) Randomly initialize particle velocities $0 \leq v_0^i \leq v_0^{\max}$ for $i = 1, \dots, p$

d) Set $k = 1$

2) Optimize

a) Evaluate function value f_k^i using design space coordinates \mathbf{x}_k^i

b) If $f_k^i \leq f_{best}^i$ then $f_{best}^i = f_k^i, \mathbf{p}_k^i = \mathbf{x}_k^i$.

c) If $f_k^i \leq f_{best}^g$ then $f_{best}^g = f_k^i, \mathbf{p}_k^g = \mathbf{x}_k^i$.

d) If stopping condition is satisfied then goto 3.

e) Update all particle velocities \mathbf{v}_k^i for $i = 1, \dots, p$

f) Update al particle positions \mathbf{x}_k^i for $i = 1, \dots, p$

g) Increment k .

h) Goto 2(a).

3) Terminate

Problem Statement. To test the optimized FLCs obtained by the bio-inspired methods; we used different linear systems. We first consider two benchmark problems called Plant 1 and Plant 2 with different levels of complexity. Fig 1 shows the Simulink bloc diagram used to simulate the Plants.

Plant 1 is given by the following second order transfer function:

$$g(s) = \frac{w_n^2}{s^2 + 2\mathcal{E} w_n s + w_n^2} \quad \mathcal{E} = 0.5, \quad w_n = 2 \quad (3)$$

where w_n is the natural frequency and \mathcal{E} is the coefficient damping.

Plant 2 is given by the following transfer function:

$$g(s) = \frac{1}{s^2 + 4} \quad (4)$$

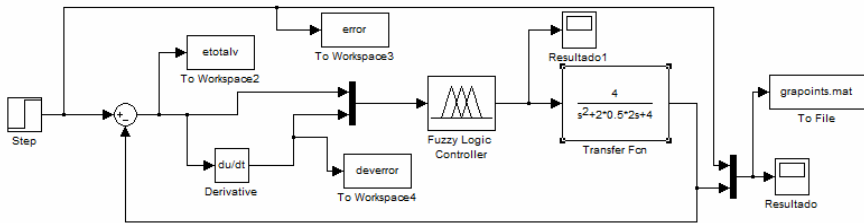


Fig. 1. Simulink bloc diagram of the Plant 1 and Plant 2

3 Fuzzy Logic Control Design

In this section we design a fuzzy logic controller (FLC) where the optimal controller was found with the first evolutionary method, which in this case is the genetic algorithm.

For the FLC a Takagi-Sugeno type of fuzzy system is used with two inputs a) error, and b) change of error, with three membership functions each input, “Negative, Zero and Positive” (Gaussian and triangular), one output which are constant values, and nine fuzzy rules (IF – THEN) [3],[4],[8],[11],[12],[13],[14]. Fig 2 shows the FLC membership functions for the plant control. Once we obtained the FLC design, we used an optimization method to find the optimal Controller.

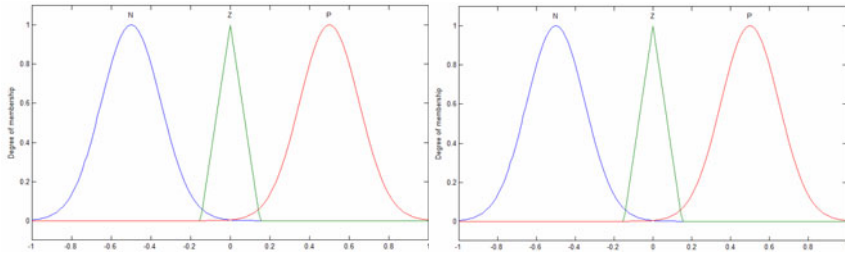


Fig. 2. a) input 1 “error”, b) input 2 “error change”

Table 1 show the parameters of the membership functions, the minimal and the maximum values in the search range for the PSO method to find the best type-1 fuzzy controller system and table 2 shows the parameters of the membership function of the type-2 FLC.

Table 1. Parameters of the membership functions for type-1 FLC

Plant 1				Plant 2			
MF Type	Point	Minimum Value	Maximum Value	MF Type	Point	Minimum Value	Maximum Value
Gaussian	a	0.3	0.6	Gaussian	a	1.8	2.8
	b	-1.2	-0.8		b	-6	-4
Triangular	a	-0.8	-0.3	Triangular	a	-3	-0.5
	b	0	0		b	0	0
	c	0.3	0.8		c	0.5	3
Gaussian	a	0.3	0.6	Gaussian	a	1.8	2.8
	b	0.8	1.2		b	4	6

Table 2. Parameters of the membership functions for type-2 FLC

Plant 1				Plant 2			
MF Type	Point	Minimum Value	Maximum Value	MF Type	Point	Minimum Value	Maximum Value
Gaussian	a	0.13	0.14	Gaussian	a	1.8	2.8
	b	-1.2	-1		b	-7	-3
	a'	0.22	0.43		a'	1.5	1.6
	b'	-1.2	-1		b'	-7	-3
Triangular	a	-1	-0.75	Triangular	a	-3	-2.5
	b	0	0		b	0	0
	c	0.15	0.25		c	0.5	2
	a'	-0.75	-0.25		a'	-2	-1
	b'	0	0		b'	0	0
	c'	0.5	0.75		c'	1	3
Gaussian	a	0.13	0.14	Gaussian	a	1.8	2.8
	b	1	1.2		b	3	7
	a'	0.22	0.43		a'	1.5	1.6
	b'	1	1.2		b'	3	7

4 Simulation Results

In this section, we evaluate, through computer simulations performed in MATLAB® and SIMULINK®, the designed FLC for the two plants using the PSO optimization method. We use random values in the cognitive (C1), social (C2) and inertia parameters of the PSO.

4.1 PSO with Type-1 FLC for Plant 1

Table 3 presents the main results of the FLC obtained with the PSO showing in the first row the best result.

Table 3. Results of the Type-1 FLC for Plant 1 obtained by PSO

No.	Swarm (Population)	Max Iterations	C1	C2	Inertia	Time exec	Average error
1	200	70	0.4542	0.5052	0.9038	0:26:26	0.081179
2	200	70	0.9667	0.3883	0.8501	0:44:46	0.086795
3	200	70	0.183	0.8722	0.9289	0:09:23	0.087324
4	200	70	0.6798	0.7371	0.6119	0:54:22	0.089289
5	200	70	0.0016	0.8994	0.6983	0:45:59	0.090177
6	200	70	0.484	0.704	0.7194	0:49:16	0.092952
7	200	70	0.7046	0.9151	0.9988	0:05:59	0.095280
8	200	70	0.1344	0.3959	0.2534	0:58:29	0.096134
9	200	70	0.1223	0.5346	0.0681	0:57:48	0.103938
10	200	70	0.1589	0.0366	0.6223	1:04:45	0.109765

Fig 3 shows the Particles behavior of PSO giving the best type-1 FLC for controlling the plant 1 and fig 4 shows the membership functions of the optimized controller.

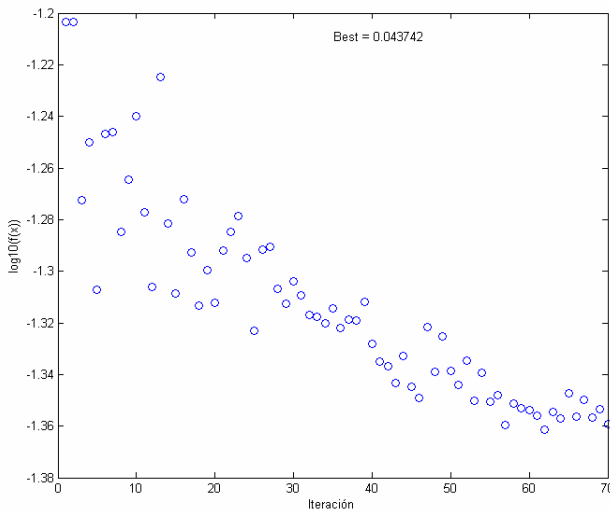


Fig. 3. Behavior of PSO particles of Type-1 FLC optimization

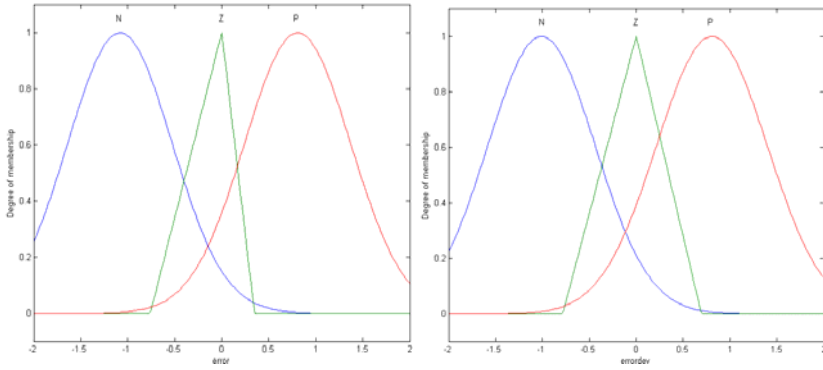


Fig. 4. Input a “error” and input b “errordev” of the optimized FLC

The simulations result of the optimized Type-1 FLC for Plant 1 is showing in the fig 5.

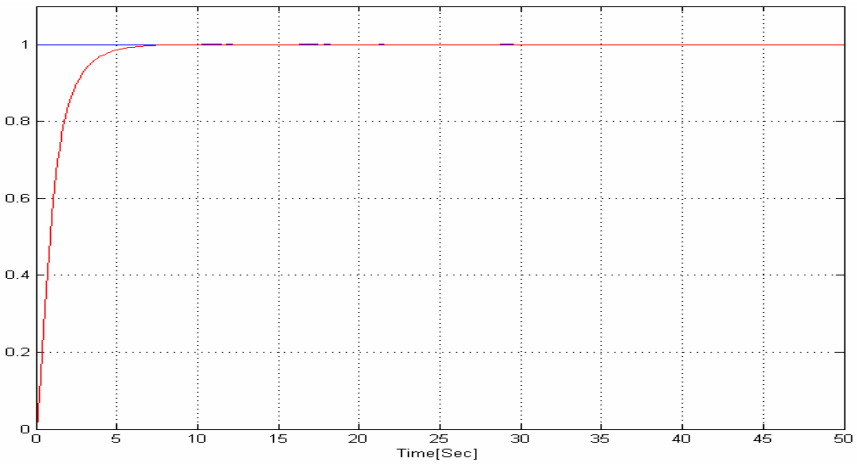


Fig. 5. Simulations result of type-1 FLC for Plant 1

4.2 PSO with Type-1 FLC for Plant 2

Table 4 presents the main results of the FLC obtained with the PSO showing in the first row the best result.

Table 4. Results of the Type-1 FLC for Plant 2 obtained by PSO

No.	Swarm (Population)	Max Iterations	C1	C2	Inertia	Time exec	Average error
1	200	70	0.0429	0.6746	0.8477	0:34:32	0.131933
2	200	70	0.7580	0.1716	0.5573	0:49:17	0.136293
3	200	70	0.7488	0.0884	0.9051	0:39:05	0.141524
4	200	70	0.8149	0.9059	0.1706	0:49:38	0.148655
5	200	70	0.0660	0.7696	0.5558	0:44:32	0.150427
6	200	70	0.8660	0.1036	0.5554	1:05:25	0.151731
7	200	70	0.5543	0.6067	0.4974	0:50:25	0.152046
8	200	70	0.7154	0.2479	0.5130	0:55:45	0.154302
9	200	70	0.0158	0.0882	0.2155	1:02:43	0.160239
10	200	70	0.4458	0.9566	0.3233	0:51:22	0.164745

Fig 6 shows the particles behavior of PSO giving the best type-1 FLC for controlling the plant 2 and fig. 7 shows the membership functions of the optimized controller.

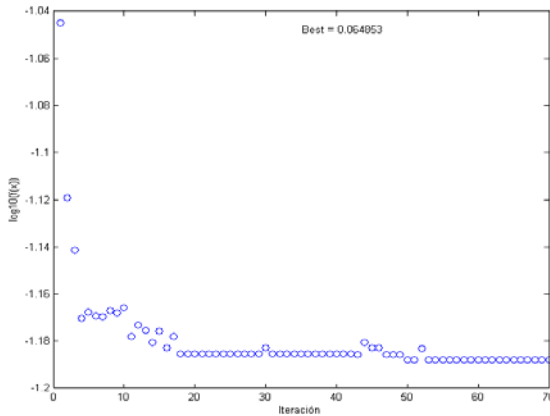


Fig. 6. Behavior of PSO particles of Type-1 FLC optimization

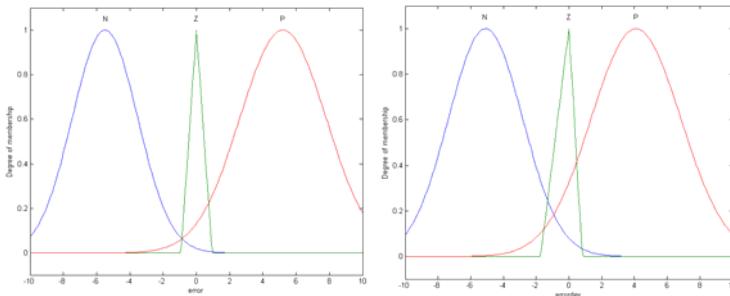


Fig. 7. Input a “error” and input b “errordev” of the optimized FLC

The simulation result of the optimized Type-1 FLC for Plant 2 is shown in fig 8.

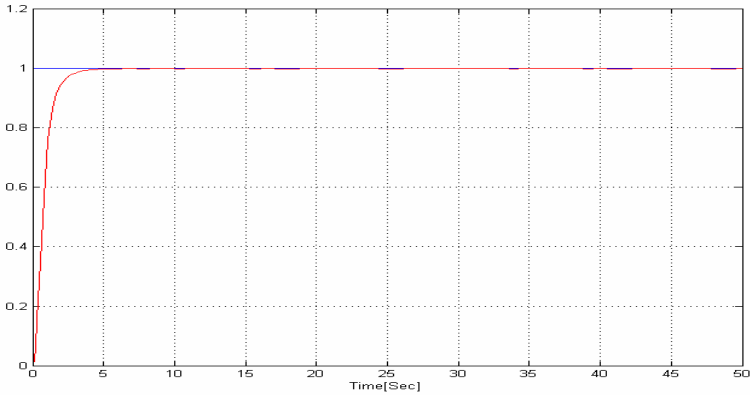


Fig. 8. Simulations result of type-1 FLC for Plant 2

4.3 PSO with Type-2 FLC for Plant 1

Table 5 presents the main results of the FLC obtained with PSO showing in the first row the best result.

Table 5. Results of the Type-2 FLC for Plant 1 obtained by PSO

No.	Swarm (Population)	Max Iterations	C1	C2	Inertia	Time exec	Average error
1	200	90	0.5149	0.3317	0.6808	7:05:20	0.08028
2	200	70	0.8149	0.9059	0.1706	12:44:19	0.08794
3	200	70	0.8129	0.8159	0.1906	10:52:09	0.08894
4	200	70	0.7646	0.9229	0.1096	9:50:45	0.12521
5	200	70	0.8168	0.9359	0.4806	11:05:43	0.12630

Fig 9 shows the particles behavior of PSO giving the best type-2 FLC for controlling the plant 1 and fig 10 shows the membership functions of the optimized controller.

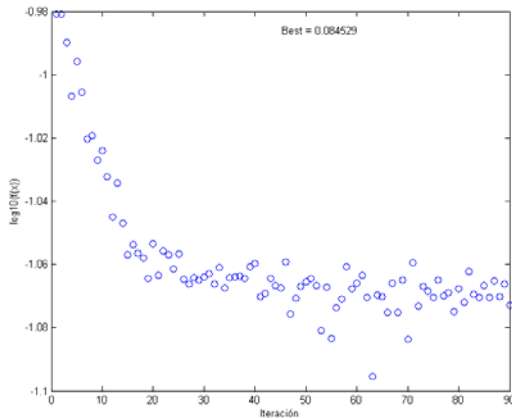


Fig. 9. Behavior of PSO particles of Type-2 FLC optimization

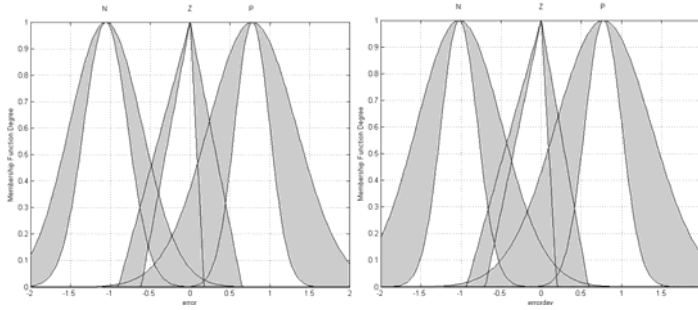


Fig. 10. Input a “error” and input b “errordev” of the optimized FLC

The simulation result of the optimized Type-2 FLC for Plant 1 is shown in fig 11.

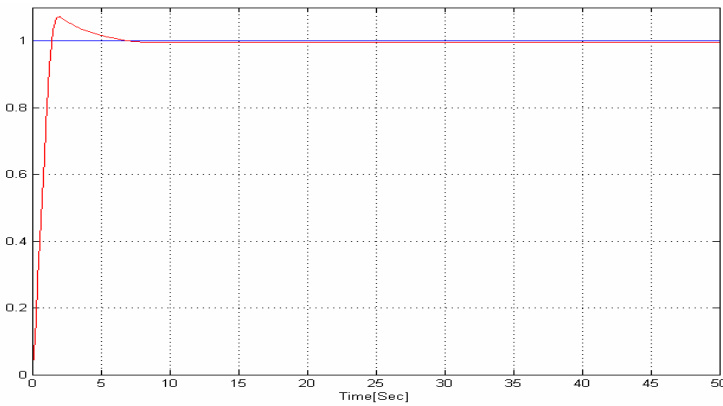


Fig. 11. Simulations result of type-2 FLC for Plant 1

4.4 PSO with Type-2 FLC for Plant 2

Table 6 presents the main results of the FLC obtained with PSO showing in the first row the best result.

Table 6. Results of the Type-2 FLC for Plant 2 obtained by PSO

No.	Swarm (Population)	Max Iterations	C1	C2	Inertia	Time exec	Average error
1	200	90	0.5149	0.3317	0.6808	7:05:20	0.08028
2	200	70	0.8149	0.9059	0.1706	12:44:19	0.08794
3	200	70	0.8129	0.8159	0.1906	10:52:09	0.08894
4	200	70	0.7646	0.9229	0.1096	9:50:45	0.12521
5	200	70	0.8168	0.9359	0.4806	11:05:43	0.12630

Fig 12 shows the Particles behavior of PSO giving the best type-2 FLC for controlling the plant 1 and fig 13 shows the membership functions of the optimized controller.

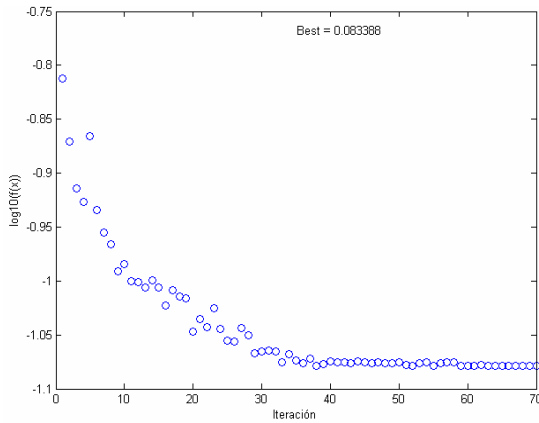


Fig. 12. Behavior of PSO particles of Type-2 FLC optimization

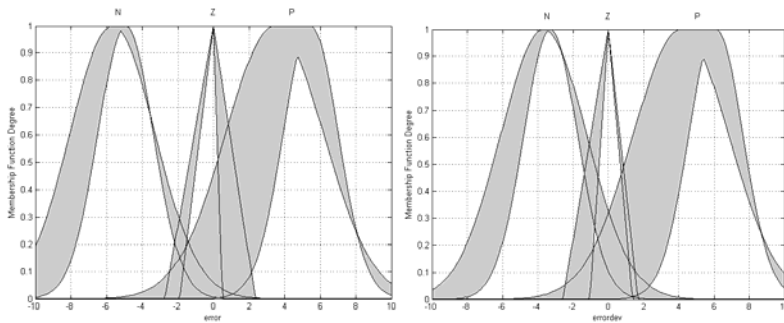


Fig. 13. Input a “error” and input b “errordev” of the optimized FLC

The simulation result of the optimized Type-2 FLC for Plant 2 is shown in fig 14.

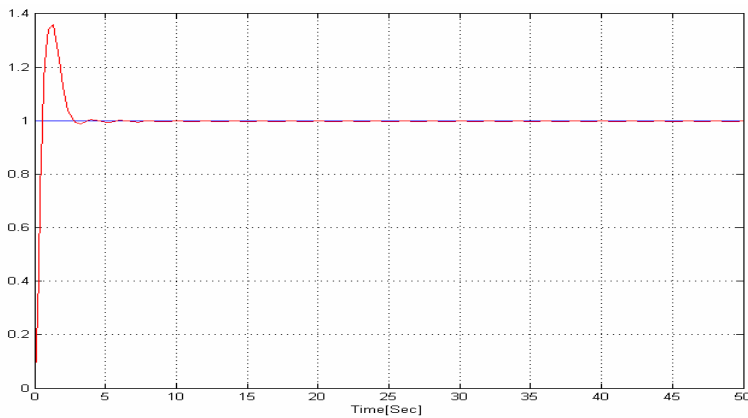


Fig. 14. Simulations result of type-2 FLC for Plant 2

5 Conclusions

We described in this paper the use of an optimization method for the simulation design of optimized FLCs. In particular we presented results for PSO in Type-1 and Type-2 FLC optimization for linear plants. The simulation results show that the closed-loop system with the type-1 FLC obtained is stabilized in less than 10 sec. On the other hand, the Type-2 FLC's obtained are better than the Type-1 FLC's because they gets stabilization in less than 5 seconds, but with more overshoot in the second Plant that the first one. The plots of the results show this difference.

We have achieved satisfactory results with PSO; the next step is to solve the problem using Type-2 FLC in a perturbed environment and considering multiple objective optimization to obtain better results. Moreover, we will extend the results to nonlinear systems, like for autonomous mobile robots.

References

1. Angeline, P.J.: Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 601–610. Springer, Heidelberg (1998)
2. Angeline, P.J.: Using Selection to Improve Particle Swarm Optimization. In: Proceedings 1998 IEEE World Congress on Computational Intelligence, Anchorage, Alaska, pp. 84–89. IEEE, Los Alamitos (1998)
3. Chi, Z., Yan, H., Pham, T.: Fuzzy Algorithms: With Applications to Image Processing and Pattern recognition. World Scientific, Singapore (1996)
4. Driankov, D., Hellendoorn, H., Reinfrank, M.: An Introduction to Fuzzy Control. Springer, Berlin (1993)
5. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan, pp. 39–43 (1995); Lu, J.-G.: Title of paper with only the first word capitalized. J. Name Stand. Abbrev. (in press)
6. Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence, England, pp. 5–129. John Wiley & Sons Ltd., Chichester (2005)
7. Fogel, D.B.: An introduction to simulated evolutionary optimization. IEEE transactions on neural networks 5(1), 3–14 (1994)
8. Fukao, T., Nakagawa, H., Adachi, N.: Adaptive Tracking Control of a NonHolonomic Mobile Robot. IEEE Trans. On Robotics and Automation 16(5), 609–615 (2000)
9. Kennedy, J., Mendes, R.: Population structure and particle swarm performance. In: Proceeding of IEEE conference on Evolutionary Computation, pp. 1671–1676 (2002)
10. Kennedy, J., Mendes, R.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2002)
11. Lee, T.H., Leung, F.H.F., Tam, P.K.S.: Position Control for Wheeled Mobile Robot Using a Fuzzy Controller. IEEE, 525–528 (1999)
12. Martinez, R., Castillo, O., Aguilar, L.T., Rodriguez, A.: Evolutionary Optimization of type-2 Fuzzy Systems Applied to Linear Plants. To appear in System, Man, and Cybernetic Conference (2009)

13. Martinez, R., Castillo, O., Aguilar, L.T.: Intelligent Control For A Perturbed Autonomous Wheeled Mobile Robot Using Type-2 Fuzzy Logic and Genetic Algorithms. *Journal of Automation, Mobile Robotics & Intelligent Systems* 2 (2008)
14. Martinez, R., Castillo, O., Aguilar, L.T.: Optimization of Interval Type-2 Fuzzy Logic Controllers for a Perturbed Autonomous Wheeled Mobile Robot using Genetic Algorithms. *Information Sciences, Informatics and Computer Science Intelligent Systems Applications and International Journal* 179(13), 2158–2174 (2009)

Optimization of Membership Functions for an Incremental Fuzzy PD Control Based on Genetic Algorithms

Yazmín Maldonado, Oscar Castillo, and Patricia Melin

Tijuana Institute of Technology, México
yaz.maldonado@gmail.com, ocastillo@tectijuana.mx,
pmelin@tectijuana.mx

Abstract. This paper proposes the optimization of an incremental fuzzy PD controller based on a genetic algorithm, which optimizes the parameters of membership functions of the inputs and outputs. The multiobjective GA is considered because it evaluates three characteristics of the controller (steady state error, overshoot and undershoot). The application of this controller is to regulate the engine speed of a direct current (DC) motor. A methodology to test and validate this controller through Matlab-Simulink is presented.

1 Introduction

Fuzzy logic controllers are used successfully in many application areas, these include control, classification, etc. [1,2,3,5, 7,10,11]. These systems based on rules incorporate linguistic variables, linguistic terms and fuzzy rules. The acquisition of these is not an easy task for the expert and are of vital importance in the operation of the controller.

Adjusting these linguistic terms and rules are usually done by trial and error, which implies a difficult task, there are methods to optimize those elements that over time have taken importance, such as genetic algorithms [6].

This work presents the optimization of membership functions for incremental Fuzzy PD control based on genetic algorithms, the controller regulates the engine speed of DC Motor, and this application is tested in Matlab-Simulink.

This paper is organized as follows, in section 2 we present an introductory explanation of Fuzzy Inference Systems and Genetic algorithms, section 3 describes the design and optimization of the fuzzy system, the experimental results of the optimized FLC are shown in Section 4. Finally, Section 5 presents the conclusions.

2 Preliminaries

Fuzzy systems are every time used more and more, because they tolerate imprecise information and can be used to model nonlinear functions of arbitrary complexity. A fuzzy system (FIS) consists of three stages: Fuzzification, Inference and Defuzzification [13]. We describe below these stages.

Fuzzification: is the interpretation of input values (numerics) by the fuzzy system, and the obtained output are fuzzy values.

Definition 1. Let $x \in X$ be a linguistic variable and $T_i(x)$ a fuzzy set associated with a linguistic value T_i . The translation of a numeric value x corresponds to a linguistic value associated with a degree of membership, $x \rightarrow \mu_{T_i}(x)$, and this is known as Fuzzification. The membership degree $\mu_{T_i}(x)$ represents a value of membership to a fuzzy set [14].

Inference: is basically like the brain of the system, here the rules of the form if-then that describe this behavior are used [2]. For example:

$$\text{If } x_1 \text{ is } A_1 \text{ and } \dots \text{ and } x_n \text{ is } A_n \text{ Then } y \text{ is } B \quad (1)$$

where $x_1 \dots x_n$ are the inputs, $A_1 \dots A_n, B$ are linguistic terms and y is the output.

Defuzzification: consists in obtaining a numeric value to the output. This stage basically selects a point that is the most representative of the action to perform [2]. There are several methods to calculate the Defuzzification, such as the Center of Height (COA), Center of Gravity (COG), etc. The COG is shown in Equation 2.

$$y = \frac{\sum_{i=1}^N h_i \mu(i)}{\sum_{i=1}^N \mu(i)} \quad (2)$$

where h_i is the maximum height of the consequent from rule i to rule N [2].

In Figure 1, the fuzzy system information processing is shown.

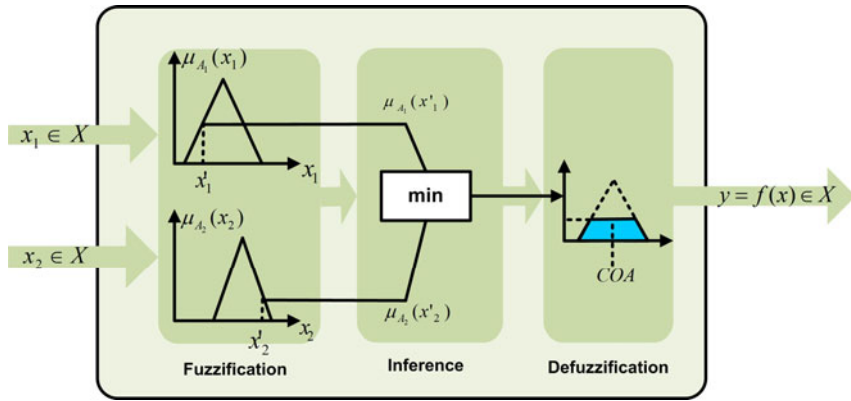


Fig. 1. Fuzzy System

A genetic algorithm (GA) is a programming technique that mimics biological evolution to solve problems, GAs have proven to be a good strategy because of its optimal results [3,8].

The GA has applications in a wide variety of fields to develop solutions to complex problems, including optimization of fuzzy systems, offering them learning and adaptation capabilities, these are commonly called genetic fuzzy systems or fuzzy system hybrids.

3 Design and Optimization of the Fuzzy Logic Controller

The Fuzzy Logic Controller (FLC) is designed for speed control of a DC motor and has two inputs and one output. The inputs are error ($e(t)$) and change of error ($e'(t)$), and the output is the control signal ($y(t)$).

The inputs are calculated as follows

$$e(t) = r(t) - y(t) \tag{3}$$

$$E'(t) = e(t) - e(t-1) \tag{4}$$

where t is the sampling time.

The reference signal $r(t)$, is given by:

$$r(t) = \begin{cases} 15 & t > 0 \\ 0 & t \leq 0 \end{cases} \tag{5}$$

Each input and output of the FIS has three linguistic terms. For the linguistic variable error and change of error, the terms are {NB, Z, PB} in this case NB is Negative Big, Z is Zero and PB is Positive Big. For the linguistic variable control

signal are {BD, H, BI} in this case BD is Big Decrement, H is Hold and BI is Big Increment.

Figure 2 shows the FIS input $e(t)$.

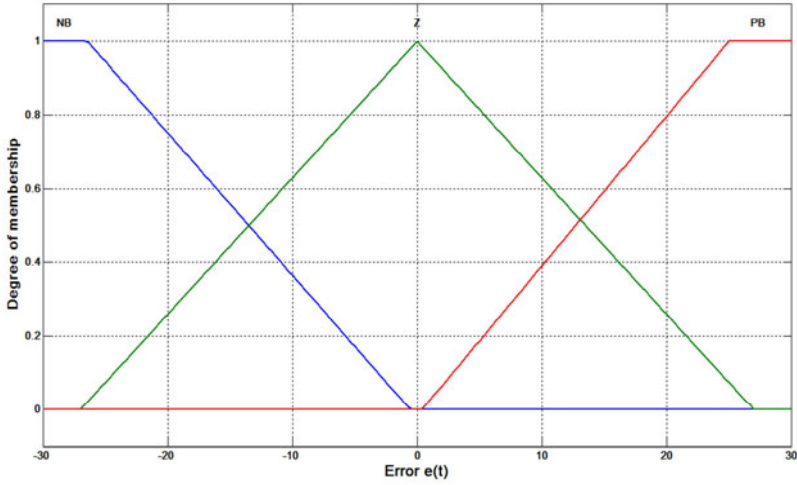


Fig. 2. FIS input $e(t)$

Figure 3 shows the FIS input change of error $e'(t)$.

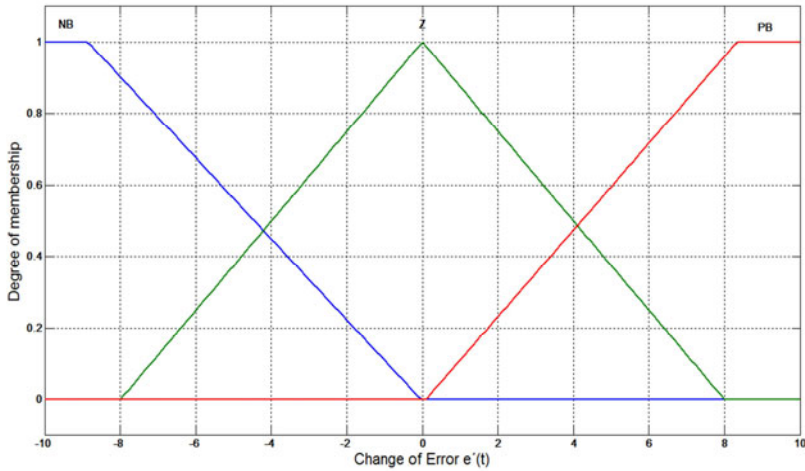


Fig. 3. FIS input $e'(t)$

Table 1 shows the rule matrix for the FIS.

Table 1. Rule matrix

e	ce	NB	Z	PB
NB		BD	BD	BD
Z		H	H	H
PB		BI	BI	BI

Figure 4 shows the FIS output ($y(t)$).

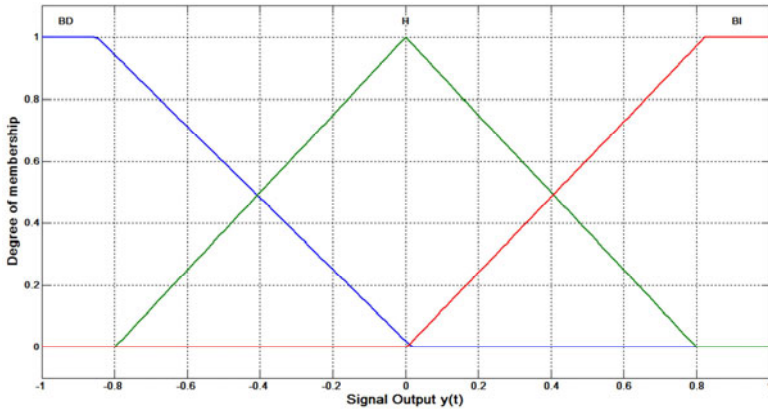


Fig. 4. FIS output $y(t)$

Figure 5 shows the control surface.

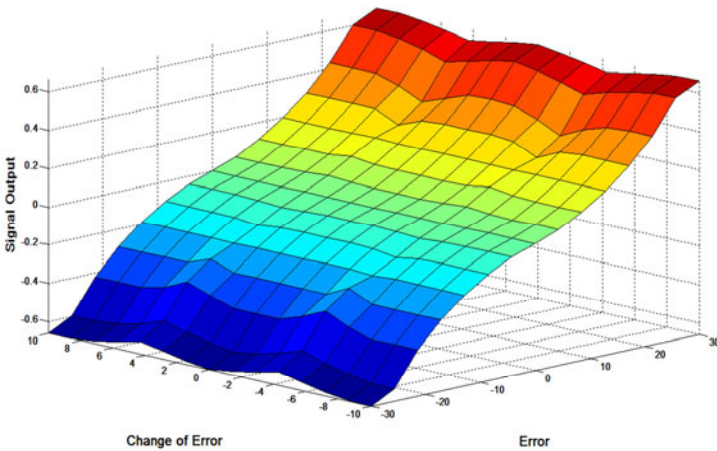


Fig. 5. Control Surface

The FIS is tested in an Incremental fuzzy PD controller [4]. Figure 6 shows the FLC architecture used.

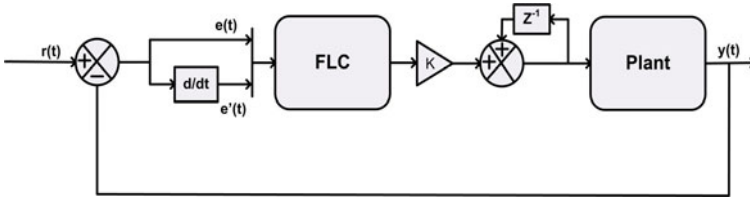


Fig. 6. Fuzzy PD Incremental

Figure 7 shows the output of the closed loop system for the previous FLC with a reference $r(t)$ of 15 revolutions per minute (rpm).

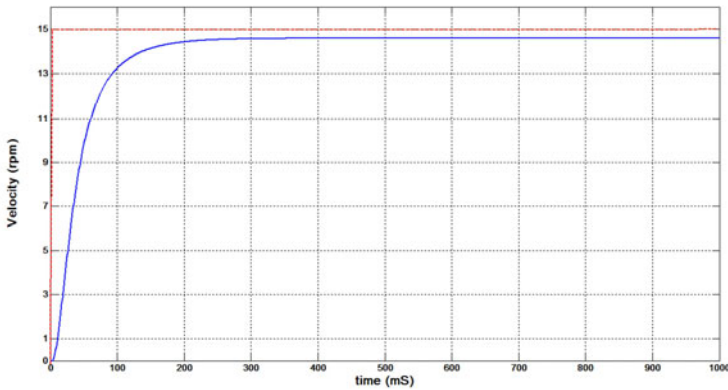


Fig. 7. Velocity for 15 rpm

As shown in figure 7, the output signal is not the best possible for this problem, the next step is to optimize the FLC to get a better response for engine speed regulation.

3.1 Optimization Method

For the optimization of the FLC using GAs, you must define the chromosome that represents the information of the individual, which in this case is related to the universe of discourse and the linguistic terms. Figure 8 shows the chromosome of the GA.

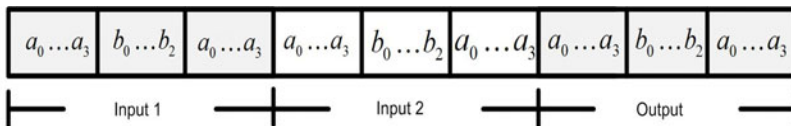


Fig. 8. GA chromosome

Figure 9 shows the triangular and trapezoidal membership functions (MF) that are used.

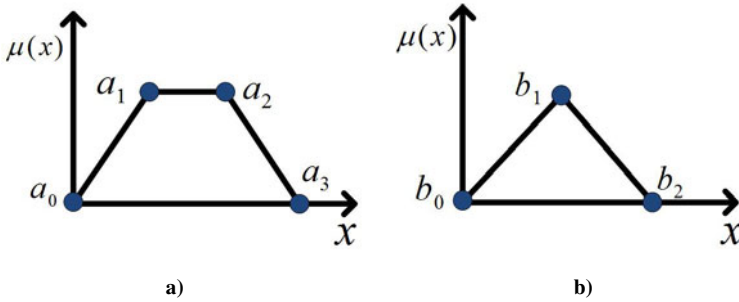


Fig. 9. Parameters of the Membership Functions. a) MF trapezoidal, b) MF triangular

The GA is of multiobjective type [9], which means that to determine the best individual three evaluations are performed:

- a) Minimum overshoot.
- b) Minimum undershoot.
- c) Minimum output steady state error (sse).

The FLC linguistic terms were optimized with the GA, but the fuzzy rules are not changed. The process of the GA is shown in Figure 10.

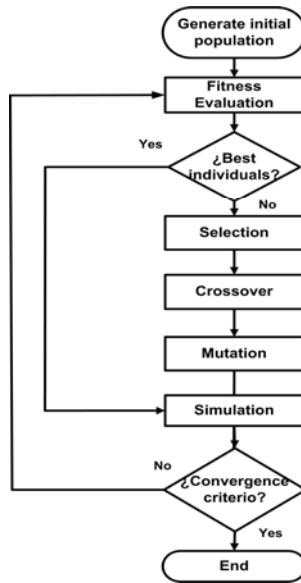


Fig. 10. Optimization GA

4 Experimental Results

To evaluate the ability of the GA, the FLC was simulated for speed control using a mathematical model of the plant in Matlab-Simulink [12], as shown in Figure 6. A series of experiments was performed that are listed in Table 2.

Table 2. GA Parameters for different experiments

No.	Generations	Crossover	Overshoot-Undershoot	SSE	Time
1	50	0.8	13.7324	0.0147	97.45621
2	50	0.8	10.0187	1.4695e-004	50.65335
3	50	0.8	14.6503	0.0103	102.4405
4	50	0.8	12.3445	0.6916	262.120341
5	30	0.8	12.9078	0.0057	65.638181
6	70	0.7	10.5311	1.0487e-004	79.378554
7	100	0.7	9.9176	0.0072	59.475111
8	100	0.7	11.1285	0.0300	176.874380
9	100	0.9	10.1559	0.0013	143.649806
10	50	0.8	15.8521	1.2145e-007	232.21931
11	200	0.8	14.7944	0.0600	269.598303
12	300	0.9	9.2882	9.1190e-004	129.769955
13	350	0.8	13.8138	0.0051	411.122418
14	70	0.9	14.2581	1.5946e-004	180.825022

In experiment No. 10 the best FLC was found because this has the lower error value. Below are the FIS characteristics for experiments 1, 2 and 10.

Figure 11 shows how the GA modified the parameters of the membership functions for the input $e(t)$.

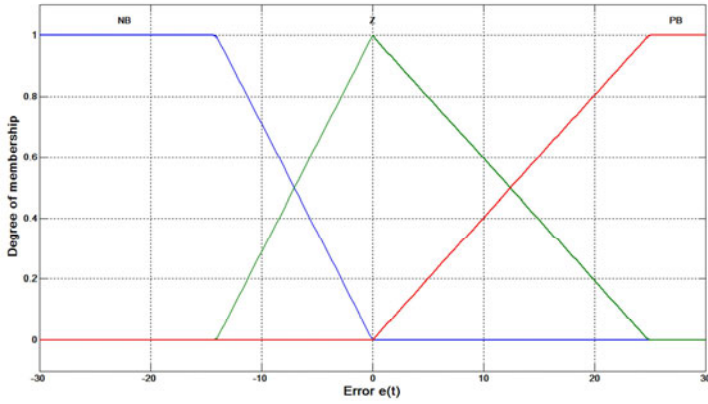


Fig. 11. FIS modified by the GA $e(t)$

Figure 12 shows the input $e'(t)$ modified by the GA.

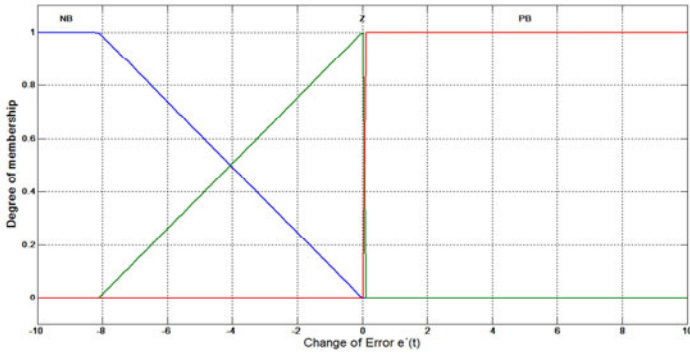


Fig. 12. FIS modified by the GA $e'(t)$

Figure 13 shows the output $y(t)$ of the FIS.

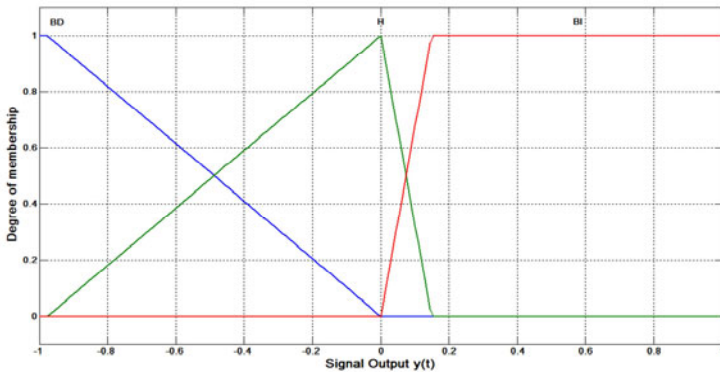


Fig. 13. Output FIS modified by the GA

Figure 14 shows the control surface modified by the GA.

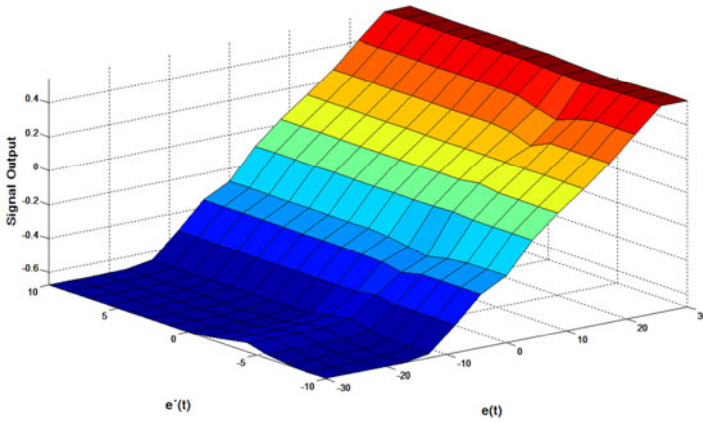


Fig. 14. Control Surface

Figure 15 shows the output signal of the FLC PD Incremental for experiment 1.

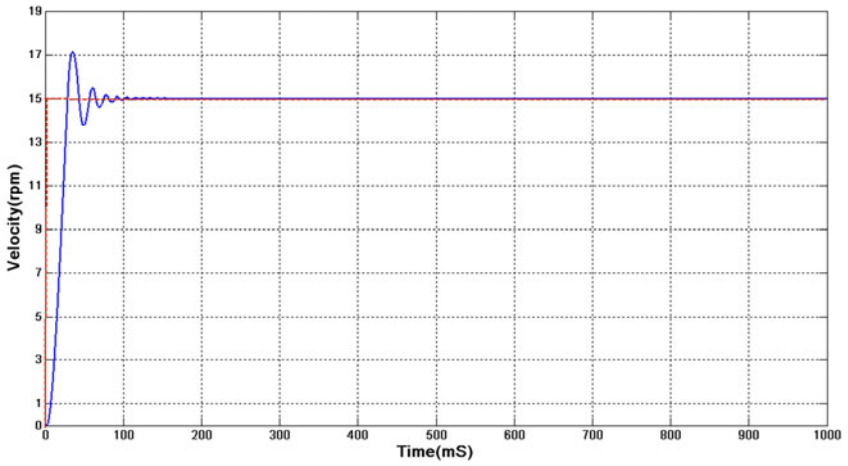


Fig. 15. Velocity

Figure 16 shows the evolution of the steady state error (sse).

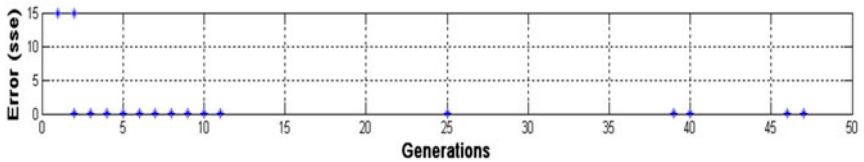


Fig. 16. Experiment 1 steady state error

Figure 17 shows the evolution of the GA fitness.

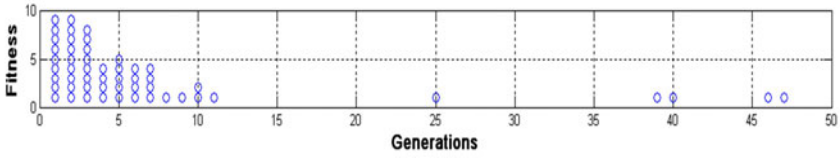


Fig. 17. Experiment 1 fitness

Figure 18 shows the evolution of the best individual.

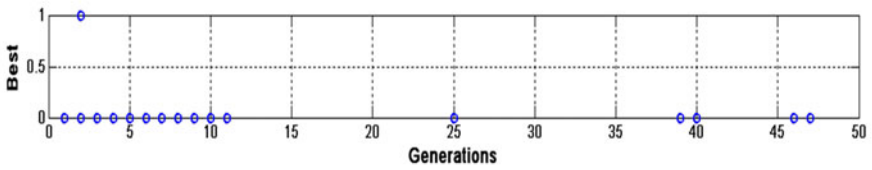


Fig. 18. Experiment 1 best individual

Below the results for experiment 2 are shown. Figure 19 shows input $e(t)$, modified by the GA.

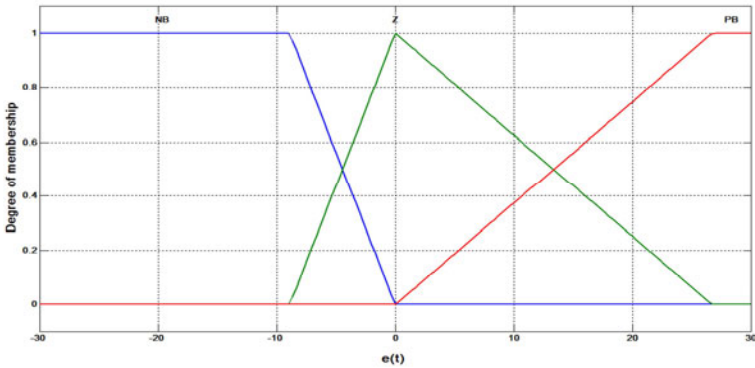


Fig. 19. FIS modified by the GA $e(t)$

Figure 20 shows the modified input $e'(t)$.

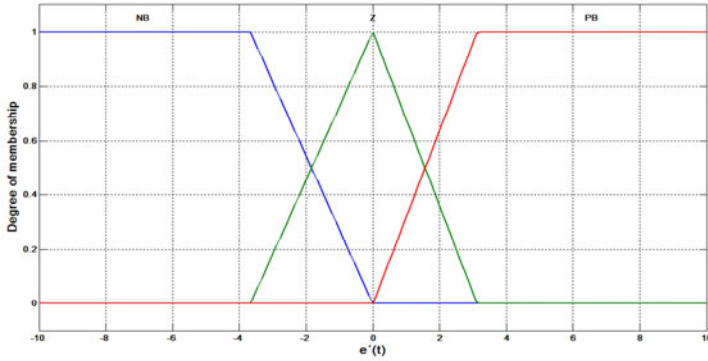


Fig. 20. FIS modified by the GA $e'(t)$

Figure 21 shows the FIS modified output.

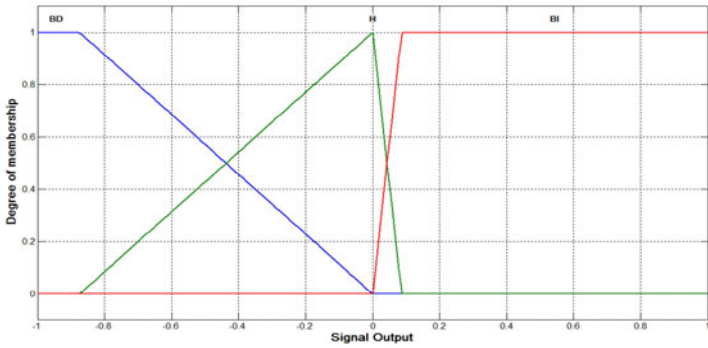


Fig. 21. FIS modified by the GA $y(t)$

Figure 22 shows the control surface modified by the GA.

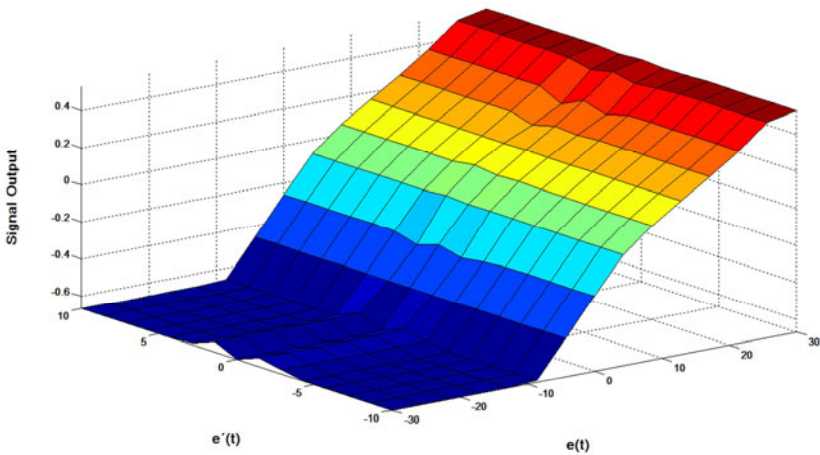


Fig. 22. Control Surface

Figure 23 shows the engine speed.

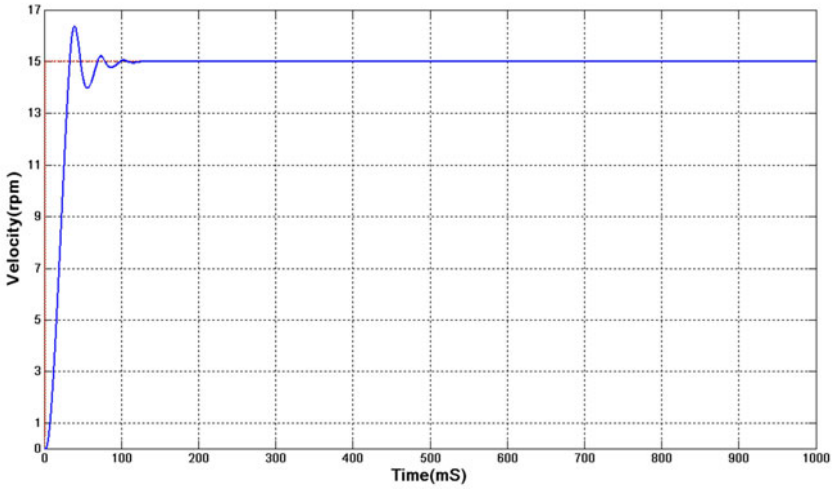


Fig. 23. Experiment 2 Velocity

Figure 24 shows the evolution of the steady state error for experiment 2.

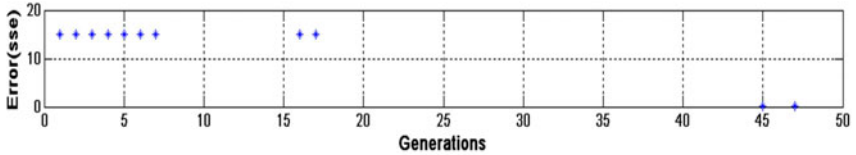


Fig. 24. Experiment 2 steady state error

Figure 25 shows the evolution of the GA Fitness.

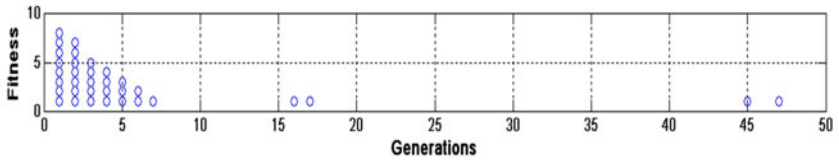


Fig. 25. Experiment 2 fitness

Figure 26 shows the evolution of the best individual.

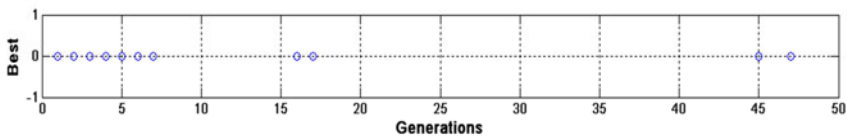


Fig. 26. Experiment 2 best individual

As shown in Table 2, the best FIS evaluated for the FLC (Fuzzy PD incremental) was the one in experiment 10. The results are shown belows.

Figure 27 shows the input $e(t)$.

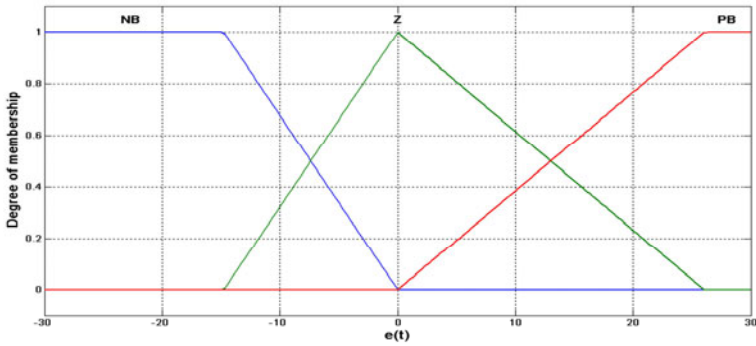


Fig. 27. FIS input $e(t)$

Figure 28 shows the input $e'(t)$.

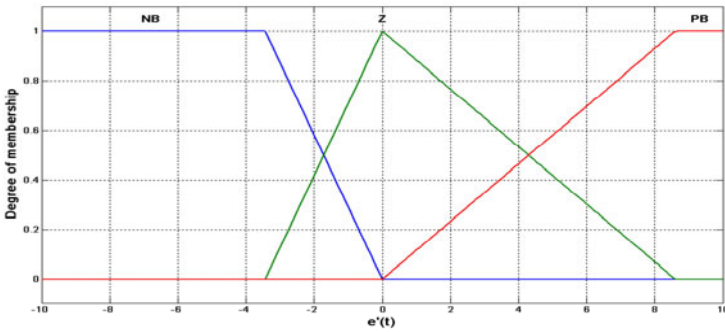


Fig. 28. FIS input $e'(t)$

Figure 29 shows FIS output.

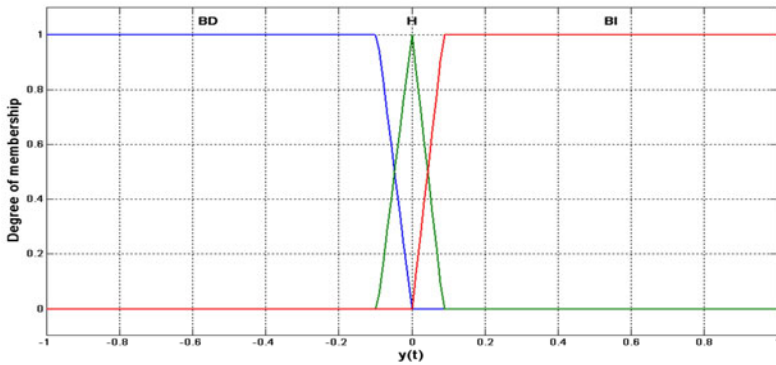


Fig. 29. FIS output $y(t)$

Figure 30 shows the control surface.

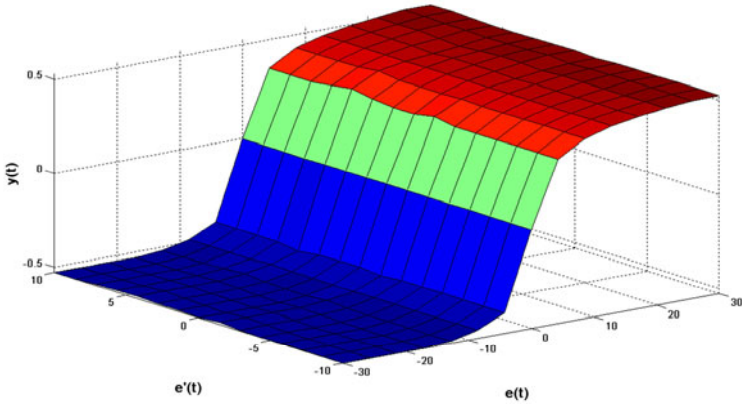


Fig. 30. Control Surface

Figure 31 shows the speed engine.

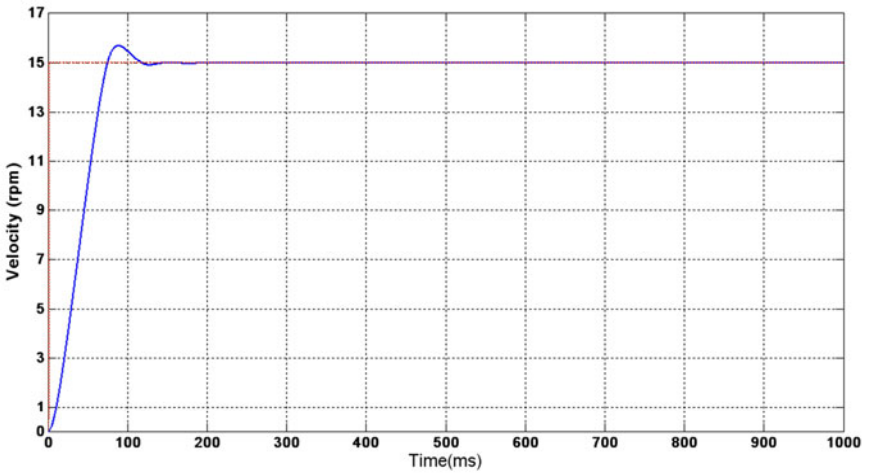


Fig. 31. Experiment 10 engine speed

Figure 32 shows the sse evolution for experiment 10.

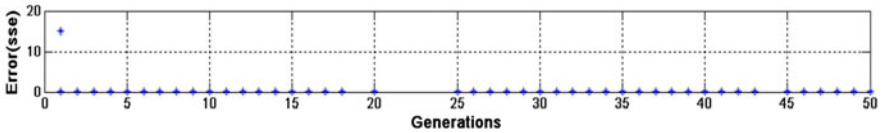


Fig. 32. Experiment 10 sse

Figure 33 shows the evolution of the GA fitness.

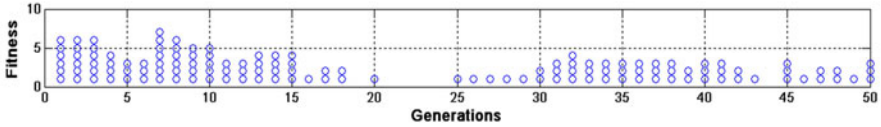


Fig. 33. Experiment 10 fitness

Figure 34 shows the evolution of the best individual.

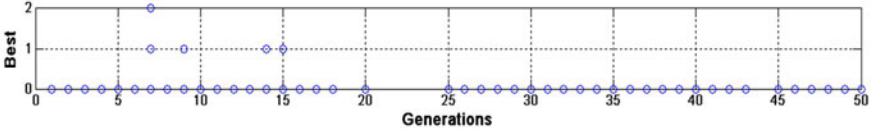


Fig. 34. Experiment 10 best individual

5 Conclusions

A FIS for the speed control of a DC motor is proposed. Due to bad response of the initial closed loop FLC, the next step was to optimize it. Three membership functions for input and output (Trapezoidal and Triangular) were optimized, but the fuzzy rules are not optimized because is thought in the future to implement this GA in hardware, so we are thinking of the speed and efficiency of the algorithm. To evaluate the controllers, the GA considered three characteristics of the FLC, overshoot, undershoot and steady state error.

Each FIS is simulated in an Fuzzy incremental PD for speed control of the DC motor.

The best FLC was obtained in 50 generations with 80% crossover, with a result of 15.85 of overshoot and undershoot, steady state error of $1.2145e-007$, in a time of 232.21 seconds with a speed of 15 rpm.

Matlab-Simulink was used to perform the simulations, but in the future a hardware implementation will be done.

References

- [1] Cirstea, M.N., Khor, J.G., McCormick, M.: Neural and fuzzy logic control of drives and power system. Newnes (2002)
- [2] Driankov, D., Hellendorn, H., Reinfrank, M.: An Introduction to Fuzzy Control. Springer, Heidelberg (1996)
- [3] Goldberg, D.E.: Genetic Algorithm in Search. Optimization & Machine Learning (1989)
- [4] Jang, J.S.R.: Fuzzy Controller Desing without Domain Experts. In: IEEE International Conference on Fuzzy systems, pp. 289–296 (1992)
- [5] Jantzen, J.: Tuning of Fuzzy PID Controllers, pp. 1–22 (1998)
- [6] Karr, C.L., Gentry, J.: Fuzzy Control of PH using Genetic Algorithm. IEE Transactions on Fuzzy Systems 1, 46–53 (1993)

- [7] Klir, G., Yuan, B.: Fuzzy Sets and Fuzzy Logic Theory and Applications. Prentice Hall, Englewood Cliffs (1995)
- [8] Koza, J.R.: Genetic Programming: on the Programming of Computers by means of natural selection (1992)
- [9] Man, K.F., Tang, K.S., Kwong, S.: Genetic Algorithms. Springer, Heidelberg (1999)
- [10] McNeilland, D., Freiburger, P.: Fuzzy Logic: The Revolutionary Computer Technology that is Changing our World. Simon & Schuster (1993)
- [11] Tsoukalas, L., Ohrig, R.E.: Fuzzy and Neural Approaches in Engineering. Wiley Interscience, Hoboken (1997)
- [12] Web page of Matlab-Simulink (2007), <http://www.mathworks.com>
- [13] Zadeh, L.A.: Fuzzy Sets. Information and Control 8, 338–353 (1965)
- [14] Zdenko, K., Stjepan, B.: Fuzzy Controller Design Theory and Applications. Taylor and Francis, Abington (2006)

Design of a Fuzzy System for the Longitudinal Control of an F-14 Airplane

Leticia Cervantes and Oscar Castillo

Tijuana Institute of Technology, Calzada Tecnológico s/n, Tijuana, México
lettyy2685@hotmail.com, ocastillo@tectijuana.mx

Abstract. In this paper we present a design of a fuzzy system for the longitudinal control of an F-14 airplane. The longitudinal control is carried out only by controlling the elevators of the airplane. To carry out such monitoring it is necessary to use the stick, the rate of elevation and the angle of attack. These 3 variables are input into the fuzzy inference system, which is of Mamdani type, and we obtain as output the value of the elevators. After designing the fuzzy inference system we turn to the simulation stage. Simulation results of the longitudinal control are obtained using a plant in Simulink and those results are compared against the PID controller.

1 Introduction

This paper focuses on the field of fuzzy logic by taking into account also the control area. Both areas can work together to solve various problems, in this case the longitudinal flight control. Over time the planes have evolved and at the same time improving their techniques for controlling their flight and avoid accidents as much as possible. For this reason, we are considering in this paper the implementation of a system that controls the horizontal position of the aircraft. We created the fuzzy system to perform longitudinal control of the aircraft and then we used a simulation tool to test the fuzzy controller under noisy conditions. We designed the fuzzy controller with the purpose of maintaining the stability in horizontal flight of the aircraft by controlling only the movement of the elevators.

2 Background and Basic Concepts

In this section we present some background of aviation and how it is changing over time and also some basic concepts for understanding this work.

2.1 Aviation History

In 1680 Giovanni Borello studied the application of the flight muscles of birds and he came to the conclusion that a person could never fly by himself. In the

nineteenth century Sir George Cayley established the modern concept of a plane and he applied the idea of a fixed wing capable of lifting the airplane [1]. Cayley established the aerodynamics foundation. In 1804 Cayley constructed a plane and realized his first unmanned flights. Cayley worked during the 5 decades following in his prototype and in 1853 an assistant of Cayley performed a flight of short duration in the plane that was in England [8]. In 1856 Jean-Marie Le Bris realized his first sky gliding flight higher than his point of departure thanks to the glider "L'Albatros Artificiel" ,which for takeoff was drawn by horses in the beach. Etienne Lenoir who was a belgian inventor has the Credit for the first internal combustion engine in 1859, he developed only one horsepower and it was inefficient, but that resulted in billions of internal combustion engines have been built since then. Many people were part of that inventions and development of the aircraft up to date is in continue growth and development for the benefit of humanity. If we want to speak about a war plane we can say something about the F-14 airplane because it is the airplane that has been used for this task. The development of the F-14 came about because of the need to improve the aging American Fighters in use in the early days of the Cold War [11]. Some form of upgrade was required if the United States was to maintain its competitive edge over the Soviet Union. As a result, the U.S. Navy and the U.S. Air Force teamed up to explore the possibility of developing a fighter plane that would, as far as possible, be appropriate for the needs of both forces [9]. The practical flight aircraft was only possible once there was the energy source capable of driving. Since then, they have created many types of engines, more powerful, small and reliable, and used as the required application [10][4] [3].

2.2 Control Surfaces

Every aircraft has control surfaces or other means which are used to generate the forces and moments required to produce the accelerations, which causes the aircraft to be steered along its three-dimensional flight path to its specified destination. A conventional aircraft is represented in Figure 1. It is shown with the usual control surfaces, namely elevator, ailerons, and rudder. Such conventional aircrafts have a fourth control, the change in thrust, which can be obtained from the engines. Many modern aircraft, particularly combat aircraft, have considerably more control surfaces, which produce additional control forces or moments. Some of these additional surfaces and motivators include horizontal and vertical canards, spoilers, variable cambered wings, reaction jets, differentially operating horizontal tails and movable fins. One characteristic of flight control is that the required motion often needs a number of control surfaces to be used simultaneously [16].

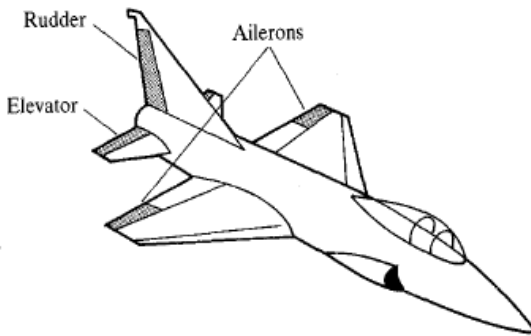


Fig. 1. Conventional Aircraft [16]

2.3 Flight Maneuvers

Any aircraft is capable of 3 possible rotations about 3 mutually perpendicular axes whose intersection point is above the center of gravity of the airplane (see Fig.2).



Fig. 2. Possible rotations around the 3 axes

As we can appreciate from Fig.2, Pitch is the type of rotation that is done on the Y axis of the plane and it can be represented as an ascent or descent of its nose. Roll is a movement around the longitudinal axis of the plane (X axis) and be seen as a lateral inclination thereof. Yaw is a rotation about the vertical axis (Z) of the plane, and produces a change in the horizontal direction of flight. Each flight maneuver is assigned to a particular type of control: side control, direction control and longitudinal control. The first one, controls the ailerons and if one of these is flexed downward, it results in an increased lift on the wing for causing the rise of the same, while if the wing is bent upward, the wing produces a sustainability decrease, encouraging the decline of the plane. The second one controls the rudder and this is done using the pedals. For a yaw movement to the right, the driver presses the right pedal, thus turning the rudder surface to the right. Finally, the longitudinal control is responsible for controlling the elevators and when we talk about longitudinal control we make reference to the pitching motion generated by the aircraft. All controlled flight consists of either one, or a combination or more than one, of these basic maneuvers. It is important to know what can happen if

you have to control the flight of an airplane because, when back pressure is applied to the elevator control, the airplane's nose rises in relation to the pilot. When forward pressure is applied to the elevator control, the airplane's nose lowers in relation to the pilot. When right pressure is applied to the aileron control, the airplane's right wing lowers in relation to the pilot. When left pressure is applied to the aileron control, the airplane's left wing lowers in relation to the pilot. When pressure is applied to the right rudder pedal, the airplane's nose moves (yaws) to the right in relation to the pilot. When pressure is applied to the left rudder pedal, the airplane's nose moves (yaws) to the left in relation to the pilot [6][19][12].

2.4 Forces Involved in Flight

Four forces act upon an aircraft in relation to straight-and level, unaccelerated flight. These forces are thrust, lift, weight, and drag (see Fig.3).



Fig. 3. The four forces [7]

Thrust is the forward force produced by the powerplant/propeller. It opposes or overcomes the drag force. As a general rule, it is said to act parallel to the longitudinal axis. This is not always the case as explained later. Drag is a backward, retarding force, and is caused by disruption of airflow by the wing, fuselage, and other protruding objects. Drag opposes thrust, and acts rearward parallel to the relative wind. Weight is the combined load of the airplane itself, the crew, the fuel, and the cargo or baggage. Weight pulls the airplane downward because of the force of gravity. It opposes lift, and acts vertically downward through the airplane's center of gravity (CG).

The lift opposes the downward force of weight, is produced by the dynamic effect of the air acting on the wing, and acts perpendicular to the flightpath through the wing's center of lift [7]. Each airplane has a standard mathematical model and usually linear models are applied whose parameters are obtained from wind tunnel tests results using a scale model [2] [17]. These models are evaluated in each operation point with the information obtained in the tests. There are more

sophisticated nonlinear models that approximate the aerodynamic coefficients by truncated Taylor series [18].

3 Related Work

The field of aeronautics has been studied extensively over the years with the aim of further improving the techniques of flying as much as possible to avoid accidents. There has been several research works in this area although each author takes their own approach to carry out the investigation.

H. Enhelen [5] mentions in his work an intelligent and autonomous flight control system for an atmospheric re-entry vehicle that is investigated, based on fuzzy logic control and aerodynamic inversion computation, also a common PD-Mamdani fuzzy logic controller is designed for all the five re-entry flight regions characterized by different actuator configurations. B. Kadmiry [13] addressed the design of a fuzzy flight controller that achieves stable and robust “aggressive” maneuverability for an unmanned helicopter. The proposed fuzzy flight controller consists of a combination of a fuzzy gain scheduler and a linguistic (Mamdani-type) controller. The fuzzy gain scheduler is used for stable and robust altitude, roll, pitch, and yaw control. The linguistic controller is used to compute the inputs to the fuzzy gain scheduler, i.e., desired values for roll, pitch, and yaw at given desired altitude and horizontal velocities. The flight controller is obtained and tested in simulation using a realistic nonlinear MIMO model of a real unmanned helicopter platform, the APID-MK3. Edgar Sanchez and Hector Becerra [22] describe different flight controllers for an X-Cell mini-helicopter. They divided their work in 2 parts. The first scheme consists of a conventional SISO PID controllers for z-position and roll, pitch and yaw angles. In the second scheme, two of the previous PID controllers are used for roll and pitch, and a linear regulator is added to control altitude and yaw angle.

D. J. Walker [25] in his work proposes a separated longitudinal and lateral controllers, each designed using H-infinity optimization. Rachman Endri, et al. [20], in their work used non-linear simulation of controller for automatic flight control system to complete control design process. After, they used the linear approach in the design and simulation of the controller (control laws) and they simulated the controller on the non-linear real aircraft model.

Jacob Reiner, et al. [21] in their work used an angle-of-attack command system for longitudinal control of a high performance aircraft. Keviczky Tamás and Balas Gary [14] in their work used a comparison between receding horizon control for the longitudinal axis control of an f-16 aircraft. As mentioned above, there are several works on aviation for their importance regarding the control, flight control range from small, trajectories, using methods such as fuzzy logic and inference systems [24][23][15][12].

4 Problem Description

The purpose of this work was to develop a fuzzy system for automatic control to maintain the aircraft in horizontal flight. The goal was to create the fuzzy system to

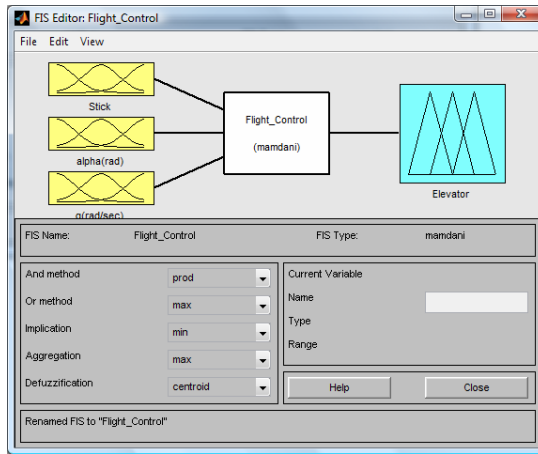


Fig. 5. Fuzzy System

The fuzzy system that we used as a controller has 3 membership functions for each of the inputs and 3 membership functions of the output. We worked with different types of membership functions, such as the Gaussian, Bell, Trapezoidal and Triangular. The rules that we used for the fuzzy system are shown in Fig.6.

1. If (Stick is bajo) and (alpha(rad) is bajo) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
2. If (Stick is bajo) and (alpha(rad) is medio) and (q(rad/sec) is medio) then (Elevator is bajo) (1)
3. If (Stick is bajo) and (alpha(rad) is alto) and (q(rad/sec) is alto) then (Elevator is medio) (1)
4. If (Stick is bajo) and (alpha(rad) is bajo) and (q(rad/sec) is medio) then (Elevator is bajo) (1)
5. If (Stick is bajo) and (alpha(rad) is bajo) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
6. If (Stick is bajo) and (alpha(rad) is medio) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
7. If (Stick is bajo) and (alpha(rad) is medio) and (q(rad/sec) is medio) then (Elevator is bajo) (1)
8. If (Stick is bajo) and (alpha(rad) is medio) and (q(rad/sec) is alto) then (Elevator is medio) (1)
9. If (Stick is bajo) and (alpha(rad) is alto) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
10. If (Stick is bajo) and (alpha(rad) is alto) and (q(rad/sec) is medio) then (Elevator is bajo) (1)
11. If (Stick is bajo) and (alpha(rad) is alto) and (q(rad/sec) is alto) then (Elevator is bajo) (1)
12. If (Stick is medio) and (alpha(rad) is bajo) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
13. If (Stick is medio) and (alpha(rad) is bajo) and (q(rad/sec) is medio) then (Elevator is bajo) (1)
14. If (Stick is medio) and (alpha(rad) is bajo) and (q(rad/sec) is alto) then (Elevator is bajo) (1)
15. If (Stick is medio) and (alpha(rad) is medio) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
16. If (Stick is medio) and (alpha(rad) is medio) and (q(rad/sec) is medio) then (Elevator is medio) (1)
17. If (Stick is medio) and (alpha(rad) is medio) and (q(rad/sec) is alto) then (Elevator is medio) (1)
18. If (Stick is medio) and (alpha(rad) is alto) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
19. If (Stick is medio) and (alpha(rad) is alto) and (q(rad/sec) is medio) then (Elevator is medio) (1)
20. If (Stick is medio) and (alpha(rad) is alto) and (q(rad/sec) is alto) then (Elevator is medio) (1)
21. If (Stick is alto) and (alpha(rad) is bajo) and (q(rad/sec) is bajo) then (Elevator is bajo) (1)
22. If (Stick is alto) and (alpha(rad) is bajo) and (q(rad/sec) is medio) then (Elevator is medio) (1)
23. If (Stick is alto) and (alpha(rad) is bajo) and (q(rad/sec) is alto) then (Elevator is medio) (1)
24. If (Stick is alto) and (alpha(rad) is medio) and (q(rad/sec) is bajo) then (Elevator is medio) (1)
25. If (Stick is alto) and (alpha(rad) is medio) and (q(rad/sec) is medio) then (Elevator is medio) (1)
26. If (Stick is alto) and (alpha(rad) is medio) and (q(rad/sec) is alto) then (Elevator is medio) (1)
27. If (Stick is alto) and (alpha(rad) is alto) and (q(rad/sec) is alto) then (Elevator is alto) (1)

Fig. 6. Fuzzy System Rules

6 Simulation Results

In this section we present the results obtained when performing the tests using the simulation plant with the PID and Fuzzy controllers. It also presents the results obtained by optimizing the fuzzy system.

6.1 Results with PID Controller

The first simulation was performed with the PID controller and we obtained the elevators behavior shown in Fig.7. We obtained an average elevator angle of 0.2967.

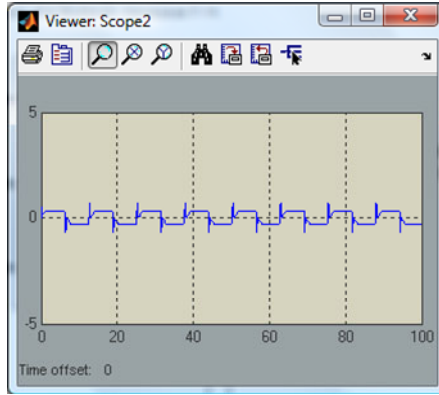


Fig. 7. Elevators Behavior

6.2 Results with the Fuzzy Controller

Once the simulation results with the PID Controller were obtained, we proceeded with our Fuzzy Controller (see Fig.8) using the fuzzy system that was created previously.

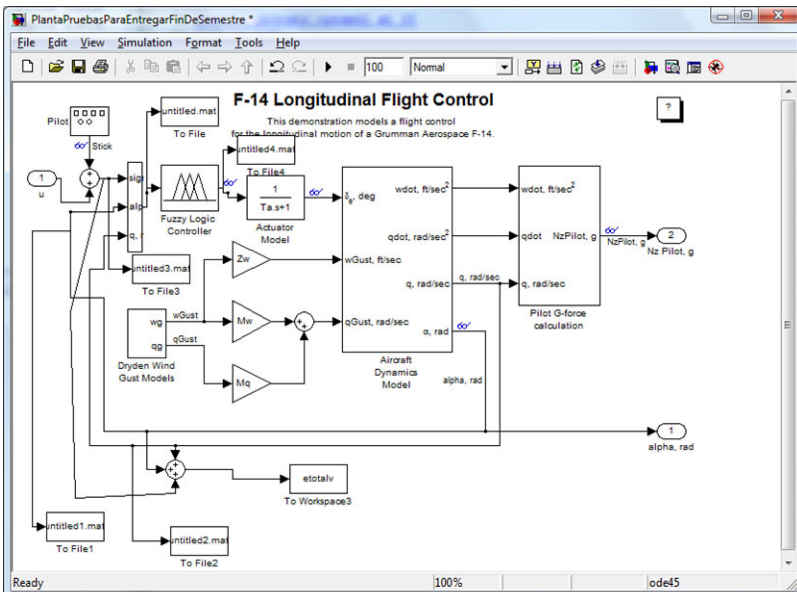


Fig. 8. Simulation Plant with Fuzzy Controller

The simulation was carried out with different types of membership functions and the results that were obtained as shown in Table 1.

Table 1. Results for simulation plant with Fuzzy Controller

Membership functions	Trapezoidal	Triangular	Gauss	Bell
Errors comparing with PID	0.1094	0.1131	0.1425	0.1222
Comments	Fast Simulation	Less Fast Simulation	Slow simulation in comparison with previous	Slow simulation in comparison with previous

Having obtained the above results, we used a genetic algorithm to optimize the membership functions of the fuzzy system and after implementing the genetic algorithm we obtained the results shown in Table 2.

Table 2. Results for simulation plant with Fuzzy Controller and Genetic Algorithm

Genetic Algorithm	Error with respect to PID
Using Trapezoidal membership functions	0.0531
Using Gauss membership functions	0.084236395
Using Bell membership functions	0.0554
Using Triangular membership functions	0.0261

Given the above results we can see that better results were obtained using genetic algorithms and in particular the best result was using Membership functions of triangular type. The best fuzzy system that the genetic algorithm produced is shown in fig.9.

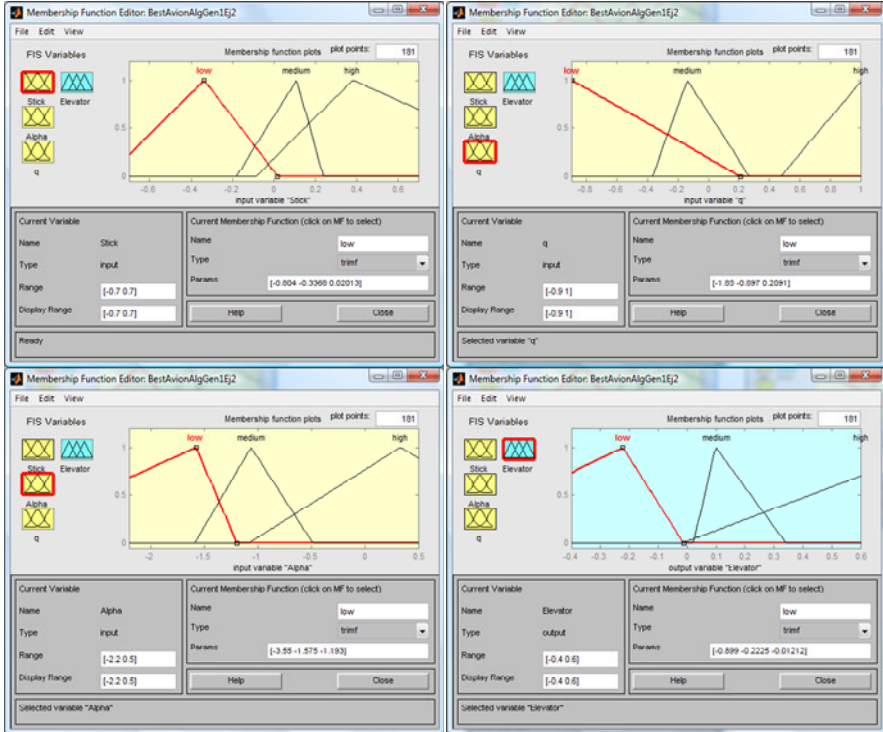


Fig. 9. The best fuzzy system

7 Conclusions

Based on the results obtained with the proposed method we can say that to achieve control of the plane in the Pitch axis, the fuzzy controller is a good option to have a good control of the elevators and maintain the aircraft on the Pitch axis as required.

The result that we obtained was better when we used the genetic algorithm optimization with a triangular type membership function. It is important to consider different membership function because you can see the differences and you can choose the best result. Future work consists in adding noise to test the fuzzy controller under disturbances.

References

- [1] Abusleme, H., Angel, C.: Control difuso de un vehiculo volador no tripulado, Phd Thesis, Pontificia University of Chile (2000)
- [2] Blakelock, J.: Automatic Control of Aircraft and Missiles. Wiley, New York (1965)
- [3] Dorf, R.: Modern Control Systems. Addison-Wesley Pub. Co., Reading (1997)
- [4] Dwinell, J.: Principles of Aerodynamics. McGraw-Hill Book Company, New York (1929)
- [5] Engelen, H., Babuska, R.: Fuzzy logic based full-envelope autonomous flight control for an atmospheric re-entry spacecraft Control Engineering Practice, vol. 11(1), pp. 11–25 (January 2003)
- [6] Federal Aviation Administration, Airplane Flying Handbook, Book (2007)
- [7] Federal Aviation Administration. Pilot's Handbook of Aeronautical Knowledge, Book (2008)
- [8] Garcia, C.: "Everything about airplane". Paper (2009), <http://los-aviones-y-su-historia.blogspot.com/2009/02/40-motor-de-combustion-interna-ano-1859.html>
- [9] Gardner, A.: U.S Warplanes The F-14 Tomcat, Book (2003)
- [10] Gibbens, P., Boyle, D.: Introductory Flight Mechanics and Performance. University of Sydney, Australia (1999)
- [11] Holmes, T.: US Navy F-14 Tomcat Units of Operation Iraqi Freedom, Book (2005)
- [12] Jamshidi, M., Vadiee, N., Ross, T.: Fuzzy Logic and Control: Software and Hardware Applications, vol. 2. University of New Mexico
- [13] Kadmiry, B., Driankov, D.: A fuzzy flight controller combining linguistic and model-based fuzzy control. Fuzzy Sets and Systems 146(3), 313–347 (2004)
- [14] Keviczky, T., Balas, G.: Receding horizon control of an F-16 aircraft: A comparative study. Control Engineering Practice 14(9), 1023–1033 (2006)
- [15] Liu, M., Naadimuthu, G., Lee, E.S.: Trayectory tracking in aircraft landind operations management using the adaptive neural fuzzy inference system. Computers & Mathematics with Applications 56(5), 1322–1327 (2008)
- [16] McLean, D.: Automatic Flight Control System, Book, Prentice Hall, Englewood Cliffs (1990)
- [17] McRuer, D., Ashkenas, I., Graham, D.: Aircraft Dynamics and Automatic Control. Princeton University Press, Princeton (1973)
- [18] Morelli, E.A.: Global Nonlinear Parametric Modeling with Application to F-16 Aerodynamics, NASA Langley Research Center, Hampton, Virginia (1995)
- [19] Nelson, R.: Flight Stability and automatic control. Department of Aerospace and Mechanical Engineering, University of Notre Dame, 2nd edn. McGraw Hill, New York (1998)
- [20] Rachman, E., Jaam, J., Hasnah, A.: Non-linear simulation of controller for longitudinal control augmentation system of F-16 using numerical approach Information Sciences, vol. 164(1-4), pp. 47–60 (August 2, 2004)
- [21] Reiner, J., Balas, G., Garrard, W.: Flight control design using robust dynamic inversion and time-scale separation. Automatic 32(11), 1493–1504 (1996)
- [22] Sanchez, E., Becerra, H., Velez, C.: Combining fuzzy, PID and regulation control for an autonomous mini-helicopter. Information Sciences 177(10), 1999–2022 (2007)

- [23] Sefer, K., Omer, C., Okyay, K.: Adaptive neuro-fuzzy inference system based auto-nomous flight control of unmanned air vehicles. *Expert Systems with Applications In Press, Corrected Proof* (June 2009)
- [24] Song, Y., Wang, H.: Design of Flight Control System for a Small Unmanned Tilt. Rotor Aircraft *Chinese Journal of Aeronautics*, vol. 22(3), pp. 250–256 (2009)
- [25] Walker, D.J.: Multivariable control of the longitudinal and lateral dynamics of a fly-by-wire helicopter. *Control Engineering Practice* 11(7), 781–795 (2003)

A Fuzzy Reactive Controller of a Mobile Robot

Abraham Meléndez, Oscar Castillo, and Arnulfo Alanis Garza

Tijuana Institute of Technology, Tijuana México

Abraham.ms@gmail.com, ocastillo@tectijuana.mx,

alanis@tectijuana.edu.mx

Abstract. This paper describes an evolutionary algorithm approach for the optimization of a reactive controller applied to a mobile robot. The algorithm will optimize the Fuzzy Inference System and the position and number of the sensors on the robot, while trying to use the minimum power possible.

1 Introduction

Robots are being more commonly used in many areas of research and a reason for this is that they are becoming more accessible economically for researchers. In this paper we consider the optimization of a fuzzy controller; that gives the ability of reaction to the robot. This may be too general so let's limit what in this paper will be described as ability of reaction, this is applied in the navigation concept, so what this means is that when the robot is moving and at some point of its journey it encounters an unexpected obstacle it will react to this stimulation avoiding and continuing on its path. The trajectory and path following are considered independent parts and are not consider on this paper [19].

There are many traditional techniques available to use in control, such as PD, PID and many more but we took a different approach in the Control of the robot, using an area of soft computing which is fuzzy logic that was introduced by Zadeh [1]. Later this idea was applied in the area of control by Mandami [2] where the concept of FLC (Fuzzy Logic Controller) originated. It's also important to mention that this is not the only area were the fuzzy concepts are applied but it's where the most work has been done, and were many people have contributed important ideas and methods like Takagi and Sugeno [2].

This paper is organized as follows: In section 2 we describe the mobile robot used in these experiments, section 3 describes the development of the evolutionary method. Section 4 shows the simulation results. Finally section 5 shows the Conclusions.

2 Mobile Robot

In this section we describe the particular mobile robot considered in this work. The robot is based on the description of the the Simulation toolbox for mobile

robots[22] assumes wheeled mobile robot consisting of one or two conventional, steered, unactuated and not-sensed wheels and two conventional, actuated, and sensed wheels (conventional wheel chair model). This type of chassis provides two DOF (degrees of freedom) locomotion by two actuated conventional non-steered wheels and one or two unactuated steered wheels. The Robot has two degrees of freedom (DOFs): y-translation and either x-translation or z-rotation [22], Figure 1 shows the robots configuration, it will have 2 independent motors located on each side of the robot and one castor wheel for support located at the form of the robot.

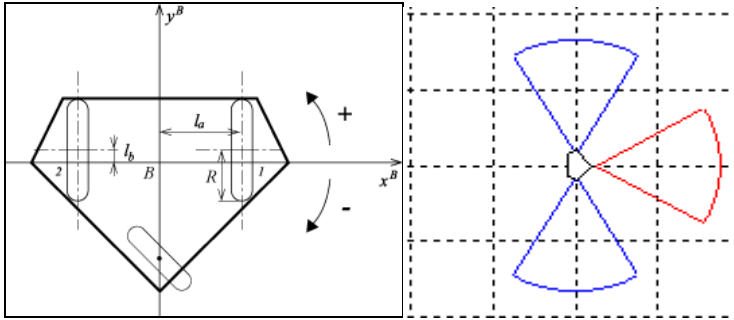


Fig. 1. Kinematic coordinate system assignments [22]

The kinematic equations of the mobile robot are as follows:

$$\begin{pmatrix} v_{B_x} \\ v_{B_y} \\ \omega_{B_z} \end{pmatrix} = \frac{R}{2l_a} \begin{pmatrix} -l_b & l_b \\ -l_a & -l_a \\ -1 & 1 \end{pmatrix} \begin{pmatrix} \omega_{W_1} \\ \omega_{W_2} \end{pmatrix} \tag{1}$$

Equation 1 represents the sensed forward velocity solution [22].

Equation 2 represents the Actuated Inverse Velocity Solution [22].

$$\begin{pmatrix} \omega_{W_1} \\ \omega_{W_2} \end{pmatrix} = \frac{1}{R(l_b^2 + 1)} \begin{pmatrix} -l_a l_b & -l_b^2 - 1 & -l_a \\ l_a l_b & -l_b^2 - 1 & l_a \end{pmatrix} \begin{pmatrix} v_{B_x} \\ v_{B_y} \\ \omega_{B_z} \end{pmatrix} \tag{2}$$

Where under the Metric system.

- V_{B_x}, V_{B_y} Translational velocities [$\frac{m}{s}$],
- ω_{B_z} Robot z-rotational velocity [$\frac{rad}{s}$],
- $\omega_{W_1}, \omega_{W_2}$ Wheel rotational velocities [$\frac{rad}{s}$],
- R Actuated wheel radius[m],
- l_a, l_b Distances of wheels from robot's axes [m].

3 Evolutionary Method Description

In this section we will cover the Genetic Algorithm applied to our problem of finding the best fuzzy reactive controller for a mobile robot.

The purpose of using an evolutionary method, is to obtain the best reactive control possible, for the robot, but also taking into consideration other desirable characteristics on the robot that we want to improve making this a multi objective [17] problem, for this we will take advantage of the HGA (Hierarchy Genetic Algorithm) intrinsic characteristic to solve multi objective problems, now let us state the main goal of our HGA. Figure 2 shows the genetic process.

The main goal is to optimize the Reactive Control taken into consideration the following:

- Fine tune the Fuzzy Memberships
- Optimize the FIS if then fuzzy rules
- The mobile robot Power Usage

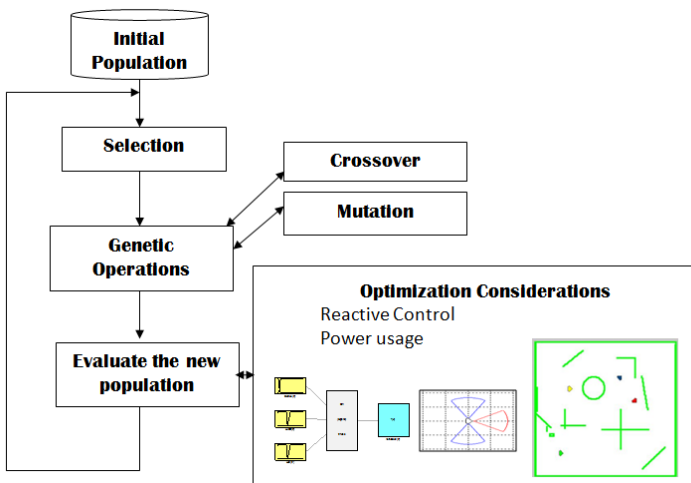


Fig. 2. Genetic Algorithm process

In Fig. 2, we show the global cycle process of the GA, under the Evaluation of the each individual, is where we are going to measure the goodness of each of the FIS (Fuzzy Inference System) represent by each Individual chromosome, in our test area, that will take place in a unknown environment (Maze [19]) to the robot where the robot's objective will be find the exit, avoiding hitting the walls and any other obstacle present.

Our criteria to measure the FIS global performance will take into consideration the following:

- Number of collisions
- Average time between hits
- Cover Distance

- Time use to cover distance
- Battery life

All of these variables are the inputs of the Evaluation FIS that we will use to obtain the fitness value of each chromosome (shown in Figure 3).

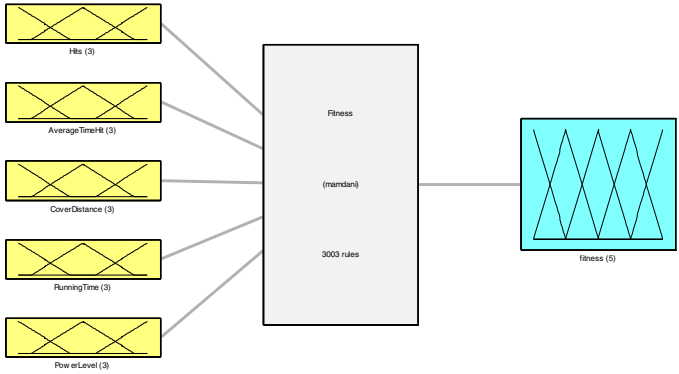


Fig. 3. Fitness FIS

The FIS that we optimize is a Mamdani type fuzzy system, consisting of 3 inputs that are the distances obtain by the robots sensors describe on section 2, and 2 outputs that control de velocity of the servo motors on the robot, all this information is encoded on each chromosome.

The chromosome architecture, shown on Fig. 4, where we have encoded the membership functions type and parameters, we have set a maximum number of 5 membership functions for each of the outputs and input variables.

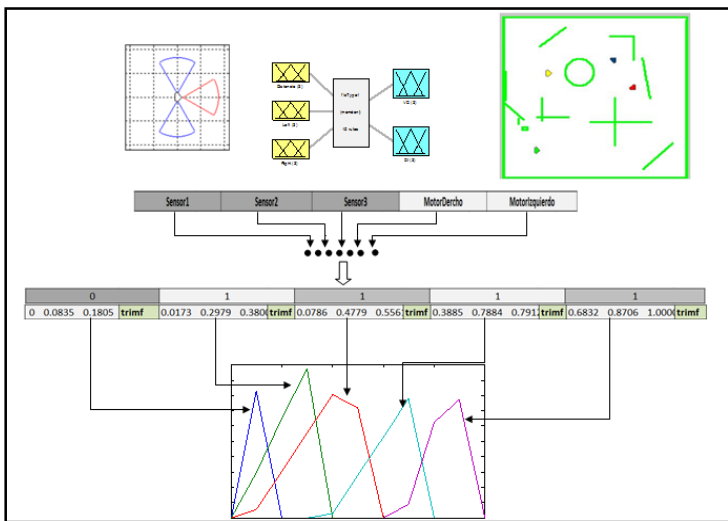


Fig. 4. Chromosome Architecture

4 Simulation Results

We have worked on the reactive control for a mobile robot before, where we use a particular maze problem to test the effectiveness of each of the reactive controls, and we did not use any optimization strategy to fine tune the controllers as it was a manually process, and from that experience is that we decided to apply GA to this problem. On Figure 5 and Table 1 we show the results obtained in the experiments.

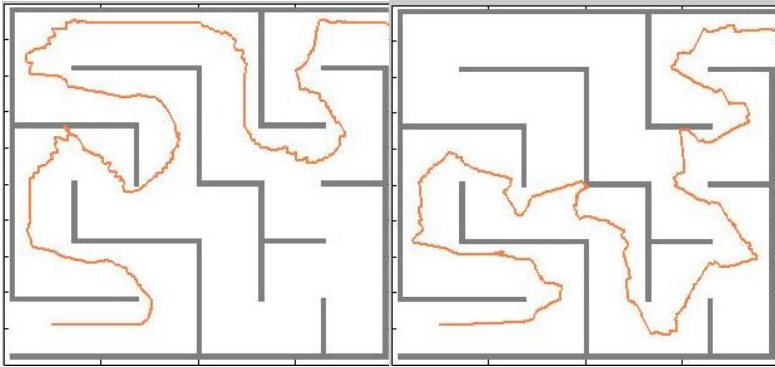


Fig. 5. Robot trajectory control by 10 Rules FIS (Left) and 27 FIS Rules (Right)

Figure 6 shows the new optimized results with the GA.

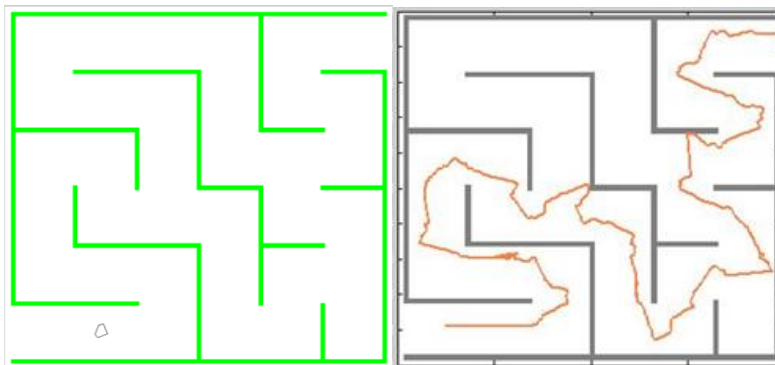


Fig. 6. Previous Result vs. New Controllers Experiments

Table 1. Reactive Control Non Optimized Results

<i>Experiment</i>	<i>LAS</i>	<i>RAS</i>	<i>Time</i>	<i>FE</i>
<i>27 Rules FIS</i>				
9	38.26	38.46	60.34	Yes
10	40.42	40.64	59.50	Yes
12	40.06	40.12	60.70	Yes
<i>Average</i>	<i>39.58</i>	<i>39.74</i>	<i>60.18</i>	
<i>Standard Deviation</i>	<i>1.16</i>	<i>1.14</i>	<i>0.62</i>	
<i>10 Rules FIS</i>				
1	31.51	47.43	55.75	Yes
2	35.55	54.33	51.80	Yes
3	36.66	53.68	55.40	Yes
4	36.66	53.68	54.10	Yes
5	36.91	54.18	49.95	Yes
6	35.44	52.51	46.65	Yes
7	36.14	51.66	49.85	Yes
8	33.48	49.58	51.76	Yes
9	37.61	51.51	55.15	Yes
<i>Average</i>	<i>35.55</i>	<i>52.06</i>	<i>52.27</i>	
<i>Standard Deviation</i>	<i>1.92</i>	<i>2.32</i>	<i>3.10</i>	

*LAS=Left Motor Average Speed

*RAS= Right Motor Average Speed

*FE= Found Exit

5 Conclusions

We are yet to conclude these experiments but we expect with the help of the HGA to improve the overall performance of the mobile robot, and improve the results obtained previously.

We will test our best reactive controller obtained with the HGA under the same maze problem this in order to establish a comparison between the controllers.

Acknowledgments

We would like to express our gratitude to the CONACYT, and Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

- [1] Aceves, A., Aguilar, J.: A Simplified Version of Mamdani's Fuzzy Controller The Natural Logic Controller. *IEEE Transactions on Fuzzy Systems* 14(1), 16–30 (2006)
- [2] Astudillo, L., Castillo, O., Aguilar, L.T.: Intelligent Control of an Autonomous Mobile Robot Using Type-2 Fuzzy Logic. *Journal of Nonlinear Studies* 14(1), 37–48
- [3] Leslie, A., Castillo, O., Aguilar, L.: Control Difuso de Robots Autónomos Móviles en Ambientes Inciertos usando Lógica Difusa. Tesis, División de Estudios y Posgrados e Investigación, ITT, México, Octubre del (2006)
- [4] Campion, G., Bastin, G., D'Andrea-Novell, B.: Structural Properties and Classification of Kinematic and Dynamic Models of Wheeled Mobile Robots. *IEEE Trans. on Robotics and Automation* 12(1), 47–62 (1996)
- [5] Cardenas, S., Castillo, O., Aguilar, L.: " Controlador de Seguimiento para un Robot Móvil empleando Lógica Difusa", Tesis, División de Estudios y Posgrados e Investigación, ITT, México (October 2005)
- [6] Conde Bento, L., Urbano, N., Abel, M., Michel, P.: Path-Tracking Controller of a bi-steerable Cybernetic Car using Fuzzy Logic. In: *Proceedings of ICAR 2003 The 11th International Conference on Advanced Robotics*, pp. 1556–1561 (2003)
- [7] Cupertino, F., Giordano, V., Naso, D., Delfino, L.: Fuzzy control of a mobile robot. *Robotics & Automation Magazine, IEEE*, 74–81
- [8] Erkkinen, T.: Embedded Coder Robot NXT,
<http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=13399>
- [9] Erkkinen, T.: Embedded Coder Robot for LEGO® Mindstorms® NXT Rev. 3.03,
<http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=13399>
- [10] Ishikawa, S.: A Method of Indoor Mobile Robot Navigation by Fuzzy Control. In: *Proc. Int. Conf. Intell. Robot. Syst.*, Osaka, Japan, pp. 1013–1018 (1991)
- [11] Thomson, A., Baltes, J.: A path following system for autonomous robots with minimal computing power, University of Auckland, Private Bag 92019, Auckland, New Zealand, Technical Report (July 2001)
- [12] Jang, J.-S.R., Sun, C.-T., Mizutani, E.: *Neuro-Fuzzy and Soft Computing a computational approach to learning and machine intelligence*. Prentice Hall, Englewood Cliffs (1997)
- [13] Jang, J.-S.R., Sun, C.-T., Mizutani, E.: *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall, Englewood Cliffs (1997)
- [14] Kulkarni, A.: *Computer Vision and Fuzzy-Neural Systems*. Prentice Hall PTR, Englewood Cliffs (Mayo 2001)
- [15] Leyden, M., Toal, D., Flanagan, C.: A Fuzzy Logic Based Navigation System for a Mobile Robot. In: *Proceedings of Automatisierungs Symposium*, Wismar, Germany (1999)
- [16] Macek, K., Petrovic, I., Siegart, R.: A control method for stable and smooth path following of mobile robots. In: *Proceedings of the 2nd European Conference on Mobile Robots - ECMR 2005*, Ancona, Italy, September 7-10, pp. 128–133 (2005)
- [17] Man, K.F.: *Genetic Algorithms: Concepts and Designs (Advanced Textbooks in Control and Signal Processing)*. Springer, Heidelberg (1999) (Corrected edition)

- [18] Martínez, R., Castillo, O., Aguilar, L.: "Control Inteligente de Robots Autónomos Móviles Bajo Pares Perturbados Usando Lógica Difusa Tipo-2" Tesis, División de Estudios y Posgrados e Investigación, ITT, México (February 2008)
- [19] Meléndez, A., Castillo, O., Soria, J.: Reactive Control of a Mobile Robot in a Distributed Environment Using Fuzzy Logic. In: NAFIPS 2008. Annual Meeting of the North American Fuzzy Information Processing Society, May 19-22, pp. 1–5 (2008)
- [20] Payton, D.W., Rosenblatt, J.K., Keirse, D.M.: Plan guided reaction. *IEEE Transactions on Systems, Man and Cybernetics* 20(6), 1370–1382
- [21] Peri, V.M., Simon, D.: Fuzzy logic control for an autonomous robot Fuzzy Information Processing Society. In: NAFIPS 2005. Annual Meeting of the North American, pp. 337–342 (2005)
- [22] Pishkenari, H.N., Mahboobi, S.H., Meghdari, A.: On the Optimum Design of Fuzzy Logic Controller for Trajectory Tracking Using Evolutionary Algorithms. In: *IEEE Conference on Cybernetics and Intelligent Systems*, December 1-3, vol. 1, pp. 660–665 (2004)
- [23] Autonomous Mobile Robotics Toolbox,
http://www.uamt.feec.vutbr.cz/robotics/simulations/amrt/imrobot_en.html

Multi-Agent System with Personality Profiles and Preferences and Learning for Autonomous Mobile Robot with Fuzzy Logic Support

Arnulfo Alanis Garza, Oscar Castillo, and José Mario García Valdez

Division of Graduate Studies and Research, Instituto Tecnológico de Tijuana,
Calzada Tecnológico, S/N, Tijuana, México
alanis@tectijuana.edu.mx

Abstract. For achieving a good and friendlier interaction between man and robot, this paper used various techniques of psychology, such as the ODD protocol and Big Five. Loyalty is the fact that helps knowledge generation, expression, and learning in humans. These models of personality are verified, in real situations, this, by its application in the situation where humans and robots interact in view of the stage.

1 Introduction

Robotics has evolved considerably since its inception, examples of the use of them seriously, in car manufacturing. In recent times the use of robots in the field of households has grown, thanks to a decline in cost and size. It is mentioned that not too distant future, robots will provide services to individuals, to our workplaces and in our homes [1]. There are currently supporting some domestic chores (vacuum cleaners).

It would be wrong to think or believe that users learn about sensors, actuators and control architectures. In support of this, you have to convey a mental model that helps users make sense of the robot's behavior and understanding that are necessary steps on their side. The proper design of the interaction between humans and robots will be crucial for the understanding and acceptance of new robotic products [2].

Some researchers [16, 17] have reported that in the human domain Affect, long-lasting affective relationship as well as event-based temporal aspects such as emotion significantly influences on the determination of human beings social behaviors. Based on the above claims, we designed the mechanism of a robot loyalty formation by computationally modeling the affective relationship between a robot and people. Some social theories such as good-feeling generation principle, balance social theory, social exchange theory are applied to our model development.

2 Preliminaries

2.1 Agents

Let's first deal with the notion of intelligent agents. These are generally defined as "software entities", which assist their users and act on their behalf. Agents make your life easier, save you time, and simplify the growing complexity of the world, acting like a personal secretary, assistant, or personal advisor, who learns what you like and can anticipate what you want or need. The principle of such intelligence is practically the same of human intelligence. Through a relation of collaboration-interaction with its user, the agent is able to learn from himself, from the external world and even from other agents, and consequently act autonomously from the user, adapt itself to the multiplicity of experiences and change its behavior according to them. The possibilities offered for humans, in a world whose complexity is growing exponentially, are enormous [20,21].

We need to be careful to distinguish between rationality and omniscience. An omniscient agent knows the actual outcome of its actions, and can act accordingly; but omniscience is impossible in reality. Crossing the street was rational because most of the time the crossing would be successful, and there was no way I could have foreseen the falling door. Note that another agent that was equipped with radar for detecting falling doors or a steel cage strong enough to repel them would be more successful, but it would not be any more rational [11].

2.2 Agent Based Simulations

As these non-linear, adaptive interactions are mostly too complex to be captured by analytical expressions, computer simulations are most often used. The basic idea of such simulations is to specify the rules of behavior of individual entities, as well as the rules of their interaction, to simulate a multitude of the individual entities using a computer model, and to explore the consequences of the specified individual-level rules on the level of population as a whole, using results of simulation runs. As the simulated entities are usually called agents, the simulations of their behavior and interactions are known as agent-based simulations. The properties of individual agents describing their behavior and interactions are known as elementary properties, and the properties emerging on the higher, collective level are known as emergent properties [18].

2.3 Basic Properties of Agent-Based Models

What makes agent-based models particularly appealing and interesting is that consequences on the collective level are often neither obvious, nor expectable, even in many cases when the assumptions on individual agent properties are very simple. Namely, the capability of generating complex and intriguing emergent properties arises.

Not so much from the in-built rules of individual agent behavior, as from the complexity of the network of interactions among the agents. Precisely this

multitude of agents, as well as the multitude and complexity of their interactions, are the main reasons why in most cases formal mathematical deduction of results of an agent-based model is not possible.

This is also the reason why the issues of complexity remained relatively under-explored until recently. Namely, as scientists regularly decide to pay attention to “problems defined by the conceptual and instrumental techniques already at hand” [4], “some facts are pushed to the periphery of scientific investigation, either because they are thought not to be relevant, or because their study would demand unavailable techniques”. Accordingly, only after recent advances in the development of computational technology have enabled massive simulation experiments, the issues of emergent complexity came closer to the focus of scientific research.

Besides the above mentioned modeling of bottom-up effects, i. e. the effects originating at the individual level and influencing the collective one, more complex agent-based models are also capable of modeling top-down effects, arising at the collective level and influencing the level of individual agents[22,23].

3 Personalities

Several experiments have demonstrated that users are naturally biased to ascribe certain personality traits to machines, to PCs, and other types of media [25]. For a product designer, it is therefore important to understand how these perceptions of personality influence the interaction and how a coherent personality can be utilized in a product. Personality is an extensively studied concept in psychology. As McAdams [5] point out, there is no “comprehensive and integrative framework for understanding the whole person”. Carver [14] gives an impression of the diversity of research on personality. They present an overview of personality theories categorized along seven perspectives, including the biological, psychoanalytic, neo-analytic, learning, cognitive self-regulation, phenomenological, and dispositional perspective. In outlines, these theories agree on the general characteristics of personality, amongst others that personality is tied to the physical body; helps to determine how the person relates to the world; shows up in patterns (recurrent and consistent); and is displayed in many ways (in behavior, thoughts, and feelings). As our work concentrates on the expression of personality as a pattern of traits, personality research on dispositional traits was considered most relevant. This dispositional perspective is based on the idea that people have relatively stable qualities (or traits) that are displayed in diverse settings. Dryer [6] stresses three focus points to maintain the coherence of the characters personality: (1) cohesiveness of behavior (2) temporal stability (3) cross-situation generality. A combination of several trait theories that focused on labeling and measuring people’s personality based on the terms of everyday language (e.g. helpful, assertive, impulsive, etc.) led to an emerging consensus on the dimensions of personality in the form of the Big-Five theory. The Big-Five is currently the theory that is supported by most empirical evidence and it is generally accepted [5]. It describes personality in five

dimensions: extraversion, agreeableness, conscientiousness, neuroticism, and openness to new experiences. Recent studies have used personality theories such as the Big Five to assess people's perceptions of robot personality (e.g. [13, 30]). However, the Big-Five theory of personality can also be used as a framework to describe and design the personality of products, and in particular of robots. Norman [18] describes personality as: 'a form of conceptual model, for it channels behavior, beliefs, and intentions into a cohesive, consistent set of behaviors. Although he admits this is an oversimplification of the complex field of human personality, the statement indicates that deliberately equipping a robot with a personality, it helps to provide people with good models and a good understanding of the behavior.

Table 1. Five-factor model: dimensions and facets

Personality dimension	Personality facets
Extraversion	warmth, gregariousness, assertiveness, activity, excitement-seeking, positive emotion
Agreeableness	trust, straightforwardness, altruism, compliance, modesty, tender-mindedness
Conscientiousness	competence, order, dutifulness, achievement, striving, self-discipline, deliberation
Openness	fantasy, aesthetics, feelings, actions, ideas, values
Neuroticism	anxiety, angry hostility, depression, self-consciousness, impulsiveness, vulnerability

4 Big Five Personality Factors

Why do we study personality?

The NEO that you have just completed looks at 5 personality traits, known as the Big Five. We will briefly look at what traits are, how these personality factors were determined, what the traits mean, what the Big Five predict about our behavior, and how these factors might relate to motivation.

What are traits?

Traits are consistent patterns of thoughts, feelings, or actions that distinguish people from one another. Traits are basis tendencies that remain stable across the life span, but characteristic behavior can change considerably through adaptive processes. A trait is an internal characteristic that corresponds to an extreme position on a behavioral dimension.

There have been different theoretical perspectives in the field of personality psychology over the years including human motivation, the whole person, and individual differences. The Big Five falls under the perspective of individual differences.

How were these personality factors determined?

The Big Five represents taxonomy (classification system) of traits that some personality psychologists suggest capture the essence of individual differences in personality. These traits were arrived at through factor analysis studies. Factor is a technique generally done with the use of computers to determine meaningful relationships and patterns in behavioral data. You begin with a large number of behavioral variables. The computer finds relationships or natural connections where variables are maximally correlated with one another and minimally correlated with other variables and then groups the data accordingly. After this process has been done many times a pattern appears of relationships or certain factors that capture the essence of all of the data. Such a process was used to determine the Big Five Personality factors. Many researchers tested factors other than the Big Five and found the Big Five to be the only consistently reliable factors [9, 22, and 26].

Strict trait personality psychologists go so far as to say our behavior is really determined by these internal traits, giving the situation a small role in determining behavior. In other words, these traits lead to an individual acting a certain way in a given situation.

Allport, Norman and Cattell were influential in formulating this taxonomy which was later refined. Allport compiled a list of 4500 traits. Cattell reduced this list to 35 traits. Others continued to analyze these factors and found congruence with self-ratings, ratings by peers and ratings by psychological staff that eventually became the Big Five factors.

The Big Five factors are:

- I – extraversion vs introversion
- II – agreeableness vs antagonism
- III – conscientiousness vs undirectedness
- IV – neuroticism vs emotional stability
- V – openness to experience vs not open to experience

There was a need for an integrative framework for measuring these factors. The NEO Personality Inventory was created by Costa and McCrae and originally measured only neuroticism, extraversion and openness. The other factors were added later. There are other measures of the Big Five, such as the BFI (Big Five Inventory) and the TDA (Traits Descriptive Adjectives). The NEO has the highest validity of the Big Five measurement devices [21].

What do the five traits mean?

Keep in mind that the traits fall on a continuum and this overhead shows characteristics associated with each of the traits. Looking at these characteristics we can formulate what each of the traits mean.

- Openness - (inventive / curious vs. cautious / conservative). Appreciation for art, emotion, adventure, unusual ideas, curiosity, and variety of experience.
- Conscientiousness - (efficient / organized vs. easy-going /careless). A tendency to show self-discipline, act dutifully, and aim for achievement; planned rather than spontaneous behavior.
- Extroversion - (outgoing / energetic vs. shy / withdrawn). Energy, positive emotions, urgency, and the tendency to seek stimulation in the company of others.
- Agreeableness - (friendly / compassionate vs. competitive / outspoken). A tendency to be compassionate and cooperative rather than suspicious and antagonistic towards others.
- Neuroticism - (sensitive / nervous vs. secure /confident). A tendency to experience unpleasant emotions easily, such as anger, anxiety, depression, or vulnerability [9, 22, 26].

5 The ODD Protocol

The basic idea of the protocol is always to structure the information about an IBM in the same sequence (Fig. 1). This sequence consists of seven elements that can be grouped in three blocks: Overview, Design concepts, and Details (as a mnemonic, this sequence can be referred to as the ODD sequence). The overview consists of three elements (purpose, State variables and scales, process overview and scheduling), which provide an overview of the overall purpose and structure of the model. Readers very quickly can get an idea of the model's focus, resolution and complexity. After reading the overview it should be possible to write, in an object oriented programming language, the skeleton of a program that implements the IBM described. This skeleton includes the declaration of all objects (classes) describing the models entities (different types of individuals or environments) and the scheduling of the model's processes. The block or element "Design concepts" does not describe the model itself, but rather describes the general concepts underlying the design of the model. The purpose of this element of the protocol is to link model design to general concepts identified in the field of Complex Adaptive Systems [29]. These concepts include questions about emergence, the type interactions among individuals, whether individuals consider predictions about future conditions, or why and how stochastic city is considered. By referring to such general design concepts, each individual-based and agent-based model is integrated into the larger framework of the science of Complex Adaptive Systems. The third part of ODD, Details, includes three elements (initialization, input, sub-models) that present the details that were omitted in the overview. In particular, the submodels implementing the model's processes are described in detail. All information required to completely re-implement the model and run the baseline

simulations should be provided here. The logic behind the ODD sequence is: context and general information is provided first (Overview), followed by more strategic considerations (Design concepts), and finally more technical details (Details). We can help readers understand our IBMs by always using this structure: a standard protocol that provides the information in an order that allows the reader to easily build on their previous understanding. Below, the seven elements of ODD are described.

We conclude that what we badly need is a standard protocol for describing IBMs which combines two elements: (1) a general structure for describing IBMs, thereby making a model's description independent of its specific structure, purpose and form of implementation [24] and (2) the language of mathematics, thereby clearly separating verbal considerations from a mathematical description of the equations, rules, and schedules that constitute the model. Such a protocol could, once widely used, guide both readers and writers of IBMs. In this article we propose a standard protocol for describing IBMs (including agent-based models, multi-agent simulation, or multi-agent systems; see Discussion). The basic idea of the protocol was proposed by [28] and then discussed during an international workshop on individual-based modeling held in Bergen, Norway, in the spring of 2004. Most participants of that workshop are among the authors of this article, leading to 28 authors from seven different countries. The work of the authors covers a wide range of fields within ecology (e.g., marine, terrestrial, plant, animal, behavior, population, forest, theory, conservation, etc.), and the authors have altogether been involved in the writing of more than 200 IBM-based papers. We agreed to test and refine the standard protocol proposed by [28] by applying it to our own models: every author, or team of co-authors, rewrote one of their existing model descriptions using the new standard protocol. The set of 19 models used in this test differs widely in scope, structure, complexity, and implementation details. As a result of the test applications, the protocol was slightly revised.

Here, we first present the standard protocol, which [7] refer to as the PSPC + 3 protocols. The abbreviation "PSPC" referred to the initials of first four elements of the protocol (purpose, structure, process, concepts) and "+3" referred to the remaining three elements. In the revised protocol, however, the names of some elements have been changed. We are therefore using a new acronym, "ODD", which stands for the three blocks of elements 'Overview', 'Design concepts', and 'Details' (fig. 1). Then we present an example application of the protocol, and summarize our experience with test applications in a list of frequently asked questions which provides practical hints for using the protocol. Finally we discuss both our experience with the test applications and ODD's potentials and limitations and how it could contribute to further unification of the formulation and implementation of IBMs.

Overview	Purpose
	State variables and scales
	Process overview and scheduling
Design concepts	Design concepts
Details	Initialization
	Input
	Submodels

Fig. 1. The seven elements of the ODD protocol, which can be grouped into the three locks: Overview, Design concepts, and Details

6 What Is Fuzzy Logic?

In this context, Fuzzy Logic (FL) is a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data acquisition and control systems. It can be implemented in hardware, software, or a combination of both. FL provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information. FL's approach to control problems mimics how a person would make decisions, only much faster [9].

FL incorporates a simple, rule-based IF X AND Y THEN Z approach to a solving control problem rather than attempting to model a system mathematically. The FL model is empirically-based, relying on an operator's experience rather than their technical understanding of the system. For example, rather than dealing with temperature control in terms such as "SP =500F", "T <1000F", or "210C <TEMP <220C", terms like "IF (process is too cool) AND (process is getting colder) THEN (add heat to the process)" or "IF (process is too hot) AND (process is heating rapidly) THEN (cool the process quickly)" are used. These terms are imprecise and yet very descriptive of what must actually happen. Consider what you do in the shower if the temperature is too cold: you will make the water comfortable very quickly with little trouble. FL is capable of mimicking this type of behavior but at very high rate [28].

7 Proposed Method

Following is the proposed architecture of the SMA-personality, which is charged with working the personality of the robot, ODD protocol is used and the theory of the Big five, to define the personality, the SMA-personality, to communicate with

diffuse SMA and control access to the values of SMA-personality, are the data obtained by the AN, and with them assess their development.

7.1 Design Concepts

Emergency: Dynamics of the population to leave the conduct of (the robot), but the individual's life cycle and behavior are fully represented by empirical rules.

For this to work, each player creates an SMA (robot), which consists of two agents, an agent that is responsible for controlling the engine and the last agent to act as coordinating multi-agent system [1,2,3,5].

We can describe the proposed method as follows:

In the operation of the model, we have 2 main blocks, which are responsible for knowledge and learning (paradigm of intelligent agents) and the vision and control (fuzzy logic).

These modulus are described below, the first module contains 3 agents, NA [Node Agent], TA [Task Agent] SA [System Agent] Agent System (AS), and will know every time the operation of the other agents.

To detect a change in the environment the Agent Node [that is in charge of the sensors (ultrasonic, camera and two light sensors)] with the ultrasonic sensor and two light sensors, starts its operation, and we start at the state called reactive control; this because you have to move and sense all the time until it finds an obstacle, this can happen once a photo is taken, which will be sent to the database of images (BDI), the image will be applied a pre-processing for recognition in order to know whether the object is found, this decision was taken on the angle (angle to take the decision to bypass the object), able to escape and move forward, trend data, they are caught in the trajectory control process, to observe the speed values. The Task Agent (which is responsible for the drive), once all the necessary parameters for the performance of the robot agent system (AS) who is the coordinator, and executes instructions in the robot.

For the development of the Multi-Agent System we used Gaia [29] and UML for the analysis and design of the agents. The completion of the Multi-Agent System (MAS) is based on the FIPA [10] standards for better utilization and greater possibility of extension.

7.2 SMA-Personality Architecture

Following is the proposed architecture of the SMA-personality, which is charged with working the personality of the robot, ODD protocol is used and the theory of the Big five, to define the personality, the SMA-personality, to communicate with diffuse SMA and control access to the values of SMA-personality, the personality will be measured in diffuse, so be empowered to have values closer to reality, is data obtained by the AN, and with them assesses its development.

The values given will be personality in the role of personality characteristic, and these values are:

At work with big five

Opening

- More inventive, less inventive, no inventiveness,
- More interesting, less funny, anything funny
- More cautious, less cautious, without caution
- More conservative, less conservative, nothing conservative

Conscientiousness

- Very efficient, less efficient, no efficient
- More organized, poorly organized, nothing organized
- Very sloppy, little careless, without caution
- More conservative, less conservative, nothing conservative

Extraversion

- Very energetic, less energetic, anything energetic,
- Very shy, somewhat shy, not shy

Kindness

- Very friendly, unfriendly, unkind,
- Very compassionate, unkind, unsympathetic
- Very cooperative, uncooperative, no operation

Neuroticism

- Very sensitive, very sensitive, no sensitive
- Very nervous, a little nervous, not nervous
- Much security, poor security, no insurance
- Very anxious, somewhat anxious, nothing anxious
- Very depressing, slightly depressed, nothing depressing.

Personality is allocated according to what you want in a home, according to the agent can learn to take some of the factors and have a different personality.

Below is the outline of the architecture of personality, which is composed by 2 agents, agent personality and consciousness, as well as several modules, such as a knowledge base that BC, contains a knowledge base and grow from According the data of the other process of the SMA, will also be a module of thought generated, which will take care of having the information generated from the consciousness of thought, emotion module, will have a set of basic emotions, the module of feeling that will have a set of basic elements, fig 2.

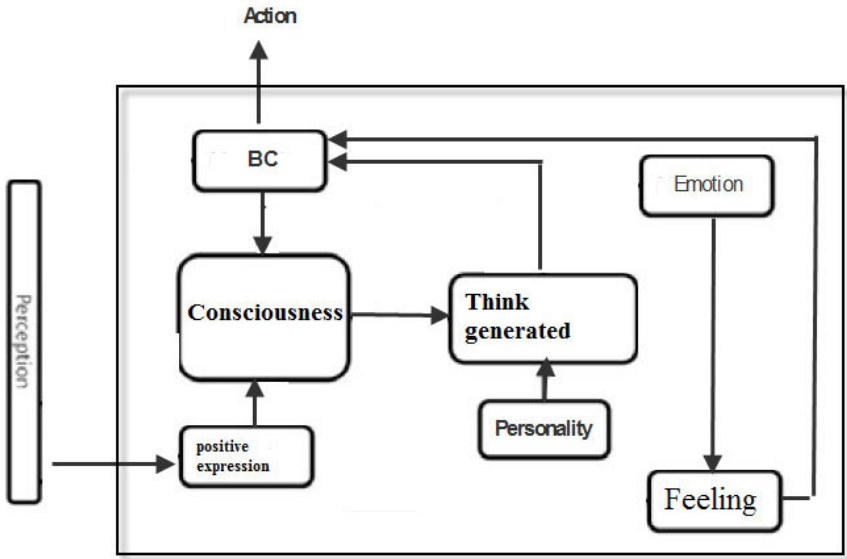


Fig. 2. SMA-Personality

8 Results

We show in Figure 3 the plant and the controller that we are used for achieving the simulation in Simulink of Mat lab.

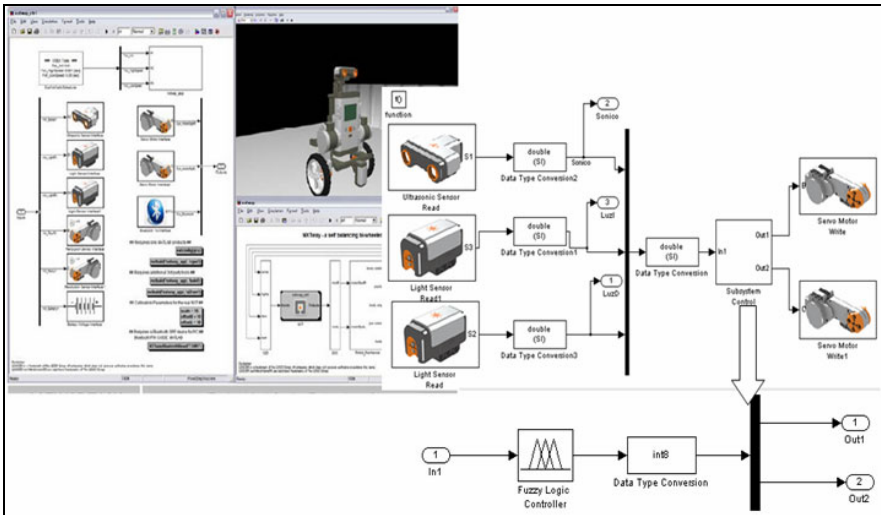


Fig. 3. Plant for simulation

Together with some fuzzy systems which were developed based on work taken as a reference [16,17] and that were the basis for the experimentation with different numbers of rules, parameters of membership functions, and types of membership functions was a good performance of our robot, achieving our goal.

In figure 4 we show the results presented above for a better understanding, which can be seen in the upper part the time it takes for each experiment out of the maze at the bottom and the approximate distance traveled to each robot in the experiment, the red dots are the results of our work, the blue dots are the results of the reference points and finally green means yes or no out of the labyrinth in this experiment.

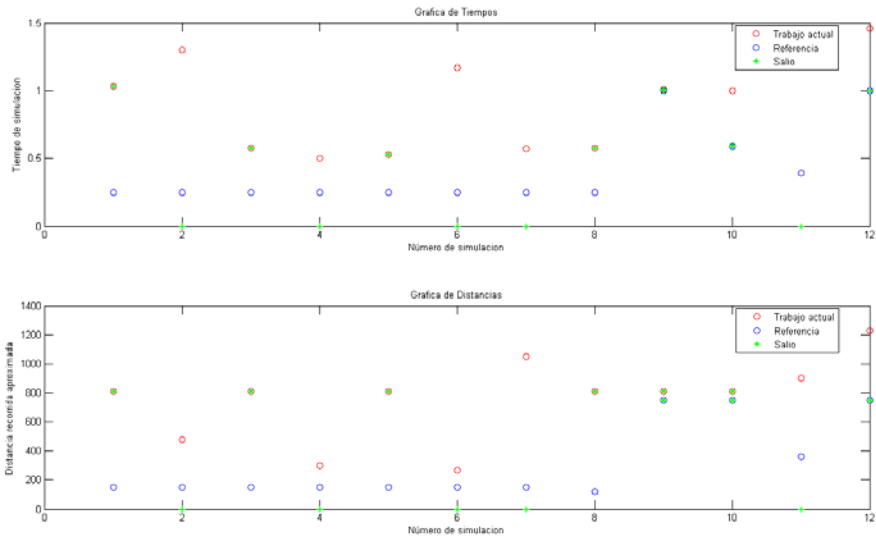


Fig. 4. Comparison between the work of references [13][14] and the current job

There are occasions when our results are very similar and therefore only seems to be the green dot that tells us whether or not we go out of the labyrinth that is the case of experiment number nine in the graph of time in which both works coincided in time output and both were successful in finding the exit of the maze.

Now we show other experiments with a fuzzy system with different number of fuzzy rules, membership functions and parameters of membership functions. It is noteworthy that although this is not a fuzzy system which is optimized for this any optimization method can be used.

The following fuzzy system in Fig.5 has three inputs, two outputs and consists of ten rules to make the inference.

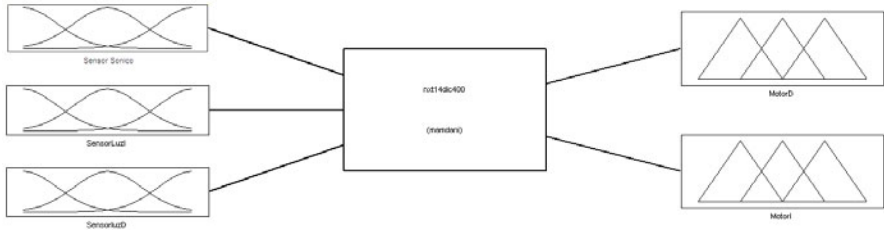


Fig. 5. Shows the fuzzy system that works for the simulations

The first input variable of the fuzzy system is the ultrasonic sensor which has three membership functions which are linguistic (close, near, far), as shown in Fig. 6.

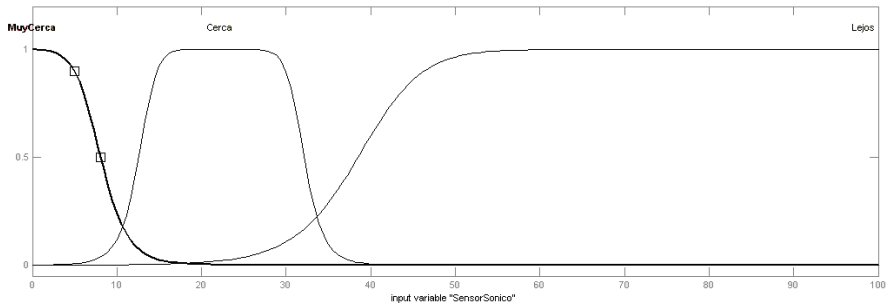


Fig. 6. The ultrasonic sensor

The second variable of the fuzzy system is the light sensor that has two functions that are membership free and wall, as shown in Fig. 7.

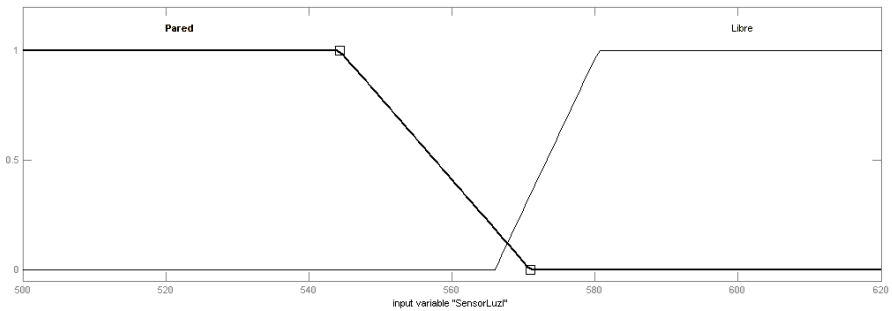


Fig. 7. The left light sensor

The third variable of the fuzzy system is the light sensor that has two functions that are membership free and wall, as shown in Fig. 8.

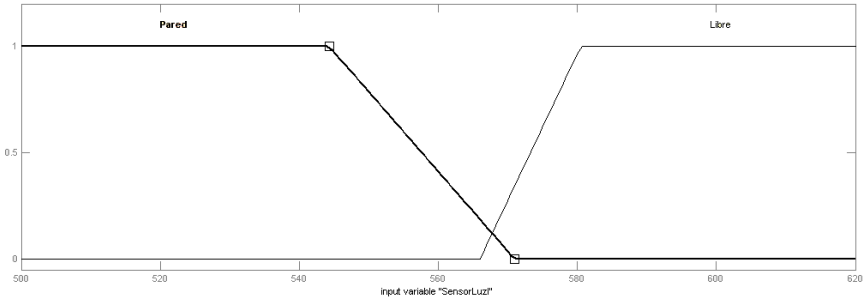


Fig. 8. The right light sensor

In Fig. 9 we show the rules that are used in the fuzzy system, which are 10 fuzzy rules.

1. If (SensorSónico is Lejos) and (SensorLuz is Pared) and (SensorLuzD is Pared) then (MotorD is AdelanteR)(MotorI is AdelanteR) (1)
2. If (SensorSónico is Cerca) and (SensorLuz is Pared) and (SensorLuzD is Pared) then (MotorD is AdelanteL)(MotorI is AdelanteL) (1)
3. If (SensorSónico is MuyCerca) and (SensorLuz is Pared) and (SensorLuzD is Pared) then (MotorD is AdelanteL)(MotorI is AdelanteL) (1)
4. If (SensorSónico is Lejos) and (SensorLuz is Libre) and (SensorLuzD is Pared) then (MotorD is AdelanteR)(MotorI is AtrasL) (1)
5. If (SensorSónico is Cerca) and (SensorLuz is Pared) and (SensorLuzD is Libre) then (MotorD is AtrasL)(MotorI is AdelanteR) (1)
6. If (SensorSónico is MuyCerca) and (SensorLuz is Pared) and (SensorLuzD is Libre) then (MotorD is AtrasL)(MotorI is AdelanteR) (1)
7. If (SensorSónico is Lejos) and (SensorLuz is Pared) and (SensorLuzD is Libre) then (MotorD is AtrasL)(MotorI is AdelanteR) (1)
8. If (SensorSónico is Cerca) and (SensorLuz is Libre) and (SensorLuzD is Libre) then (MotorD is AdelanteR)(MotorI is Cero) (1)
9. If (SensorSónico is Lejos) and (SensorLuzD is Pared) then (MotorD is AdelanteL)(MotorI is AdelanteL) (1)
10. If (SensorSónico is MuyCerca) and (SensorLuz is Libre) and (SensorLuzD is Pared) then (MotorD is AdelanteR)(MotorI is AtrasL) (1)

Fig. 9. Rules of the fuzzy system (FIS)

Having modified and improved the fuzzy systems, we proceeded with a slight modification to our Multi-Agent System, which consists in adding a new agent which has the task of giving a second option in case our robot gets stuck somewhere in the world, for example corners in which our sensors cannot do some action, then come into action. The new agent called Support Agent enters into action as I mentioned in the case that the sensor values do not become active some rule of fuzzy systems that we throw the speed of the drive to get out of where we are stuck with only works with the task agent (TA) for which our multi-agent system would be as shown in Fig. 10.

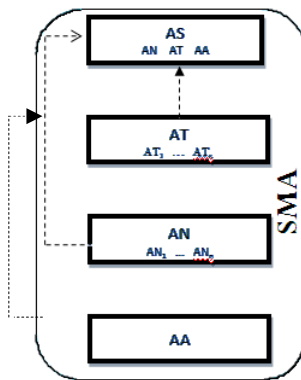


Fig. 10. New scheme of Multi-Agent System

Integrating it to our general scheme we would be as shown in Fig 11.

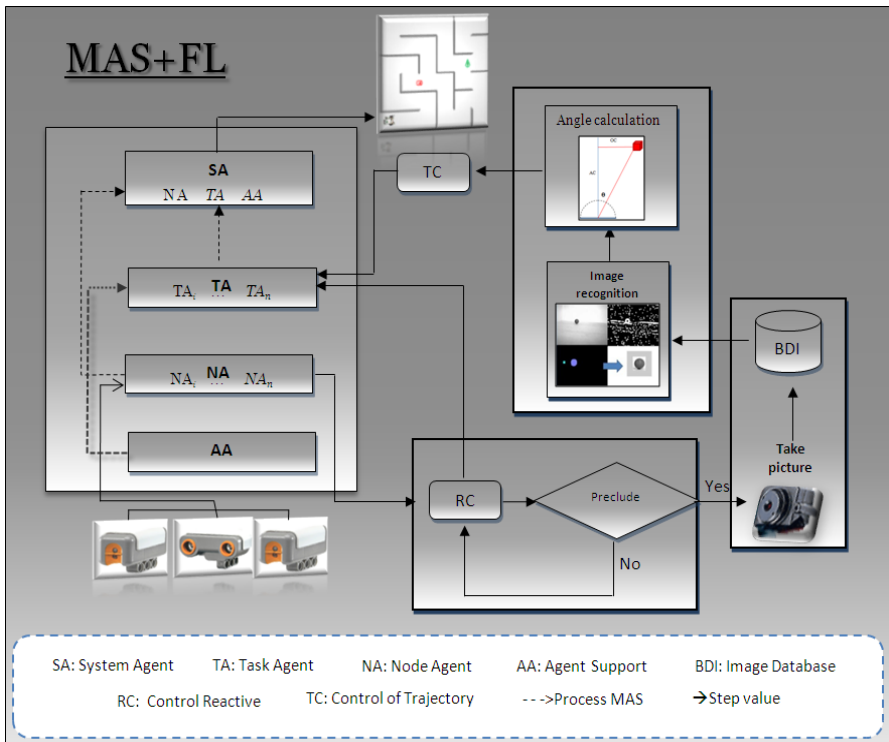


Fig. 11. New architecture of the complete system [5]

Now with these changes we had better control of movements of the robot in the search for the exit of the maze, which can be seen in the graphs of the following experiments, fig. 12, 13, and 14.

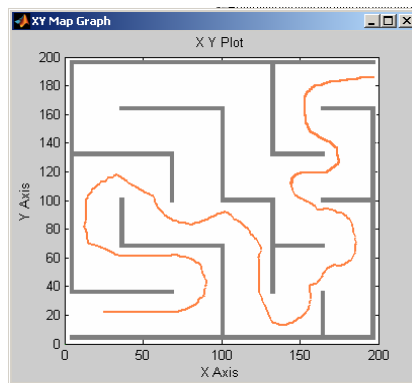


Fig. 12. Simulation-1 duration of 00:50 minutes

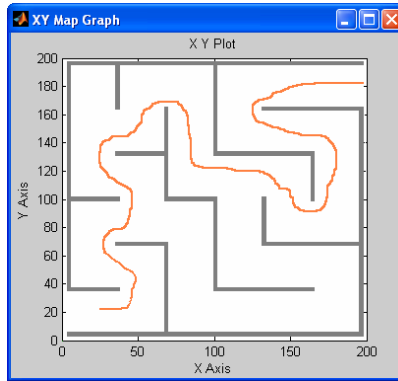


Fig. 13. Simulation-2 duration of 00:45 minutes

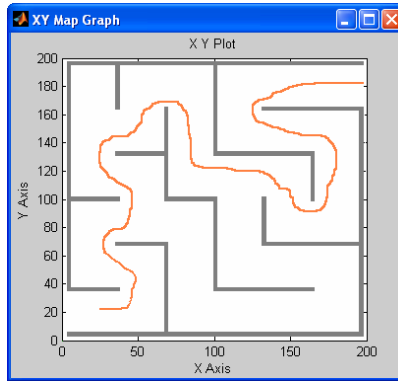


Fig. 14. Simulation-3 duration of 00:45 minutes

9 Conclusions

The development of intelligent applications is varied, in this particular case, the implementation of multi-agent systems is complex because it consists of several agents, whit the incorporation of other techniques, making it even more complex.

One of the main objectives of this investigation is to open a new area of intelligent agents backed by human psychology, and with the support of fuzzy logic, which hitherto has not been the approach intended here, which gives a degree of utilization of this paradigm of intelligent agents, but more research is needed to defend this innovative idea.

Given the non-optimized fuzzy systems, we have shown better results than were obtained in the work taken as reference [5]. From what we have achieved, we can say that a good percentage of the objectives in the area of control and learning paths. Future work consists in testing the SMA-personality, adding fuzzy

logic to the SMA, resulting in a fuzzy SMA with broader personality, both for decision making, and the personality.

References

- [1] Alanis, A.: Contribution to the design of Fault Tolerant Distributed Systems for industrial control: proposal for a new paradigm based on Intelligent Agents, PhD, Universidad Politecnica de Valencia (November 2007)
- [2] Alanis, A., López, M., Parra, B., Serrano, M., Ayala, E., Solano, C.: Multi-agent system for search and object recognition using vision, in a Lego NXT robot XXIX. In: 1 International Congress of Engineering in electronics, electro 2008 Instituto Tecnológico de Chihuahua, division of graduate studies and research, Chihuahua, Chih. Mexico, October 29, 31 (2008)
- [3] Alanis, A., López, M., Parra, B., Serrano, M., Ayala, E., Solano, C.: Multi-agent system for search and object recognition using vision, in a Lego NXT robot 1.8 °. In: National Congress of Electrical and Electronics Engineering of the Mayab conieem 2008 Instituto Tecnológico de Mérida, of Merida, Yucatan, Mexico April 21-25 (2008)
- [4] Azam, F.: Biologically Inspired Modular Neural Networks, Electrical and Computer Engineering, Blacksburg, Virginia (2000)
- [5] Carver, C., Scheier, M.: Perspectives on personality. Allyn and Bacon, Boston (1995)
- [6] Dryer, D.C.: Getting personal with computers: How to design personalities for agents. *Applied Artificial Intelligence* 13(3), 273–295 (1999)
- [7] Grimm, V.: Visual debugging: a way of analyzing, understanding, and communicating bottom-up simulation models in ecology. *Nat. Res. Model.* 15, 23–38 (2002)
- [8] Grimm, V., Railsback, S.F.: *Individual-Based Modeling and Ecology*. Princeton University Press, Princeton (2005)
- [9] Hogan, R., Johnson, J., Briggs, S. (eds.): *Handbook of personality psychology*. Academic Press, California (1997) 9
- [10] FIPA is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies, <http://www.fipa.org/>
- [11] David, H.: Japan starts Fuzzy Logic II. *Machine Design*, 16 (September 10, 1992)
- [12] Jang, S.J., Mizutani, E.: *Neuro-Fuzzy and soft computing: a computational approach to learning and machine intelligence*. Prentice-Hall, Englewood Cliffs (1997)
- [13] Kiesler, S.B., Goetz, J.: Mental models of robotic assistants. *CHI Extended Abstracts*, 576–577 (2002)
- [14] McAdams, D., Pals, J.: A New Big Five: Fundamental Principles for an Integrative Science of Personality. *American Psychologist* 61(3), 204–217 (2006)
- [15] Melendez, A., Castillo, O., y Soria, J.: Reactive Control of a Mobile Robot in a Distributed Environment Using Fuzzy Logic. In: *NAFIS 2008* (2008)
- [16] Melendez, A.: Control and monitoring reagent for a mobile robot using fuzzy logic, thesis work, ITT (August 2008)
- [17] Morgan, D.P., Scofield, C.L.: *Neural Networks and Speech Processing*. Kluwer Academic Publishers, Dordrecht (1991)

- [18] Norman, D.: How humans might interact with Robots (2001), http://www.jnd.org/dn.mss/how_might_human.htm (retrieved September 4, 2008)
- [19] Norving, P., Russell, S.: Artificial intelligence a modern approach. Prentice Hall, Australia (1996)
- [20] Cohen, P.R., et al.: An Open Agent Architecture. In: working Notes of the AAAI Spring symp. Software Agent, pp. 1–8. AAAI Press, Cambridge (1994)
- [21] Pervin, L., John, O. (eds.): Handbook of personality: theory and research. Gilford, New York (1999)
- [22] Pitt, W.C., Box, P.W., Knowlton, F.F.: An individual-based model of canid populations: modelling territoriality and social structure. *Ecol. Model.* 166, 109–121 (2003)
- [23] Potkay, C., Allen, B.: Personality: theory, research, and applications. Brooks/Cole, California (1986)
- [24] Railsback, S.F.: Concepts from complex adaptive systems as a framework for individual-based modeling. *Ecol. Model.* 139, 47–62 (2001)
- [25] Reeves, B., Nass, C.: The media equation: How people treat computers, televisions, and new media like real people and places. Cambridge University Press, New York (1996)
- [26] Russell, S., Norvig, P.: Intelligent Agent. In: Artificial Intelligence to Modern Approach, Prentice Hall series in intelligence., pp. 31–52 (1994)
- [27] Aragon, S.C.: Multi-agent system of fuzzy control for autonomous mobile robots, Masters Dissertation, Instituto Tecnológico de Tijuana, Mexico (September 2009)
- [28] University of Vigo, Department of Computer Science, Problems of search and optimization (October 2004)
- [29] Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G.: A standard protocol for describing individual-based and agent-based, September 15, vol. 198(1-2), pp. 115–126 (2006)
- [30] Walters, M.L., Syrdal, D.S., Dautenhahn, K., Te Boekhorst, R., Koay, K.L.: Avoiding the uncanny valley: robot appearance, personality and consistency of behavior in an attention-seeking home scenario for a robot companion. *Autonomous Robots* 24(2) (2008)

Composite Recurrent Neural Networks for Long-Term Prediction of Highly-Dynamic Time Series Supported by Wavelet Decomposition

Pilar Gomez-Gil¹, Angel Garcia-Pedrero¹, and Juan Manuel Ramirez-Cortes²

¹ Department of Computational Science
pgomez@acm.org, agarciapedrero@gmail.com

² Department of Electronics,
National Institute of Astrophysics, Optics and Electronics,
Luis Enrique Erro No. 1 Tonantzintla, Puebla, 72840. Mexico
jmramirez@ieee.org

Abstract. Even though it is known that chaotic time series cannot be accurately predicted, there is a need to forecast their behavior in many decision processes. Therefore several non-linear prediction strategies have been developed, many of them based on soft computing. In this chapter we present a new neural network architecture, called Hybrid and based-on-Wavelet-Reconstructions Network (HWRN), which is able to perform recursive long-term prediction over highly dynamic and chaotic time series. HWRN is based on recurrent neural networks embedded in a two-layer neural structure, using as a learning aid, signals generated by wavelets coefficients obtained from the training time series. In the results reported here, HWRN was able to predict better than a feed-forward neural network and that a fully-connected, recurrent neural network with similar number of nodes. Using the benchmark known as NN5, which contains chaotic time series, HWRN obtained in average a SMAPE = 26% compared to a SMAPE = 61% obtained by a fully-connected recurrent neural network and a SMAPE = 49% obtained by a feed forward network.

1 Introduction

The use of long-term prediction as a tool for complex decision processes involving dynamical systems has been of high interest for researchers in the last years. Some current prediction strategies approximate a model of the unknown dynamical system analyzing information contained in a solely time-series, which is supposed to describe the system's behavior. A time series may be defined as an ordered sequence of values observed from a measurable phenomena: x_1, x_2, \dots, x_n ; such observations are sensed at uniform time intervals and may be represented as integer or real numbers [25]. Once defined, an approximated model may be used to predict the trend of the system behavior or to predict as much specific values of the time series as desired. As usual, such model will be just as good as the information used

to construct it and as the capability of the modeler to represent important information embedded in the time series being analyzed.

Time series prediction consists on estimating future values of a time series $x_{t+1}, x_{t+2}..$ using past time series values $x_1, x_2...x_t$. One-step or short-term prediction occurs when several past values are used to predict the next unknown value of the time series. If no exogenous variables are considered, one-step prediction may be defined as [19]:

$$\bar{x}_{t+1} = \phi(x_{t-1}, x_{t-2}...x_{t-p}) \quad (1)$$

where ϕ is a approximation function used to predict. Similarly, long term prediction may be defined as:

$$\bar{x}_{t+h}... \bar{x}_{t+2}, \bar{x}_{t+1} = \Phi(x_{t-1}, x_{t-2}...x_{t-p}) \quad (2)$$

where h denotes the prediction time horizon, that is, the number of future values to be obtained by the predictor at once. Long term prediction may also be achieved by recursive prediction, which consists of recursively using equation (1) by feeding back past predicted values to the predictor to calculate the new ones.

The construction of models able to predict highly nonlinear or chaotic time series is of particular interest in this research. A chaotic time series is non-stationary, extremely sensitive to initial conditions of the system and contains at least one positive Lyapunov Exponent [15]. It is claimed that chaotic time series may only be short-term predicted [20]. Even though, in some cases it is possible to approximate a dynamical model with similar characteristics to that found in the non-linear time series and to use it for long-term prediction. There are many techniques used to build predictors; they may be linear or non-linear, statistical or based on computational or artificial intelligence. For example, ARMA, ARIMA and Kalman filters are linear methods [21]; k-nearest neighbors, genetic algorithms and artificial neural networks are examples of non-linear methods. Only non-linear methods are useful to forecast non-linear time series.

The use of fully-connected, recurrent neural networks for long-term prediction of highly-dynamical or chaotic time series has been deeply studied [23]. In spite of the powerful capabilities of these models to represent dynamical systems, their practical use is still limited, due to constraints found in defining an optimal number of hidden nodes for the network and the long time required to train such networks. As a way to tackle these problems, complex architectures with a reduced number of connections, better learning abilities and special training strategies have been developed [13]; examples of such works are found at [2,3, 4,10,11,13,15,25,26, 29, 30,31] among others. From the vast number of strategies used to improve the long term prediction ability of neural networks, Wavelet Theory is used either to modify neuron architectures (for example [2,6,12,31,33]) or as a pre-processing aid applied to training data (for example [11,26,28,29]). When wavelet theory is used to modify the neuron architecture, normally it is done using

a wavelet function as the activation function [33]. Other works combine wavelet decomposition (as a filtering step) and neural networks to provide an acceptable prediction value [11,29].

In this chapter we present a novel neural prediction system called HWRN (**H**ybrid and based-on-**W**avelet-**R**econstructions **N**etwork). HWRN is based on recurrent neural networks, inspired at the Hybrid complex neural network [15] and with a particular kind of architecture and training scheme supported by wavelet decomposition. In the experiments reported here, HWRN was able to learn and predict as far as 56 points of two highly-dynamical time series, obtaining better performance than a fully-connected recurrent neural network and a three-layer, feed-forward neural network with similar number of nodes than the HWRN. This chapter is organized as follows: section two describes the main characteristics, general structure and training scheme of the model. In the same section some details are given related to reconstruction of some signals that are used for supporting training, which is based on discrete wavelet transforms. Criteria used to evaluate the performance of the system are presented at section three. Section four describes the experiments performed and their results; it also includes a description of the time series used to evaluate the model. Last section presents some conclusions and ongoing work.

2 Model Description

HWRN is built using several small, fully-connected, recurrent neural networks (SRNN) attached to a recurrent layer and an output layer. Figure 1 shows the general architecture of HWRN. The SRNN are used to learn signals obtained from the training time series that contain different frequency-time information. Outputs of the SRNN are fed to a recurrent layer, which is able to memorize time information of the dynamical system. The last layer acts as a function approximator builder.

The output of each node i at HWRN and SRNN is defined as:

$$\frac{dy_i}{dt} = -y_i + \sigma(x_i) + I_i \quad (3)$$

where:

$$x_i = \sum_{j=1}^m y_j w_{ji} \quad (4)$$

represents the inputs to the i -th neuron coming from other m neurons,

I_i is an external input to i -th neuron,

w_{ji} is the weight connecting neuron i to neuron j ,

$\sigma(x)$ is the node's transfer function; it is a sigmoid for all layers except output layer, for which transfer function is linear.

In order to be solved, equation 3 may be approximated as [27]:

$$y_i(t + \Delta t) = (1 - \Delta t)y_i(t) + \Delta t\sigma(x_i(t)) + \Delta tI_i(t) \quad (5)$$

for a small Δt , where:

$$x_i(t) = \sum_{j=1}^m y_j(t)w_{ji} \quad (6)$$

For the results reported here, initial conditions of each node $y_i(t=0)$, are set as small random values. Indeed, there are no external inputs to nodes, that is $I_i(t) = 0$ for all i , all t .

Training of a HWRN predictor contains three main phases:

1. Pre-processing of the training time series and generation of reconstructed signals,
2. Training of the SRNN,
3. Training of the HWRN.

After being trained, HWRN receives as input k past values of a scaled time series, then recurrent prediction is applied to obtain as many futures values as required. Each training phase is described next.

2.1 Phase 1: Preprocessing

HWRN requires a time series with enough information of the dynamical behavior in order to be trained. Such time series may contain integer or real values and the magnitude of each element must be scaled to the interval $[0,1]$. This is required in order to use sigmoid transfer functions for the nodes in the network. To achieve this, the time series may be normalized or linearly scaled; in this research a linear scale transformation was applied, as recommended for financial time series by [7]. The linear transformation is defined as:

$$z_t = lb + \frac{x_t - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})}(ub - lb) \quad (7)$$

where:

- ub is the desired upper bound; in this case $ub = 1$,
- lb is the desired lower bound; in this case $lb = 0$,
- $\max(\mathbf{x})$ is the maximum value found at the time series,
- $\min(\mathbf{x})$ is the minimum value found at the time series.

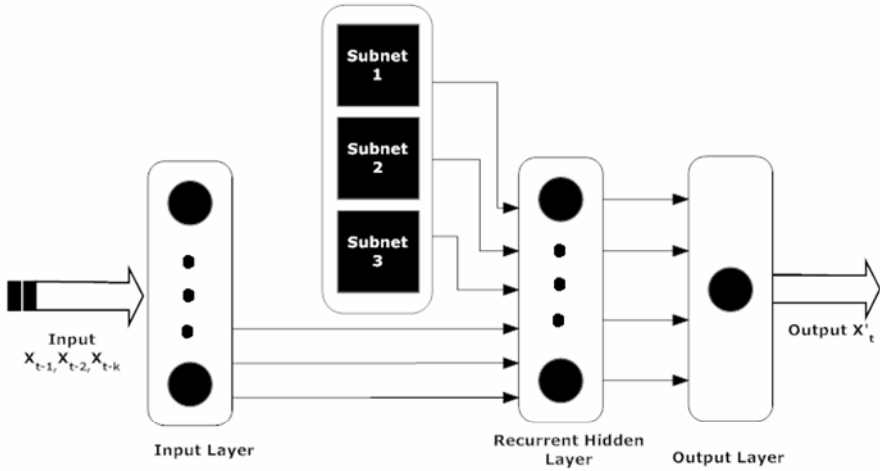


Fig. 1. A Hybrid and based-on-Wavelet-Reconstructions Network HWRN (adapted from [12])

If the original times series has missing values, they are approximated as the mean of their two nearest neighbors. No further processing is applied.

An important challenge forecasting nonlinear and chaotic time series is the complexity found to represent its non-stationary characteristics. To tackle this, the HWRN learns frequency information related to different times using different components. It is known that wavelet analysis has been used to represent local frequency information in a signal. Such analysis calculates the correlation among a signal and a function $\psi(\cdot)$, called wavelet function. Similarity among both functions is calculated for different time intervals, getting a two dimensional representation: time and frequency [1]. In this work, a multi-scale decomposition of the training signal is performed using the sub-band coding algorithm of the Discrete Wavelet Transform [22]. This algorithm uses a filter bank to analyze a discrete signal $x(t)$. This bank is made of low-pass $L(z)$ and high-pass $H(z)$ filters, separating frequency content of the input signal in spectral bands of equal width. Figure 2 shows a one-level filter bank. After performing a down-sampling with a factor of two, signals $cA(t)$ and $cD(t)$ are obtained. These signals are known as approximation and detail coefficients, respectively. This process may be executed iteratively forming a wavelet decomposition tree up to any desired resolution level. A three-level decomposition wavelet tree, used for the experiments presented in this paper, is shown in Figure 3. The original signal $x(t)$ may be reconstructed back using the Inverse Discrete Wavelet Transform (iDWT), adding up the outputs of synthesis filters. Similarly it is possible to reconstruct not only the original signal, but also approximation signals that contain low-frequency information of the original signal and therefore more information about long-term behavior. In the same way, detail signals can be reconstructed; they contain information about short-term changes in the original signal. Using the decomposition wavelet tree at figure 3, four different signals may be reconstructed (one approximation and three

detail signals) using the coefficients shown at the leaves of such tree. For the rest of this chapter, these signals are referred as “reconstructed signals.”

For example, figure 4(a) shows a chaotic time series called NN5-101 (see section 4); figure 4(b) shows its most general approximation obtained using coefficients cA_3 (see figure 3); figure 4(c) shows the most general detail signal obtained using coefficients cD_3 ; figure 4(d) shows detail signal at level 2 obtained using coefficients cD_2 ; figure 4(e) shows detail signal at maximum level obtained using coefficients cD_1 .

During the predictor training, a set of these reconstructed signals is selected and independently learned by a set of SRNN. In order to figure out which reconstructed signals contain the most important information, all possible combinations of reconstructed signals are created; next, signals in each combination are added up and the result is compared with the original signal using Mean Square Error (see equation 8). The reconstructed signals in the combination with the smallest MSE are selected to be learnt by the SRNN.

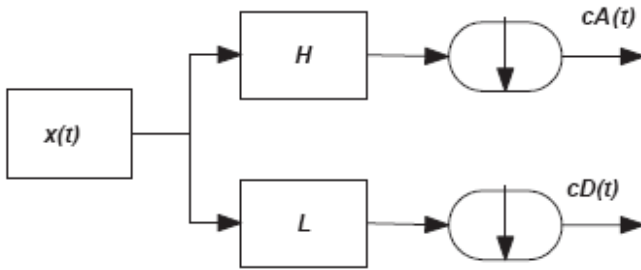


Fig. 2. An analysis filter bank

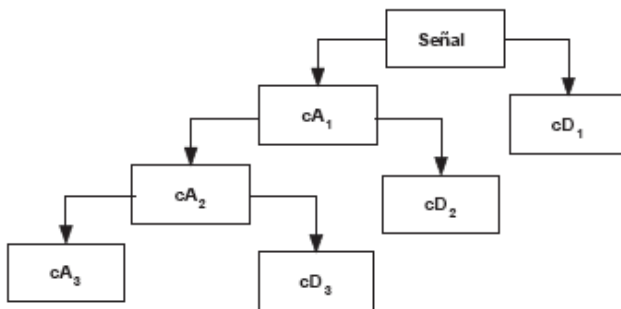


Fig. 3. A three-level decomposition wavelet tree

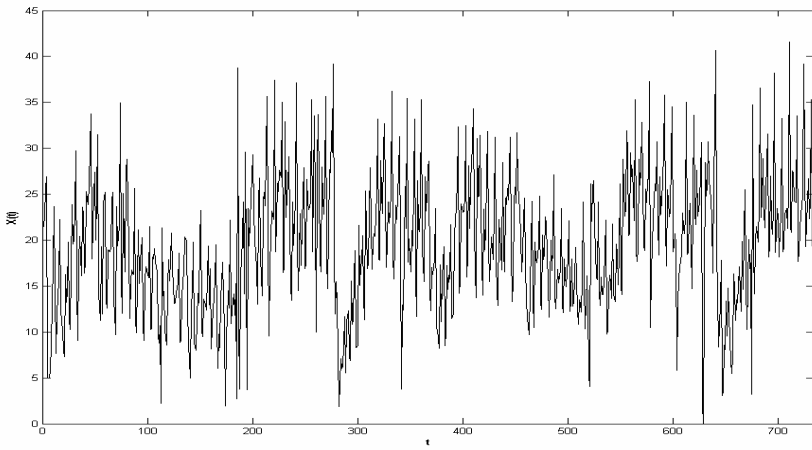


Fig. 4. (a) Original signal NN5-101 (data taken from [7])

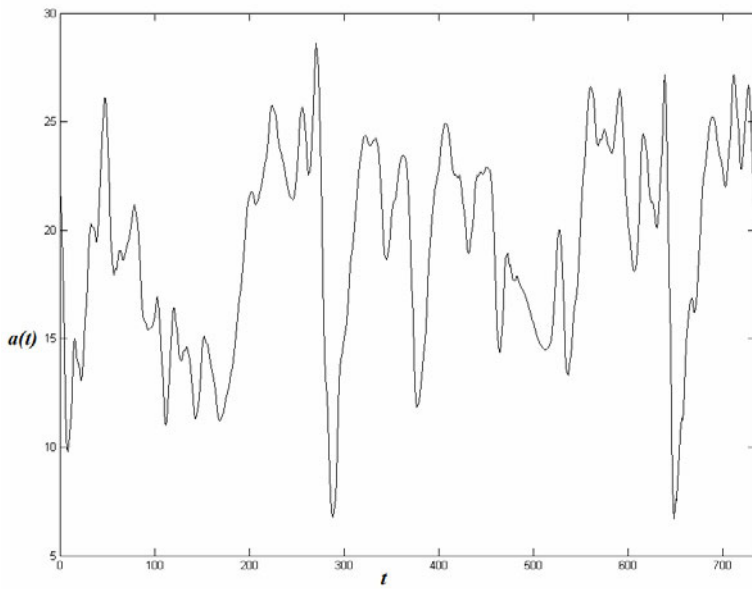


Fig. 4. (b) Most general approximation signal obtained from NN5-101

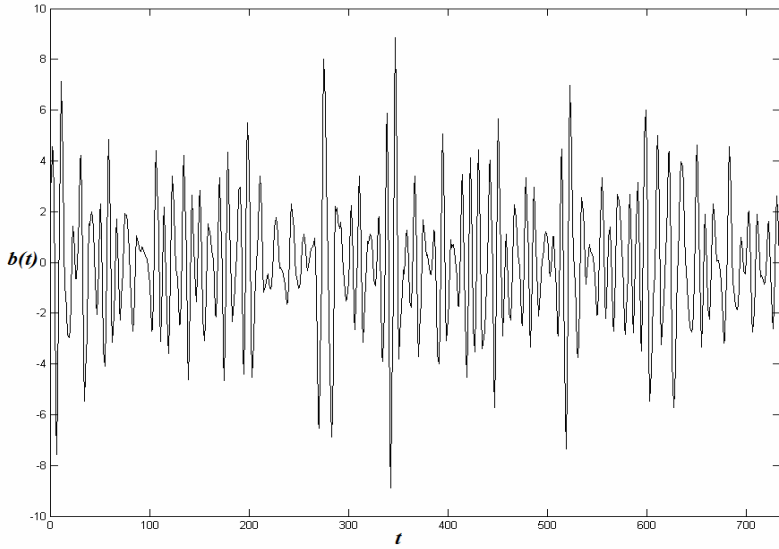


Fig. 4. (c) Most general detail signal obtained from NN5-101

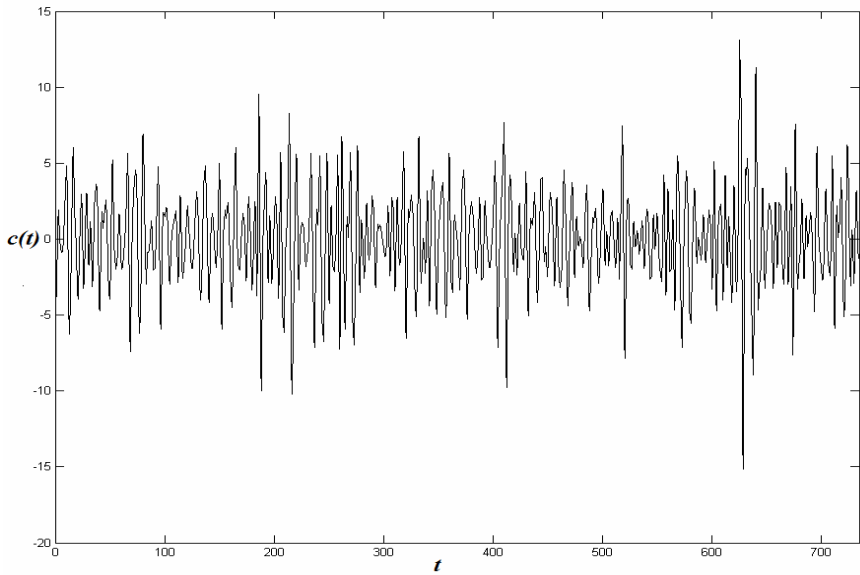


Fig. 4. (d) Detail signal at level 2 obtained from NN5-101

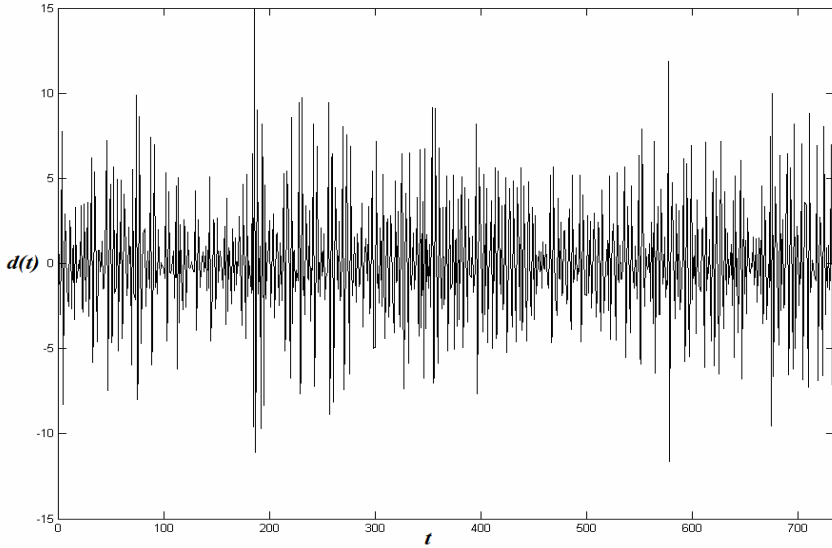


Fig. 4. (e) Detail signal at maximum level obtained from NN5-101

2.2 Phase 2: Training the SRNN

SRNN are trained to predict one point in each selected reconstructed signal; they receive as input k values of the corresponding reconstructed signal and predict the next one. Once trained, the SRNN require only the first k values of the reconstructed signal; the rest values are generated using recursive prediction as long as the predictor works. These k values are stored as free parameters of the system, to use them when prediction of the time series is taking place.

Training of all SRNN is performed using the algorithm “*Real-time real-learning based on extended Kalman filter (RTRL-EKF)*” [16]. This algorithm contains 2 parts: gradient estimation and weights adjustment. The first part is done using the Real-Time, Real-Learning Algorithm proposed by Williams and Zipser [32]; second part is done using an extended Kalman Filter. RTRL-EKF has a complexity of $O(n^4)$, where n is the number of neurons in the neural network [12].

2.3 Phase 3: Training the HWRN

After training all SRNN, their weights are imbedded in the architecture of the HWRN (see figure 1) which also contains a hidden layer with recurrent connections and an output layer with feed-forward connections. The complete architecture is trained to predict one point of the original signal, keeping fixed the weights of sub-networks SRNN. As in the case of SRNN, training is performed using “*Real-time real-learning based on extended Kalman filter (RTRLEKF) algorithm*” [16].

3 Metrics for Performance Evaluation

The prediction ability of the proposed architecture and comparative models was measured using three metrics: Mean Square Error (MSE), Symmetrical-Mean Absolute Percentage Error (SMAPE) and Mean Absolute Scaled Error (MASE). Next each metric is explained.

“Mean Square Error” is defined as:

$$MSE = \frac{1}{n} \sum_{t=1}^n (x_t - \hat{x}_t)^2 \quad (8)$$

The “Symmetrical-Mean Absolute Percentage Error” is scale-independent; therefore it is frequently used to compare performances when different time series are involved [17]. This is the official metric used by the “NN5 forecasting competition for artificial neural networks & computational Intelligence” [8]. SMAPE is defined as:

$$SMAPE = \frac{1}{n} \sum_{t=1}^n \left(\frac{|x_t - \hat{x}_t|}{(x_t + \hat{x}_t)/2} \right) (100\%) \quad (9)$$

It is important to point out that SMAPE cannot be applied over time series with negative values.

Other popular metric is the “Mean Absolute Scaled Error,” defined as:

$$MASE = \frac{1}{n} \sum_{t=1}^n \left| \frac{x_t - \hat{x}_t}{\frac{1}{n-1} \sum_{i=2}^n |x_i - x_{i-1}|} \right| \quad (10)$$

where x_t is the original time series and \hat{x}_t is the predicted time series.

4 Experiments and Results

The proposed architecture and training scheme were tested using two benchmark time series; they are:

a) The time series generated by Matlab function *sumsin()*, available at version 7.4 and commonly used in Matlab demos [24]. It is defined as:

$$s(t) = \sin(3t) + \sin(0.3t) + \sin(0.03t) \quad (11)$$

Figure 5 shows an example of 735 points of *sumsin()* time series.

b) Eleven of the time series found in the database of the “NN5 Forecasting Competition for Artificial Neural Networks and Computational Intelligence” [8]. These time-series correspond to cash drawbacks occurred daily in teller machines at England from 1996 to 1998; these series may be stationary, have local tendencies or contain zeroes or missing values. Figure 4 (a) shows the first time-series of such database, identified as “NN5-101”. The eleven time-series used here correspond to what is called the “reduced set” in such competition. In order to determine if these series were chaotic, the maximum Lyapunov Exponent (LE) of each one was calculated using the method proposed by Sano and Sawada [18]. Table 1 shows the maximum LE of each time series; notice that all are positive, an indication of chaos.

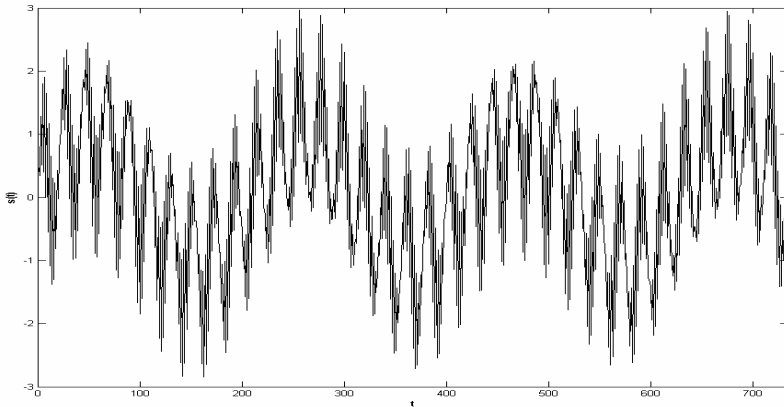


Fig. 5. 735 points of the time series $\text{sumsin}()$

Table 1. Maximum LE of reduced set series NN5 [12]

Series ID	Maximum LE
NN5-101	0.0267
NN5-102	0.6007
NN5-103	0.0378
NN5-104	0.0565
NN5-105	0.0486
NN5-106	0.0612
NN5-107	0.0678
NN5-108	0.0384
NN5-109	0.8405
NN5-110	0.0621
NN5-111	0.0220

The HWRN contains 3 SRNN; the number of nodes at each SRNN was from 6 to 10, determined experimentally depending upon the reconstructed signal being learnt; the hidden layer has 10 nodes. The performance of HWRN was compared with a three layer, feed-forward neural network (5-26-1) and a fully-connected recurrent neural network with 5 input nodes, 26 hidden nodes and one output node. These architectures have a similar number of nodes as the HWRN. All architectures receive as input 5 values ($k = 5$) of the time series and predict next value. Recurrent prediction is used to generate 56 futures values, following rules of the “NN5 forecasting competition for artificial neural networks & computational Intelligence” [8]. The architecture was implemented using Matlab V7.4, C++, and public libraries for the training algorithm available at [5]. For both cases, four reconstructed signals were generated using DWT with wavelet function Daubechies ‘*db10*’ available at Matlab. Three of the reconstructed signals were selected using the strategy described at section 2.1.

Twelve experiments were executed for each time series and each neural model. For each experiment, a different random initial set of weights was used. All trainings were made of 300 epochs. The first 635 values of each series were employed to train all models and the next 56 values were used as a testing set to compare the performance of the proposed architecture with respect to the other two models. The last 56 values of the series were used as a validation set in order to compare the performance of this architecture with respect to the competition results published by [9].

Table 2 shows the results obtained using recursive prediction of 56 values (validation set) by the 12 experiments over series *sumsim()*; the metric MAPE is not shown because it is not valid for negative values, as is with *sumsin()*. Figure 6 plots 56 predicted values (validation set) of series NN5-109, which was the series at NN5 dataset that obtained the best prediction results, with a SMAPE = 20.6%. Figure 7 plots 56 predicted values (validation set) of series NN5-107, which was the worst case obtained with series NN5, with a SMAPE = 40.5%.

Table 3 summarizes the average results obtained for the two cases, all experiments, all architectures predicting the validation set. For the three metrics in the two tested cases, HWRN got, in average, better results than the feed-forward and the fully-connected recurrent architectures. HWRN got a average SMAPE of 54% for the *sumsinn()* time series and 27% for the NN5 time series. It is important to point out that, with respect to contest results published by [9] using NN5 reduced test, HWRN could be located between the 16th and 17th place in the category of “neural networks and computational intelligence methods.”

Notice at table 3 the high Standard Deviation found in the performance measured by MASE for the three architectures. This may be due to the facts that these series are chaotic (see table 1), and that the ability of the learning algorithm RTRLEKF to find the best solution space depend, among other factors, upon the initial set of weights randomly generated. However, it may be noticed that HWRN got the smallest Standard Deviation for these cases.

Table 2. Twelve experiments predicting validation set over series $s_{umsin}()$. For a definition of MSA and MASE see equations (8) and (10)

Experiment Number	Feed-forward network		Recurrent Network		HWRN	
	MSE	MASE	MSE	MASE	MSE	MASE
1	0.112	60.827	0.236	100.004	0.110	73.184
2	0.089	49.823	0.083	56.638	0.092	66.314
3	0.070	47.931	0.036	40.038	0.370	41.687
4	0.099	58.184	0.191	86.501	0.034	41.807
5	0.061	48.613	0.023	34.78	0.029	37.863
6	0.165	80.531	0.090	53.566	0.040	43.689
7	0.096	59.478	0.003	12.399	0.132	78.450
8	0.049	49.816	0.521	107.336	0.052	49.518
9	0.104	74.631	0.146	78.909	0.054	49.677
10	0.063	55.017	0.064	53.904	0.116	67.361
11	0.063	50.018	0.087	73.610	0.099	68.531
12	0.160	84.994	0.086	61.378	0.021	32.832
Mean	0.094	59.989	0.131	63.255	0.068	54.243
St. deviation	0.038	13.044	0.140	27.553	0.039	15.551

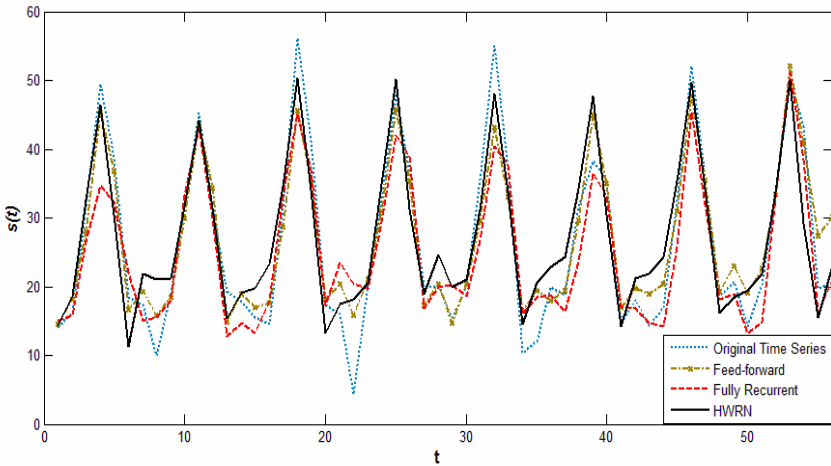


Fig. 6. Best Prediction Case using NN5, SMAPE = 20.6%, series NN5-109

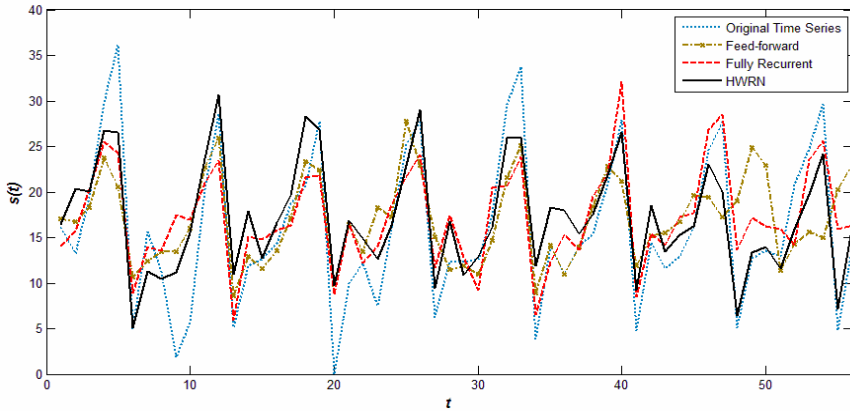


Fig. 7. Worst prediction case using NN5, SMAPE = 40.5%, series NN5-109

Table 3. Prediction errors obtained by the proposed architecture and two other architectures using 56 values ahead

Time Series	Metric	Neural Architecture		
		Feed-forward	Recurrent	HWRN
<i>sumsin()</i>	MSE	0.09 ± 0.04	0.13 ± 0.14	0.07 ± 0.04
	MASE	59.99 ± 13.04	63.25 ± 27.55	54.24 ± 15.55
Eleven examples of NN5 time series	MSE	250.12 ± 226.05	198.69 ± 131.12	34.05 ± 20.12
	SMAPE	$49.28\% \pm 12.36$	$60.75\% \pm 13.05$	$27.22\% \pm 8.27$
	MASE	$517.50 \pm 1,079.68$	$546.31 \pm 1,218.95$	194.99 ± 387.22

5 Conclusions

We presented a novel neural network predictor, called HWRN, based on a combination of small, fully-connected recurrent sub-networks, called SRNN, that are embedded in a composite neural system. This system is able to generate as many future values as desired using recursive prediction. HWRN was able to predict up to 56 points ahead of several non-linear time series, as shown by experiments done using the time series generated by Matlab's function *sumsin()* and the time series found at the reduced set of the "NN5 Forecasting Competition for Artificial Neural Networks and Computational Intelligence" [8]. The SRNN's are trained to reproduce selected reconstructed signals that represent different frequencies at different times of the original one. The reconstructed signals are obtained using the Discrete Wavelet Transform and the Inverse Discrete Wavelet Transform [1]. In average, the HWRN obtained a Symmetrical-Mean Absolute Percentage Error (SMAPE) of 27% when predicting in a recursive way 56 points ahead of 11 chaotic NN5 time series. This performance was better than the obtained with a fully-connected recurrent neural network (SMAPE= 61%) and a feed-forward network (SMAPE = 49%), both with similar number of nodes and weights. The main drawback of this system is the time required to train it. Currently our research group is looking for ways to train this system faster and for a efficient method to select the reconstructed signals generated by the iDWT.

References

1. Addison, P.S.: *The Illustrated Wavelet Transform Handbook: Introductory Theory and Applications in Science, Engineering, Medicine and Finance*. IOP Publishing, UK (2002)
2. Alarcon-Aquino, V., Garcia-Treviño, E.S., Rosas-Romero, R., y Ramirez-Cruz, J. F.: Learning and approximation of chaotic time series using wavelet networks. In: *Proceedings of the Sixth Mexican International Conference on Computer Science ENC 2005*, pp. 182–188 (2005)
3. Beliaev, I., Kozma, R.: Time series prediction using chaotic neural networks on the cats benchmark. *Neurocomputing* 70(13-15), 2426–2439 (2007)
4. Cai, X., Zhang, N., Venayagamoorthy, G.K., Wunsch II, D.C.: Time series prediction with recurrent neural networks trained by a hybrid PSO-EA algorithm. *Neurocomputing* 70(13-15), 2342–2353 (2007)
5. Cernansky, M.: Michal Cernansky's homepage downloads (2008), <http://www2.fiit.stuba.sk/~cernans/main/download.html> (last accessed January 2009)
6. Chen, Y., Yang, B., Dong, J.: Time-series prediction using a local linear wavelet neural network. *Neurocomputing* 69, 449–465 (2006)
7. Crone, S.F., Guajardo, J., Weber, R.: The impact of preprocessing on support vector regression and neural networks in time series prediction. In: *Proceedings of the 2006 International Conference on Data Mining, DMIN 2006*, pp. 37–44 (2006)
8. Crone, S.F.: NN5 forecasting competition for artificial neural networks & computational Intelligence (2008), <http://www.neural-corecasting-competition.com/> (last consulted at March 2009)
9. Crone, S.F.: NN5 forecasting competition results (2009), <http://www.neural-forecasting-competition.com/NN5/results.htm> (last consulted at July 2009)
10. Espinoza, M., Suykens, J., De Moor, B.: Short term chaotic time series prediction using symmetric LS-SVM regression. In: *Proceedings of the 2005 International Symposium on Nonlinear Theory and Applications (NOLTA)*, pp. 606–609 (2005)
11. Gao, H., Sollacher, R., Kriegel, H.P.: Spiral recurrent neural network for online learning. In: *Proceedings of the European Symposium on Artificial Neural Networks*, pp. 483–488 (2007)
12. García-Pedrero, A.: *Arquitectura Neuronal Apoyada en Señales Reconstruidas con Wavelets para predicción de Series de Tiempo Caóticas (A neural architecture supported by wavelet's reconstructed signals for chaotic time series prediction)*. Master Thesis (in Spanish), Computational Department, National Institute of Astrophysics, Optics and Electronics (2009)
13. García-Treviño, E.S., Alarcon-Aquino, V.: Chaotic time series approximation using iterative wavelet-networks. In: *Proceedings of the 16th International Conference on Electronics, Communications and Computers CONIELECOMP 2006*, pp. 19–24. IEEE Computer Society, Los Alamitos (2006)
14. Gomez-Gil, P.: Long Term Prediction, Chaos and Artificial Neural Networks. Where is the meeting point? *Engineering Letters* 15(1) (2007), http://www.engineeringletters.com/issues_v15/issue_1/EL_15_1_10.pdf

15. Gomez-Gil, P., Ramirez-Cortes, M.: Experiments with a Hybrid-Complex Neural Networks for Long Term Prediction of Electrocardiograms. In: Proceedings of the IEEE 2006 International World Congress of Computational Intelligence, IJCNN (2006), doi:10.1109/IJCNN.2006.246952
16. Haykin, S.: *Neural Networks: A Comprehensive Foundation*. Macmillan College Publishing Company, New York (1994)
17. Hyndman, R.J.: Another look at forecast accuracy metrics for intermittent demand. *Foresight: The International Journal of Applied Forecasting* 4, 43–46 (2006)
18. Kantz, H., Schreiber, T.: *Nonlinear Time Series Analysis*. Cambridge University Press, New York (2003)
19. Lendasse, A., Wertz, V., Simon, G., Verleysen, M.: Fast Bootstrap applied to LS-SVM for Long Term Prediction of Time Series. In: Proceedings of the 2004 International Joint Conference on Neural Networks, IJCNN 2004, pp. 705–710 (2004)
20. Lillekjendlie, B., Kugiumtzis, D., Christophersen, N.: Chaotic time series part II: System identification and prediction. *Modeling, Identification and Control* 15(4), 225–243 (1994)
21. Makridakis, S.G., Wheelwright, S.C., McGee, V.E.: *Forecasting: Methods and Applications*. John Wiley & Sons, Inc., New York (1983)
22. Mallat, S.G.: A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11(7), 674–693 (1989)
23. Mandic, D.P., Chambers, J.: *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*. John Wiley & Sons, New York (2001)
24. Misiti, M., Misii, Y., Oppenheim, G., Poggi, J.M.: *Wavelet Toolbox 4 User's Guide*. The MathWorks, Edition 4.3 (2008)
25. Palit, A.K., Popovic, D.: *Computational Intelligence in Time Series Forecasting: Theory and Engineering Applications*. In: *Advances in Industrial Control*, Springer-Verlag New York, Inc., Secaucus (2005)
26. Dong-Chul, P., Chung, N.T., Yunsik, L.: Multiscale BiLinear Recurrent Neural Networks and Their Application to the Long-Term Prediction of Network Traffic. In: Wang, J., et al. (eds.) *ISNN 2006*. LNCS, vol. 3973, pp. 196–201. Springer, Heidelberg (2006)
27. Pearlmutter, B.A.: *Dynamic Recurrent Neural Networks*. Technical Report CMU-CS-90-196, School of Computer Science, Carnegie Mellon University, Pittsburgh (1990)
28. Sarmiento, H.O., Villa, W.M.: Artificial intelligence in forecasting demands for electricity: an application in optimization of energy resources. *Revista Colombiana de Tec-nologías de Avanzada* 2(12), 94–100 (2008)
29. Soltani, S.: On the use of the wavelet decomposition for time series prediction. *Neuro-computing* 48, 267–277 (2002)
30. Soltani, S., Canu, S., Boichu, C.: Time series prediction and the wavelet transform. In: *International Workshop on Advanced Black Box modelling*, Leuven, Belgium (1998)
31. Wei, H.L., Billings, S.A., Guo, L.: Lattice Dynamical Wavelet Neural Networks Implemented Using Particle Swarm Optimization for Spatio-Temporal System Identification. *IEEE Transactions on Neural Networks* 20(1), 181–185 (2009)
32. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. *Neural Computing* 1(2), 270–280 (1989)
33. Zhang, Q., Benveniste, A.: Wavelet networks. *IEEE Transactions on Neural Networks* 3(6), 889–898 (1992)

An Interval Type-2 Fuzzy Neural Network for Chaotic Time Series Prediction with Cross-Validation and Akaike Test

Juan R. Castro¹, Oscar Castillo², Patricia Melin², Olivia Mendoza¹,
and Antonio Rodríguez-Díaz¹

¹ Computer Science in the Universidad Autonoma de Baja California,
Tijuana, B. C., México
jrcastror@uabc.edu.mx, omendoz@uabc.edu.mx, ardiaz@uabc.mx

² Computer Science in the Graduate Division Tijuana, Institute of Technology Tijuana,
B.C., Mexico
ocastillo@tectijuana.mx, pmelin@tectijuana.mx

Abstract. A novel homogeneous integration strategy of an interval type-2 fuzzy inference system (IT2FIS) with Takagi-Sugeno-Kang reasoning (TSK IT2FIS) is presented. This TSK IT2FIS is represented as an adaptive neural network (NN) with hybrid learning (IT2FNN:BP+RLS) in order to automatically generate an interval type-2 fuzzy logic system (TSK IT2FLS). Consequent parameters are updated with recursive least-square (RLS) algorithm; antecedent parameters with back-propagation (BP) algorithm. Mackey-Glass chaotic time series forecasting results are presented ($\tau=17, 30, 100$) with different signal noise ratio (SNR). Soundness for uncertainty, adaptability and learning and generalization capabilities is shown using 10-fold Cross Validation, Akaike Information Criteria (AIC) and F-Test.

1 Introduction

System's modeling is considered one of the most important problems in science. Models are abstractions of reality that can be applied to improve our understanding of real world phenomena. Finding functional relations between variables that intervene in phenomena is a problem that arises in several areas like economy, engineering, biology, sociology and medicine. Approximation of unknown functions from experimental data, e.g. time series prediction, is particularly important. In this case a sequence of time sample sets is used to predict the system's behavior in short and long-term. Several types of uncertainty are originated by improper data, bad modeling or wrong interpretations given to models. Also, the nature of phenomena might generate fluctuating data or variables, given thus uncertain data. Human mistakes and bad equipment calibration can generate incorrect data [1, 2,6]. Imprecision is the cause of

language granularity. Vagueness is considered a new imperfect information category. Inconsistence is originated by data redundancy that generate conflict or contradictions. Uncertainty levels vary depending on context. This uncertainty can be caused by noise in data [3,4,47,48]. Type-1 (T1FLS) and interval type-2 (IT2FLS) fuzzy logic systems, combined with techniques like neural networks have shown their ability to solve different kinds of problems like control [10-12, 14, 15, 16, 27, 32, 42, 44], prediction [12, 17, 20], signal and images processing [12, 13, 18, 28, 29], industrial processes [21, 22], etc. In many cases, success was achieved due to adding human expert knowledge. There is consensus between researchers that more intelligent systems can be developed by hybrid soft computing methodologies [41,45]. Type-1 Fuzzy Neural Network (T1FNN) [9, 40, 41] and Interval Type-2 Fuzzy Neural Network (IT2FNN) [22-26]; type-1 [31, 33, 35, 36, 38] and type-2 [32] fuzzy evolutionary systems are typical hybrid systems in softcomputing. These systems combine T1FLS generalized reasoning methods [30, 41, 42, 44, 45] and IT2FLS [19-21] with neural networks learning capabilities [41, 43, 46, 49] and evolutionary algorithms [34, 37, 39, 43], respectively. In this paper, four main architectures (IT2FNN-0, IT2FNN-1, IT2FNN-2 and IT2FNN-3) for integrating a first order TSK IT2FIS, with real consequents (A2C0) and interval consequents (A2C1), are proposed. Integration strategies for process elements of TSK IT2FIS are analysed for each architecture (fuzzification, knowledge base, type reduction and defuzzification).

2 Interval Type-2 Fuzzy Neural Network (IT2FNN)

An IT2FNN combines an TSK IT2FIS and an adaptive neural network in order to take advantage of characteristics of each model. Adaptive networks [40,41] provide a theoretical frame that unifies almost all kinds of neural networks with learning capabilities, and their fundamental properties are a key element for understanding other paradigms.

An IT2FNN is characterized by a directed graph $G = (V, E)$, where fan-in zero input nodes and fan-out zero output nodes. Each node $n \in V$ represents a processing unit with an associated static function f_n , and each edge $e \in E$ indicates causal relation between connected nodes. Node set V is divided into two separate subsets, $V = (A \cup N)$. Nodes $n \in A$ are called adaptive and their outputs depend not only on their inputs, but also on internal modifiable parameters $\{\xi_1^n, \xi_2^n, \dots\}$ internal to $n \in A$. On the other hand, nodes $n \in N$ whose functions depend only on inputs are called non adaptive. In general, when representing IT2FNN graphically, rectangles are used to represent adaptive nodes and circles to represent non-adaptive nodes. Output values of pair nodes (green color) and odd nodes (blue color) represent uncertainty intervals (Fig. 1–6). In this kind of interval type-2 neurofuzzy adaptive networks, nodes represent processing units called neurons, which can be classified into crisp and fuzzy neurons. In IT2FNN, type-1 fuzzy neuron model (T1FN) proposed by Hirota and Pedrycz [7, 8] is extended into an interval type-1 fuzzy neuron (IT1FN) and interval type-2 fuzzy neuron (IT2FN).

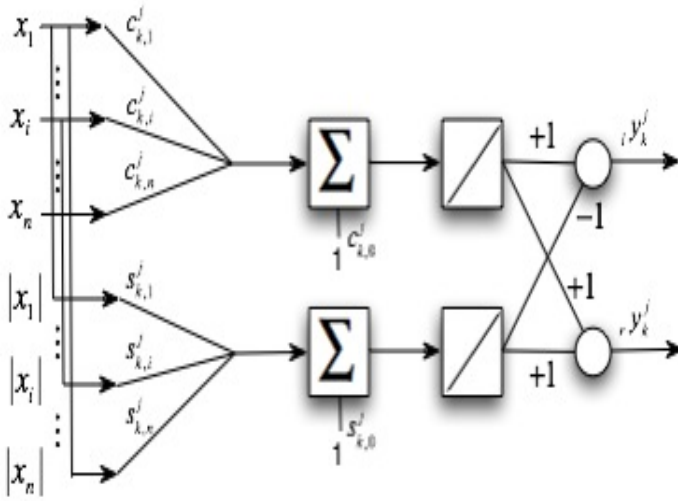


Fig. 1. Interval Type-1 Fuzzy Neuron (IT1FN)

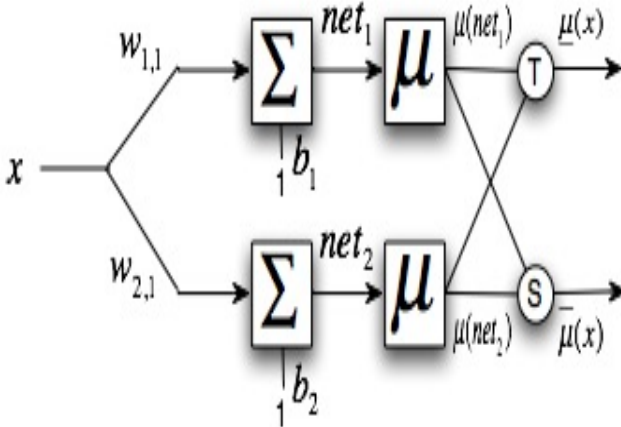


Fig. 2. Interval Type-2 Fuzzy Neuron (IT2FN)

IT2FNN-0 architecture has 7 layers. Layer 1 has adaptive nodes for fuzzifying inputs; layer 2 has non-adaptive nodes with interval fuzzy values. Layer 3 (rules) has non-adaptive nodes for generating firing strength of TSK IT2FIS rules. Layer 4, lower and upper values of rules firing strength are normalized. Adaptive nodes in layer 5 (consequent) are connected to layer 0 for generating rules consequents. Non-adaptive nodes in layer 6 evaluate values from left-right interval. Non-adaptive node in layer 7 (defuzzification) evaluates average of interval left-right values.

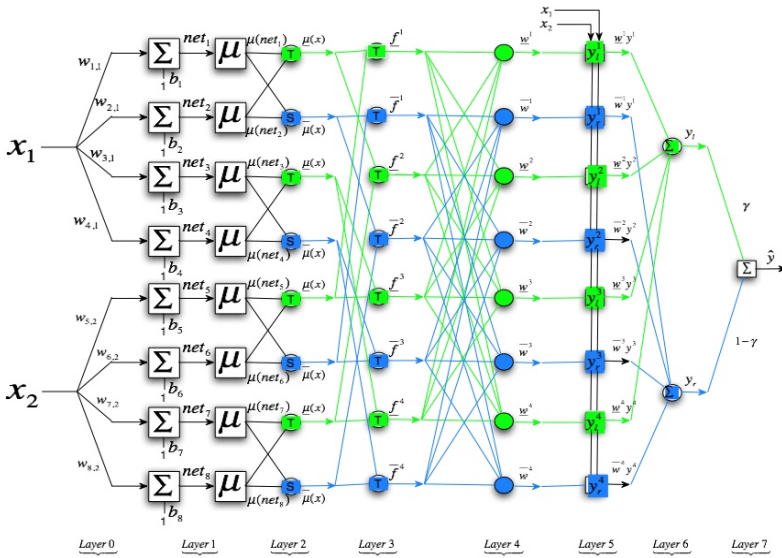


Fig. 3. IT2FNN-0 Architecture

IT2FNN-1 architecture has 5 layers, consists of adaptive nodes with equivalent function to lower-upper membership in fuzzification layer (layer 1). Non-adaptive nodes in rules layer (layer 2) interconnect with fuzzification layer (layer 1) in order to generate TSK IT2FIS rules antecedents. Adaptive nodes in consequent layer (layer 3) are connected to input layer (layer 0) to generate rules consequents. Non-adaptive nodes in type-reduction layer (layer 4) evaluate left-right values with Karnik and Mendel (KM)[19-21] algorithm. Non-adaptive node in defuzzification layer (layer 5) average left-right values.

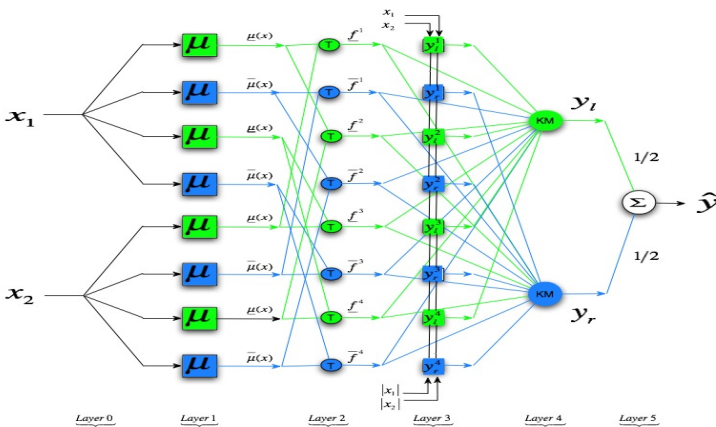


Fig. 4. IT2FNN-1 Architecture

In IT2FNN-2 case, consists on 6 layers, uses IT2FN for fuzzifying inputs (layers 1-2). Non-adaptive nodes in rules layer (layer 3) interconnect with lower-upper linguistic values layer (layer 2) to generate TSK IT2FIS rules antecedents. Non-adaptive nodes in consequents layer (layer 4) are connected with input layer (layer 0) to generate rules consequents. Non-adaptive nodes in type-reduction layer (layer 5) evaluate left-right values with KM algorithm. Non-adaptive node in defuzzification layer (layer 6) averages left-right values.

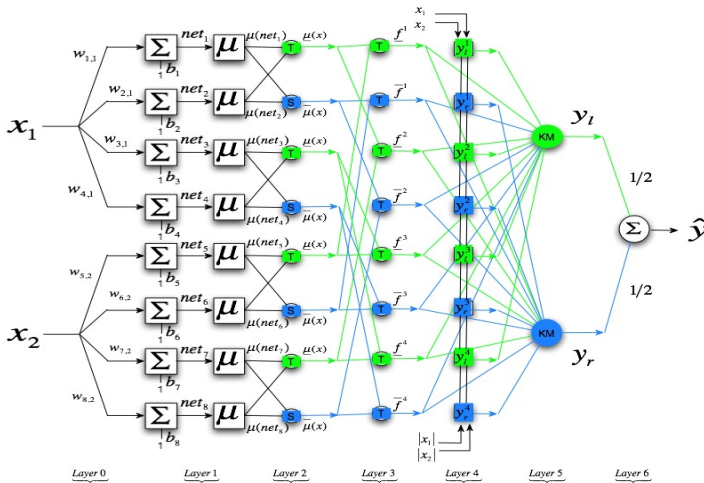


Fig. 5. IT2FNN-2 Architecture

IT2FNN-3 architecture has 8 layers, uses IT2FN for fuzzifying inputs (layers 1-2). Non-adaptive nodes in rules layer (layer 3) interconnect with lower-upper linguistic values layer (layer 2) to generate TSK IT2FIS rules antecedents. Adaptive nodes in layer 4 adapt left-right firing strength, biasing rules lower-upper trigger forces with synaptic weights between layers 3-4. Layer 5's non-adaptive nodes normalize rules lower-upper firing strength. Non-adaptive nodes in consequent layer (layer 6) interconnect with input layer (layer 0) to generate rules consequents. Non-adaptive nodes in type-reduction layer (layer 7) evaluate left-right values adding lower-upper product of lower-upper triggering forces normalized by rules consequent left-right values. Node in defuzzification layer is adaptive and its output \hat{y} is defined as biased average of left-right values and parameter γ . Parameter γ (0.5 by default) adjusts uncertainty interval defined by left-right values $[\hat{y}_l, \hat{y}_r]$.

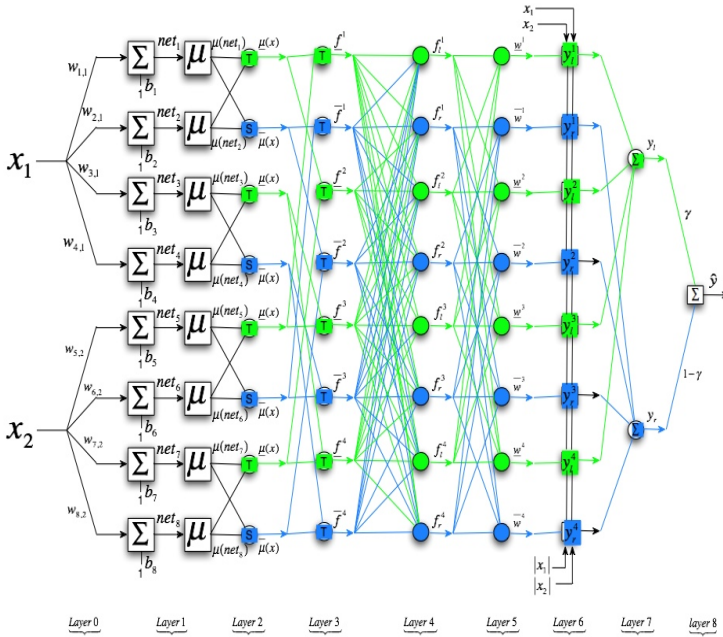


Fig. 6. IT2FNN-3 Architecture

3 Mackey-Glass Forecasting Chaotic Time Series

In this chapter, results from simulations using ANFIS, IT2FNN-0, IT2FNN-1, IT2FNN2 and IT2FNN3 are presented for forecasting Mackey-Glass chaotic time series [50] with $\tau = 17, 30, 100$ and different signal noise ratio values, $SNR(\text{dB}) = 0, 10, 20, 30$ as uncertainty source. Levenberg-Marquardt hybrid method is used for adapting IT2FNN models parameters. Proposed IT2FNN architectures are validated using 10-fold cross-validation [53, 54] considering sum square error (*SSE*) or root mean square error (*RMSE*) in training or test phase; Akaike information criteria (*AIC*) and F test [51, 52]. Cross-validation procedure evaluation is done using Matlab’s *crossvalind* function. Noise is added by Matlab’s *awgn* function. Also, *AIC* and F test are evaluated using statistical Matlab’s Toolbox. In Mackey-Glass chaotic time series, 1200 data sets were generated with initial conditions $x(0)=1.2$ and $\tau= 17, 30$ and 100 using Runge-Kutta 4th order method and several uniform noise added. An input-output vector was taken with format: $[x(\mathbf{t}-18), x(\mathbf{t}-12), x(\mathbf{t}-6), x(\mathbf{t}); x(\mathbf{t}+6)]$. For identifying ANFIS [40, 41] and IT2FNN models, an IT2FNN model with 4 inputs and one output is used with Mackey-Glass chaotic time series using: 16 rules, 2 igaussmtype2 IT2MF for each input, 50 epochs, 500 training data and 500 test data with 10-fold cross-validation. Tables 1-9 and figures 7-12 show *RMSE* (*TRN* and *CHK*) with 10-fold cross validation and the number of points (ζ) out of

uncertainty interval $\tilde{Y}(x) \in [\hat{y}_l(x), \hat{y}_r(x)]$ results between IT2FNN models and ANFIS. It can be seen that IT2FNN architectures forecast better than Mackey-Glass chaotic time series when noise is present.

Table 1. RMSE (TRN/CHK) and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=17$

SNR(dB)		0	10	20	30	free
ANFIS	TRN	0.2643	0.0902	0.0316	0.0110	0.0020
	CHK	0.2917	0.0989	0.0345	0.0118	0.0021
	ζ	NA	NA	NA	NA	NA
IT2FNN-0	TRN	0.2357	0.0776	0.0289	0.0088	0.0017
	CHK	0.2576	0.0837	0.0327	0.0107	0.0019
	ζ	28	23	17	13	7
IT2FNN-1	TRN	0.2089	0.0672	0.0268	0.0056	0.0016
	CHK	0.2334	0.0875	0.0318	0.0078	0.0018
	ζ	25	19	15	11	6
IT2FNN-2	TRN	0.1883	0.0540	0.0189	0.0066	0.0014
	CHK	0.1947	0.0592	0.0207	0.0071	0.0015
	ζ	24	18	14	10	4
IT2FNN-3	TRN	0.1558	0.0427	0.0134	0.0041	0.0011
	CHK	0.1632	0.0549	0.0148	0.0043	0.0013
	ζ	20	15	11	8	3

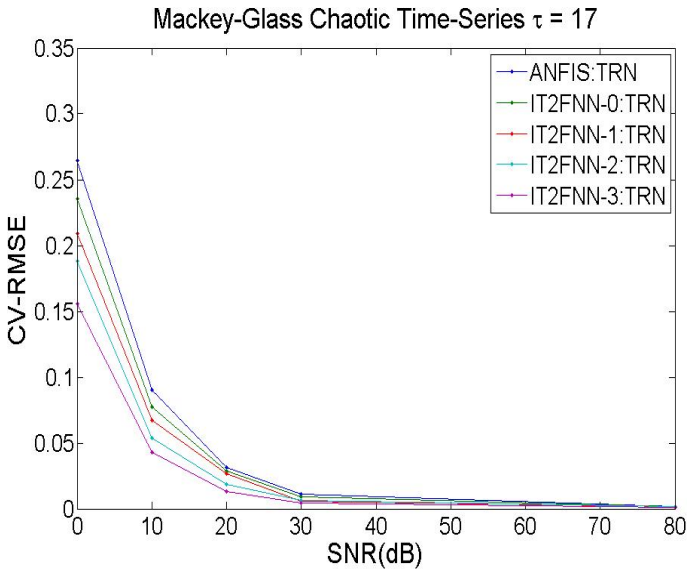


Fig. 7. RMSE (TRN) values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=17$

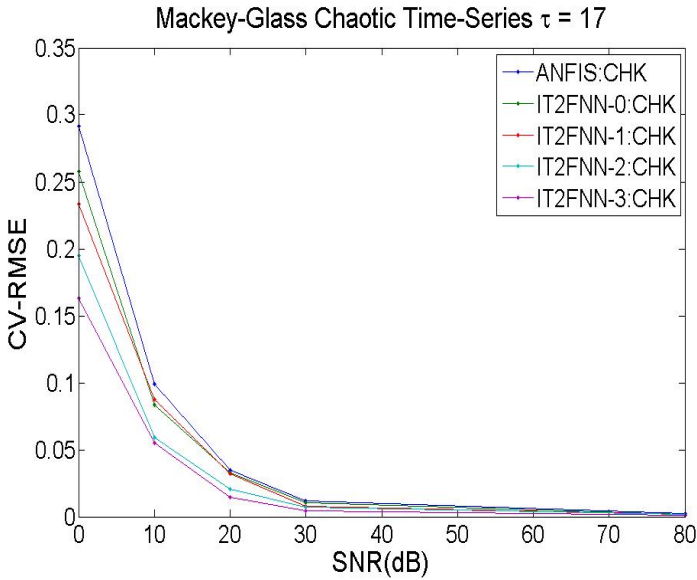


Fig. 8. RMSE (CHK) values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=17$

Table 2. Akaike Information Criteria (AIC) of TRN/CHK and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=17$

SNR(dB)		0	10	20	30	free
ANFIS	TRN	1968.63	893.58	-155.29	-1210.56	-2915.30
	CHK	2067.27	985.66	-67.49	-1140.35	-2866.51
	ζ	NA	NA	NA	NA	NA
IT2FNN-0	TRN	1872.11	761.12	-226.61	-1415.70	-3059.82
	CHK	1960.96	836.79	-103.08	-1220.21	-2948.60
	ζ	28	23	17	13	7
IT2FNN-1	TRN	1911.40	777.22	-142.05	-1707.68	-2960.45
	CHK	2022.30	1041.19	29.02	-1376.33	-2842.66
	ζ	25	19	15	11	6
IT2FNN-2	TRN	1807.59	558.53	-491.29	-1543.38	-3093.98
	CHK	1841.01	650.47	-400.32	-1470.36	-3024.99
	ζ	24	18	14	10	4
IT2FNN-3	TRN	1618.12	323.75	-835.20	-2019.46	-3335.14
	CHK	1664.53	575.06	-735.82	-1971.84	-3168.09
	ζ	20	15	11	8	3

Shaded cells in Table 2 show, based on AIC, that architecture IT2FNN-3 (TRN/CHK) forecasts better Mackey-Glass chaotic series with $\tau=17$ than ANFIS. In Table 3 it is shown that there is no significant improvement in architectures IT2FNN-1 (CHK) at SNR=10, 20 and noise-free compared with ANFIS.

Table 3. Statistic F (TRN/CHK) and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=17$

SNR(dB)		0	10	20	30	free
ANFIS/IT2FNN-0 $F_{0.95}(9,395) = 1.9036$	TRN	11.2972	15.4097	8.5838	24.6875	16.8570
	CHK	12.3887	17.3879	4.9648	9.4877	9.7261
	ζ	28	23	17	13	7
ANFIS/IT2FNN-1 $F_{0.95}(89,315) = 1.3070$	TRN	2.1262	2.8374	1.3814	10.1169	1.9909
	CHK	1.9890	0.9823	0.6265	4.5609	1.2781
	ζ	25	19	15	11	6
ANFIS/IT2FNN-2 $F_{0.95}(89,315) = 1.3070$	TRN	3.4336	6.3359	6.3547	6.2921	3.6838
	CHK	4.4051	6.3387	6.2921	6.2368	3.3978
	ζ	24	18	14	10	4
ANFIS/IT2FNN-3 $F_{0.95}(89,315) = 1.3070$	TRN	6.6461	12.2542	16.1434	21.9371	8.1609
	CHK	7.7678	7.9467	15.6932	23.1138	5.6964
	ζ	20	15	11	8	3

Table 4. RMSE (TRN/CHK) and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=30$

SNR(dB)		0	10	20	30	free
ANFIS	TRN	0.3416	0.1582	0.0876	0.0635	0.0585
	CHK	0.3800	0.1714	0.0943	0.0672	0.0619
	ζ	NA	NA	NA	NA	NA
IT2FNN-0	TRN	0.2543	0.1278	0.0633	0.0512	0.0444
	CHK	0.2722	0.1385	0.0719	0.0555	0.0467
	ζ	36	29	25	19	9
IT2FNN-1	TRN	0.2097	0.1123	0.0517	0.0413	0.0326
	CHK	0.2605	0.1203	0.0611	0.0429	0.0359
	ζ	33	27	22	17	8
IT2FNN-2	TRN	0.1719	0.0812	0.0415	0.0301	0.0297
	CHK	0.1801	0.0892	0.0447	0.0318	0.0289
	ζ	30	26	20	16	6
IT2FNN-3	TRN	0.1521	0.0733	0.0338	0.0249	0.0212
	CHK	0.1708	0.0773	0.0419	0.0302	0.0241
	ζ	27	23	18	8	5

Shaded cells in Table 5 show, based on AIC, that architecture IT2FNN-3 (TRN/CHK) forecasts better Mackey-Glass chaotic series with $\tau=30$ than ANFIS. In Table 6 it is shown that there is a significant improvement in all architectures IT2FNN (TRN/CHK) model for all SNR values with respect to ANFIS.

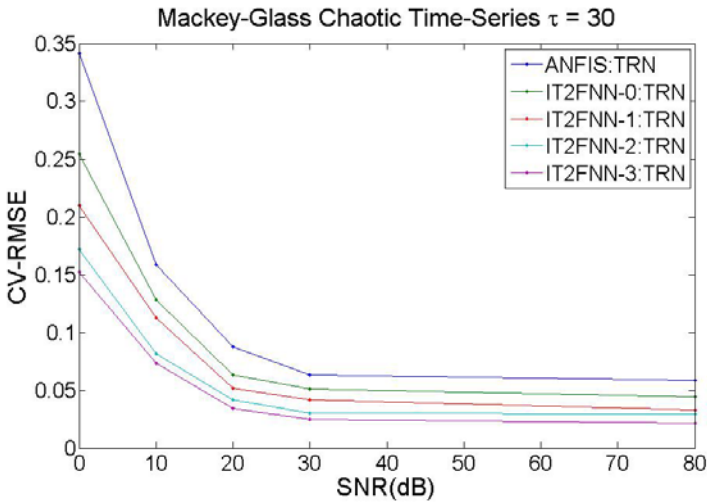


Fig. 9. RMSE (TRN) values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=30$

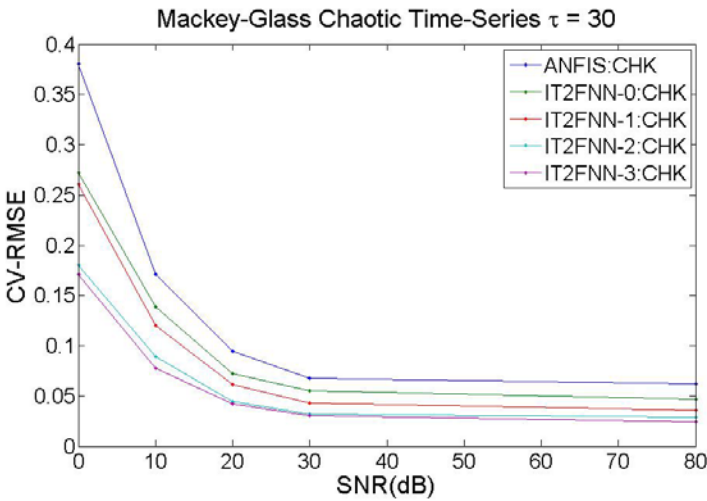


Fig. 10. RMSE (CHK) values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=30$

Table 5. Akaike Information Criteria (AIC) of TRN/CHK and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=30$

SNR(dB)		0	10	20	30	free
ANFIS	TRN	2225.19	1455.41	864.33	542.59	460.58
	CHK	2331.72	1535.55	938.03	599.22	517.07
	ζ	NA	NA	NA	NA	NA
IT2FNN-0	TRN	1948.06	1260.02	557.43	345.29	202.79
	CHK	2016.09	1340.42	684.83	425.93	253.29
	ζ	36	29	25	19	9
IT2FNN-1	TRN	1915.23	1290.72	515.01	290.41	53.86
	CHK	2132.15	1359.54	682.06	328.42	150.29
	ζ	33	27	22	17	8
IT2FNN-2	TRN	1716.46	966.46	295.24	25.93	39.30
	CHK	1763.06	1060.43	369.52	29.02	66.61
	ζ	30	26	20	16	6
IT2FNN-3	TRN	1594.09	864.11	90.01	-215.58	-376.45
	CHK	1710.04	917.24	308.83	-22.61	-248.24
	ζ	27	23	18	8	5

Table 6. Statistic F (TRN/CHK) and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=30$

SNR(dB)		0	10	20	30	free
ANFIS/IT2FNN-0 $F_{0.95}(9,395) = 1.9036$	TRN	35.3061	23.3632	40.1645	23.6202	32.3015
	CHK	41.6464	23.3277	31.6065	20.4550	33.2196
	ζ	36	29	25	19	9
ANFIS/IT2FNN-1 $F_{0.95}(89,315) = 1.3070$	TRN	5.8527	3.4845	6.6219	4.8276	7.8578
	CHK	3.9920	3.6454	4.8913	5.1452	6.9830
	ζ	33	27	22	17	8
ANFIS/IT2FNN-2 $F_{0.95}(89,315) = 1.3070$	TRN	10.4374	9.8952	12.2307	12.2126	10.1922
	CHK	12.2172	9.5288	12.2124	12.2661	12.6977
	ζ	30	26	20	16	6
ANFIS/IT2FNN-3 $F_{0.95}(89,315) = 1.3070$	TRN	14.3131	12.9471	20.2343	19.4788	23.4108
	CHK	13.9798	13.8620	14.3880	13.9852	19.8096
	ζ	27	23	18	8	5

Table 7. RMSE (TRN/CHK) and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=100$

SNR(dB)		0	10	20	30	free
ANFIS	TRN	0.4253	0.1839	0.1035	0.0706	0.0617
	CHK	0.4677	0.2035	0.1227	0.0758	0.0669
	ζ	NA	NA	NA	NA	NA
IT2FNN-0	TRN	0.3017	0.1445	0.0828	0.0563	0.0415
	CHK	0.3322	0.1497	0.0898	0.0681	0.0483
	ζ	50	42	38	29	13
IT2FNN-1	TRN	0.2258	0.1133	0.0687	0.0406	0.0307
	CHK	0.2976	0.1235	0.0779	0.0473	0.0397
	ζ	48	40	36	27	11
IT2FNN-2	TRN	0.1751	0.0724	0.0412	0.0313	0.0189
	CHK	0.1899	0.0783	0.0570	0.0398	0.0296
	ζ	46	38	34	24	8
IT2FNN-3	TRN	0.1311	0.0527	0.0258	0.0179	0.0155
	CHK	0.1377	0.0554	0.0311	0.0212	0.0133
	ζ	39	31	28	19	6

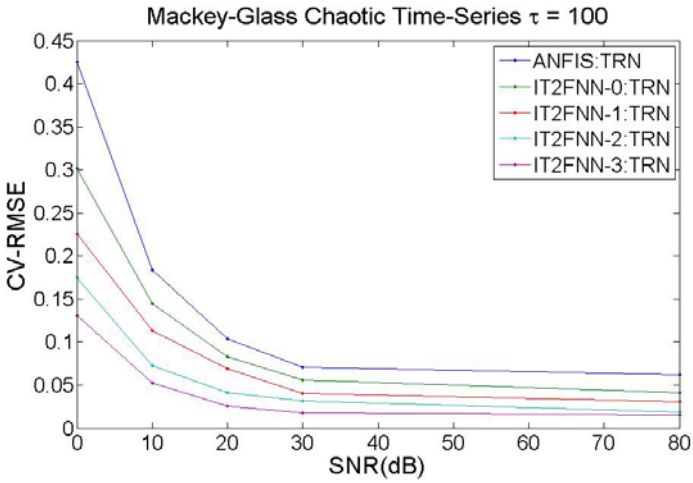


Fig. 11. RMSE (TRN) values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=100$

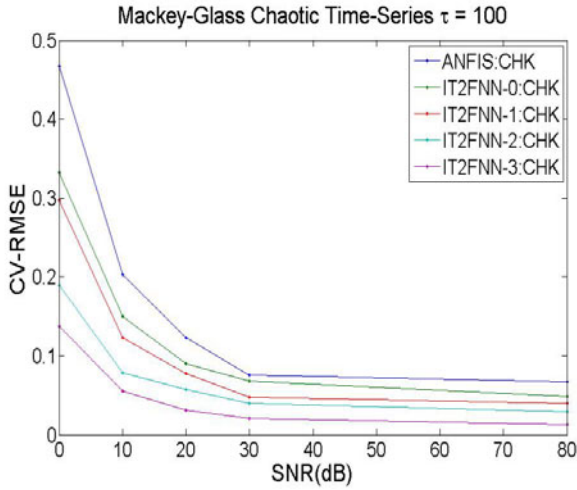


Fig. 12. RMSE (CHK) values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=100$

Table 8. Akaike Information Criteria (AIC) of TRN/CHK and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=100$

SNR(dB)		0	10	20	30	free
ANFIS	TRN	2444.34	1605.94	1031.12	648.58	513.83
	CHK	2539.38	1707.21	1201.29	719.65	594.75
	ζ	NA	NA	NA	NA	NA
IT2FNN-0	TRN	2118.98	1382.83	825.98	440.24	135.24
	CHK	2215.29	1418.18	907.13	630.53	286.98
	ζ	50	42	38	29	13
IT2FNN-1	TRN	1989.20	1299.59	799.30	273.32	-6.19
	CHK	2265.30	1385.79	924.97	426.06	250.90
	ζ	48	40	36	27	11
IT2FNN-2	TRN	1734.91	851.76	287.99	13.17	-491.29
	CHK	1816.05	930.10	612.60	253.42	-42.68
	ζ	46	38	34	24	8
IT2FNN-3	TRN	1445.51	534.16	-180.08	-545.65	-689.61
	CHK	1494.63	584.13	6.76	-376.45	-842.69
	ζ	39	31	28	19	6

Table 9. Statistic F (TRN/CHK) and ζ values determined in models ANFIS and IT2FNN with 10-fold cross-validation for forecasting Mackey-Glass chaotic time series with $\tau=100$

SNR(dB)		0	10	20	30	free
ANFIS/IT2FNN-0 $F_{0.95}(9,395)=1.9036$	TRN	43.3268	27.1968	24.6875	25.1267	53.1238
	CHK	43.1053	37.2147	38.0502	10.4860	40.3112
	ζ	50	42	38	29	13
ANFIS/IT2FNN-1 $F_{0.95}(89,315)=1.3070$	TRN	9.0170	5.7851	4.4939	7.1630	10.7567
	CHK	5.2022	6.0705	5.2415	5.5501	6.5113
	ζ	48	40	36	27	11
ANFIS/IT2FNN-2 $F_{0.95}(89,315)=1.3070$	TRN	17.3411	19.2960	18.7967	14.4677	34.1803
	CHK	17.9294	20.3677	12.8613	9.2985	14.5403
	ζ	46	38	34	24	8
ANFIS/IT2FNN-3 $F_{0.95}(89,315)=1.3070$	TRN	33.7089	39.5592	53.4196	51.5191	52.5432
	CHK	37.2914	44.2169	51.5527	41.7074	86.0115
	ζ	39	31	28	19	6

Shaded cells in Table 8 show, based on AIC, that architecture IT2FNN-3 (TRN/CHK) forecasts better Mackey-Glass chaotic series with $\tau=100$ than ANFIS. In Table 9 it is shown that there is significant improvement in architectures IT2FNN-1 (CHK) model for all SNR values with respect to ANFIS.

4 Conclusions

This paper models interval type-2 fuzzy neural networks (IT2FNN) architectures for forecasting Mackey-Glass chaotic time series. Combining neural networks and interval type-2 fuzzy logic systems improves learning, uncertainty handle and discovery, which are desirable characteristics for intelligent processing of imperfect information. Results show higher efficiency due to better representation capability and scalability. Mackey-Glass results analyzed with cross-validation, AIC and F tests, show that IT2FNN is a generalization of autoregressive models in times series case.

Cross validation, AIC and F test contrast analysis with statistical models show that IT2FNN models have better results under high uncertainty or same results under low uncertainty as ANFIS models.

References

- [1] Klir, G.J., Folger, T.A.: Fuzzy Sets, Uncertainty and Information. Prentice Hall, Englewood Cliffs (1988)
- [2] Klir, G.J., Wierman, M.J.: Uncertainty-Based Information: Elements of Generalized Information Theory. Physica-Verlag / Springer, Heidelberg (1999)
- [3] Davis, T.J., Keller, C.P.: Modelling uncertainty in natural resource analysis using fuzzy sets and Monte Carlo simulation: slope stability prediction. International Journal of Geographical Information Science 11(5), 409–434 (1997)
- [4] Gil Aluja, J.: Elements for a theory of decision in uncertainty. Kluwer Academic Publishers, Boston (1999)

- [5] Hagrass, F., Roberts, H., Callaghan, D.: A Type-2 Fuzzy Based System for Handling the Uncertainties in Group Decisions for Ranking Job Applicants within Human Resources Systems. In: FUZZ-IEEE 2008, Hong Kong (June 2008)
- [6] Hwang, C., Rhee, F.C.-H.: Uncertain Fuzzy Clustering: Interval Type-2 Fuzzy. *IEEE Transaction on Fuzzy System* 15(1), 107–120 (2007)
- [7] Hirota, K., Pedrycz, W.: Knowledge-based networks in classification problems. *Fuzzy Sets and Systems* 51, 1–27 (1992)
- [8] Hirota, K., Pedrycz, W.: OR/AND neuron in modeling fuzzy set connectives. *IEEE Transactions on Fuzzy Systems* 2, 151–161 (1994)
- [9] Horikawa, S., Furuhashi, T., Uchikawa, Y.: On fuzzy modeling using fuzzy neural networks with the backpropagation algorithm. *IEEE Transactions on Neural Networks* 3 (1992)
- [10] Chaoui, H., Gueaieb, W.: Type-2 Fuzzy Logic Control of a Flexible-Joint Manipulator. *Journal Of Intelligent And Robotic Systems* 51(2), 159–186 (2008)
- [11] Wu, D., Tan, W.W.: A simplified type-2 fuzzy logic controller for real-time control. *Isa Transactions* 45(4), 503–516 (2006)
- [12] Castillo, O., Melin, P., Kacprzyk, J., Pedrycz, W.: Type-2 Fuzzy Logic theory and applications. In: *Proceedings of Granular Computing 2007*, Silicon Valley, CA, USA, November 2007, pp. 145–150 (2007)
- [13] Zeng, J., Liu, Z.-Q.: Type-2 fuzzy sets for pattern recognition: The state-of-the-art. *Journal Uncertain System* 1(3), 163–177 (2007)
- [14] Sepulveda, R., Castillo, O., Melin, P., Rodriguez-Diaz, A., Montiel, O.: Experimental study of intelligent controllers under uncertainty using type-1 and type-2 fuzzy logic. *Information Sciences* 177(10), 2023–2048 (2007)
- [15] Singh, M., Srivastava, S., Gupta, J.R.P., Hanmandlu, M.: A new algorithm-based type-2 fuzzy controller for diabetic patient. *International Journal of Biomedical Engineering And Technology* 1(1), 18–40 (2007)
- [16] Astudillo, L., Castillo, O., Melin, P., Alanis, A., Soria, J., Aguilar, L.: Intelligent Control of an Autonomous Mobile Robot using Type-2 Fuzzy Logic. *Journal of Engineering Letters* 13(2), 93–97 (2006)
- [17] Baguley, P., Page, T., Koliza, V., Maropoulos, P.: Time to market prediction using type-2 fuzzy sets. *Journal of Manufacturing Technology Management* 17(4), 513–520 (2006)
- [18] Zeng, J., Liu, Z.-Q.: Type-2 Fuzzy Hidden Markov Models and Their Application to Speech Recognition. *IEEE Transactions On Fuzzy Systems* 14(3), 454–467 (2006)
- [19] Mendel, J.M.: *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*. Prentice Hall, NJ (2001)
- [20] Karnik, N.N., Mendel, J.M.: Applications of type-2 fuzzy logic systems to forecasting of time-series. *Inform. Sci.* 120, 89–111 (1999)
- [21] Wu, D., Mendel, J.M.: A Vector Similarity Measure for Interval Type-2 Fuzzy Sets and Type-1 Fuzzy Sets. *Information Sciences* 178, 381–402 (2008)
- [22] Lee, C.-H., Lin, Y.-C.: Type-2 Fuzzy Neuro System Via Input-to-State-Stability Approach. In: Liu, D., Fei, S., Hou, Z., Zhang, H., Sun, C. (eds.) *ISNN 2007*. LNCS, vol. 4492, pp. 317–327. Springer, Heidelberg (2007)
- [23] Lin, Y.-C., Lee, C.-H.: System Identification and Adaptive Filter Using a Novel Fuzzy Neuro System. *International Journal of Computational Cognition* 5(1) (March 2007)
- [24] Lee, C.H., Hong, J.L., Lin, Y.C., Lai, W.Y.: Type-2 Fuzzy Neural Network Systems and Learning. *International Journal of Computational Cognition* 1(4), 79–90 (2003)

- [25] Wang, C.H., Cheng, C.S., Lee, T.-T.: Dynamical optimal training for interval type-2 fuzzy neural network (T2FNN). *IEEE Trans. on Systems, Man, and Cybernetics Part B: Cybernetics* 34(3), 1462–1477 (2004)
- [26] Hagrais, H.: Comments on Dynamical Optimal Training for Interval Type-2 Fuzzy Neural Network (T2FNN). *IEEE Transactions on Systems Man And Cybernetics Part B* 36(5), 1206–1209 (2006)
- [27] Mendez, G.M., Leduc, L.: Hybrid Learning Algorithm for Interval Type-2 Fuzzy Logic Systems. *Control and Intelligent Systems Journal, Special Issue on Nonlinear Adaptive PID Control Part 2* 34(3), 206–215 (2006)
- [28] Own, C.-M., Tsai, H.-H., Yu, P.-T., Lee, Y.-J.: Adaptive type-2 fuzzy median filter design for removal of impulse noise. *Imaging Science* 54(1), 3–18 (2006)
- [29] Cao, X.-Q., Zeng, J., Yan, H.: Modeling Uncertain Speech Sequences Using Type-2 Fuzzy Hidden Markov Models. In: Ip, H.H.-S., Au, O.C., Leung, H., Sun, M.-T., Ma, W.-Y., Hu, S.-M. (eds.) *PCM 2007*. LNCS, vol. 4810, pp. 315–324. Springer, Heidelberg (2007)
- [30] Pedrycz, W.: *Fuzzy Modelling: Paradigms and Practice*. Kluwer Academic Press, Dordrecht (1996)
- [31] Wang, C.H., Liu, H.L., Lin, C.T.: Dynamic optimal Learning rate of A Certain Class of Fuzzy Neural Networks and Its Applications with Genetic Algorithm. *IEEE Trans. Syst. Man, Cybern.* 31(3), 467–475 (2001)
- [32] Wu, D., Wan Tan, W.: Genetic learning and performance evaluation of interval type-2 fuzzy logic controllers. *Engineering Applications of Artificial Intelligence* 19(8), 829–841 (2006)
- [33] Ascia, G., Catania, V., Panno, D.: An Integrated Fuzzy-GA Approach for Buffer Management. *IEEE Trans. Fuzzy Syst.* 14(4), 528–541 (2006)
- [34] Bonissone, P.P., Subbu, R., Eklund, N., Kiehl, T.R.: Evolutionary Algorithms + Domain Knowledge = Real-World Evolutionary Computation. *IEEE Trans. Evol Comput.* 10(3), 256–280 (2006)
- [35] Chiou, Y.-C., Lan, L.W.: Genetic fuzzy logic controller: an iterative evolution algorithm with new encoding method. *Fuzzy Sets Syst.* 152(3), 617–635 (2005)
- [36] Pedrycz, W.: *Fuzzy Evolutionary Computation*. Kluwer Academic Publishers, Dordrecht (1997)
- [37] Deb, K.: *A population-based algorithm-generator for real-parameter optimization*. Springer, Heidelberg (2005) (in press)
- [38] Ishibuchi, H., Nozaki, K., Yamamoto, N., Tanaka, H.: Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Trans. Fuzzy Syst.* 3, 260–270 (1995)
- [39] Engelbrecht, A.P.: *Fundamentals of computational swarm intelligence*. John Wiley & Sons, Ltd., Chichester (2005)
- [40] Jang, J.-S.R.: ANFIS: adaptive-network-based fuzzy inference system. *IEEE Trans. Syst., Man, Cybern.* 23(3), 665–684 (1993)
- [41] Jang, J.S.R., Sun, C.T., Mizutani, E.: *Neuro-fuzzy and Soft Computing*. Prentice-Hall, New York (1997)
- [42] Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. *Int. J. Man-Mach. Stud.* 7, 1–13 (1975)
- [43] Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*. Prentice-Hall, NJ (2003)
- [44] Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst., Man, Cybern.* 15, 116–132 (1985)

- [45] Zadeh, L.A.: Fuzzy Logic, Neural Neural Networks and Soft Computing. Communications of the ACM 37(3), 77–84 (1994)
- [46] Hagan, M.T., Demuth, H.B., Beale, M.H.: Neural Network Design. PWS Publishing, Boston (1996)
- [47] Zadeh, L.A.: Towards a generalized theory of uncertainty (GTU)—an outline. Information Sciences 172, 1–40 (2005)
- [48] Zadeh, L.A., Kacprzyk, J. (eds.): Computing With Words in Information/Intelligent Systems, vol. 1 & 2. Physica-Verlag, New York (1999)
- [49] Haykin, S.: Adaptive Filter Theory. Prentice Hall, Englewood Cliffs (2002) ISBN 0-13-048434-2
- [50] Mackey, M.C., Glass, L.: Oscillation and Chaos in Physiological Control Systems. Science 197, 287–289 (1977)
- [51] Ljung, L.: System Identification: Theory for the User, pp. 278–280. Prentice-Hall, Englewood Cliffs (1987)
- [52] Akaike, H.: A new look at the statistical model identification. IEEE Transactions on automatic control AC 19, 716–723 (1974)
- [53] Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, pp. 214–216. Springer, Heidelberg (2001)
- [54] Theodoridis, S., Koutroumbas, K.: Pattern Recognition, pp. 341–342. Academic Press, London (1999)

Chaotic Time Series Prediction Using Ensembles of ANFIS

Jesus Soto, Oscar Castillo, and José Soria

Tijuana Institute of Technology, Tijuana México
charay_10@hotmail.com, ocastillo@tectijuana.mx

Abstract. This paper presents the proposed architecture for Ensembles of ANFIS (adaptive Network based fuzzy inference system), with emphasis on its application to prediction of chaotic time series (like the Mackey-Glass), where the goal is to minimize the prediction error. The methods used for the integration of the Ensembles of ANFIS are: Integrator by average and the integrator of weighted average. The performance obtained with the Ensemble architecture overcomes several standard statistical approaches and neural network models reported in the literature by various researchers. In the experiments we changed the type of membership functions and the desired error, thereby increasing the complexity of the training.

1 Introduction

System modeling based on conventional mathematical tools (e.g., differential equations) is not well suited for dealing with ill-defined and uncertain systems. By contrast, a fuzzy inference system employing fuzzy if-then rules can model the qualitative aspects of human knowledge and reasoning processes without employing precise quantitative analyses.

This fuzzy modeling or fuzzy identification, first explored systematically by Takagi and Sugeno [12], has found numerous practical applications in control, prediction and inference. However, there are some basic aspects of this approach, which are in need of better understanding. More specifically:

- 1) No standard methods exist for transforming human knowledge or experience into the fuzzy rule base and database of a fuzzy inference system.
- 2) There is a need for effective methods for tuning the membership functions (MF's) so as to minimize the output error measure or maximize a performance index.

In this perspective, the aim of this paper is to suggest a new architecture for Ensembles of Adaptive-Network-based Fuzzy Inference System, or simply Ensembles of ANFIS, which can serve as a basis for constructing a set of fuzzy if-then rules with appropriate membership functions to generate the estipulated input-output pairs. The architecture of adaptive fuzzy inference systems proposed by [6] uses the gradient descent backpropagation [5] and least squares to achieve the learning ability of ANFIS. This paper reports the results of the simulations, where the Ensembles of ANFIS was used to predict chaotic time series of Mackey-Glass [8], where the results for each ANFIS were evaluated by the criteria of the root mean square error (RMSE). For the integration of the results of each Ensembles of ANFIS two different integration methods were used: integrator by average, and integration of weighted average. The selection of this time series for the simulation was simply because it is a benchmark problem that has been widely quoted in the literature by different researchers, which allows to compare results with other approaches such as neural networks and linear regression.

In the next section we describe the basics of ANFIS and the ANFIS Ensemble learning, like the architecture proposed in this paper. Section 3 presents the chaotic time series, as well as the adjustment parameters of the ANFIS Ensemble. Section 4 presents the methods of integration of the Ensembles of ANFIS. Section 5 presents the simulation and the results obtained in this work. Section 6 presents conclusions and future work.

2 ANFIS and Ensemble Learning

This section presents the basic concepts of ANFIS, to give us an idea of how the operation of ANFIS really is. In the second part we consider the learning, for understanding how they apply to our proposal. In the last part we describe the architecture or structure we are proposing for the Ensembles of ANFIS.

2.1 Basics Aspects of ANFIS

This section presents the basic architecture of ANFIS and the hybrid learning that it uses.

For simplicity, consider a fuzzy inference system with two rules of Takagi-Sugeno type [13]:

Rule 1: If x is A_1 and y is B_1 , then $f_1 = p_1 + p_1y + r_1$

Rule 2: If x is A_2 and y is B_2 , then $f_2 = p_2 + p_2y + r_2$

Fig. 1. (a) and (b) illustrates the reasoning mechanism and the corresponding ANFIS architecture, respectively. The functions of the nodes in the same layer of ANFIS are of the same family of functions, as described below. (In subsequent, O_i^j denotes the i th output node of the layer j .)

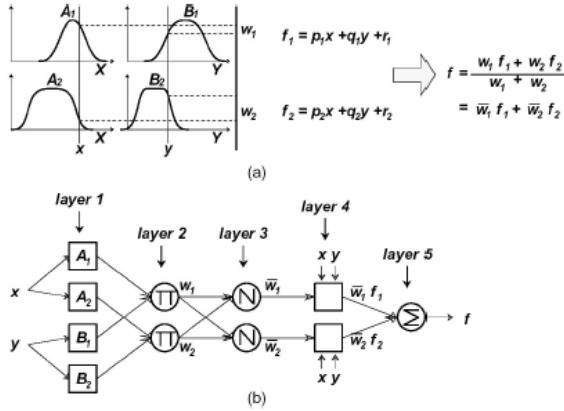


Fig. 1. (a) fuzzy model; (b) equivalent ANFIS architecture

Layer 1: Each node in this layer corresponds to a linguistic label and the output node is equal to the value of membership in this linguistic label. The parameters of a node can change the shape of the membership function used to characterize the linguistic label. For example, the function of the i th node is

$$O_i^1 = \mu A_i(x) \frac{1}{1 + [(\sum_i \omega_i f_i)^2]^{b_i}} \tag{1}$$

where k is the input node i ; A_i is linguistic label *small, large, etc.* Associated with this node, and $\{a_i, b_i, c_i\}$ is the set of parameters. Parameters in this layer are called *premise parameters*.

Layer 2: each node in this layer calculates the firing power of each rule:

$$O_i^2 = \omega_i = \mu A_i(x) \times \mu B_i(y), i = 1, 2. \tag{2}$$

Layer 3: the i th node of this layer calculates the ratio of the firing strength of the i th rule of the sum of all the firing strength:

$$O_i^3 = \bar{\omega}_i = \frac{\omega_i}{\omega_1 + \omega_2}, i = 1, 2 \tag{3}$$

Layer 4: node i in this layer has the following node function

$$O_i^4 = \bar{\omega}_i f_i = \bar{\omega}_i (p_i x + q_i y + r_i) \quad (4)$$

Layer 5: the single node in this layer computes the overall output as the sum of all input signals:

$$\text{overall output} = O_1^5 = \Sigma \bar{\omega}_i f_i = \frac{\Sigma i \omega_i f_i}{\Sigma i \omega_i f_i} \quad (5)$$

In this way an adaptive network has been built (Fig. 1. (b)), which is functionally equivalent to inference systems (Fig. 1. (a)), so it is called ANFIS, referring to fuzzy inference systems based on adaptive networks. The basic learning rule of ANFIS is the gradient descent backpropagation [15], which calculates the error rates (defined as the derivative of the squared error for each output node) recursively from the output to the input nodes. This is the same learning rule used in neural networks [11]. In the basic architecture of Fig. 1. it may be noted that given the values of premise parameters, the total output f can be expressed as linear combinations of the parameters accordingly:

$$\begin{aligned} f &= \bar{\omega}_1 f_1 + \bar{\omega}_2 f_2 \\ &= (\bar{\omega}_1 x) p_1 + (\bar{\omega}_1 y) q_1 + (\bar{\omega}_1) r_1 \\ &= (\bar{\omega}_2 x) p_2 + (\bar{\omega}_2 y) q_2 + (\bar{\omega}_2) r_2 \end{aligned} \quad (6)$$

As a result we have a hybrid learning algorithm [7], which combines the gradient descent and least-squares estimation. More specifically, the step forward from the hybrid learning algorithm, functional signals (output nodes) are in progress towards layer 4 and the parameters of consequence are identified by least squares. In the step back and premise parameters are updated by gradient descent.

2.2 Ensemble Learning

Ensemble is a learning paradigm, where multiple component learners are trained for a same task, and the predictions of the component learners are combined for dealing with future instances [14]. Since an ensemble is often more accurate than its component learners, such a paradigm has become a hot topic in recent years and has already been successfully applied to optical character recognition, face recognition, scientific image analysis, medical diagnosis, etc [16].

In this paper, the learning of the total output of each Ensemble was calculated with integrators of average and weighted average, just as the output of each ANFIS is calculated with the root mean squared error RMSE.

2.3 Structure of the Ensemble of ANFIS

In this paper, we propose the structure illustrated in Fig. 2.

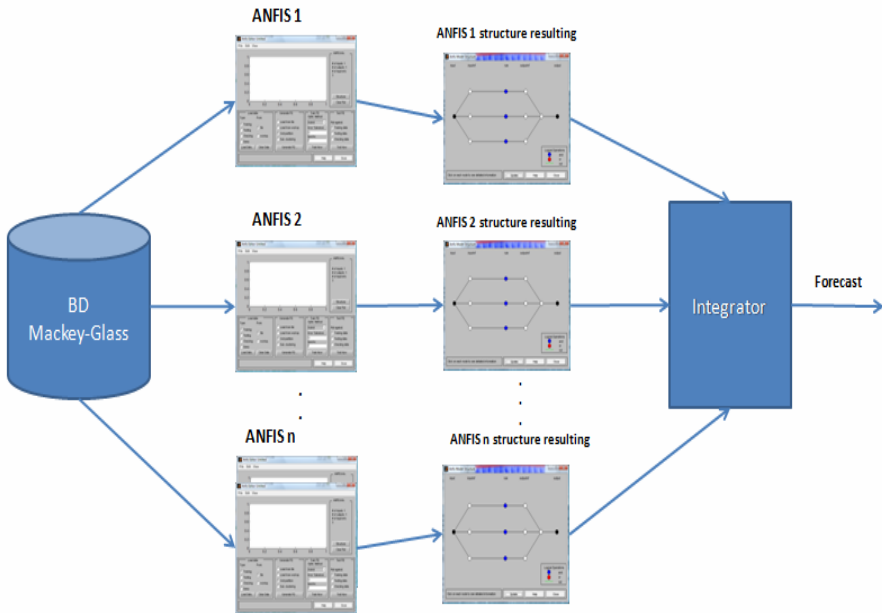


Fig. 2. The structure of the Ensembles de ANFIS

This structure is divided into 5 sections, where the first section represents the data base to simulate in our Ensembles of ANFIS, which in our case is the data base of the Mackey-Glass time series [8] the second section training and validation will be done sequentially in each ANFIS, where the number of ANFIS be trained can be from 1 to n depend on what the user wants to test, but in our case we are dealing with a set of 3 ANFIS in the Ensemble, in the third section we have to generate the results of each ANFIS trained in the previous section and the fourth section will integrate the overall results of each ANFIS, such integration will be done by direct average and weighted average is obtained and finally the outcome or the final prediction of the Ensembles ANFIS learning.

The main idea of Ensemble learning in ANFIS, is that each ANFIS has different ways to train and thus to be simulated as if it was an expert system, i.e., give different points of view and then predict the time series, then take decisions or the results of each ANFIS and reach the same conclusion, then integrating the results and obtain the best prediction of the time series that is being simulated, to avoid future unexpected events.

3 Chaotic Time Series Prediction

The problem of predicting future values of this time series has been a point of reference for many researchers. The aim is to use the values of the time series known at point $x = t$ to predict the value of the series at some future point $x = t + P$. The standard method for this type of prediction is to create a mapping from D points spaced time series in Δ , is $(x(t - (D - 1)\Delta), \dots, x(t - \Delta), x(t))$, to a predicted future value $x(t + P)$. To allow a comparison with previous results in this work [3], [9,10], [4] the values $D = 4$ and $\Delta = P = 6$, were used.

The used chaotic time series data used is defined as the Mackey-Glass time series (5), whose differential equation is defined as:

$$\frac{dx(t)}{dt} = \frac{0.2x(-t - \tau)}{1 + x^{10}(t - \tau)} - 0.1x(t) \quad (7)$$

For obtaining the values of the time series for each point, we applied the Runge-Kutta method for the solution of equation 7. The integration step was set at 0.1, with initial condition $x(0) = 1.2$, $\tau = 17$, $x(t)$ is then obtained for $0 \leq t \leq 1200$, which is illustrated in Fig. 3. (We assume $x(t) = 0$ for $t < 0$ in the integration.)

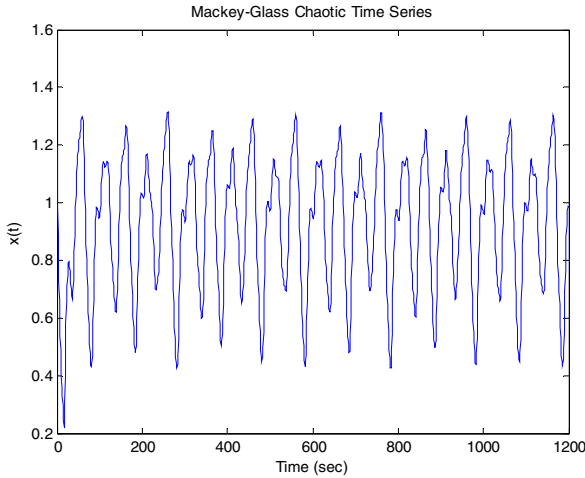


Fig. 3. The chaotic time series

From the Mackey-Glass time series we extracted 800 pairs of data points. Similar to [7], we predict $x(t)$ from the three past values of the chaotic time series, that is, $x(t-18)$, $x(t-12)$, and $x(t-6)$. Therefore the format of the training data is

$$[x(t-18), x(t-12), x(t-6) ; x(t)] \quad (8)$$

Where $t = 19$ to 818. The first 400 pairs of data are used to train the ANFIS, while the other 400 pairs of data are used to validate the model identification. The number of membership functions assigned to each entry of the ANFIS is fixed arbitrarily at 2, so that the number of rules is 8. Fig. 4. (a) shows the initial membership functions for each input variable and Fig 4. (b) illustrates the output membership functions after learning, but in our case the membership function type is varied. Note that the ANFIS used here contains a total of 50 parameters of adjustment, of which 32 are consequent (linear) parameters and 18 premise (nonlinear) parameters.

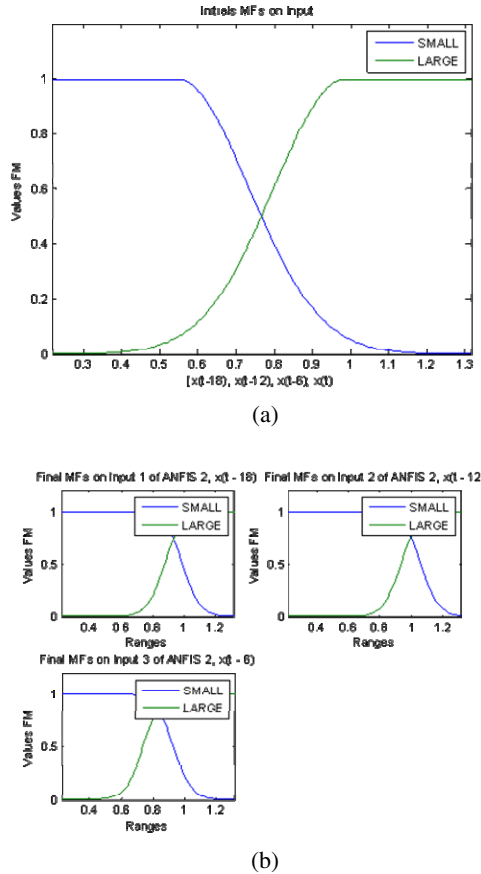


Fig. 4. (a) Initial membership functions before learning; (b) Final membership functions after learning

4 Methods of Integration

There exists a diversity of methods of integration or aggregation of information, and we mention some of these methods below:

Integration by average: this method is used in the Ensembles of ANFIS. This integration method is the simplest and most straightforward, consists in the sum of the results generated by each ANFIS is divided by the sum of number of ANFIS, and the disadvantage is that there are cases in which the prognosis is not good.

Integration of weighted average: this method is an extension of the integration by average, with the main difference that the weighted average assigns importance weights to each of the ANFIS. These weights are assigned to a particular ANFIS based on several factors; the most important is the knowledge product experience. This integration method belongs to the well known aggregation operators.

5 Simulation Results

This section presents the results obtained through experiments on the structure of the ANFIS Ensemble, which show the performance that was obtained from each experiment to simulate the chaotic time series. Similarly, it is showing the results for each set of ANFIS that integrate our Ensembles of ANFIS. The results obtained by the integration by averages, and integration of weighted averages are shown in the last section.

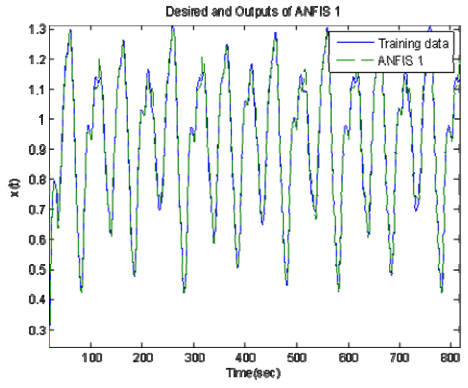
5.1 Ensemble

In this Ensemble we have a set of 3 ANFIS learning, it is noteworthy that the type of membership functions the were assigned to each desired ANFIS were varied and the goal error was assigned to each ANFIS to 0.001 and the output of each ANFIS is evaluated by the following equation:

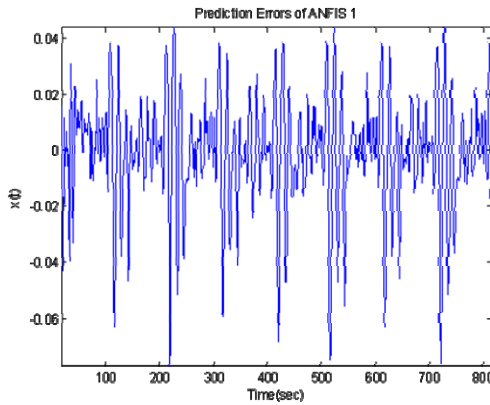
$$\text{RMSE} = \sqrt{\sum_{t=1}^n \frac{(x_{1t} - x_{2t})^2}{n}} \quad (9)$$

5.1.1 ANFIS 1

In this ANFIS 1 the triangular membership function was used, the prediction of output generated by the ANFIS 1 is shown in Fig. 5. (a), showing the trained data and the output of ANFIS 1, their differences are shown in Fig. 5. (b), which shows the prediction error between the desired data and data predicted by the ANFIS 1.



(a)



(b)

Fig. 5. (a) Desired (solid line) and Output (Dashed line) of ANFIS 1; (b) Prediction error

5.1.2 ANFIS 2

In this ANFIS 2 the sigmoid membership function was used, the prediction of output generated by the ANFIS 2 is shown in Fig. 6. (a), showing the trained data and the output of ANFIS 2, their differences are shown in Fig. 6. (b), which shows the prediction error between the desired data and data predicted by the ANFIS 2.

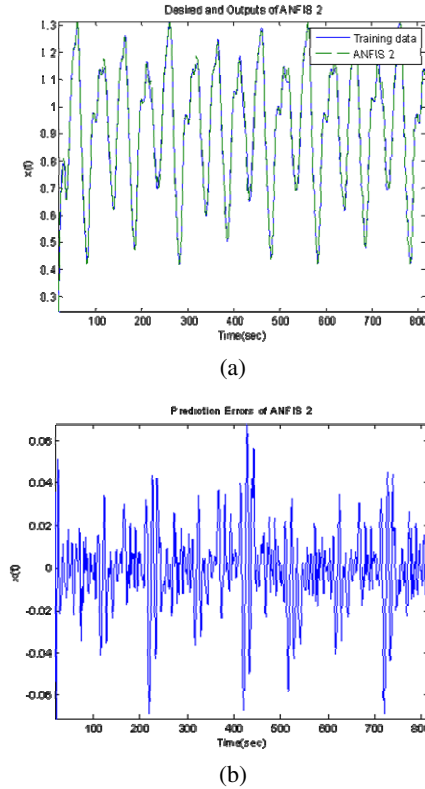
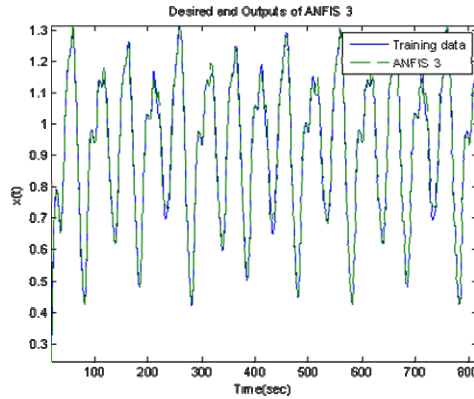


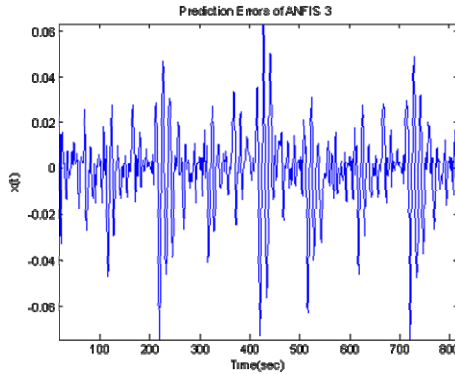
Fig. 6. (a) Desired (solid line) and Output (Dashed line) of ANFIS 2; (b) Prediction error

5.1.3 ANFIS 3

In this ANFIS 3 the Π -shaped built-in membership function was used, the prediction of output generated by the ANFIS 3 is shown in Fig. 7. (a), showing the trained data and the output of ANFIS 3, their differences are shown in Fig. 7. (b), which shows the prediction error between the desired data and data predicted by the ANFIS 3.



(a)



(b)

Fig. 7. (a) Desired (solid line) and Output (Dashed line) of ANFIS 3; (b) Prediction error

5.2 Integration

This section presents the results obtained by the methods de integration of average and weighted average, which one used for simulation in our experiments of the Ensembles of ANFIS in order to obtain a good forecast of the simulated time series.

5.2.1 Integration by Average

For obtaining the Ensemble average result of ANFIS, the integration method of average was implemented, where the results of each ANFIS are add and divided between the same numbers of ANFIS, this result is shown in Fig. 8. (a), which compares the actual data trained against the integrator settles for average, their differences are shown in Fig. 8. (b), which shows the prediction error between the desired data and data predicted by integrating on average.

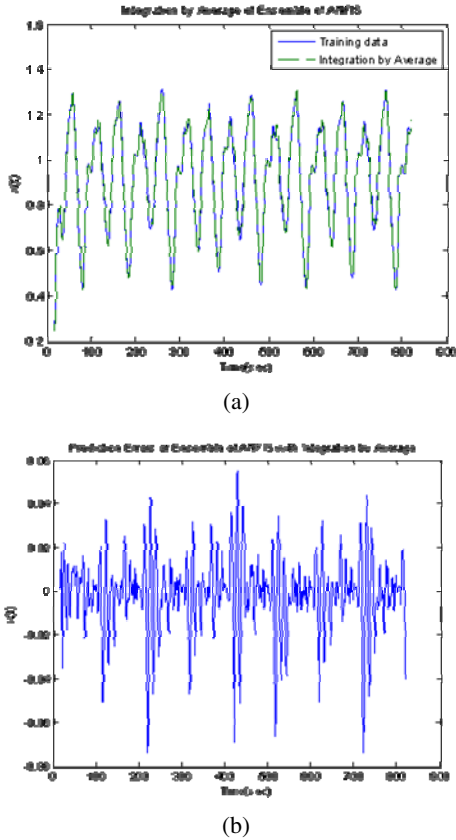
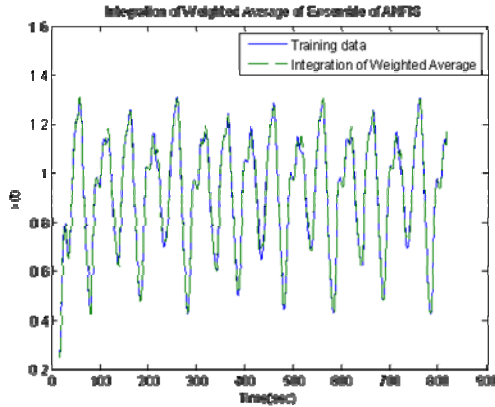


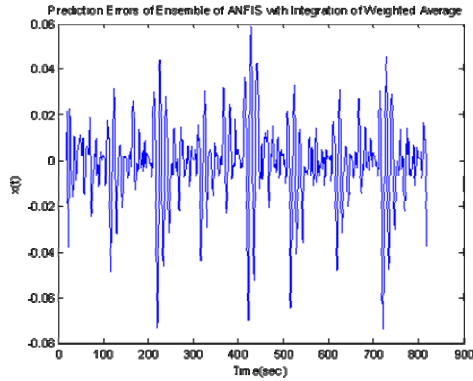
Fig. 8. (a) Desired (solid line) and Output (Dashed line) of integrator by average; (b) Prediction error

5.2.2 Integration of Weighted Average

For obtaining the result in a more accurate Ensembles of ANFIS, in this case the method of integration by weighted average is implemented, which was given a weight depending on the results obtained from each ANFIS, these weights were assigned manually, where the lowest error among the ANFIS output is assigned a weight of 0.50, the ANFIS that had the second best average error was assigned 0.30 and the ANFIS with biggest error was assigned a weight of 0.20, thus obtaining 100% of the weights assigned among the Ensembles ANFIS. Fig. 9. (a), shows the comparison between the training data and output data generated by the integration by weighted average, their differences are shown in Fig. 9. (b), which shows the prediction error between the desired data and data predicted by integrative of weighted average.



(a)



(b)

Fig. 9. (a) Desired (solid line) and Output (Dashed line) of integrator of weighted average; (b) Prediction error

5.3 Results

The results shown in Table 1 are the 5 best experiments obtained with the Ensembles of ANFIS, it is noteworthy that in each experiment we varied the type of membership functions and the goal error, thereby increasing the complexity to Ensemble ANFIS.

Table 1. Results of different Ensembles of ANFIS

Ensemble	ANFIS	Error Goal	Kind of MF	RMSE	Integration by Average	Integration of Weighted Average
1	ANFIS 1	0.01	gbellmf	0.0187	0.0183	0.0182
	ANFIS 2		gaussmf	0.0189		
	ANFIS 3		gauss2mf	0.0179		
2	ANFIS 1	0.001	trimf	0.0202	0.0174	0.0173
	ANFIS 2		dsigmf	0.0186		
	ANFIS 3		pimf	0.0176		
3	ANFIS 1	0.0001	gauss2mf	0.0181	0.0176	0.018
	ANFIS 2		gbellmf	0.0187		
	ANFIS 3		dsigmf	0.0186		
4	ANFIS 1	0.00001	gaussmf	0.0189	0.0183	0.0182
	ANFIS 2		gauss2mf	0.0181		
	ANFIS 3		gbellmf	0.0187		
5	ANFIS 1	0.000001	trimf	0.0202	0.0178	0.0175
	ANFIS 2		gauss2mf	0.0181		
	ANFIS 3		pimf	0.0176		

6 Conclusions

In conclusion we can say that the results obtained with the proposed Ensembles of ANFIS architecture have been good and positive in predicting time series (like the Mackey-Glass), as it has managed to minimize the prediction error of the time series against the results obtained by other researchers.

These results obtained by our Ensembles of ANFIS architecture, managed to reach 98% of prediction, which offers an efficient outcome and avoids us prevent unexpected events in the future.

As future work we will also consider the Mexican Stock Exchange and the Dow Jones time series, to verify that the proposed approach can provide good results and that they can be better than those obtained by other researchers.

References

- [1] Brocklebank, J.C., Dickey, D.A.: SAS for Forecasting Series, pp. 6–140. SAS Institute Inc., Cary (2003)
- [2] Brockwell, P.D., Richard, A.D.: Introduction to Time Series and Forecasting, pp. 1–219. Springer, New York (2002)
- [3] Castro, J., Castillo, O., Melin, P., Rodriguez, A.: A Hybrid Learning Algorithm for Interval Type-2 Fuzzy Neural Networks: The Case of Time Series Prediction, vol. 15a, pp. 363–386. Springer, Heidelberg (2008)
- [4] Filev, D., Yager, R.: On the issue of obtaining OWA operator weights. *Fuzzy Sets and Systems* 94(2), 157–169 (1998)
- [5] Jang, J.-S.R.: Fuzzy modeling using generalized neural networks and Kalman filter algorithm. In: Proc. of the Ninth National Conference on Artificial Intelligence, AAAI 1991, pp. 762–767 (1991)
- [6] Jang, J.-S.R.: Rule extraction using generalized Neural Networks. In: Proc. of the 4th IFSA World Congress, pp. 82–86 (1991)
- [7] Jang, J.-S.R.: ANFIS: Adaptive-network-based fuzzy inference systems. *IEEE Trans. on Systems, Man, and Cybernetics* 23, 665–685 (1992)
- [8] Mackey, M.C., Glass, L.: Oscillation and chaos in physiological control systems. *Science* 197, 287–289 (1997)
- [9] Mackey, M.C.: Mackey-Glass. McGill University, Canada (September 5, 2009), http://www.scholarpedia.org/-article/Mackey-Glass_equation
- [10] MATLAB 2007, Historical Mackey Glass data (September 1, 2008)
- [11] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
- [12] Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. on Systems, Man, and Cybernetics* 15, 116–132 (1985)
- [13] Takagi, T., Sugeno, M.: Derivation of fuzzy control rules from human operation control actions. In: Proc. of the IFAC Symp. on Fuzzy Information, Knowledge Representation and Decision Analysis, pp. 55–60 (1983)
- [14] Thomas, G.D.: Machine Learning Research: Four Current Directions. *Artificial Intelligence, Magazine* 18(4), 97–136 (1997)
- [15] Werbos, P.: Beyond regression: New tools for prediction and analysis in the behavioral sciences. PhD thesis, Harvard University (1974)
- [16] Zhou, Z.-H., Wu, J., Tang, W.: Ensembling neural networks: many could be better than all. *Artificial Intelligence* 137(1-2), 239–263 (2002)

Modeling Facial Expression of Intelligent Virtual Agents

María Lucila Morales-Rodríguez, Fabián Medellín-Martínez,
and Juan J. González B.

Instituto Tecnológico de Ciudad Madero, División de Estudios de Posgrado e Investigación,
Cd. Madero, Tamaulipas, Mexico
{lmoralesrdz, fabermed}@gmail.com,
jjgonzalezbarbosa@hotmail.com

Abstract. An intelligent virtual agent is an intelligent agent with a digital representation and endowed with conversational capabilities; this interactive character exhibits human-like behavior and communicates with humans or virtual interlocutors using verbal and nonverbal modalities such as gesture and speech. Building credible characters requires a multidisciplinary approach, the behaviors that must be reproduced are very complex and must be decoded by their interlocutors. In this paper we introduce a kinesics model in order to select the nonverbal expression of intelligent virtual agents able to express their emotions and purposes using gestures and face expressions. Our model select the nonverbal expressions based on causal analysis, and data mining algorithms applied to perceptual studies in order to identify significant attributes in emotional face expressions.

1 Introduction

The intelligent virtual agents are increasingly present on the user interfaces because they try to simulate the forms of human communication and thus to improve the human-machine interaction. Their design involve a multidisciplinary approach combining computer sciences, psychological, sociological and philosophical issues [1], the challenge is to endow them with humans characteristics like emotions and personality, and thus to allow the suspension of the disbelief.

An intelligent virtual agent is an intelligent agent with a digital representation and endowed with conversational capabilities; when this interactive character exhibits human-like behavior and communicates with humans or virtual interlocutors using verbal and nonverbal modalities such as gesture and speech they are known as Embodied Conversational Agent (ECA) [2]. These kinds of agents try to transmit an illusion of life and reproduce the richness and the dynamics of the human

behavior [3]. Characters able to transmit this illusion could more easily produce an immersion's experience in a virtual world. They are being used in educative applications like virtual tutors, or as presenters or guides in visits to virtual museums. Also they could be found as salesmen or real estate agents, personal trainers and therapists, among others applications.

These agents become visible in the social interfaces like animated actors who represent human beings and are used to improve the interactivity with the users. In order to improve this interactivity, it is not enough that the virtual character looks real to be believable and catch the interest of the user. The expression of the interaction must be given in a natural way, for that reason, the character must behave and communicate in the same way that a human do, using verbal and nonverbal expressions. Gratch et al [1] identify that the face-to-face conversation, emotions and personality, and human figure animation are the key issues that must be addressed in creating virtual humans.

Following these issues, the creation of virtual characters has evolved, starting with the creation of fantasy characters until virtual humans. The first projects developed worked with a specific set of facial expressions from a corpus of facial expressions product of the work conducted by Ekman and his colleagues[4]. These projects were talking faces trying to reproduce emotional expressions [5-7] and specific nonverbal functions in order to synchronize the verbal expression [8]. Other works express blend of emotions of different types and performative of the communicative act being performed [9].

The interest of our work is the selection of the facial expression of a virtual character that can express itself in a nonverbal way his emotional context coherence. The inherent difficulty to this task occurs because the human being can experience a variety of emotions, which are represented by a wide range of expressions, and are influenced by the traits of personality and the context.

In this paper we introduce a kinesics model in order to model the nonverbal expression of intelligent virtual agents able to express their emotions and purposes using gestures and face expressions. Our model is based on causal analysis, and data mining algorithms applied to perceptual studies in order to identify significant attributes in emotional face expressions and thus design a selection model of facial expressions.

2 Expressing Emotional Behavior

There are several ideas about what can give believability to virtual agents. For Badler [3], what makes a virtual human « human » is not just a well-executed exterior design but movements, reactions, and decision-making which appear « natural », appropriate, and contextually-sensitive. To be able to produce this behavior, these life-like characters must communicate credibility in their emotional expressions (coherent reaction), exhibit personality and be able to express their indexical

and reflexive behavior, which are transmitted by verbal, paraverbal and non verbal activity, showed through facial, vocal and gestural expressions.

In this work we used a behavioral model proposed by Morales [10] to handle the perceptions of the agent and determine its actions, on the basis of his emotional state and traits of personality. This model works with a series of attributes that define the emotion, mood, attitude, intention and objective of the agent. These elements models the behavior and are constantly updated, we characterized and used them for the selection of the emotional expression.

The Fig. 1 shows the general architecture of a virtual character, where the output data of the behavioral model are taken by a Kinesics model in order to select and integrate the no verbal expression of the virtual character.

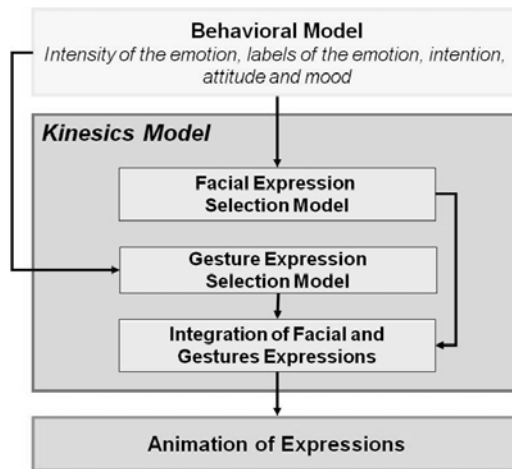


Fig. 1. Components of a general architecture for the animation of a virtual agent

An interesting detail to be mentioned is the way in that the behavioral model handles the emotional state of the agent. In order to handle and to represent the emotions of the agent a circumflex model is used, divided in eight regions by four axes that represent basic psychological concepts. According to this approach, an emotion is represented by a set of psychological characteristics that allow to define it and to classify it. The circumflex in Figure 2 shows the distribution of the top four recognized emotions: fear, anger, sadness and joy.

The emotion label selection is based on its position in the space delimited by the four axes. For example, the emotion of anger is in the portion of space characterized by positive values of stress, arousal and stance, as well as a negative value of valence.



Fig. 2. Classification of the emotions in four dimensions according to the proposed circumflex model in Morales [10]

In this paper we detail the process related to the facial expression in the kinetics model, we examined the existing relations between the face expressions and the attributes given by the behavioral model like intention, objective, emotion, mood and attitude, these relations are going to be used to determine the design of the selection model.

The facial expressions are created by all the possible combinations of a total of 5 eyebrows and 4 mouths shapes, giving a total of 20 different images (see Fig. 3).

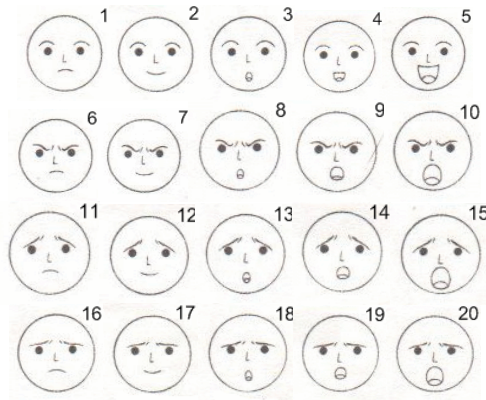


Fig. 3. The twenty resulting faces of the combinations of the different eyebrows and mouth shape

3 Characterization of Behavioral Attributes

In order to analyse the existing relations between the face expressions and the attributes given by the behavioral model, we implemented a survey where the

participants characterized the set of twenty images using the attributes attitude, intention, mood and emotion of the behavioral model.

The information generated, was used to develop a causal analysis to observe the weight of each attribute. We did the causal analysis using Tetrad¹ (Fisher's Z test, alpha value of 0.01) and we obtain that the attribute « mood » is the most significant of all (Fig. 4).

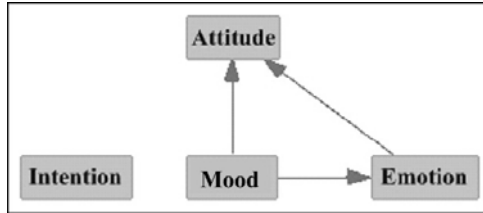


Fig. 4. Causal analysis of the attributes that characterized the face expressions

We also apply data mining using Naive-Bayes and ID3 classifiers to identify the best method to predict the expression that corresponds with the attributes of the behavioral model. The Fig. 5 shows the percentage of success of these classifiers, we note that the Naive-Bayes classifier was the best classifier with a 50% of success.

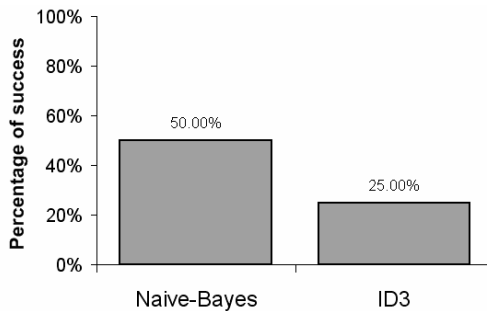






















Fig. 5. Success percentage of classifiers

Although the Naive-Bayes classifier has low performance, we can observe that the predicted images do not vary too much from the expected one (Table 1).

¹ In <http://www.phil.cmu.edu/projects/tetrad> it is possible to find more information and free download software of Tetrad Project.

Table 1. Comparison between the results obtained by the Naive-Bayes classifier with the image that we expected in the cases in that the classifier failed

Expected	Obtained	Expected	Obtained
 4	 5	 11	 15
 5	 2	 13	 14
 7	 6	 16	 14
 8	 6	 18	 3
 9	 10	 20	 19

In order to distinguish the face expressions, we develop another survey that ask to the participants the identification of the emotion expressed by the face through the characterization of the four axes showed in the circumflex. With the new information we can built a new circumflex, with the most popular answers of the survey (See Fig. 6).



Fig. 6. New allocations of the emotional labels in the circumflex

This last survey, gives us the information to verify the interpretation of an emotion based on the four dimensions that characterize the circumflex. Now we are able to use the data of the emotional axes to identify an emotion and their intensity. This information has been used in the design of the selection model.

4 Design of the Face Expression Selection Model

The selection model of face expressions works with the information generate by the behavioral model and with the corpus product of the surveys.

The model uses the corpus that has been classified by the Mood and by the four axes that characterizes an emotion. This corpus is processed in two layers in order to reduce it and treated by a Naïve Bayes algorithm.

The face expressions selection model can be appreciated in Figure 7 and their main algorithm in Table2.

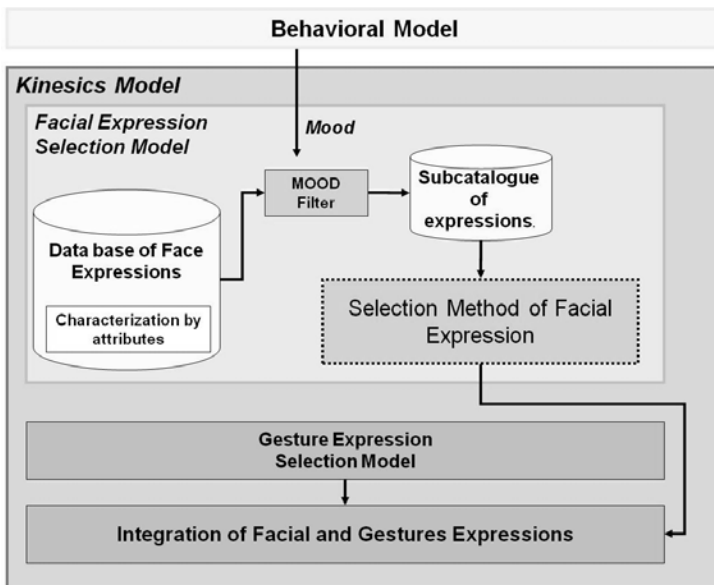


Fig. 7. Structure of the Selection Model of Facial Expression

The first layer of the model apply a mood filter, where we will choose only the images that were characterized with the current mood value of the behavioral model.

We have chosen mood as the filter attribute, because of the causality tests. The result of this filter will give us a sub catalogue of expressions.

Table 2. Algorithm of the selection model of face expressions

Facial_Expression_Model(Facial_Ex_DB, attributes)	
1.	For each gesture \in Facial_Ex_DB such that moodgesture = moodattributes
2.	add gesture into First_Candidates_Facial_Ex
3.	end for each
4.	Final_Facial_Ex \leftarrow ExpressionSelection (Facial_Ex_DB , First_Candidates_Gestures, attributes)
5.	return Final_Facial_Ex

This sub catalogue and the attributes of attitude, intention and emotion are used in the second layer : (see Fig. 8).

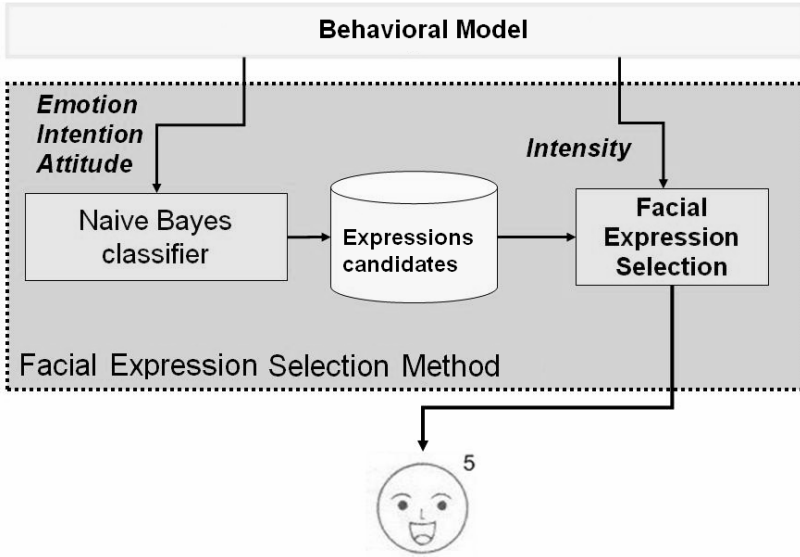


Fig. 8. Structure of the Selection Method of Facial Expression

This second process selects from the database the face expression that better match to the label of the emotion, intensity and attitude. In Table 3 we observed the detail of the selection process.

Table 3. Function ExpressionSelection

ExpressionSelection (First_Candidates_Facial_Ex, attributes)	
1.	Second_Candidates_Facial_Ex \leftarrow TopRankingNaiveBayes(First_Candidates_Facial_Ex, attributes)
2.	Final_Facial_Ex \leftarrow FaceExpSelection(Second_Candidates_Facial_Ex, attributes)
3.	return Final_Facial_Ex

This method calls a Naive Bayes Classifier to select the three better expressions (function TopRankingBayes, see Table 4), these face expressions could be variations of a same emotion affected by its intensity.

Table 4. Function that return the three Top Ranking of NaiveBayes

TopRankingNaiveBayes (Facial_Ex, A)
1. For each $f \in$ Facial_Ex
2. Probabilities $f \leftarrow 0$
3. For each $a \in A$
4. Probabilities $f \leftarrow$ Probabilities f + ObtainProbability(a)
5. end for each
6. End for each
7. Order Facial_Ex by Probabilities
8. return Facial_Ex 0 and Facial_Ex 1 and Facial_Ex 2

The next step uses the intensity of the emotion to select between the group of three facial expressions with the highest Naive Bayes probabilities, see the function FaceExpSelection (Table 5).

Table 5. Algorithm with the final selection

FaceExpSelection (Facial_Ex_DB , Candidates_Facial_Ex, A)
1. Intensity \leftarrow EmotionIntensity(ArousalA, StressA, StanceA, ValenceA, α)
2. For each gesture \in Candidates_Facial_Exgesture
3. sum $\leftarrow 0$
4. For each emotion \in Gestureemotions
5. Probabilities \leftarrow Probabilityemotion * Emotionintensity
6. sum \leftarrow sum + Probabilities
7. end for each
8. Probabilitygesture \leftarrow sum
9. end for each
10. return Max Probabilitygesture \in Candidates_Facial_Exgesture

The intensity of the emotions involved in the three better facial expressions is calculated by the operation described in Table 6. This calculus take the axes values related with the definition of the emotion. The intensity is obtained from a sum of the values of the four axes, where the axes are going to have a different weight. The adjacent axes to the label of the emotion have a smaller weight than the nonadjacent axes.

















Table 6. Intensity calculation of each emotion label

Emotion Label	Intensity Calculation
Anger	$(\alpha)[(-1)(Valence) + Stance] + (1-\alpha)[Stress + Arousal]$
Surprise	$(\alpha)[Valence + Stress] + (1-\alpha)[Stance + Arousal]$
Joy	$(\alpha)[Arousal + (-1)(Stress)] + (1-\alpha)[Stance + Valence]$
Pleasure	$(\alpha)[(-1)(Arousal) + Stance] + (1-\alpha)[(-1)(Stress) + Arousal]$
Relaxation	$(\alpha)[Valence + (-1)(Stance)] + (1-\alpha)[(-1)(Stress) + (-1)(Arousal)]$
Sadness	$(\alpha)[(-1)(Valence) + (-1)(Stress)] + (1-\alpha)[(-1)(Stance) + (-1)(Arousal)]$
Distress	$(\alpha)[(-1)(Arousal) + Stress] + (1-\alpha)[(-1)(Stance) + (-1)(Valence)]$
Fear	$(\alpha)[Arousal + (-1)(Stance)] + (1-\alpha)[Stress + (-1)(Valence)]$

5 Experimental Results

In Table 7, we present a sequence of executions of the Behavioral Model and the experimental results obtained.

Table 7. Executions of the system

Input	Axes	Naïve Bayes Candidate Images			Final Selection
Mood: Negative	Stress : 0.639	 20	 19	 18	 18
Emotion: Fear	Arousal: 9.0	Naive Bayes 0.00291307382	Naive Bayes 0.001356211169	Naive Bayes 0.00120193359	
Attitude: Directive	Stance: -1.164	Intensity 0.0523153323	Intensity 0.013809930878	Intensity 0.083002777777	
Intention: Demand	Valence: -4.343				
Mood: Negative	Stress: 0.513	 20	 13	 18	 18
Emotion: Fear	Arousal : 9.0	Naive Bayes 0.00145653691	Naive Bayes 0.000997146937	Naive Bayes 0.000801289062	
Attitude: Directive	Stance : -0.156	Intensity -0.032766633	Intensity 0.014757436922	Intensity 0.063547222222	
Intention: Inform	Valence : -1.156				
Mood: Neutral	Stress : 0.41	 7	 14	 13	 7
Emotion: Fear	Arousal : 9.0	Naive Bayes 0.0000000495	Naive Bayes 0.00000002828	Naive Bayes 0.0000000165	
Attitude: Directive	Stance : -0.125	Intensity -0.03043611111	Intensity -0.0778872237	Intensity -0.0611533859	
Intention: Demand	Valence : -0.978				
Mood: Neutral	Stress : 0.363	 11	 14	 13	 13
Emotion: Fear	Arousal : 9.0	Naive Bayes 0.00015470312	Naive Bayes 0.000000028288	Naive Bayes 0.000000090008	
Attitude: Directive	Stance : -0.123	Intensity -0.1152013888	Intensity -0.0794237301	Intensity -0.06280523702	
Intention: Reprimand	Valence : -0.907				

The table shows the inputs to the selection model, the three images with the best probabilities according to Naive Bayes classifier and the image of the final selection. Each image presents their probabilities and emotional intensity values. The final selection is the image with the maximum intensity.

In these examples we observe that in some cases, the final image selected does not correspond with the image of the maximum probability of the Naive Bayes classifier, this could be influenced by the use of several inputs (attributes) with same weight in the classifier. The addition of the emotional intensity gave to the final selection a greater importance to the emotional state and define the final selection. This procedure follows the results of the causal analysis, where the emotion is second attribute with a greater weight.

6 Conclusions

In this paper we present a model of facial selection as part of a kinesics model for intelligent virtual agents able to express their emotions and purposes using gestures and face expressions.

The difficulty of this task occurs because the human being can experience a variety of emotions, which are represented by a wide range of expressions, which are influenced by the traits of personality and the context. Our model selects the face expressions influenced by a causal analysis of the attributes implied in the behavioral model of the virtual character and by a Naive Bayes Classifier. The selection model processing a face corpus according to the weight of these attributes and intensity. The model searches provide a contribution to the improvement of the virtual humans believability.

Acknowledgements

This works has been product of the project 2457.09-P supported by the Dirección General de Educación Superior Tecnológica (DGEST) de la Secretaría de Educación Pública (SEP) and the Instituto Tecnológico de Ciudad Madero.

References

1. Gratch, J., et al.: Creating Interactive Virtual Humans: Some Assembly Required. *IEEE Intelligent Systems* 17(4), 54–63 (2002)
2. Cassell, J., et al. (eds.): *Embodied Conversational Agents*, vol. 426. MIT Press, Cambridge (2000)
3. Badler, N.: Virtual humans for animation, ergonomics, and simulation. In: *IEEE Workshop on Non-Rigid and Articulated Motion*, Puerto Rico (1997)
4. Ekman, P., Friesen, W.V., Hager, J.C. (eds.): *Facial Action Coding System: The Manual and User Guide*. Salt Lake City, Human Face (2002)
5. Bui, T.D., et al.: Generation of Facial Expressions from Emotion Using a Fuzzy Rule Based System. In: *Australian Joint Conference on Artificial Intelligence*. Springer, Adelaide (2001)
6. De Rosis, F., et al.: From Greta's Mind to her Face: Modelling the Dynamics of Affective States in a Conversational Embodied Agent. *International Journal of Human-Computer Studies* 59(1-2), 81–118 (2003)
7. Pelachaud, C., Bilvi, M.: *Computational Model of Believable Conversational Agents* (2002)
8. DeCarlo, D., et al.: Specifying and animating facial signals for discourse in embodied conversational agents. *Computer animation and virtual worlds* 15, 27–38 (2004)
9. Pelachaud, C., Poggi, I.: Facial Performative in a Conversational System. In: *Workshop on Embodied Conversational Characters*, WECC 1998, Lake Tahoe, USA (1998)
10. Morales Rodríguez, M.L.: *Modèle d'interaction sociale pour des agents conversationnels animés. Application à la rééducation de patients cérébro-lésés*. PhD Thesis. PhD These: Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier, p. 108 (2007)

Agent Communication Using Semantic Networks

Iván Espinoza-Hernández, Dora-Luz Flores, Antonio Rodríguez-Díaz,
Manuel Castañón-Puga, and Carelia Gaxiola

Universidad Autónoma de Baja California, Facultad de Ciencias Químicas e
Ingeniería, Tijuana, Baja California, México, CP. 22390
{espinoza.ivan,dflores,ardiaz,puga,cgaxiola}@uabc.edu.mx

Abstract. The ability to understand the world around us is crucial to successfully act within an environment. For such a task it is important to understand relations and context between objects and actions. The goal is to have an agent capable of understanding its environment and by using knowledge of how it works to take actions. The term environment is defined as physical places, virtual place and social situations an agent can encounter. Current agent architectures do not specify how knowledge should be handled from its environment or how to generate knowledge from experiences. This work focuses on using semantic networks to represent information an agent acquires from its environment, allowing communication in a more meaningful way by transferring information from the semantic network. An agent is able to have an abstraction of the environment to the extent that it is capable of interacting within it. Experiments with this method are done using a custom tool called Wiinik, this tool allows creating hierarchical agents using scripts as behavior rules. Semantic networks, although not standardized, have great potential to model any kind of knowledge, regardless of an agent's capability. In this paper we describe how knowledge modeled as semantic networks is beneficial for agent decisions and facilitates the transfer of information between agents. Additionally, we describe the architecture of a software tool developed for social simulations and we discuss two case studies that were implemented using this software.

Keywords: Semantic networks, Communication, multi-agent systems, rule engine, fuzzy systems.

1 Introduction

Humans are social beings by nature. We rely on others to perform specialized tasks within a community and each individual plays a role. Only the ones with the correct knowledge and ability can perform certain tasks adequately even though most members are physically capable of performing such tasks. The complexities of the interactions of any modern or primitive society go well beyond these

statements, but they do provide a good starting point for developing behavior models and software tools that would allow us to implement simple agents that are capable of interacting with each other within a limited environment.

Acquiring and sharing information amongst a group is a vital part in the development of any society. Each discovery will lead a group closer to a common goal and thus ensuring the group's survival or well being. Being able to represent an agent's acquired knowledge in such a way that could be easily extended to different yet similar domains would greatly increase the ability to solve problems. Furthermore, the ability to transmit knowledge between individuals requires certain abilities and a common knowledge of basic concepts that can be represented by sounds, written signs or gestures. Simulating these abilities in a limited environment presents many challenges, amongst which are the general representation of any kind of knowledge, the extraction of relevant information from that knowledge and transmitting in a way that can be understood by any number of intended receivers.

One of the reasons for developing Wiinik is to create a platform that would allow us to experiment with different behavior theories and knowledge representations that can ultimately result in agents capable of understanding their environment and use knowledge of how it works to take action. The ability to explore different possibilities helps us answer questions as to which approaches perform best in specialized situations and in general situations. Ultimately being able to optimize general situation methods to specific situations would allow agents to adapt to changing environment and work with partial information.

Probability theory has been used in agent-based modeling in order to model social processes and dynamics [11]. These models are based on the interactions of individual entities, named "agents". These agents use simple behavioral rules, local to their simulated environment [12]. The use of Fuzzy Logic to model personality in agents has previously shown satisfactory results [18]. Computational models such as AvatarSim, FLAME, and PETEEI use fuzzy logic to simulate emotions and personality in agents [5]. In Wiinik, an agent's behavior is determined by a fuzzy inference system (FIS), which is designed to define the agent's psychological elements. Semantic networks are used to represent the information acquired by the agent from its environment, allowing the agent to communicate in a more meaningful way. Semantic networks, although not standardized, have great potential to model any kind of knowledge, regardless of an agent's capability.

Another important factor for social developments and interactions is the surroundings. The surrounding environment determines what options are available and what goals becomes a priority, among other things. In order to work with a simplified model yet still have results that can relate to real situations we consider three types of environments: physical, virtual and social. A physical environment represents what an agent can perceive as input from the physical world. A virtual world is a combination of concepts that can be based on previous physical environment experiences or fictional experiences. Lastly, a social environment represents the rules that determine what relation objects have to a current situation, what is appropriate to do, and other similar information. An agent is able to have an abstraction of the environment to the extent that it is capable of interacting within it.

Our work on the Wiinik platform has been focused on addressing all of these issues. We are able to work with multiple agents and environments interacting and behaving according to scripts based on logic rules. Behavior scripts have great flexibility that allows the implementation of different behavior patterns and mechanisms for decision-making. Similar work can be found for language processing[13] and behavior modeling[10], since each is a monumental task on its own, few alternatives are available for experimenting with both.

In the following sections, we present a brief description of important concepts used throughout this document, a description of the Wiinik platform is discussed and in the final sections we present a discussion of two case studies that have been implemented using Wiinik.

2 Basic Definitions

In this section basic definitions for semantic networks and agents are provided before describe how each concept contributes toward simulating human behavior.

2.1 Semantic Networks

A semantic network is a simple representation scheme that uses a graph of labeled nodes and labeled directed arcs to encode knowledge. Nodes represent either objects, concepts or events and arcs represent relationships between nodes [16]. As an extension of directed graphs, semantic networks can be formally defined as

$$G = (V, E). \quad (1)$$

where V is the set of nodes and E is the set of arcs denoting the different unions between nodes. Different types of semantic networks have been defined such as IS-A, conceptual graph and frames in [2,14,19]. Unfortunately one of the greatest strengths of semantic networks for representing knowledge is also one of its greatest weaknesses, almost any idea that is understood can be expressed in a semantic network but this greatly depends on the interpretation of the designer and there is no single correct answer to the description. This facilitates descriptions of objects and phenomena at the expense of complicating interpretation, data extraction and transmission.

2.2 Definition of Agents

An agent is defined as a software entity which functions continuously and autonomously in a particular environment, often inhabited by other agents and processes. A formal definition of an agent A_g is given as

$$A_g = \langle Sit, Act, Dat, f_{A_g} \rangle. \quad (2)$$

where:

1. *Sit* is a set of situations an agent can be in,
2. *Act* is a set of actions that can be performed by an agent,
3. *Dat* is a set of values that an agent internal memory can have, and
4. f_{Ag} is the behavior function defined as $f_{Ag} : Sit \times Dat \rightarrow Act$ depends on the current situation and the current memory values to determine the next action [6].

The main focus of this work is on the element *Dat*, this component will consist of a semantic network containing all the knowledge an agent has acquired. Relevant information from this set is extracted as needed either to decide future actions and to exchange ideas with other agents. For this work every agent needs to have an understanding of basic concepts that are defined within *Dat*. The amount of information that this set can hold depends greatly on the application or experiment that an agent is designed for.

The *Act* set limits an agent to perform only actions that are defined within this set. Changing this set would have significant effects on f_{Ag} results and thus changing the experiences that are lived within an environment. This parameter is useful in order to simulate different types of beings or same beings with slightly different physical abilities.

f_{Ag} is defined as a behavior rule script given to each agent. As knowledge is acquired and new methods are discovered, an agent can add new rules or modify old rules in order to adapt to changing circumstances. This also allows for agents that were identical at one point to behave differently with time.

3 Description of Wiinik Tool

Wiinik is a software tool developed to help model human behavior, interactions and reasoning in an effort to imitate as closely as possible the constraints that exist in real life as best as they are understood.

This software allows for many possibilities by combining several technologies, such as Multi-Agent Systems (MAS), rule-based programming, fuzzy logic, and more.

Wiinik follows the agent model proposed by [4], where each agent is made up of one *Persona* agent and one or more cognition agents. In section 3.1 we will describe this type of agent and others that we used. This allows for the Transactional Analysis theory [1,20] to be easily implemented and experimented on. Each agent may have a unique composition giving greater diversity on the types of situations and theories that can be simulated.

3.1 Definition of a MAS

Wiinik makes use of the open source platform Java Agent Development framework (JADE) [3] that handles all of the MAS operations. These include communication and mobility among others. JADE was chosen as it is fully implemented in Java, compliance with the Foundation for Intelligent Physical Agents (FIPA) standards [7], and its role as middle-ware allows quick and organized development.

3.2 Rule Engine

The rule engine is a very important approach of this project's design goal. Making use of such a tool allows for an agent to be easily customized without affecting any other and can even change its own code during the simulation.

Jade Expert Shell System (JESS) [8] is what was chosen for this project. Because it is implemented in Java, it can be easily integrated with the JADE platform. JESS is used as the scripting language to handle the knowledge base and interaction rules for each agent. When an agent is created it is given a name and a JESS script to control its behavior.

3.3 Agent Mechanics

Each agent is designed to resemble a real person as closely as possible, thus its behaviors are designed to represent physical actions instead of fixed instructions to perform. Fig. 1 shows a diagram of an internal agent structure and interaction.

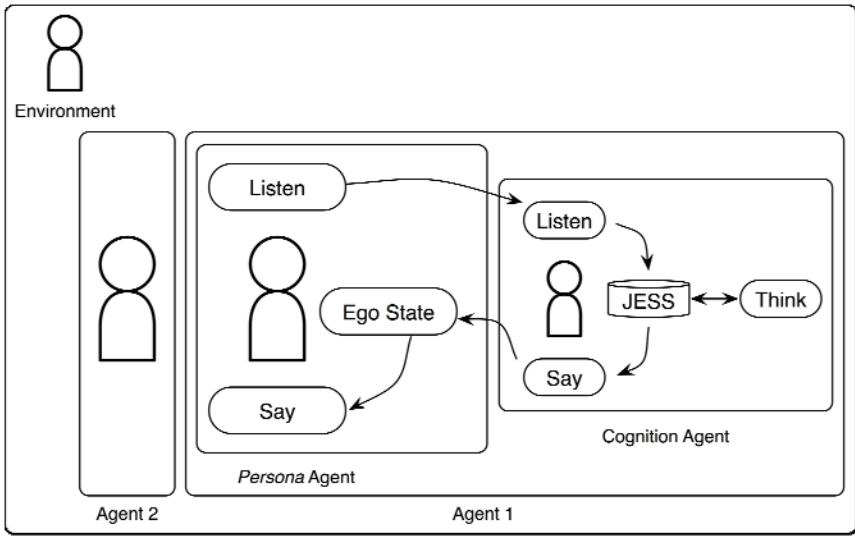


Fig. 1. Internal agent structure

An agent action set Act from tuple 2 is defined as $Act = \{Say, Listen\}$ for message passing only. But, if we include all possible actions then we have

$$Act = \{Say, Listen, Think, Move\}. \tag{3}$$

where:

1. *Say* behavior. This behavior is scheduled when an agent needs to communicate. It will arrange the message in a proper format and send it to the corresponding receiver.

2. *Listen* behavior. This behavior is responsible for receiving any incoming messages and either resend to all cognition agents or add message to knowledge base. Only messages coming from other *Persona* agents are processed in this way.
3. *Think* behavior. Whenever there is new input to process, this behavior will allow the rule engine to execute and if there is any output it will process it accordingly.
4. *Move* behavior. Represents the ability of an agent to move between different environments.

Figure 1 shows an Ego State behavior; here all cognition agent messages are received by this behavior. Once all agents have responded to an event, it takes the message with greater value and creates a Say behavior so that it can be sent. Finally, rule engine component is what gives an agent the ability to process the input collected from the environment and output actions to be performed or generates new knowledge based on information already obtained.

Definition of *Persona* Agent. *Persona* agents are named after the Latin word “Persona” which translates to mask. The purpose of this type of agents is to manage all the actions that are sent by Cognition agents and choose which to perform. It consists three main behaviors: Listen, Ego State, and Say. Received messages are sent to all grouped cognition agents and their output is received by the Ego State behavior for evaluation, once a decision is made, the agent takes the appropriate action.

Definition of Cognition Agent. The Cognition agent allows the agent as a whole to think and act. Even though its architecture is very similar to a *Persona* agent, there are a few key differences. The knowledge base receives input when a message is received from the *Persona* agent or when it is informed of a new agent being created. The rule engine is activated for every new input. If there is any output, a Say behavior will be assigned to inform the *Persona* agent and have it evaluate the decision.

Definition of Environment Agent. When dealing with agent interaction within an environment, there is a need to keep track of some values. This agent can handle any information necessary such as keeping track of the population, weather and traffic among others and can be consulted.

3.4 Message Structure

A message M is formally defined as a 7-tuple that holds information that allows a deeper interpretation of messages beyond the literal meaning of the contents. M holds information that is based on perceptions and can be integrated into the knowledge base adding context information for any given event, we have:

$$M = (Id, T, P, C, BPNM, A, Ag_{RS}). \quad (4)$$

where:

1. *Id* is a unique identification value used to associate a message with an event allowing an agent to interact without any confusion when processing many concurrent events.
2. *T* is the topic of the conversation. It is used to identify the context in which a message is used so that it can be interpreted accordingly.
3. *P* is the performative, it states the reason, purpose or intention of the message.
4. *C* is the content of the message. This element can be a simple text message or formatted in such a way that it can be directly integrated into an agent's knowledge base and processed as data with behavior rules. The content can also be formatted as a new behavior rule to be added and thus expand an agent's ability to act upon new data or correct behavior patterns.
5. Behavior Profile Node Message $BPNM = (\lambda, ES_i, ES_u, \pi_i, \pi_u)$. The *BPNM* element is proposed by [9], where:
 1. $\lambda = [0..1]$. λ represents the perception an agent has over a specific event. It is a single real value between 0 and 1 inclusive calculated using fuzzy logic.
 2. $ES = \{Parent, Adult, Child\}$. ES_i and ES_u represent the ego state an agent is currently in and perceives respectively. The way an agent acts, what goals it decides to pursue and what information will be relevant for decisions depends greatly on this value.
 3. $\pi = [0..1]$. π_i and π_u represent the psychological posture an agent has and perceives respectively. It is a single real value between 0 and 1 inclusive calculated using fuzzy logic.
6. *A* is used to identify what messages have been processed. This item helps identify old messages allowing agents to keep a record of previous experiences.
7. Ag_{RS} identifies the sender or intended receiver of the message.

3.5 Behavior Scripts

Agent actions are modeled through behavior scripts that are in the form of logic rules. JESS is used as the rule engine and is the main component for decision-making. JESS scripts can be given to several types of agents for different purposes not limited to controlling actions, they may also facilitate tasks such as agent creation. Fig. 2 shows an example on how to create four agents using JESS scripts and Fig. 3 shows part of what would be an agent's behavior script.

It is possible to specify more than one script for each *Persona* agent, this will create a cognition agent for each script and each will receive whatever input is meant for his or her persona agent. For actions with agents with more than one cognition agent, each resulting action is weighted against every other resulting action and the most urgent action is chosen.

4 Study Cases

Each of the following cases of study focuses on certain features of the Wiinik platform that allow us to simulate a variety of scenarios.

```

(assert (Load (Type TimerAgent)
              (Name "kronos")))
;-----
(assert (Load (Type PersonAgent)
              (Name "agent1")
              (Script ./Scripts/DonJuan/DonJuan.clp)))
;-----
(assert (Load (Type PersonAgent)
              (Name "agent2")
              (Script ./Scripts/DonJuan/Victim.clp)))
;-----
(assert (Load (Type PersonAgent)
              (Name "agent3")
              (Script ./Scripts/DonJuan/Victim.clp)))

```

Fig. 2. Sample script for loading four agents

```

(defrule receive-job "receive job"
  ?h <-(Hear
        (attended ?a&:(= ?a FALSE))
        (id ?msgId)
        (performative "REQUEST")
        (topic "Work")
        (content ?job)
        (sender ?id))
  (me (aid ?aid) (bpNode ?bp))
  =>
  (bind ?jj (assert-string ?job))
  (modify ?jj (MID ?id))
  (printout t "Asserting job " ?job crlf))

```

Fig. 3. Sample behavior rule

4.1 Case 1. “Don Juan” Syndrome

This case of study is based on the pathology described in [15] and on previous work described in [9]. The agents are modeled to follow the typical behavior described in the Don Juan syndrome only when their behavior profiles match and they are in the correct ego state. The behavior scripts also implemented the concept of cathexis [17] which allows the ability to switch between priorities in a stable manner.

The first step in this scenario involves a greeting; this allows each agent to exchange information on how each perceives the other. Based on this information an agent will decide how to act toward each other and decides how to proceed with

the communication. If an agent is not suitable for one's current needs then no further message exchange is done, otherwise the script will continue its course. The information used to decide this behavior is contained within the Behavior Profile Node Message (BPNM).

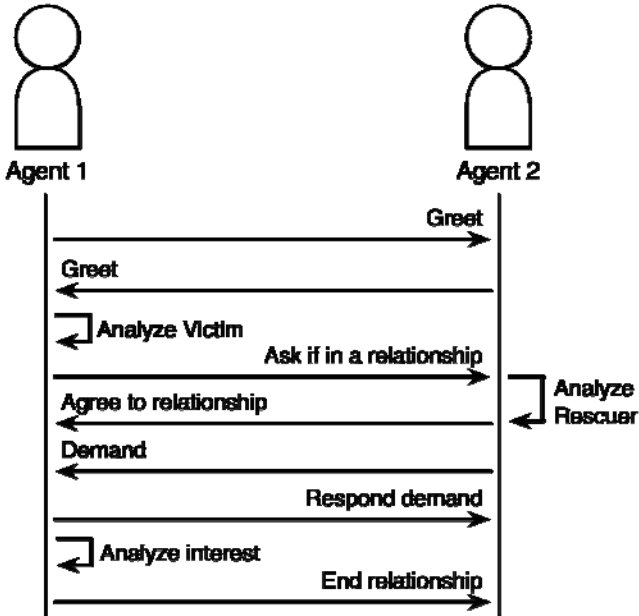


Fig. 4. Don Juan syndrome case study script flow

If the agent acting as a Don Juan should decide that the recently greeted agent is suitable and is not currently already in a relationship then the next step is to propose a relationship to this agent. The second agent would receive the message and decide whether or not the first agent is perceived as a savior and if that is what is desired at the moment.

When the agents agree to be in a relationship a series of messages are exchanged in which the one playing the role of the victim will start demanding things from the Don Juan agent, this increase in commitment will reduce the interest the Don Juan agent has for the victim and will shortly end with the relationship forcing the victim to hate the Don Juan agent and this agent will go on to wait for another agent to start a relationship with.

During this scripted behavior many internal values have changed, mainly in the perception that each has for each other and themselves that allow this sort of behavior to continue. Any number of things could alter the course and may result in the script not being followed to completion if there are any alterations caused by external or internal influences.

Figure 4 shows an uninterrupted exchange of messages between a Don Juan agent and another agent that takes the role of victim.

4.2 Case 2. A Company's Workers

When working toward a common goal, it is important for each member to play their role and to have shared knowledge on important concepts. The success or failure will be decided by the group effort and not by individual achievements. This case of study demonstrates how key information can be standardized among the group yet given slightly different interpretations or give priorities to different parts of the information.

In this case of study, a kind of agent called Manager must first have a record of how many agents type Employees are available to work and how capable they are for the job (this is measured by how tired an employee agent is). Each Employee agent will contact its Manager agent as soon as possible with that information.

Once a manager knows how many employees are available, it can then send a request message to the environment asking for work. The Environment agent will reply with job specifications and await confirmation. Depending on the size of the job and how many employees are available for it, a manager must either accept or decline this job. If the job is declined then the manager will lose credibility and will receive smaller jobs, this is preferred to risking not completing the job on time.

If a job is accepted then a manager will divide the work and give each employee a fair share, not necessarily equal shares. Since a manager agent has knowledge on each employee status, it may choose to give consideration to how capable or tired each agent is, or disregard this information and assign work to tired employees agents. This will cause tired agents to produce many errors and the job may take longer to complete.

When agents are assigned work they will perform their task as best as their current level allows them to. Each employee can have different priorities, some may be workaholics, and some may prefer to rest if tired or have different levels for deciding when they are tired. If an agent is not within a certain level of health then they will have a high probability of producing errors when work is assigned to them, and may choose not to work for some time until they consider themselves fit for work. Once they finish their workload they report to the manager.

When a manager receives enough finished tasks to complete the job then this information is communicated to the environment agent and it is rated depending on the job being completed on time. The cycle then repeats with the manager requesting another job, again considering whether or not the employees are fit to complete it or not.

Figure 5 depicts a simplified case consisting of only one agent of each type. Each manager agent is to acquire jobs and decide whether it is capable of finishing on time with the given amount of workers. On the other hand, worker agents will prioritize their well being before job completion. An environment agent will only care whether an accepted job was done on time and how often a job is refused. Each agent is a representation of the knowledge that is passed and how each views a job in different ways.

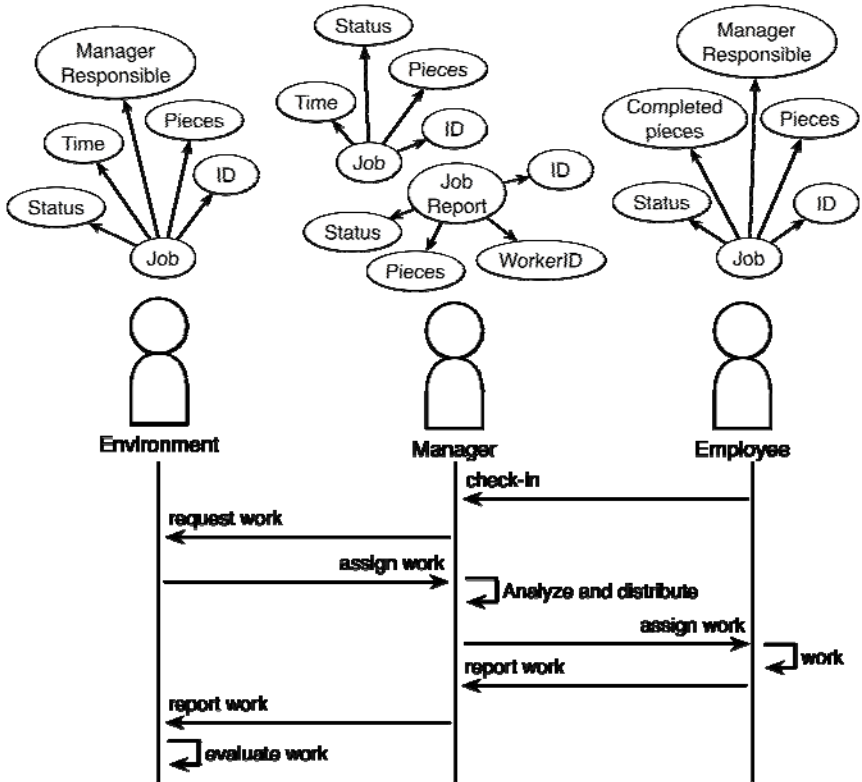


Fig. 5. Script flow of the company workers study case

5 Conclusions and Future Work

Semantic networks can provide richer information when communicating than simple instructions given as code or strings. Important additional information can be associated with each message and each agent will evaluate the meaning giving priority to its very specific needs. This type of information opens the way for different interpretations and exploring how messages can become unclear in given situations or when there is a lack of information needed to interpret the meaning.

Wiinik provides a flexible platform for experimenting with different behavior models and types of interaction. The JADE software provides powerful tools for inspection message passing and the JESS rule engine provides useful instructions for modeling behaviors.

Future work will focus on implementing advanced techniques for decisions and learning such as different types of neural network architectures and fuzzy systems.

Acknowledgements

The authors would like to thank CONACYT. The student Iván Espinoza-Hernández is supported by a scholarship from CONACYT.

References

1. Berne, E.: *Principles of Group Treatment*. Oxford University Press, New York (1964)
2. Brachman, R.J.: What is-a is and isn't: An analysis of taxonomic links in semantic networks. *Computer* 16(10), 30–36 (1983)
3. Caire, G., Bellifemine, F.L., Greenwood, D.: *Developing Multi-Agent Systems with JADE*. Wiley, Chichester (2007)
4. Castañón-Puga, M., Rodríguez-Díaz, A., et al.: Social Systems Simulation Person Modeling as Systemic Constructivist Approach. In: Castillo, O., Melin, P., Kacprzyk, J., Pedrycz, W. (eds.) *Soft Computing for Hybrid Intelligent Systems*, vol. 154. Springer, Heidelberg (2007)
5. El-Nasr Seif, M., Yen, J.: Agents, Emotional Intelligence and Fuzzy Logic. In: *Proceedings of NAFIPS, Florida, US* (1998)
6. Denzinger, J., Hamdan, J.: Improving observation based modeling of other agents using tentative stereotyping and compactation through kd-tree structuring. *WIAS* 4(3), 255–270 (1999)
7. FIPA. FIPA Abstract Architecture Specification (2002)
8. Friedman-Hill, E.J.: *Jess, The Java Expert System Shell*. Distributed Computing Systems, Sandia National Laboratories, Livermore, CA (2000)
9. Flores, D.L., Rodríguez-Díaz, A., Castro, J.R., Gaxiola, C.: TA-Fuzzy Semantic Networks for Interaction Representation in Social Simulation. In: *Evolutionary Design of Intelligent Systems*. *SCI*, pp. 257–270, 213–225 (2009)
10. Ghasem-Aghae, N., Kaedi, M., Ören, T.I.: Effects of Cognitive Complexity in Agent Simulation: Fuzzy Rules and an Implementation. In: *Proceedings of: CM&SC - Conceptual Modeling and Simulation Conference 2005, within I3M 2005 - International Mediterranean Multiconference, Marseille, France, SCS, San Diego, CA, October 20-22* (2005)
11. Gilbert, N., Troitzsch, K.G.: *Simulation for the Social Scientist*. Open University Press, Buckingham (1999)
12. Gorman, D.M., Mezic, J., Mezic, I., Gruenewald, P.J.: Agent-based modeling of drinking behavior: a preliminary model and potential applications to theory and practice. *Am. J. Public Health*. 96, 2055–2060 (2006)
13. Majumdar, A., Sowa, J.F., Stewart, J.: Pursuing the goal of language understanding. In: Eklund, P., Haemmerlé, O. (eds.) *ICCS 2008. LNCS (LNAI)*, vol. 5113, pp. 21–42. Springer, Heidelberg (2008)
14. Minsky, M.: *A framework for representing knowledge*. Technical report, Cambridge, MA, USA (1974)
15. Novellino, M.: The Don Juan Syndrome: The Script of the Great Losing Lover. *Transactional Analysis Journal* 36(1), 3343 (2006)
16. Quillian, M.R.: *Semantic Memory*. MIT Press, Cambridge (1968)

17. Rodríguez-Díaz, A., Cristobal-Salas, A., Castañón-Puga, M., Jauregui, C., Gonzalez, C.: Personality and Behaviour Modelling Based on Cathexis Flux. In: Proceedings of the Joint Symposium on Virtual Social Agents AISB 2005: Social Intelligence and Interaction in Animals, Robots and Agents, ED. AISB The Society for the Study of Artificial Intelligence and the Simulation of Behaviour UH, pp. 130–136. The University of Hertfordshire (2005)
18. Sharma, S., Singh, H., Prakash, A.: Multi-agent modeling and simulation of human behavior in aircraft evacuations. *IEEE Transactions on Intelligent Transportation Systems* 44(4), 1477–1488 (2008)
19. Sowa, J.F.: *Conceptual structures: information processing in mind and machine*. Addison-Wesley Longman Publishing Co., Inc., Boston (1984)
20. Woollams, S., Brown, M.: *TA: the total handbook of transactional analysis*. Prentice Hall, London (1979)

Fuzzy Cellular Model for Predator-Prey Interaction Applied to the Control of Plagues in a Peppers Cropping

Cecilia Leal-Ramirez¹, Oscar Castillo², and Antonio Rodriguez-Diaz¹

¹ Universidad Autonoma de Baja California, Tijuana, BC, Mexico
cecilialr@uabc.mx, ardiaz@uabc.mx

² Instituto Tecnológico de Tijuana, Mexico
ocastillo@hafsamx.org

Abstract. The control of plagues is the regulation and the handling of some species referred to as plagues, normally these are species that affect the ecology and the economy of a certain location. The search for solutions to the important economic incidence of the plagues in the croppings has had an evolution throughout the last two decades. In the studies of population dynamics of plagues in croppings, it is fundamental to consider the predator-prey systems. We formulated a fuzzy cellular model for modeling predator-prey interactions and to study the control of plagues in a croppings of pepper affected by *Frankliniella* species. We consider as predators for the biological control of these plagues two species to the *Orius* species. Our model can be used to calculate the introduction of natural enemies (predators) for fighting against the injurious species (preys). The results show that the proposed model works as a tool to achieve plagues control. The combination of predator and prey numbers allows us to stabilize the pepper density level on a suitable bio-economic equilibrium level.

1 Introduction

The control of plagues is the regulation and the handling of some species referred as plagues, normally for being species that affect the ecology, the economy, etc. The control of plagues is important because the quality of the product and its value in the market depend on them. Two typical procedures for the control of plagues exist at the moment. The first is the chemical control that uses insecticides that are highly effective although the risk exists of contaminating the product with substances

detrimental for the human health. The second is the biological control, which consists of the use of beneficial insects for the plants, as natural predators. The product that is obtained using biological control has a higher value in the market. One of the first models to incorporate interactions between predators and prey was proposed by [12] and [22]. This model is based on differential equations, and actually, the general models, of the predator-prey type, are modifications or extensions from these equations. In the present work, we formulated a fuzzy cellular model of the predator-prey type to study the control of a plague on a pepper croppings, which is defined using the differential equations that characterize the model proposed by [12] and [22], but expressed in terms of a fuzzy cellular structure plus terms of emigration and immigration for the predator. In the study of multispecies problems, normally one of the most important goals is to find the conditions for the bio-economic equilibrium of the resource in order to maximize the revenues earned from them maintaining the ecological balance amongst the species [18]. In our study the goal is maintaining the pepper cropping yield on a bioeconomically suitable equilibrium level combining the predators and preys density in the population dynamics. The equilibrium level is defined as the acceptable peppers density after of the damage to the cropping produced by the plague during a time interval. The data on the pepper croppings were taken from [6] and [3], and the change coefficients that govern the predator-prey interaction were taken from [16]. Our interest is in obtaining a characterization of the fuzzy cellular predator-prey model using fuzzy logic systems based on fuzzy sets [23] to evaluate the parameters defined as change coefficients in the model. This characterization allows us to evaluate the effectiveness of depredation produced by the calculation of the introduction of natural enemies (*Orius* species) for fighting against the injurious species (*Frankliniella* species). The basic concepts of fuzzy set theory are presented in section 2. The fuzzy cellular predator-prey model structure is formulated step by step in section 3. In section 4 a bravely description on the predator and prey species which affect to the pepper cropping is made. In section 5, the fuzzy logic system is developed using the Matlab fuzzy logic toolbox. The results generated by several simulations are shown in section 6, where our model shows how ecology, economics and mathematics can be combined into a fuzzy cellular predator-prey model to identify management strategies that will make the work of plagues control more efficiently done with respect to the models defined by using complex differentials equations. Finally, in section 7 the conclusions are presented.

2 Basic Concepts of Fuzzy Sets

Fuzzy sets are considered elements that have degrees of membership for belonging to the set. Fuzzy sets were introduced by [23] as an extension of the classical notion of a set. In classical set theory, the membership of elements in a set is assessed in binary terms according to a bivalent condition. By contrast, fuzzy set theory allows the gradual assessment of the membership of elements in a set; this is described with the aid of a membership function valued in the real unit interval $[0, 1]$. If X is a set

of objects denoted by x and A is a fuzzy set in X , then A is defined as an ordered pairs set

$$\tilde{A}(x) = \{(x, \mu_{\tilde{A}}) | x \in X\}, \tag{1}$$

where $\tilde{A}(x)$, is called membership function for the fuzzy set A . The construction of a fuzzy set depends on the identification of an appropriate universe and the specification of a membership function with appropriate linguistic meaning [23]. Fuzzy set theory can be used in a wide range of domains in which information is incomplete or imprecise [8]. In the following section, we will use the fuzzy sets to define the change coefficients of the multispecies fuzzy cellular model and to study its dynamics.

3 Fuzzy Cellular Predator-Prey Structure

The mathematical models that describe predator and prey relationships are used to study interactions between two species, when one of them depends on the other for food and for survival. Such dynamic relationships between predators and preys are prominent areas of study in Ecology [5], [17]. The model proposed by [12] and [22] is the simplest model of interactions between predator and prey, and it is expressed as a differential equation system:

$$\begin{aligned} \frac{dH}{dt} &= rH - aHP \\ \frac{dP}{dt} &= bHP - mP \end{aligned} \tag{2}$$

where H and P represent the number of preys and of predators respectively, at time t , and r = intrinsic rate of preys population increase in the absence of predation, a = predation rate, b = reproduction rate of predators, m = predator mortality rate in the absence of preys. In the present study, the constants a, b, r and m are called 'change coefficients' of the classical predator-prey model. Model (2) considers that in the absence of predators the number of preys grows exponentially and also that in the absence of preys, the number of predators decreases exponentially. The terms $(-aHP)$ and (bHP) describe the prey-predator encounters, which are favourable to predators and fatal to preys. It assumed that the change in the population density is mainly given by the difference of two related terms with the birth and the death of the individuals, and that both terms are proportional to the population density [13] and to the amount of resources available [21].

In general, the model behaviour is unnatural showing no asymptotic stability. However, numerous modifications of this model exist, which make it more realistic and are used today in the analysis of population dynamics. A representation of model (2) as a deterministic model is given by

$$\begin{aligned} H(t + \Delta t) &= H(t) + (r - aP(t))H(t) \\ P(t + \Delta t) &= P(t) + (bH(t) - m)P(t) \end{aligned} \tag{3}$$

In models (2) and (3) it is frequent to assume that all individuals of the population interact with the same probability independently of the abundance, frequency or space-position of individuals of a particular space. This assumption can be more or less justified for concrete cases in animal populations, but this is at all unjustified in plant populations due to the competition process [26]. Models in space have been proposed, and could be the more significant advance of the recent Ecology to understand the role of the local process into population space-temporal organization [26].

Now we proceed to introduce a space-temporal representation into the equations (3) using a cellular structure. This structure is composed by a rectangular region of $M \times N$ cells. The union of all the cells defines the cellular space in which the evolution of each cell depends on its present state and the states of its immediate neighboring cells. Let C be a single cell, then $C(i, j)$ is the cell centred at the (i, j) coordinates, where $1 \leq i \leq M$ and $1 \leq j \leq N$. Each cell is a fundamental element for the population density change, reason why the contribution of all the cells determines the change of the total population density in the system's dynamics. Therefore, modifying expression (3) according to the cellular structure definition, we obtain

$$\begin{aligned} H_{ij}(t + \Delta t) &= H_{ij}(t) + (r - aP_{ij}(t))H_{ij}(t) \\ P_{ij}(t + \Delta t) &= P_{ij}(t) + (bH_{ij}(t) - m)P_{ij}(t) \end{aligned} \tag{4}$$

A population may increase rapidly from a low level under favorable environmental conditions, but this increase in numbers will eventually approach the level where resources cannot support the continue increase, this is, indefinite increases in population density do not occur [20].

If we take into account that the imprecision and uncertainty are intrinsic concepts in bio-systems [8], then we can define the change coefficients, under which the populations fluctuate, as functions that depend on two variables characterized as fuzzy sets,

$$\begin{aligned} H_{ij}(t + \Delta t) &= H_{ij}(t) + (r(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t)) - a(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))P_{ij}(t))H_{ij}(t) \\ P_{ij}(t + \Delta t) &= P_{ij}(t) + (b(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))H_{ij}(t) - m(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t)))P_{ij}(t) \end{aligned} \tag{5}$$

where $\tilde{H}_{ij}(t)$ and $\tilde{P}_{ij}(t)$ are the prey and predator densities respectively located in cell (i, j) at time t .

The prey and predator look for optimal environment conditions, and even though all the conditions are not the optimal ones, preys and predators will always try to be located into environments where each of these conditions is within tolerable limits that allow their normal development. Let $\Pi_\alpha(i, j)$ be the neighborhood of the cell $C(i, j)$ in terms of a radius α

$$\Pi_\alpha = \left\{ C(k, l) \mid \begin{array}{l} \max\{|k - i|, |l - j|\} \leq \alpha \\ 1 \leq k \leq M; 1 \leq l \leq N, \end{array} \right\} \tag{6}$$

where α is a positive integer number and (k, l) are the coordinates of another cell where the magnitude of the difference between (i, k) and (j, l) does not exceed the value of α . The population that emigrates towards the neighboring cells from a cell $C(i, j)$ is determined by a factor that is directly proportional to the population size. Taking in account that in studies on control of plagues on croppings, the predator will emigrate to find preys, and the prey will always have unlimited resource to survival, now we retake the equations (5) and (6) to define the emigration considering that only the predators can emigrate,

$$\begin{aligned}
 H_{ij}(t + \Delta t) &= H_{ij}(t) + r(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))H_{ij}(t) - a(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))P_{ij}(t)H_{ij}(t) \\
 P_{ij}(t + \Delta t) &= P_{ij}(t) + b(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))H_{ij}(t)P_{ij}(t) - m(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))P_{ij}(t) \quad (7) \\
 &\quad - \beta(\tilde{H}_{ij}(t), \tilde{P}_{ij}(t))P_{ij}(t) + \prod_a^p(k, l, t)
 \end{aligned}$$

where $H_{ij}(t) > 0, P_{ij}(t) > 0, \beta(\cdot)$ is the emigration function of the predators and $\prod_a^p(k, l, t)$ is the predators population that immigrates from any neighboring cell $C(k, l)$ of the cell $C(i, j)$ to the cell $C(i, j)$. Thus, we have formulated a fuzzy cellular model of the predator-prey type (7).

4 Description of the Predator and Prey Species

Works on plagues in croppings under plastic have been made as application of the predator-prey models [15]. For example, authors in [6] analyzed from the mathematical point of view, the sufficient conditions for the existence of global attractors, for the solutions to the possible differential systems (predator-prey) on two croppings, pepper and tomato. These croppings are generally affected by three groups of plague species of great economic severity: "heliotis of the tomato", "white fly" and "trips western of the flowers". Predators for the biological control of these plagues species are the *Orius* and *Macrolophus* species [6]. In addition, they made adjustments with their proposed model and analyzed if the adjustment was good by means of statistical coefficients.

We have taken as a case study, the control of the plague known as the *Frankliniella* species commonly denominated "trips western of the flowers" using the *Orius* predator species, in peppers cropping made under the method proposed by [6]. We assume that the interaction between predator and prey is governed by the change coefficients obtained by [16] and not by the ones obtained by [6] (see Table I).

Table 1. Change coefficients values obtained from [16]

<i>r</i>	<i>a</i>	<i>b</i>	<i>m</i>
0.012	0.0003	0.2	0.01

5 Fuzzy Logic System

Fuzzy logic was proposed as a generalization of classic logic [23], [25], and is used to model vagueness and uncertainty in the real world. Fuzzy logic systems (FLS) have four basic components: an input processor, a fuzzy rules base, a fuzzy inference mechanism and an output processor (Fig. 1).

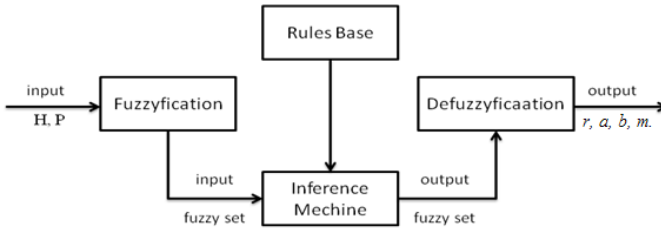


Fig. 1. Basic components of a fuzzy logic system

In the FLS, the input and output variables have fuzzy sets as values with partial overlap, which create qualitative groups of values. Through the linguistic variables it is possible to define new fuzzy sets based on the existing sets using the adjectives "much", "little", "very", etc [24]. The fuzzy rules have the IF-THEN form, these connect hypothesis to conclusions [25]. The FLS initiates the computation by mapping the input values to the fuzzy sets that characterize the input variables. These fuzzy sets are used as input to the inference mechanism through the fuzzy rules. These are combined in the inference mechanism to produce a fuzzy output, which is mapped to a numerical value by means of a defuzzification process to produce an output value. A particular type of fuzzy inference is the Mamdani method. This type of inference combines the membership degrees associated with each input value by the minimum operator and aggregates the rules through the maximum operator [14]. We now proceed to introduce, step by step, the development of a fuzzy logic system to evaluate the change coefficients, using the Matlab fuzzy logic toolbox, with the fuzzy inference given by the Mamdani method, and the center of area method for the defuzzification.

It is possible to interpret that the environment of a species is variable by defining the change coefficients as fuzzy [10]. In our model the predator depends on the prey for survival. Taking in to account the model definition (7), we developed one fuzzy logic system to evaluate its change coefficients. The system consists of two input variables, five output variables and 9 fuzzy rules. The input variables correspond to the prey and predator densities located in the cell (i, j) at time t , which are characterized by three states: Few, Moderate and Much (Fig. 2 and Fig. 3).

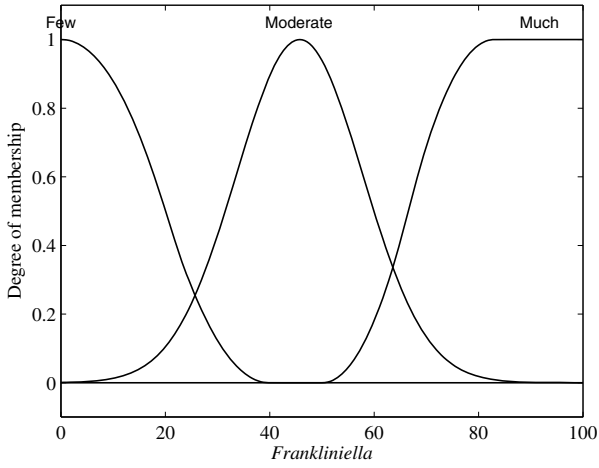


Fig. 2. Membership functions characterizing the input variable *Frankliniella*

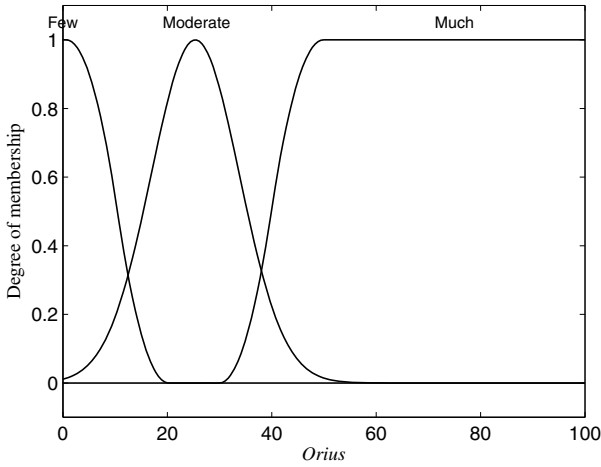


Fig. 3. Membership functions characterizing the input variable *Orius*

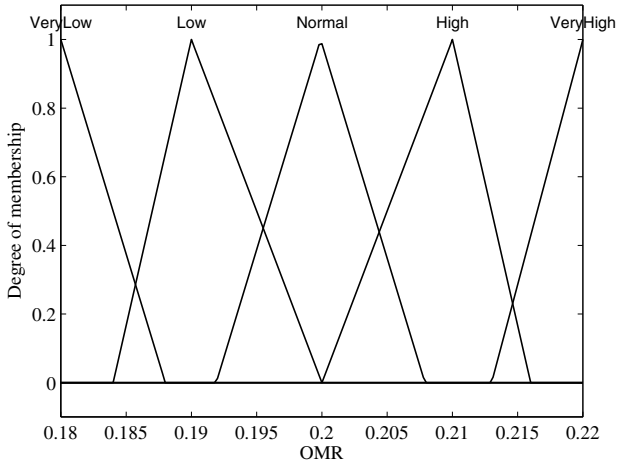


Fig. 4. Membership functions characterizing the output variable OMR (*Orius* Mortality Rate)

There is no defined pattern to determine the membership functions that characterize the input variables. Therefore, we used the patterns more appropriate for characterization of the input and output variables. The output variables are the change coefficients expressed in model (7). These variables are characterized by five states: VeryLow, Low, Moderate, High, VeryHigh (Figs. 5-8).

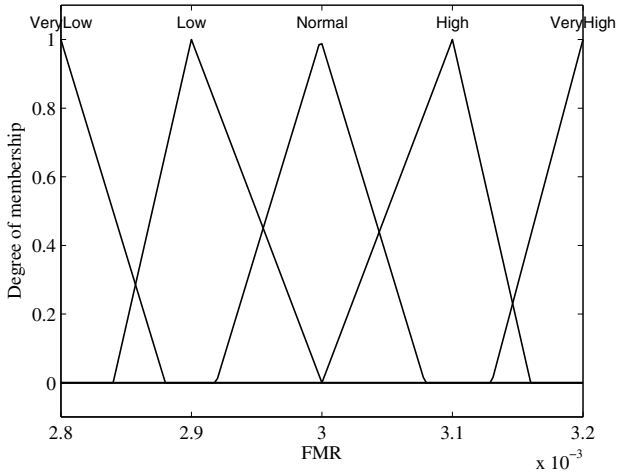


Fig. 5. Membership functions characterizing the output variable FMR (*Frankliniella* Mortality Rate)

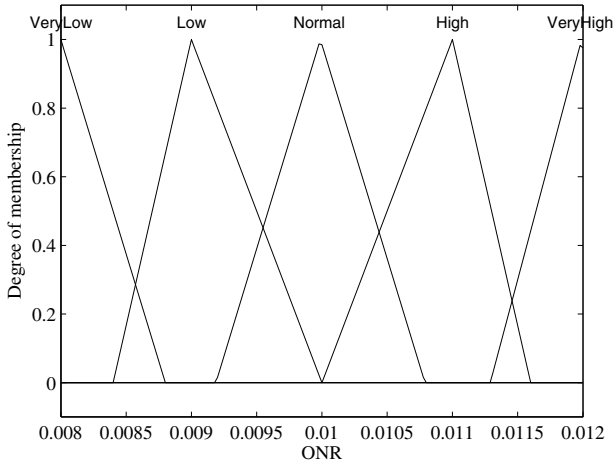


Fig. 6. Membership functions characterizing the output variable ONR (*Orius* Natalty Rate)

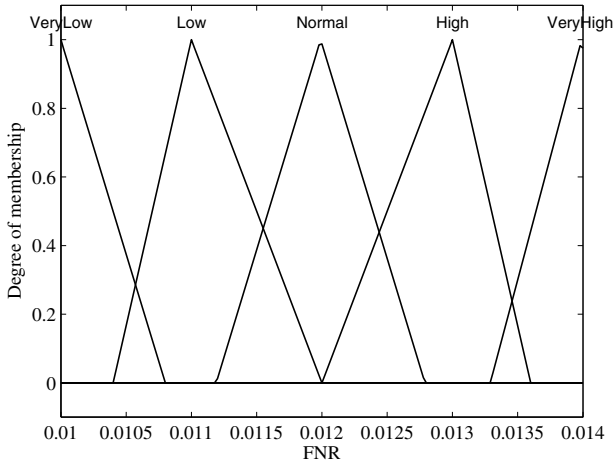


Fig. 7. Membership functions characterizing the output variable FNR (*Frankliniella* Natalty Rate)

The membership functions domain can be changed according to the desired behavior. We constructed a total of 9 fuzzy rules to describe the natural behavior generated by the interaction between prey and predator when one of them depends

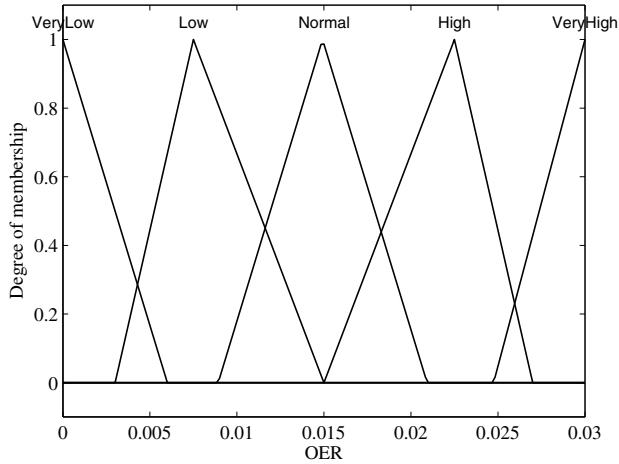


Fig. 8. Membership functions characterizing the output variable OER (*Orius* Emigration Rate)

on the other for survival. The fuzzy rules set was constructed from the interpretation on the life cycle of both species, and are expressed as follows,

1. If (*Frankliniella* is Few) and (*Orius* is Few) then (OMR is VeryHigh)(FMR is VeryLow)(ONR is VeryLow)(FNR is VeryHigh)(OER is Normal).
2. If (*Frankliniella* is Few) and (*Orius* is Moderate) then (OMR is VeryHigh)(FMR is Low)(ONR is VeryLow)(FNR is High)(OER is High).
3. If (*Frankliniella* is Few) and (*Orius* is Much) then (OMR is VeryHigh)(FMR is Normal)(ONR is VeryLow)(FNR is Normal)(OER is VeryHigh).
4. If (*Frankliniella* is Moderate) and (*Orius* is Few) then (OMR is High)(FMR is Low)(ONR is Low)(FNR is High)(OER is Low).
5. If (*Frankliniella* is Moderate) and (*Orius* is Moderate) then (OMR is Normal)(FMR is Normal)(ONR is Normal)(FNR is Normal)(OER is Normal).
6. If (*Frankliniella* is Moderate) and (*Orius* is Much) then (OMR is Low)(FMR is High)(ONR is High)(FNR is Low)(OER is High).
7. If (*Frankliniella* is Much) and (*Orius* is Few) then (OMR is Normal)(FMR is High)(ONR is VeryHigh)(FNR is Low)(OER is VeryLow).
8. If (*Frankliniella* is Much) and (*Orius* is Moderate) then (OMR is Low)(FMR is VeryHigh)(ONR is Normal)(FNR is VeryLow)(OER is Low).
9. If (*Frankliniella* is Much) and (*Orius* is Much) then (OMR is VeryLow)(FMR is VeryHigh)(ONR is VeryLow)(FNR is VeryLow)(OER is Normal).

Where FNR = r , FMR = a , OMR = b , ONR = m , OER = β .

6 Simulation Results

In the work of [6] the experimental data were obtained from a pepper cropping with area of $2000m^2$, which we represent by cells in the present work. Each cell represents $20m^2$ of the total area, and there are two pepper plants on $1m^2$. Therefore, there are 40 pepper plants on $20m^2$ and 4000 pepper plants on $2000m^2$. Our goal is studding the damage of the pepper generated by the plague (*Frankliniella*) using model (7) and the method of [6] under the hypothesis that the interaction between *Orius-Frankliniella* is governed by the change coefficients shown in Table 1.

Model (7) describes the dynamics of the biological system in which two species interact, the predator and the prey (see Fig. 9 and Fig. 10).

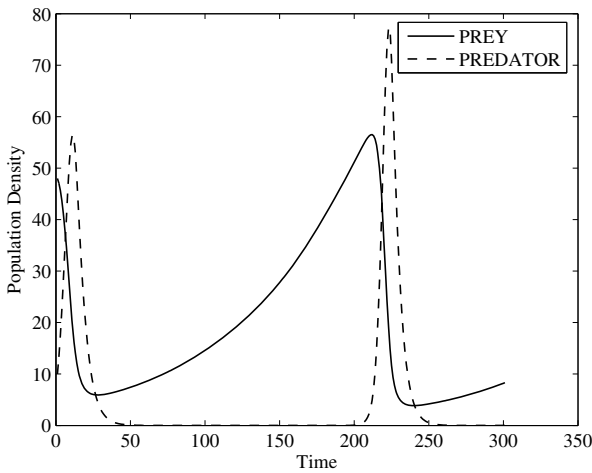


Fig. 9. Prey and predator densities generated by the model (7)

Fig. 9 shows the changes in the prey and predator densities, while in Fig. 10 we show the closed lines formed by the prey and predator densities, which demonstrate that the model (7) has no asymptotic stability like model (2), it does not converge to an attractor. Nevertheless, in nature, the populations tend to the regulation or asymptotic stabilization [7]. To control the plagues in croppings it is not necessary to know the interaction between prey and predator indefinitely, because the croppings have a time interval wherein the resource grows until its harvest. The harvest period in a production cycle under a greenhouse is from 4 to 7 months [3]. Therefore, in the present study, we consider only a time interval of $[1 - 30]$ weeks to generate the interaction between the species *Orius* and *Frankliniella*, using model (7). Although this model has no asymptotic stability, the interaction between prey and predator will determine the damage level produced by the prey density on the plant in each

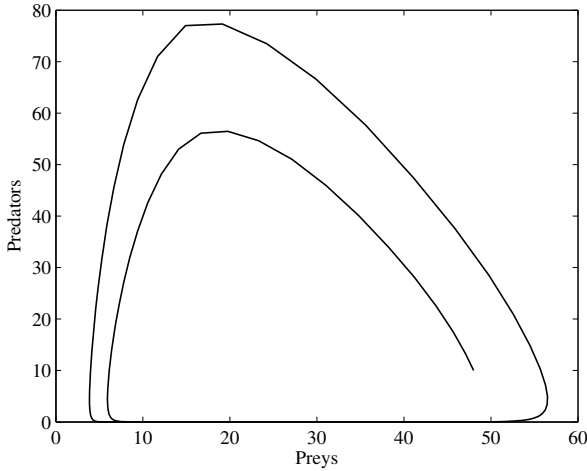


Fig. 10. Closed lines formed by the prey and predator densities generated by the model (7)

cell. We expect that the damage on the cropping tends to the asymptotic stability converging towards an attractor, which should be on a suitable bioeconomically equilibrium [3]. The bioeconomical equilibrium is defined as the cropping production acceptable after of the damage generated by the plagues on a time interval. Hypothetically we have considered the damage caused by a prey of $1 \times 10^{-6}\%$ on one plant, per week.

The results are presented as averages obtained by 20 simulations, and these are classified in 4 case studies, each one determines a region on the total area wherein an initial number of preys is distributed, simulating the contaminated area by plagues (Tables 2, 3, 4 and 5), and the initial number of predators is distributed on the total.

Table 2. Damage produced by the preys on the cropping of pepper plants, considering the 10% on the total area as the contaminated region by preys at the beginning

<i>Frankliniella</i> \ <i>Orius</i>	20 × Cell	30 × Cell	40 × Cell	50 × Cell	60 × Cell	70 × Cell
[0 – 20] × Cell	0.00216	0.00222	0.00191	0.00136	0.00135	0.00106
[10 – 20] × Cell	0.00342	0.00294	0.00254	0.00217	0.00184	0.00155
[30 – 40] × Cell	0.00474	0.00384	0.00323	0.00279	0.00242	0.00207

We only will present in the graphics the two more representatives combinations per each Table, to show how the interaction produced by preys and predators, determines the asymptotic growth of the damage caused by the preys on the plants per cell.

Fig. 11 and Fig. 13 show the interactions between preys and predator generated by model (7) using the combination of the prey and predator initial number shown in Table 2. The Fig. 12 and Fig. 14 show the damage caused by the preys on the plants per cell, which is generated by Fig. 11 and Fig. 13 respectively.

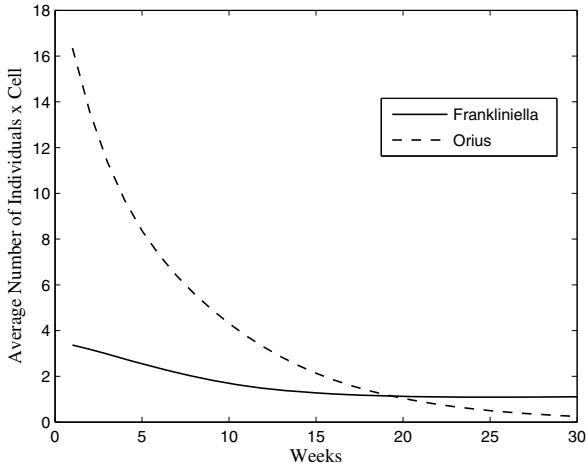


Fig. 11. Prey and predator densities generated by the model (7) using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 10% of the total area (see Table 2).

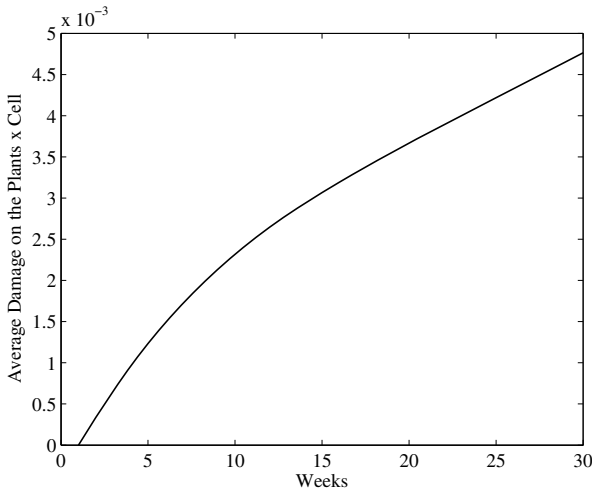


Fig. 12. Damage caused by the preys on the plants per cell. The preys were initially distributed on 10% of the total area, using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 2).

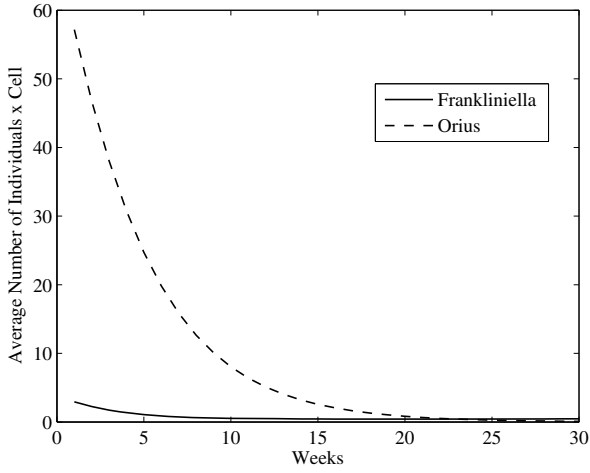


Fig. 13. Prey and predator densities generated by the model (7) using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 10% of the total area (see Table 2).

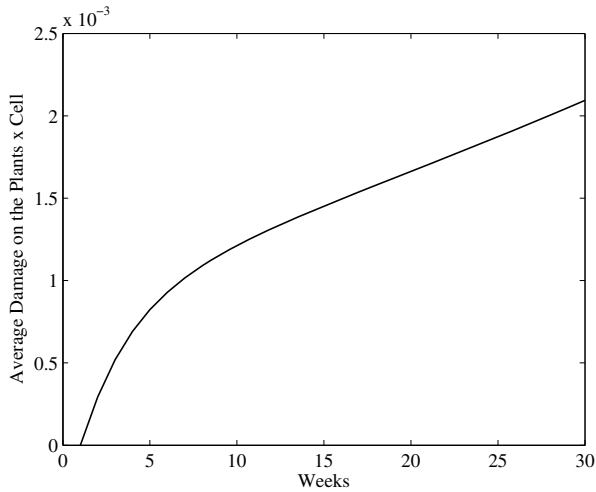


Fig. 14. Damage caused by the preys on the plants per cell. The preys were initially distributed on 10% of the total area, using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 2).

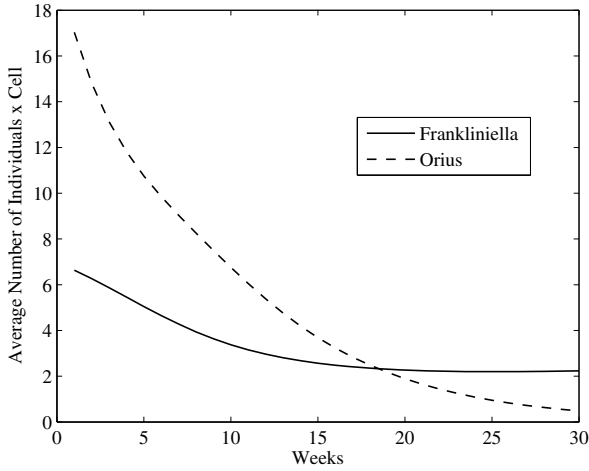


Fig. 15. Prey and predator densities generated by the model (7) using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 20% of the total area (see Table 3).

Table 3. Damage produced by the preys on the cropping of pepper plants, considering the 20% on the total area as the contaminated region by preys at the beginning

<i>Frankliniella</i> \ <i>Orius</i>	20 × Cell	30 × Cell	40 × Cell	50 × Cell	60 × Cell	70 × Cell
[0 – 20] × Cell	0.00515	0.00446	0.00355	0.00272	0.00222	0.00241
[10 – 20] × Cell	0.00696	0.00619	0.00491	0.00427	0.00373	0.00322
[30 – 40] × Cell	0.00984	0.00774	0.00642	0.00551	0.00479	0.00421

Fig. 15 and Fig. 17 show the interactions between preys and predator generated by model (7) using the combination of the prey and predator initial number shown in Table 3. Fig. 16 and Fig. 18 show the damage caused by the preys on the plants per cell, which is generated by Fig. 15 and Fig. 17 respectively.

Table 4. Damage produced by the preys on the cropping of pepper plants, considering the 30% on the total area as the contaminated region by preys at the beginning

<i>Frankliniella</i> \ <i>Orius</i>	20 × Cell	30 × Cell	40 × Cell	50 × Cell	60 × Cell	70 × Cell
[0 – 20] × Cell	0.00777	0.00679	0.00526	0.00446	0.00361	0.00359
[10 – 20] × Cell	0.01053	0.00912	0.00738	0.00619	0.00557	0.00482
[30 – 40] × Cell	0.01427	0.01165	0.00965	0.00828	0.00720	0.00631

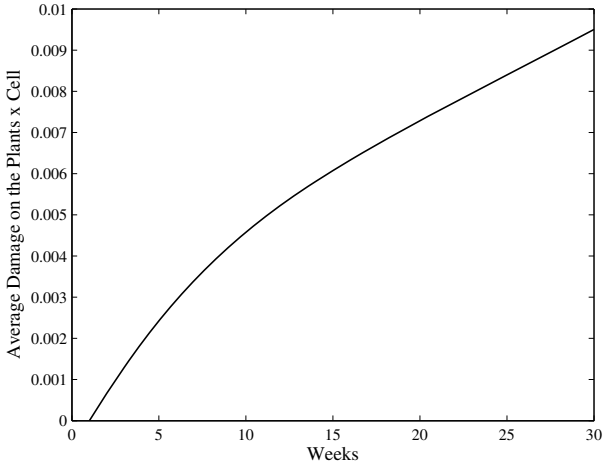


Fig. 16. Damage caused by the preys on the plants per cell. The preys were initially distributed on 20% of the total area, using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 3).

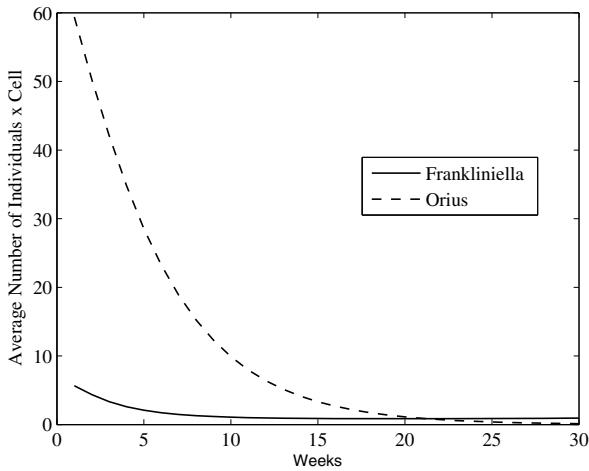


Fig. 17. Prey and predator densities generated by the model (7) using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 20% of the total area (see Table 3).

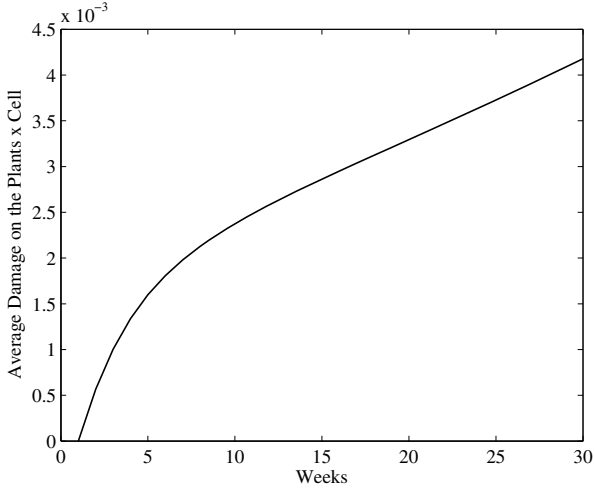


Fig. 18. Damage caused by the preys on the plants per cell. The preys were initially distributed on 20% of the total area, using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 3).

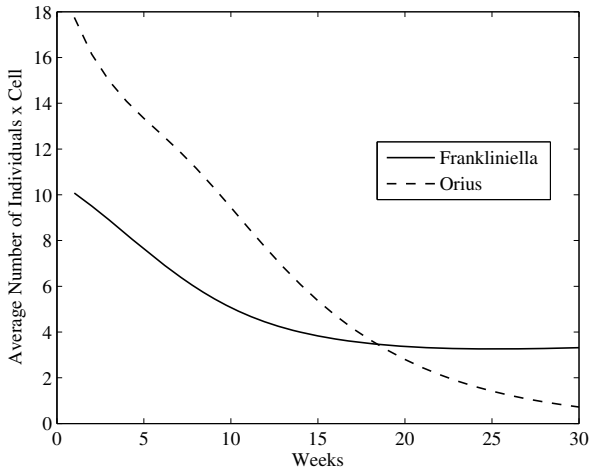


Fig. 19. Prey and predator densities generated by the model (7) using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 30% of the total area (see Table 4).

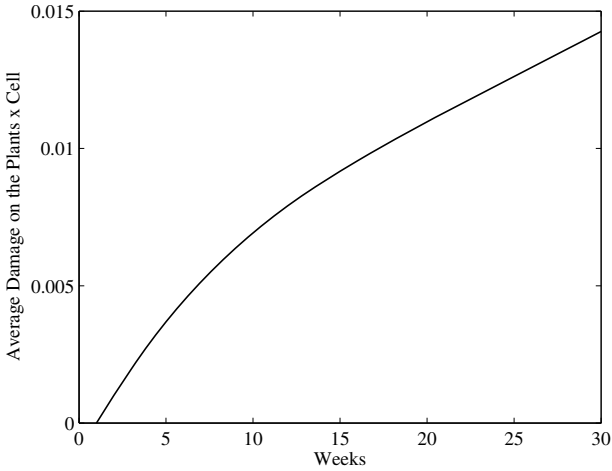


Fig. 20. Damage caused by the preys on the plants per cell. The preys were initially distributed on 30% of the total area, using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 4).

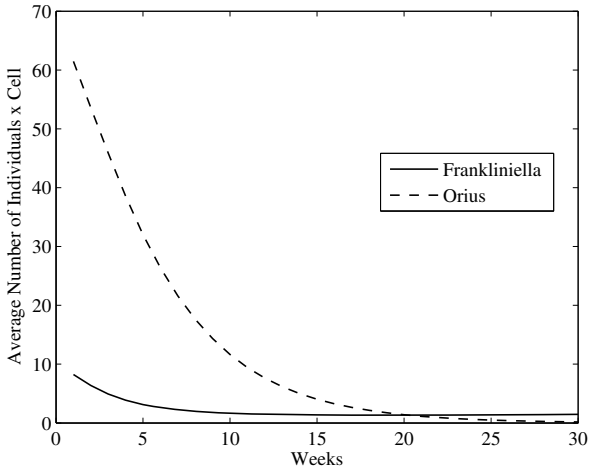


Fig. 21. Prey and predator densities generated by the model (7) using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 30% of the total area (see Table 4).

Fig. 19 and Fig. 21 show the interactions between preys and predator generated by model (7) using the combination of the prey and predator initial number shown in Table 3. Fig. 20 and Fig. 22 show the damage caused by the preys on the plants per cell, which is generated by Fig. 19 and Fig. 20 respectively.

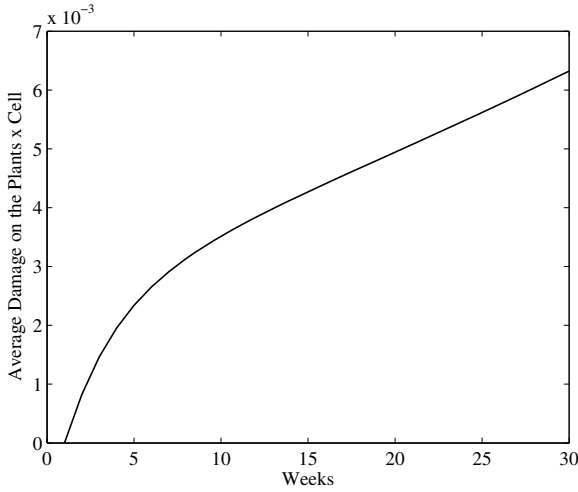


Fig. 22. Damage caused by the preys on the plants per cell. The preys were initially distributed on 30% of the total area, using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 4).

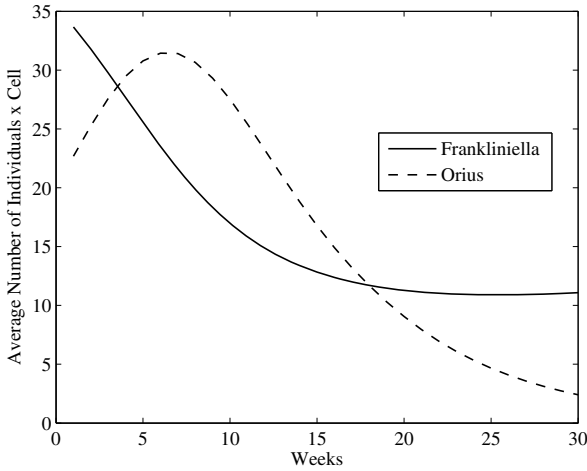


Fig. 23. Prey and predator densities generated by the model (7) using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 100% of the total area (see Table 5).

Fig. 23 and Fig. 25 show the interactions between preys and predator generated by model (7) using the combination of the prey and predator initial number shown in Table 3. Fig. 24 and Fig. 26 show the damage caused by the preys on the plants per cell, which is generated by Fig. 23 and Fig. 25 respectively.

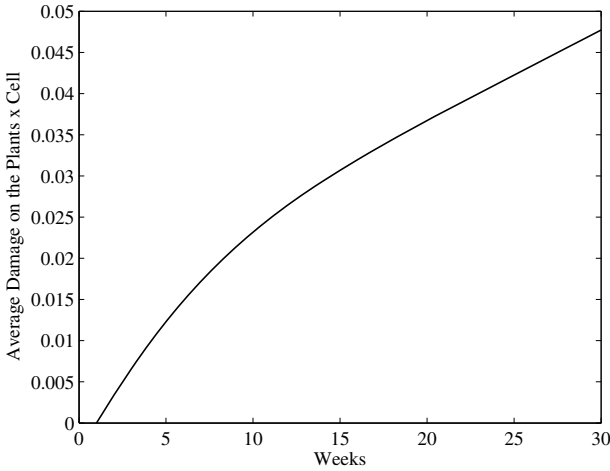


Fig. 24. Damage caused by the preys on the plants per cell. The preys were initially distributed on 100% of the total area, using the combination of 20 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 5).

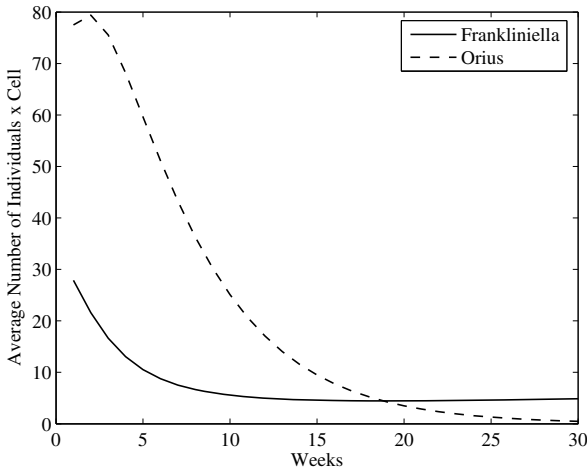


Fig. 25. Prey and predator densities generated by the model (7) using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell. The preys were initially distributed on 100% of the total area (see Table 5).

Inside certain limits, the increment in the population's density causes decreases in the yield for the plants [1], [19], [4], [2], [11]. In the work presented by [3], the authors made a study on the yield of a cropping combining the population's density

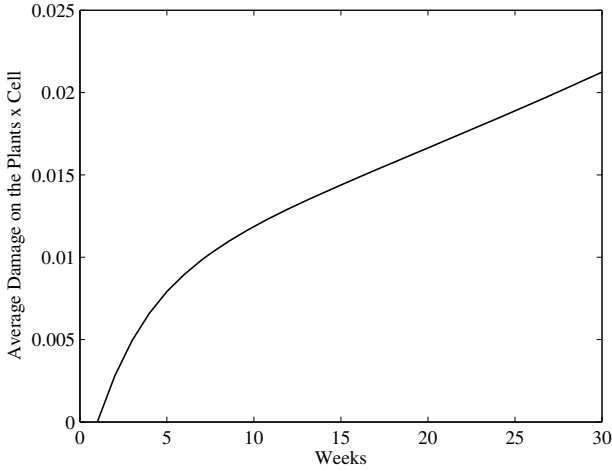


Fig. 26. Damage caused by the preys on the plants per cell. The preys were initially distributed on 100% of the total area, using the combination of 70 *Orius* per cell and from 30 to 40 *Frankliniella* per cell (see Table 5).

per m^2 , the early pruning of the terminal buds and the length of harvest period. Their results showed that the cropping total yield is of 6.27 peppers per plant with no pruning. According to this result, we present the cropping total yield associated to the area initially affected by the plague and considering that the yield of a plant is of 8 peppers (see Tables 2-5). We would expect to have a total yield of 32000 peppers in case of that the cropping does not have plague during 30 weeks. Nevertheless, all the croppings tend to have plagues. Therefore the croppings must be protected of plagues using methods as the biological control to maximize their yield.

Table 5. Damage produced by the preys on the cropping of pepper plants, considering the 100% on the total area as the contaminated region by preys at the beginning

<i>Frankliniella</i> \ <i>Orius</i>	20 × Cell	30 × Cell	40 × Cell	50 × Cell	60 × Cell	70 × Cell
[0 – 20] × Cell	0.0240	0.0214	0.0183	0.0141	0.0111	0.0121
[10 – 20] × Cell	0.0354	0.0299	0.0254	0.0218	0.0185	0.0160
[30 – 40] × Cell	0.0478	0.0389	0.0324	0.0278	0.0243	0.0212

The agriculture under greenhouse conditions requires producing croppings of commercial high value and that are economically portable. Therefore its objective is to maximize the yield of the croppings, which implies that the yield of cropping must stay on an equilibrium level economically portable. The equilibrium level is defined as the acceptable peppers density after of the damage produced by the plague during a time interval. For the case study presented in this work, we could

determine the suitable procedure to control the plague on the desired yield of the Table 6. For example, if we want to have a yield above 80% then we must take into account from 20 to 70 predators to fight the dispersed prey on 10% of the total area (see Table 6). For the case of 20% affected by plague, we must consider more than 70 predators to fight the plague. Thus, we will be able to maintain the cropping yield on an equilibrium level of 80%.

Table 6. Cropping yield according to the area affected by the plague

Area(%)	<i>Orius</i>	<i>Frankliniella</i>	Damage Area (%)	Harvest Peppers (%)
10	20	[30-40]	0.47	99.52
10	70	[30-40]	0.21	99.79
20	20	[30-40]	0.99	99.01
20	70	[30-40]	0.43	99.57
30	20	[30-40]	1.43	98.57
30	70	[30-40]	0.64	99.36
100	20	[30-40]	4.78	95.22
100	70	[30-40]	2.12	97.88

7 Conclusion

We have modeled the predator-prey dynamics without using explicit complex differential equations applied to control of plagues. Our model calculates the introduction of natural enemies (predators) for fighting against the injurious species (preys). We considered the change coefficients from [16] to define the predator-prey interaction. The results show that the proposed model works as a tool to plagues control. The combination of predator and prey numbers allows us to stabilize the resource level on a suitable bio-economic equilibrium level. The use of the model to evaluate the effectiveness of predation remains to be developed. Nevertheless, which is observed, at the moment is that the present model can be useful in this evaluation, that constitutes a problem in all the processes of biological control. We consider that the use of fuzzy logic is a great contribution to the construction of models as the one proposed in the present work. Fuzzy theory allows modeling many forms of biological organization in a more realistic way [9]. In addition, the input and output sets of the fuzzy systems can be easily constructed with the help of specialists in the field, that is, a specialist will know when the population of a particular species is small, large, and so forth.

References

1. Batal, K., Smittle, D.: Response of bell pepper to irrigation, nitrogen, and plant-population. *J. Am. Soc. Hort. Sci.* 106, 259–262 (1981)
2. Cebula, S.: Optimization of plant and shoot spacing in greenhouse production of sweet pepper. *Acta Hort.* 412, 321–329 (1995)

3. Cruz-Huerta, N., Sanchez del Castillo, F., Ortiz-Cereceres, J., Mendoza-Castillo, M.: High plant stand with early pruning on yield and harvest period in bell pepper. *Agricultura Tecnica en Mexico* 31(1), 73–80 (2009)
4. Decoteau, D., Graham, H.: Plant spatial arrangement affects growth, yield, and pod distribution of cayenne peppers. *Hortsci* 29, 149–151 (1994)
5. Edelstein-Keshet, L.: *Mathematical Models in Biology*. McGraw-Hill, Inc., New York (1987)
6. Gamez, M., Carreno, R., Andujar, A., Barranco, P., Cabello, T.: Modelos matematicos de depredador-presa en cultivos hortícolas en invernadero en el sudeste de la península ibérica. *Biol. San. Veg. Plagas* 26, 665–672 (2002)
7. Gotelli, N.: *A primer of Ecology*. Sinauer Associates (1998) (incorporated)
8. Huang, Z., Chen, S., Xia, Y.: Incorporate intelligence into an ecological system: An adaptive fuzzy control approach. *Applied Mathematics and Computation* 177, 243–250 (2006)
9. Schaefer, J.A., Wilson, C.: The fuzzy structure of populations. *Can. J. Zool.* 80, 2235–2241 (2002)
10. Leal-Ramírez, C., Castillo, O., Rodríguez-Díaz, A.: Fuzzy cellular model applied to the dynamics of a uni-specific population induced by environment variations. In: *Proceedings of Ninth Mexican International Conference on Computer Science*, pp. 210–220 (2008)
11. Lorenzo, P., Castilla, N.: Bell pepper yield response to plant density and radiation in unheated plastic greenhouse. *Acta Hort.* 412, 330–334 (1995)
12. Lotka, A.: *Elements of physical biology*. Williams and Wilkins Co (1925)
13. Malthus, T.R.: *An Essay on the Principle of Population*. London (1798)
14. Mendel, J.: *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*. Prentice-Hall, NJ (2001)
15. Mills, N., Getz, W.: Modelling the biological control of insect pests: a review of host-parasitoid models. *Ecological Modelling* 92, 121–143 (1996)
16. Morales, J., Buranr, J.V.: Interactions between cycloneda sanhuine and the brown citrus aphid: adult feeding and larval mortality. *Environ. Entomol.* 14(4), 520–522 (1985)
17. Murray, J.: *Mathematical Biology*. Springer, Berlin (1990)
18. Sadhukhan, D., Sahoo, L., Mondal, B., Maiti, M.: Food chain model with optimal harvesting in fuzzy environment. *J. Appl. Math. Comput.* 1(2), 1–2 (2009)
19. Stoffella, P., Bryan, H.: Plant population influences growth and yields of bell pepper. *J. Am. Soc. Hort. Sci.* 113, 835–839 (1998)
20. Turchin, P.: Does population ecology have general laws? *Oikos* 94, 17–26 (2001)
21. Verlhust, P.: Notice sur la loi que la population suit dans son accroissement. *Corr. Math. Phys.* 10, 113–121 (1932)
22. Volterra, V.: Variazioni e fluttuazioni del numero d'individui in specie animali conviventi. *Mem. R. Accad. Naz. dei Lincei. Ser. VI* 2 (1926)
23. Zadeh, L.: Fuzzy sets. *Inform. Control* 8, 338–353 (1965)
24. Zadeh, L.: The concept of a linguistic variable and its application to approximate reasoning, parts 1, 2, and 3. *Information Sciences* 8, 199–249, 301–357 (1975)
25. Zadeh, L.: Fuzzy logic. *Computer* 1(4), 83–93 (1988)
26. Zavala, M., Diaz-Sierra, R., Purves, D., Zea, G., Urbietta, I.: Modelos espacialmente explicitos. *Ecosistemas* 15(3), 88–99 (2006)

Using Gaussian Copulas in Supervised Probabilistic Classification

Rogelio Salinas-Gutiérrez, Arturo Hernández-Aguirre,
Mariano J.J. Rivera-Meraz, and Enrique R. Villa-Diharce

Center for Research in Mathematics (CIMAT), Guanajuato, México
{rsalinas, artha, mrivera, villadi}@cimat.mx

Abstract. This chapter introduces copula functions and the use of the Gaussian copula function to model probabilistic dependencies in supervised classification tasks. A copula is a distribution function with the implicit capacity to model non linear dependencies via concordance measures, such as Kendall's τ . Hence, this chapter studies the performance of a simple probabilistic classifier based on the Gaussian copula function. Without additional preprocessing of the source data, a supervised pixel classifier is tested with a 50-images benchmark; the experiments show this simple classifier has an excellent performance.

Keywords: Gaussian copula, supervised classification.

1 Introduction

In Pattern Recognition applications many algorithms and models have been proposed for many tasks, specially for *clustering*, *regression* and *classification*. Applications in which a *training data set* with categories and attributes is available and the goal is to assign a new object to one of a finite number of discrete categories are known as *supervised classification* problems [2, 12, 15]. In this work we present the use of the Gaussian copula function as an alternative for modeling dependence structure in a supervised probabilistic classifier.

Copula functions are suitable tools in statistics for modeling multiple dependence, not necessarily linear dependence, in several random variables. For this reason, copula functions have been widely used in economics and finance [5, 7, 9, 25, 26]. More recently copula function have been used in other fields such as climate [22], oceanography [6], hydrology [10], geodesy [1], reliability [17], evolutionary computation [20, 21] and engineering [11]. By using copula

theory, a joint distribution can be built with a copula function and, possibly, several different marginal distributions. Copula theory has been used also for modeling multivariate distributions in *unsupervised learning* problems such as image segmentation [4, 8] and retrieval tasks [16, 19, 24]. In [13], the bivariate copula functions Ali-Mikhail-Haq, Clayton, Frank and Gumbel are used for unsupervised classification. These copulas are well defined for two variables but when extended to three or more variables several complications arise (for instance, undefined copula parameters), preventing their generalization and applicability. For the Gaussian copula however, there exist a simple “general formula” for any number of variables. This work introduces the use of Gaussian copula in supervised classification, and compares an independent probabilistic classifier with a copula-based probabilistic classifier.

The content of the chapter is the following: Section 2 is a short introduction to copula functions, Section 3 presents a copula based probabilistic model for classification. Section 4 presents the experimental setting to classify an image database, and Section 5 summarizes the conclusions.

2 Copula Functions

The copula concept was introduced 50 years ago by Sklar [23] to separate the effect of dependence from the effect of marginal distributions in a joint distribution. Although copula functions can model linear and nonlinear dependencies, they have been barely used in computer science applications where nonlinear dependencies are common and need to be represented.

Definition 1. *A copula C is a joint distribution function of standard uniform random variables. That is,*

$$C(u_1, \dots, u_d) = P(U_1 \leq u_1, \dots, U_d \leq u_d) \ ,$$

where $U_i \sim U(0, 1)$ for $i = 1, \dots, d$.

For a more formal definition of copula functions, the reader is referred to [14, 18]. The following result, known as Sklar’s theorem, states how a copula function is related to a joint distribution function.

Theorem 1 (Sklar’s theorem). *Let F be a d -dimensional distribution function with marginals F_1, F_2, \dots, F_d , then there exists a copula C such that for all x in $\overline{\mathbb{R}}^d$,*

$$F(x_1, x_2, \dots, x_d) = C(F_1(x_1), F_2(x_2), \dots, F_d(x_d)) \ ,$$

where $\overline{\mathbb{R}}$ denotes the extended real line $[-\infty, \infty]$. If $F_1(x_1), F_2(x_2), \dots, F_d(x_d)$ are all continuous, then C is unique. Otherwise, C is uniquely determined on $\text{Ran}(F_1) \times \text{Ran}(F_2) \times \dots \times \text{Ran}(F_d)$, where Ran stands for the range.

According to Theorem 1, any joint distribution function F with continuous marginals F_1, F_2, \dots, F_d has associated a copula function C . Moreover, the associated copula C is a function of the marginal distributions F_1, F_2, \dots, F_d . An important consequence of Theorem 1 is that the d -dimensional joint density f and the marginal densities f_1, f_2, \dots, f_d are also related:

$$f(x_1, \dots, x_d) = c(F_1(x_1), \dots, F_d(x_d)) \cdot \prod_{i=1}^d f_i(x_i) , \quad (1)$$

where c is the density of the copula C . The Equation (1) shows that the product of marginal densities and a copula density builds a d -dimensional joint density. Notice that the dependence structure is given by the copula function and the marginal densities can be of different distributions. This contrasts with the usual way to construct multivariate distributions, which suffers from the restriction that the marginals are usually of the same type. The separation between marginal distributions and a dependence structure explains the modeling flexibility given by copula functions.

2.1 Gaussian Copula Function

There are several parametric families of copula functions, such as Student's t copula and Archimedean copulas. One of these families is the Gaussian copula function.

Definition 2. *The copula associated to the joint standard Gaussian distribution is called Gaussian copula.*

According to Definition 2 and Theorem 1, if the d -dimensional distribution of a random vector (Z_1, \dots, Z_d) is a joint standard Gaussian distribution, then the associated Gaussian copula has the following expression:

$$C(\Phi(z_1), \dots, \Phi(z_d); \Sigma) = \int_{-\infty}^{z_1} \dots \int_{-\infty}^{z_d} \frac{e^{-\frac{1}{2}t' \Sigma^{-1}t}}{(2\pi)^{(n/2)} |\Sigma|^{1/2}} dt_d \dots dt_1 ,$$

or equivalently,

$$C(u_1, \dots, u_d; \Sigma) = \int_{-\infty}^{\Phi^{-1}(u_1)} \dots \int_{-\infty}^{\Phi^{-1}(u_d)} \frac{e^{-\frac{1}{2}t' \Sigma^{-1}t}}{(2\pi)^{(n/2)} |\Sigma|^{1/2}} dt_d \dots dt_1 ,$$

where Φ is the cumulative distribution function of the marginal standard Gaussian distribution and Σ is a symmetric matrix with main diagonal of ones. The elements outside the main diagonal of matrix Σ are the pairwise correlations ρ_{ij} between variables Z_i and Z_j , for $i, j = 1, \dots, d$ and $i \neq j$. It can be noticed that a d -dimensional standard Gaussian distribution has mean vector zero and a correlation matrix Σ with $d(d-1)/2$ parameters.

Algorithm 1. Pseudocode for estimating parameters

- 1: for each random variable X_i , $i = 1, \dots, d$, estimate its marginal distribution function \hat{F}_i using the observed values x_i . The marginal distribution function can be parametric or nonparametric
 - 2: determine $u_i = \hat{F}_i(x_i)$, for $i = 1, \dots, d$
 - 3: calculate $z_i = \Phi^{-1}(u_i)$ where Φ is the cumulative standard Gaussian distribution function, for $i = 1, \dots, d$
 - 4: estimate the correlation matrix $\hat{\Sigma}$ for the random vector (Z_1, \dots, Z_d) using pseudo observations (z_1, \dots, z_d)
-

The dependence parameters ρ_{ij} of a d -dimensional Gaussian copula can be estimated using the maximum likelihood method. To do so, we follow the steps of Algorithm 1.

Due to Equation (1), the d -dimensional Gaussian copula density can be calculated as:

$$\begin{aligned}
 c(\Phi(z_1), \dots, \Phi(z_d); \Sigma) &= \frac{\frac{1}{(2\pi)^{(d/2)}|\Sigma|^{1/2}} e^{-\frac{1}{2}z'\Sigma^{-1}z}}{\prod_{i=1}^d \frac{1}{(2\pi)^{1/2}} e^{-\frac{1}{2}z_i^2}} \\
 &= \frac{1}{|\Sigma|^{1/2}} e^{-\frac{1}{2}z'(\Sigma^{-1}-I)z} . \tag{2}
 \end{aligned}$$

Given that a Gaussian copula is a distribution function it is possible to simulate data from it. The main steps are the following: once a correlation matrix Σ is specified, a data set can be generated from a joint standard Gaussian distribution. The next step consists of transforming this data set using the cumulative distribution function Φ . For random vectors with a Gaussian copula associated to their joint distribution, the first step is to generate data from the copula and then determining their quantiles by means of their cumulative distribution functions. Algorithm 2 and Figures 1, 2 and 3 illustrate the sampling procedure for different correlations.

Algorithm 2. Pseudocode for generating data with Gaussian dependence structure

- 1: simulate observations (z_1, \dots, z_d) from a joint standard Gaussian distribution with matrix correlation Σ
 - 2: calculate $u_i = \Phi(z_i)$ where Φ is the cumulative standard Gaussian distribution function, for $i = 1, \dots, d$
 - 3: determine x_i using quasi-inverse $F_i^{-1}(u_i)$, where F_i is a cumulative distribution function, for $i = 1, \dots, d$
-

Figure 1-(a) shows 500 bivariate data with correlation $\rho = -0.5$ drawn from a bivariate standard Gaussian distribution (step 1, Algorithm 2). The histogram on the vertical axis and the histogram on the horizontal axis

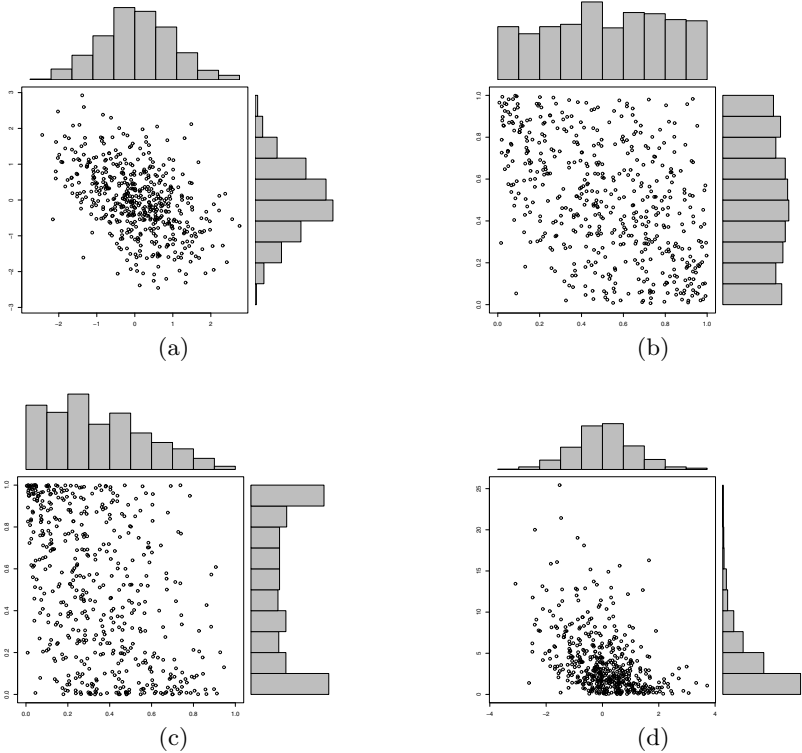


Fig. 1. (a) A sample of 500 points from a standard Gaussian distribution with parameter $\rho = -0.50$. (b) The corresponding sample for a Gaussian copula. (c) The associated sample for a joint distribution with marginal Beta distributions with parameters (1, 2) (histogram on the horizontal axis) and (0.5, 0.5) (histogram on the vertical axis). (d) The associated sample for a joint distribution with marginal t-Student distribution with 8 degrees of freedom (histogram on the horizontal axis) and marginal exponential distribution with mean 4 (histogram on the vertical axis).

illustrate that both marginals are univariate standard Gaussian distributions. This data set is used to obtain a sample from a Gaussian copula, as shown in Figure 1-(b) (step 2, Algorithm 2). Both histograms illustrate that marginals are uniform, according to Definition 1. Figure 1-(c) shows a sample from a joint distribution with Gaussian copula and Beta marginals (step 3, Algorithm 2). This sample is obtained using the data set of Figure 1-(b). Figure 1-(d) shows a sample from a joint distribution with Gaussian copula, Student's t marginal distribution and exponential marginal distribution (step 3, Algorithm 2). This sample is also obtained from the data set of Figure 1-(b). In order to appreciate how the correlation parameter modifies

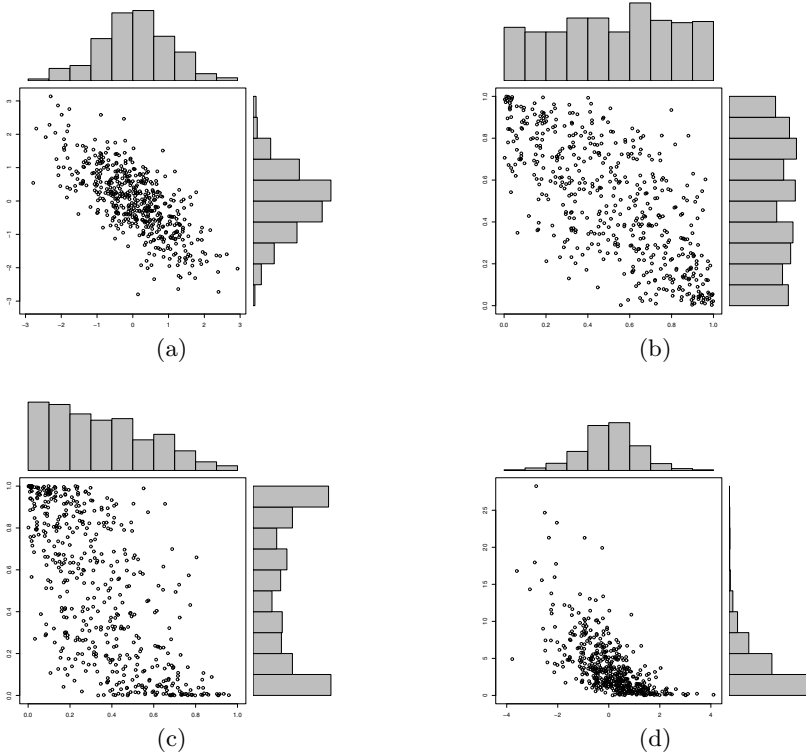


Fig. 2. (a) A sample of 500 points from a standard Gaussian distribution with parameter $\rho = -0.70$. (b) The corresponding sample for a Gaussian copula. (c) The associated sample for a joint distribution with marginal Beta distributions with parameters (1,2) (histogram on the horizontal axis) and (0.5,0.5) (histogram on the vertical axis). (d) The associated sample for a joint distribution with marginal t-Student distribution with 8 degrees of freedom (histogram on the horizontal axis) and marginal exponential distribution with mean 4 (histogram on the vertical axis).

the dependence structure, Figures 2 and 3 show the same information as Figure 1 with $\rho = -0.7$ and $\rho = -0.95$, respectively.

Although the correlation is used to generate data from a Gaussian copula, it is not necessary the same for joint distributions with Gaussian copula and non-Gaussian marginals. However, the data sets in Figure 1 have the **same concordance value** measured in Kendall's τ . This important result for parametric bivariate copulas (see [18]) is explained through the equation:

$$\tau(X_1, X_2) = 4 \int_0^1 \int_0^1 C(u_1, u_2; \theta) dC(u_1, u_2; \theta) - 1 \quad (3)$$

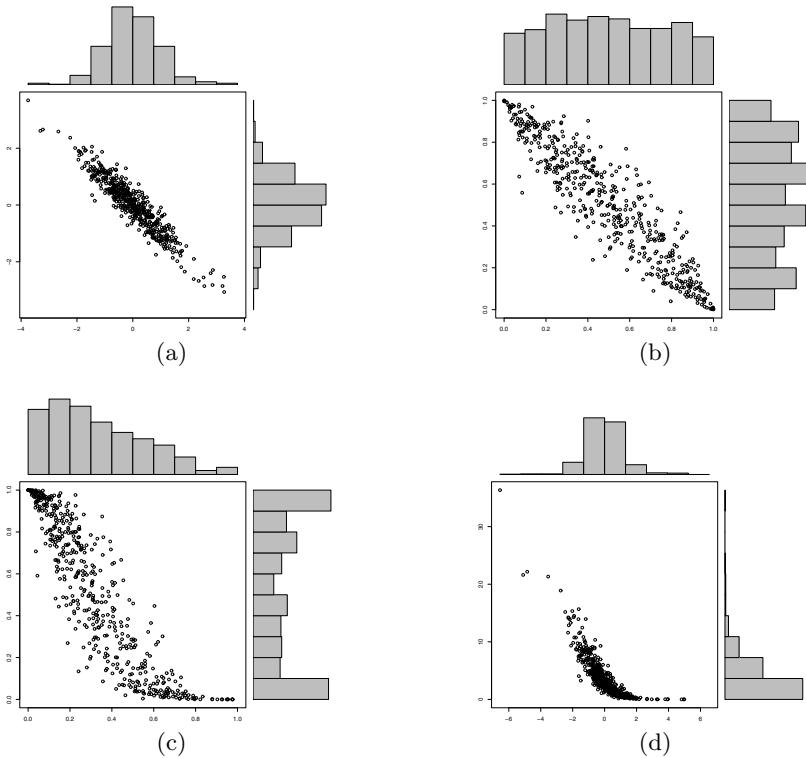


Fig. 3. (a) A sample of 500 points from a standard Gaussian distribution with parameter $\rho = -0.95$. (b) The corresponding sample for a Gaussian copula. (c) The associated sample for a joint distribution with marginal Beta distributions with parameters (1,2) (histogram on the horizontal axis) and (0.5,0.5) (histogram on the vertical axis). (d) The associated sample for a joint distribution with marginal t-Student distribution with 8 degrees of freedom (histogram on the horizontal axis) and marginal exponential distribution with mean 4 (histogram on the vertical axis).

which relates the dependence parameter θ of a copula and Kendall's τ . For a bivariate Gaussian copula, Equation (3) can be written as

$$\tau = \frac{2}{\pi} \arcsin(\rho) . \tag{4}$$

Given that is well established how to estimate correlation matrixes, evaluate densities, and calculate integrals for the multidimensional Gaussian distribution, the Gaussian copula function is relatively easy to implement.

3 The Probabilistic Classifier

As noted, the aim of this work is to introduce the use of Gaussian copula functions in supervised classification. According to Theorem 1, we can employ a copula function in a probabilistic classifier, such as a Bayesian classifier. In this section we present a three dimensional probabilistic model based on three empirical distribution functions and a trivariate dimensional Gaussian copula function.

The Bayes' theorem states the following:

$$P(K = k|E = e) = \frac{P(E = e|K = k) \times P(K = k)}{P(E = e)}, \quad (5)$$

where $P(K = k|E = e)$ is the posterior probability, $P(E = e|K = k)$ is the likelihood function, $P(K = k)$ is the prior probability and $P(E = e)$ is the data probability.

The Equation (5) has been used as a tool in supervised classification. A probabilistic classifier can be designed comparing the posterior probability that an object belongs to class K given its attributes E . The object is then assigned to the class with the highest posterior probability. For practical reasons, the data probability $P(E)$ does not need to be evaluated for comparing posterior probabilities. Furthermore, the prior probability $P(K)$ can be substituted by an uniform distribution if the user does not have an informative distribution.

3.1 The Probabilistic Classifier Based on Gaussian Copula Function

For continuous attributes, a Gaussian copula function can be used for modeling the dependence structure in the likelihood function. In this case, the Bayes' theorem can be written as:

$$P(K = k|e) = \frac{c(F_1(e_1), \dots, F_n(e_n)|k, \Sigma) \times \prod_{i=1}^n f_i(e_i|k) \times P(K = k)}{f(e_1, \dots, e_n)}, \quad (6)$$

where F_i and f_i are the marginal distribution functions and the marginal densities of attributes, respectively. The function c is a d -dimensional Gaussian copula density defined by Equation (2). As can be seen in Equation (6), each category determines a likelihood function.

3.2 The Probabilistic Classifier Based on Independent Model

By considering conditional independence among the attributes in Equation (6), or equivalently, an independent structure in the likelihood function given a category, a probabilistic classifier can use the following expression in order to calculate posterior probabilities:

$$P(K = k|e) = \frac{\prod_{i=1}^n f_i(e_i|k) \times P(K = k)}{f(e_1, \dots, e_n)} . \quad (7)$$

Equation (7) uses an independent structure given by a copula density equals to one. This independent copula density can be also obtained by a Gaussian copula function when matrix Σ is the identity matrix I .

3.3 An Application Example

Consider the following specific classification problem: assign a pixel to a certain class according to its color attributes. If we have information about the color distribution of each class, then we can use this information and the Bayes' theorem in order to classify new pixels. This is an example of supervised classification. For a red-green-blue (RGB) color space and two classes, a Gaussian copula based classifier can be written as

$$P(k|r, g, b) = \frac{c(F_R(r), F_G(g), F_B(b)|k, \Sigma) f_R(r|k) f_G(g|k) f_B(b|k) \times P(k)}{f(r, g, b)} , \quad (8)$$

where c is a trivariate Gaussian copula density.

In order to classify a pixel, we use in Equation (8) a prior probability $P(K = k)$ based on the uniform distribution, nonparametric marginal densities \hat{f} based on histograms to approximate $f_R(r|k)$, $f_G(g|k)$ and $f_B(b|k)$, and nonparametric marginal distributions \hat{F} based on empirical cumulative distribution functions to approximate $F_R(r)$, $F_G(g)$ and $F_B(b)$. For modeling the dependence structure of the likelihood function $f(r, g, b|k)$ we present the use of a trivariate Gaussian copula function.

4 Experiments

We use two probabilistic models in order to classify pixels of 50 test images. The first model is an independent probabilistic model (I-M) based on the product of marginal distributions. The second model is a copula-based model (GC-M) that takes into account a dependence structure by means of a trivariate Gaussian copula. The image database was used in [3] and is available online [27]. This image database provides information about two classes: the foreground and the background. The training data and the test data are contained in the labelling-lasso files [27], whereas the correct classification is contained in the segmentation files. Figures 4, 5 and 6 show the description of three images from the database. Table 3 shows a description for each image. Although the database is used for segmentation purposes, the aim of this work is to introduce the use of the Gaussian copula function in supervised color pixel classification. We use the information for supervised color pixel classification, without taking into account the spatial information.

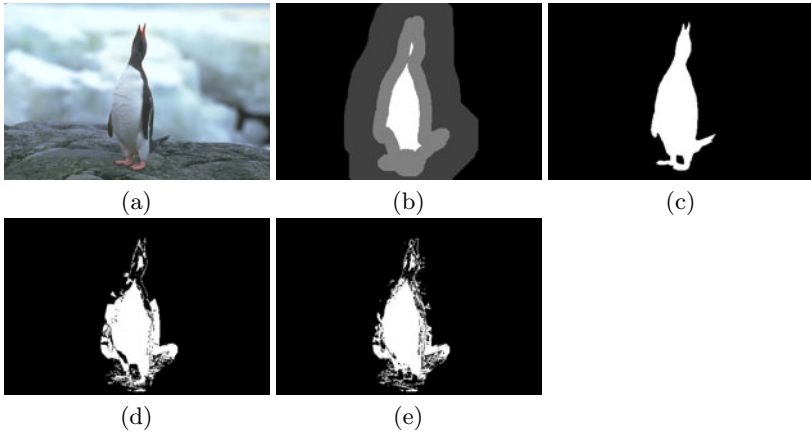


Fig. 4. (a) The color image. (b) The labelling-lasso image with the training data for background (dark gray), for foreground (white) and the test data (gray). (c) The correct classification with foreground (white) and background (black). (d) Classification made by I-M. (e) Classification made by GC-M.

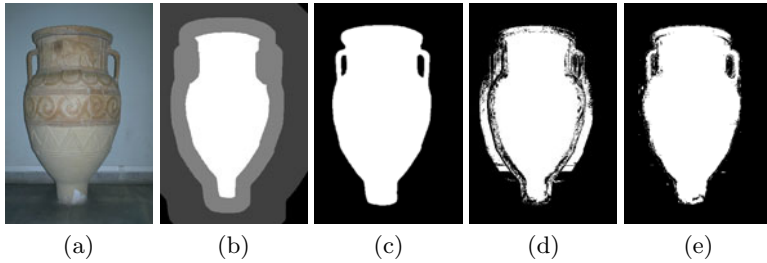


Fig. 5. (a) The color image. (b) The labelling-lasso image with the training data for background (dark gray), for foreground (white) and the test data (gray). (c) The correct classification with foreground (white) and background (black). (d) Classification made by I-M. (e) Classification made by GC-M.

Two evaluation measures are used in this work: *accuracy* and *Tanimoto coefficient*. The accuracy is described in Figure 7. We define the positive class as foreground and the negative class as background.

The Tanimoto coefficient (TC) is also known as *Jaccard similarity measure*. This measure, TC, is defined as:

$$TC(k) = \frac{V_{m \cap g}(k)}{V_{m \cup g}(k)},$$

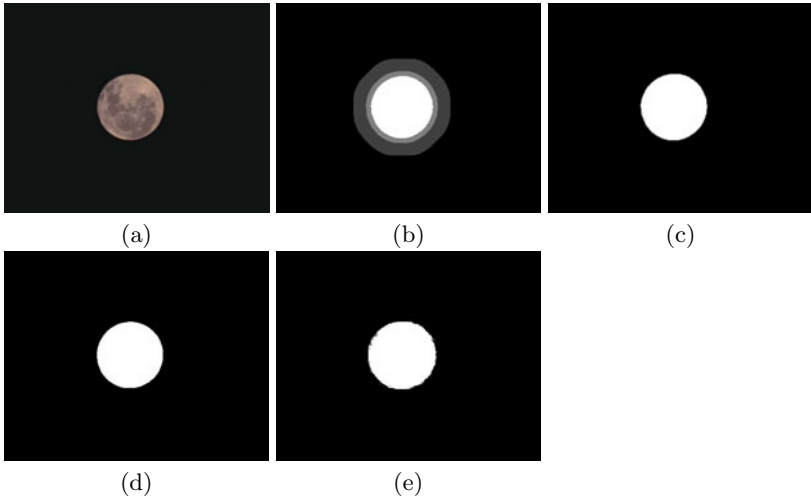


Fig. 6. (a) The color image. (b) The labelling-lasso image with the training data for background (dark gray), for foreground (white) and the test data (gray). (c) The correct classification with foreground (white) and background (black). (d) Classification made by I-M. (e) Classification made by GC-M.

		Truth	
		Positive	Negative
Model	Positive	tp	fp
	Negative	fn	tn

$$accuracy = \frac{tp + tn}{tp + fp + fn + tn}$$

(a)

(b)

Fig. 7. (a) A confusion matrix for binary classification, where tp are true positive, fp false positive, fn false negative, and tn true negative counts. (b) Definition of accuracy used in this work.

where $V_{m \cap g}(k)$ denotes the number of pixels classified as class k by both the model and the ground truth and $V_{m \cup g}(k)$ denotes the number of pixels classified as class k by either the model or the ground truth.

4.1 Numerical Results

In Table 1 we summarize the measure values reached by the independent probabilistic model (I-M) and the copula-based model (GC-M). The information about the number of pixels well classified for each class is reported in

Table 3. We include in Table 4 the performances of I-M and GC-M for each image.

Table 1. Descriptive results for all evaluation measures. BG stands for the background class and FG stands for the foreground class.

Measure	Minimum	Median	Mean	Maximum	Std. deviation
I-M					
Tanimoto coefficient – BG	0.369	0.690	0.695	0.955	0.143
Tanimoto coefficient – FG	0.341	0.633	0.638	0.953	0.168
Accuracy	0.571	0.792	0.795	0.976	0.107
GC-M					
Tanimoto coefficient – BG	0.450	0.816	0.797	0.976	0.118
Tanimoto coefficient – FG	0.375	0.780	0.758	0.972	0.141
Accuracy	0.587	0.889	0.871	0.987	0.083

To properly compare the performance of the probabilistic models, we conducted a hypothesis test based on a Bootstrap method for the differences between the means of accuracy and Tanimoto coefficients, for both probabilistic models. Table 2 shows the confidence interval for the means, and the corresponding p-value.

Table 2. Results for the difference between evaluation measure means in each model. A 95% confidence interval and a p-value are obtained through a Bootstrap technique. BG stands for the background class and FG stands for the foreground class.

Measure	95% Interval		p-value
Tanimoto coefficient – BG	-1.52E-01	-5.18E-02	2.67E-04
Tanimoto coefficient – FG	-1.80E-01	-5.96E-02	3.67E-04
Accuracy	-1.14E-01	-3.94E-02	2.67E-04

4.2 Discussion

According to Table 1, the GC-M shows the best behaviour for all evaluation measures. For instance, the mean accuracy for the I-M, 79.5%, is less than the mean accuracy for the GC-M, 87.1%. This means that using a I-M approximately has 8% more error rate than using a GC-M.

The average of the Tanimoto coefficient for the background class is greater than the average of the Tanimoto coefficient for the foreground class, for both I-M and GC-M (see Table 1). These coefficients are shown for each image in Figure 8-(a) and Figure 8-(b). Notice the Tanimoto coefficients for GC-M on the background and foreground are very similar, denoting a better classification than I-M. In most of the images and for each class, we can see in Figure 8-(c) and 8-(d) (also in Table 4) that GC-M outperforms I-M. In

average, according to Table 1, the GC-M improves the I-M in both classes. For the foreground class from 63.8% to 75.8%, and for the background class from 69.5% to 79.7%.

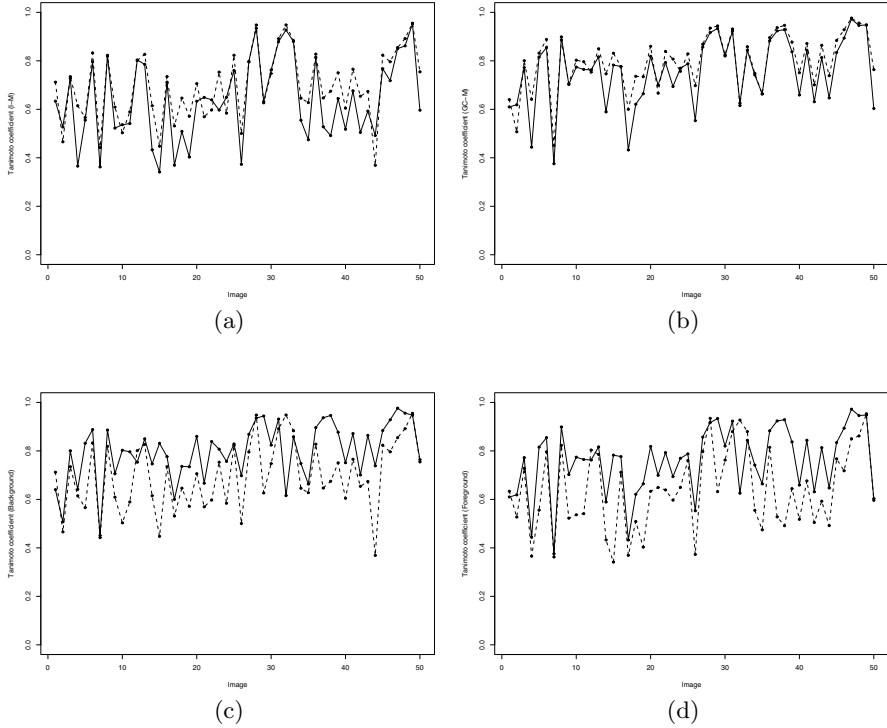


Fig. 8. Tanimoto coefficient for the supervised pixel classification on the 50 images of [3]. Image order is the same as in Table 4. (a) Background (dashed line) and foreground (solid line) for the I-M. (b) Background (dashed line) and foreground (solid line) for the GC-M. (c) I-M (dashed line) and GC-M (solid line) for the background class. (d) I-M (dashed line) and GC-M (solid line) for the foreground class.

Table 1 also shows information about the standard deviations for each evaluation measure. For all cases, the standard deviation indicates that using a GC-M in pixel classification is more consistent than using an I-M.

In order to statistically compare the performance of the probabilistic models, Table 2 shows confidence intervals and p-values that confirm differences between the models. None of confidence intervals include the 0 value and all p-values are less than $\alpha = 0.05$.

5 Conclusions

In this work we introduce the use of Gaussian copulas in supervised pixel classification. According to numerical experiments the selection of a Gaussian copula for modeling structure dependence can help achieve better classification results. An specific example is the image *227092*, which appears in Figure 5, its accuracy for the I-M classifier is 57.1%, whereas its accuracy for the GC-M classifier is 89.5%. For this image, the Gaussian copula improves its accuracy.

Although we model the dependence structure for each image with the same copula function, this is not necessary. There are many copula functions and the Gaussian copula has been chosen due to its practical usefulness and easy implementation. However, having more than one copula at hand may improve the performance of the copula-based classifier. In such case, a copula selection procedure is necessary. The evaluation results are the consequence of the selected dependence structure and marginals. For instance, on the image *106024*, Figure 4, the performance of the I-M classifier is 57.6% accurate (accuracy), whereas the GC-M classifier is 58.7% accurate. For most applications better results can be obtained by selecting the best fitted copula function from a set of available copulas. For example, in the experiment reported, the performance of the I-M classifier is better than GC-M for image *fullmoon*, Figure 6. However, the GC-M is expected to improve the performance of the I-M classifier if we used the proper copula.

Acknowledgments. The first author acknowledges support from the National Council of Science and Technology of México (CONACyT) through a scholarship to pursue graduate studies in the Department of Computer Science at the Center for Research in Mathematics.

References

1. Bacigál, T., Komorníková, M.: Fitting Archimedean copulas to bivariate geodetic data. In: Rizzi, A., Vichi, M. (eds.) *Compstat 2006 Proceedings in Computational Statistics*, pp. 649–656. Physica-Verlag, Heidelberg (2006)
2. Bishop, C.M.: *Pattern Recognition and Machine Learning*. Springer, Heidelberg (2007)
3. Blake, A., Rother, C., Brown, M., Perez, P., Torr, P.: Interactive Image Segmentation using an adaptive GMMRF model. In: *Proc. European Conference in Computer Vision (ECCV)*, Springer, Heidelberg (2004)
4. Brunel, N., Pieczynski, W., Derrode, S.: Copulas in vectorial hidden Markov chains for multicomponent image segmentation. In: *Proceedings of the 2005 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 717–720 (2005)
5. Cherubini, U., Luciano, E., Vecchiato, W.: *Copula Methods in Finance*. Wiley, Chichester (2004)
6. De-Waal, D.J., Van-Gelder, P.H.A.J.M.: Modelling of extreme wave heights and periods through copulas. *Extremes* 8(4), 345–356 (2005)

7. Dowd, K.: Copulas in Macroeconomics. *Journal of International and Global Economic Studies* 1(1), 1–26 (2008)
8. Flitti, F., Collet, C., Joannic-Chardin, A.: Unsupervised Multiband Image Segmentation using Hidden Markov Quadtree and Copulas. In: *IEEE International Conference on Image Processing* (2005)
9. Frees, E.W., Valdez, E.A.: Understanding relationships using copulas. *North American Actuarial Journal* 2(1), 1–25 (1998)
10. Genest, C., Favre, A.C.: Everything You Always Wanted to Know about Copula Modeling but Were Afraid to Ask. *Journal of Hydrologic Engineering* 12(4), 347–368 (2007)
11. Grigoriu, M.: Multivariate distributions with specified marginals: Applications to Wind Engineering. *Journal of Engineering Mechanics* 133(2), 174–184 (2007)
12. Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, Heidelberg (2009)
13. Jajuga, K., Papla, D.: Copula Functions in Model Based Clustering. In: *Proceedings of the 29th Annual Conference of the Gesellschaft für Klassifikation e.V.*, pp. 606–613. Springer, Heidelberg (2005)
14. Joe, H.: *Multivariate models and dependence concepts*. Chapman and Hall, London (1997)
15. Mackay, D.J.C.: *Information Theory, Inference, and Learning Algorithms*, Cambridge (2008)
16. Mercier, G., Bouchemakh, L., Smara, Y.: The Use of Multidimensional Copulas to Describe Amplitude Distribution of Polarimetric SAR Data. In: *IGARSS (2007)*
17. Monjardin, P.E.: *Análisis de dependencia en tiempo de falla*. Master’s thesis, Centro de Investigación en Matemáticas. Guanajuato, México (2007) (in Spanish)
18. Nelsen, R.B.: *An Introduction to Copulas*. Springer, Heidelberg (2006)
19. Sakji-Nsibi, S., Benazza-Benyahia, A.: Multivariate indexing of multichannel images based on the copula theory. In: *IPTA 2008* (2008)
20. Salinas-Gutiérrez, R., Hernández-Aguirre, A., Villa-Diharce, E.R.: Using Copulas in Estimation of Distribution Algorithms. In: *MICAI 2009: Advances in Artificial Intelligence*, pp. 658–668. Springer, Heidelberg (2009)
21. Salinas-Gutiérrez, R., Hernández-Aguirre, A., Villa-Diharce, E.R.: D-vine EDA: a new Estimation of Distribution Algorithm based on Regular Vines. In: *Genetic and Evolutionary Conference, GECCO 2010* (2010) (accepted for publication)
22. Schölzel, C., Friederichs, P.: Multivariate non-normally distributed random variables in climate research – introduction to the copula approach. *Nonlinear Processes in Geophysics* 15(5), 761–772 (2008)
23. Sklar, A.: Fonctions de répartition à n dimensions et leurs marges. *Publications de l’Institut de Statistique de l’Université de Paris* 8, 229–231 (1959)
24. Stitou, Y., Lasmar, N., Berthoumieu, Y.: Copulas based multivariate gamma modeling for texture classification. In: *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1045–1048 (2009)
25. Trivedi, P.K., Zimmer, D.M.: *Copula Modeling: An Introduction for Practitioners*. Foundations and Trends[®] in Econometrics Now Publishers, vol. 1 (2007)

26. Venter, G., Barnett, J., Kreps, R., Major, J.: Multivariate Copulas for Financial Modeling. *Variance* 1(1), 103–119 (2007)
27. Image and Video Editing,
<http://research.microsoft.com/en-us/um/cambridge/projects/visionimagevideoediting/segmentation/grabcut.htm>

Appendix

Table 3. Description of images used in this work. BG stands for the background class and FG stands for the foreground class. Columns 3 and 4 give the size of test data. The last 4 columns give the number of pixels well classified for each class and for each probabilistic classifier.

Image name	Image size	Test pixels		I-M		GC-M	
		BG	FG	BG	FG	BG	FG
21077	321 × 481	4322	3353	3648	2551	3144	2763
24077	321 × 481	11529	10983	6577	8399	6348	10000
37073	321 × 481	8260	6642	6404	6187	7115	6014
65019	321 × 481	9853	8398	9181	3317	9099	4061
69020	321 × 481	25203	22634	17561	16813	22798	20421
86016	321 × 481	3271	2215	2765	2166	2937	2179
106024	321 × 481	9093	7368	5528	3961	5574	4087
124080	321 × 481	18286	18773	16487	16924	16307	18653
153077	321 × 481	13851	12098	11072	7774	10806	10638
153093	321 × 481	12027	11809	7617	8699	11414	9615
181079	481 × 321	23845	23110	18650	15320	22494	18705
189080	481 × 321	23363	23523	20726	21020	19722	20707
208001	481 × 321	10227	9530	9994	7669	10064	7914
209070	321 × 481	6696	4075	5117	2447	5894	2874
227092	481 × 321	19656	17321	12869	8229	19129	13966
271008	321 × 481	10909	9216	8934	7967	8800	8795
304074	481 × 321	7239	4794	5017	2591	5534	2810
326038	321 × 481	10781	7680	8730	4952	9488	5571
376043	481 × 321	13654	13485	12022	6094	13072	9343
388016	481 × 321	17800	15592	15633	11248	17596	12929
banana1	480 × 640	29983	24052	17120	23964	20285	23601
banana2	480 × 640	27433	21518	17063	20378	25373	18698
banana3	480 × 640	26205	20164	25588	12405	26035	14115
book	480 × 640	26087	21474	15689	20699	19852	21325
bool	450 × 520	20123	16850	19500	13279	18726	14373
bush	600 × 450	32513	22099	21072	12504	27734	14870
ceramic	480 × 640	30549	25709	24809	25069	27328	24791
cross	600 × 450	34602	25733	32824	25703	32918	25132
doll	549 × 462	18866	15106	12976	13269	17947	14960
elefant	480 × 640	27858	22787	20918	22656	23158	22540
flower	450 × 600	16125	13246	14612	12977	15036	13225

Continued on next page

Table 3. (continued)

Image name	Image size	Test pixels		I-M		GC-M	
		BG	FG	BG	FG	BG	FG
fullmoon	350 × 442	1580	1043	1498	1043	983	1026
grave	600 × 450	12294	12832	11977	11567	12219	10889
llama	371 × 513	8930	8445	7783	5322	7547	7287
memorial	600 × 450	14853	12598	12900	6902	10936	10964
music	480 × 640	23945	19494	20457	18723	21794	19112
person1	450 × 600	19092	16384	16452	10041	18831	15372
person2	450 × 600	12796	9595	11492	5358	12465	9219
person3	600 × 450	14649	11450	13494	8122	14022	10112
person4	450 × 600	19250	16631	15230	10691	18197	11653
person5	600 × 450	13990	11332	13009	8327	13025	10377
person6	600 × 450	19015	15645	16753	9038	16071	11732
person7	600 × 450	12110	9634	9934	6998	11795	8093
person8	480 × 640	16684	12741	6740	11157	14690	9534
scissors	480 × 640	30768	23335	28152	19910	30181	19960
sheep	600 × 450	5331	3733	4750	3098	5243	3415
stone1	480 × 640	18716	15635	16087	15525	18376	15528
stone2	480 × 640	22002	18489	21556	16315	21788	17692
teddy	398 × 284	13892	13739	13790	13191	13426	13466
tennis	472 × 500	19471	13129	18054	8673	18322	8613

Table 4. Evaluation measures for each image. TC stands for the Tanimoto coefficient, BG stands for the background class and FG stands for the foreground class. Columns 2, 3 and 4 give the results for the independent probabilistic model. The last 3 columns give the results for the Gaussian copula-based probabilistic model.

Image name	I-M			GC-M		
	TC-BG	TC-FG	Accuracy	TC-BG	TC-FG	Accuracy
21077	0.712	0.633	0.808	0.640	0.610	0.770
24077	0.466	0.527	0.665	0.507	0.619	0.726
37073	0.735	0.728	0.845	0.801	0.772	0.881
65019	0.615	0.366	0.685	0.641	0.444	0.721
69020	0.566	0.555	0.719	0.832	0.816	0.903
86016	0.833	0.796	0.899	0.888	0.855	0.933
106024	0.442	0.362	0.576	0.450	0.375	0.587
124080	0.819	0.823	0.902	0.886	0.899	0.943
153077	0.609	0.523	0.726	0.706	0.703	0.826
153093	0.503	0.536	0.685	0.803	0.774	0.882
181079	0.590	0.541	0.723	0.796	0.765	0.877
189080	0.801	0.804	0.890	0.753	0.762	0.862
208001	0.827	0.786	0.894	0.850	0.816	0.910
209070	0.615	0.433	0.702	0.746	0.589	0.814
227092	0.448	0.341	0.571	0.831	0.782	0.895

Continued on next page

Table 4. (continued)

Image name	I-M			GC-M		
	TC-BG	TC-FG	Accuracy	TC-BG	TC-FG	Accuracy
271008	0.735	0.712	0.840	0.777	0.777	0.874
304074	0.531	0.369	0.632	0.600	0.432	0.693
326038	0.646	0.509	0.741	0.736	0.621	0.816
376043	0.571	0.403	0.668	0.735	0.664	0.826
388016	0.706	0.633	0.805	0.860	0.818	0.914
banana1	0.569	0.649	0.760	0.667	0.699	0.812
banana2	0.597	0.639	0.765	0.839	0.793	0.900
banana3	0.753	0.597	0.819	0.807	0.694	0.866
book	0.584	0.649	0.765	0.757	0.770	0.866
bool	0.823	0.760	0.887	0.829	0.788	0.895
bush	0.500	0.373	0.615	0.698	0.553	0.780
ceramic	0.795	0.797	0.887	0.868	0.857	0.926
cross	0.948	0.934	0.970	0.935	0.917	0.962
doll	0.627	0.632	0.773	0.944	0.934	0.969
elefant	0.747	0.762	0.860	0.824	0.820	0.902
flower	0.891	0.879	0.939	0.931	0.923	0.962
fullmoon	0.948	0.927	0.969	0.616	0.626	0.766
grave	0.883	0.880	0.937	0.858	0.844	0.920
llama	0.646	0.555	0.754	0.748	0.741	0.854
memorial	0.628	0.474	0.721	0.663	0.664	0.798
music	0.828	0.815	0.902	0.896	0.883	0.942
person1	0.647	0.528	0.747	0.937	0.924	0.964
person2	0.675	0.492	0.753	0.946	0.929	0.968
person3	0.751	0.644	0.828	0.877	0.837	0.925
person4	0.605	0.518	0.722	0.751	0.659	0.832
person5	0.765	0.676	0.843	0.872	0.844	0.924
person6	0.654	0.505	0.744	0.701	0.631	0.802
person7	0.674	0.593	0.779	0.864	0.813	0.915
person8	0.369	0.492	0.608	0.739	0.647	0.823
scissors	0.823	0.767	0.888	0.884	0.834	0.927
sheep	0.796	0.718	0.866	0.928	0.894	0.955
stone1	0.855	0.850	0.920	0.976	0.972	0.987
stone2	0.892	0.862	0.935	0.956	0.946	0.975
teddy	0.955	0.953	0.976	0.948	0.948	0.973
tennis	0.755	0.596	0.820	0.764	0.603	0.826

Subjective Colocalization Analysis with Fuzzy Predicates

Pablo Rivas-Perea, Jose Gerardo Rosiles, and Wei Qian

The University of Texas El Paso,
Department of Electrical and Computer Engineering, El Paso TX 79968, USA

Abstract. Understanding the protein-to-protein interactions at the subcellular level, as well as other organic molecules, is crucial to explain cellular functions and to elucidate disease mechanisms. These interactions can be captured visually by overlapping the fluorescent microscopic images of two proteins tagged with fluorescent labeling agents that react to green and red wavelengths respectively. Interaction is determined by subjectively assessing the amount colocalization of green and red on the image composite based on the amount of yellow present in the image composite (i.e., green and red form yellow). Attempts to reduce the subjectivity of this process have focused on the computation of statistical coefficients and related methods. Even though statistical colocalization coefficients give a degree of correlation among the imaged proteins, they still need to be interpreted with subjective qualifiers like "high", "low", "strong", etc. Hence, there is no current agreement on the meaning of these coefficients among researchers. In this paper we propose the use of fuzzy linguistic variables to model the subjective interpretation of co-localization coefficients. Based on interpretations found in the literature, we produce a set of rules that map the coefficient values to a linguistic interpretation. The result of this work is a tool that provides an descriptive ensemble of coefficient interpretations that could guide researchers to a uniform interpretation colocalization criteria.

1 Introduction

Fluorescence microscopy makes possible the functional analysis of proteins and other sub-cellular structures [1, 2, 3]. This imaging technique consists of tagging proteins of interest using fluorescent labeling agents which react to a specific wavelength. The spatial distribution of a protein over a cell is captured through an image sensor tuned to the fluorochrome wavelength.

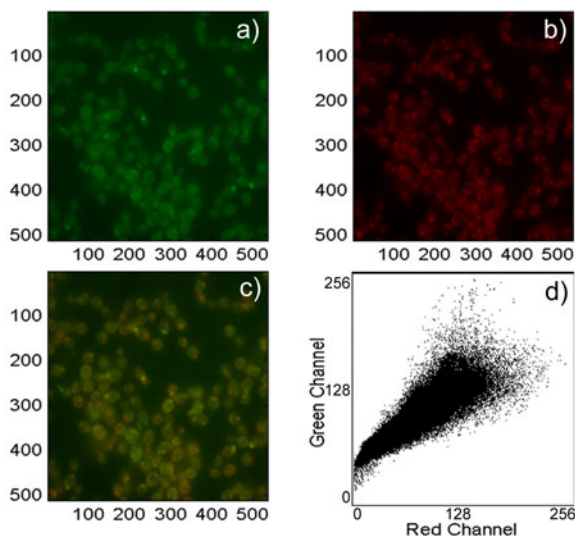


Fig. 1. An example image from the Yeast database. In (a) is shown the green channel. In (b) is shown the red channel, and in (c) the two joint channels. In (d) we have the scatter diagram showing the pixels distribution across the two channels.

Among the many uses of fluorescent microscopy images, it is possible to assess the interaction among two or more proteins at the sub-cellular level. Understanding these interactions is crucial to explain cellular functions and to elucidate disease mechanisms. The simultaneous presence of two proteins in the same sub-cellular compartment indicates a functional relationship for some specific biological process. Imaging the same specimen with two proteins tagged to express at different wavelengths (e.g., green and red), allows us to register the spatial distribution of the proteins simultaneously. By overlaying both images on the same plane, yellow regions would appear on those regions where red and green pixels spatially overlap. This overlaying method and the process of assessing the relative overlap (i.e., yellow regions) among the two proteins is known as *colocalization*.

An example using images from the yeast database [4, 5, 6] is shown in Figure 1, including the highlight of areas with strong visual colocalization. Biologists typically do a subjective assessment of colocalization through visual inspection of image overlays. Hence, colocalization results can vary from one person to another, or from one study to another. Attempts to reduce the subjectivity of this process have focused on the computation of statistical coefficients [7, 8, 9, 10] and related methods [11, 12, 13]. Even though statistical colocalization coefficients give a degree of correlation among the imaged proteins, they still need to be interpreted with subjective qualifiers like "high", "low", "strong", etc. Hence, there is no current agreement on the

meaning of these coefficients among researchers. In addition, they seem to be strongly influenced by the acquisition and post-processing steps applied to the images.

In this paper we propose the use of fuzzy linguistic variables [14, 15, 16, 17, 18] to model the subjective interpretation of co-localization coefficients. Based on interpretations found in the literature, we produce a set of rules that map the coefficient values to a linguistic interpretation. The result of this work is a tool that provides a descriptive ensemble of coefficient interpretations that could guide researchers to a uniform interpretation colocalization criteria. The proposed scheme can be described in two steps. First, using fluorescence microscopy images, a quantitative description of colocalization is obtained through the use of statistical coefficients reported in the literature [7, 8, 11, 12, 13, 19, 20] as features to fuzzy models. Second, based on the qualitative interpretation of different colocalization case studies, we constructed fuzzy linguistic variables to represent the quantitative results. This was followed by membership functions modeling. We tested our fuzzy model over the yeast and A431 cells image databases. The proposed model outputs natural language fuzzy predicates providing a consistent description of quantitative colocalization analysis results.

The paper is organized as follows. In Section 2 we give a brief introduction to the fluorescence microscopy imaging process. The quantitative colocalization feature extraction process is discussed in Section 3. In Section 4 we discuss the design our fuzzy logic-based subjective colocalization analysis model. Examples of the model utilization are presented in Section 5. We discuss our conclusions and subsequent work in Section 6.

2 An Overview of Fluorescence Microscopy

Fluorescence microscopy is an imaging technique where proteins (or some other molecule) are tagged using fluorescent labeling agents (fluorophores) which react to a specific wavelength. When different proteins on the same specimen are tagged, a multichannel image is generated, where each channel captures the contribution of each fluorophore. The image formation process can be approximated as a linear spatially-invariant system where the intensity distribution is modeled as [19]

$$I(n_1, n_2, n_3) \propto \int_{\mathbb{R}^3} \left| h_\lambda \left(\frac{n_1}{\hat{\varsigma}} - u, \frac{n_2}{\hat{\varsigma}} - v, \frac{n_3}{\hat{\varsigma}} - w \right) \right|^2 \chi(u, v, w) dudvdw, \quad (1)$$

where $\lambda = \frac{\iota \ell}{E_{em}}$ is the emitted light wavelength, E_{em} is the energy level difference during light emission, ι is Plank's constant, ℓ is the speed of light, $\hat{\varsigma}$ is the magnification of the microscope objective, χ is an object dependent function related to the light emission properties of the fluorophore.

The function $|h_\lambda|^2$ is known as the point spread function (PSF) of the microscope. It can be measured experimentally or modeled using the physical properties of the system. A common mathematical model is given by [19]

$$h_\lambda(n_1, n_2, n_3) = \int_{\mathbb{R}^2} P(u, v) e^{j2\pi n_3 \frac{u^2+v^2}{2\lambda\alpha^2}} e^{-j2\pi \frac{n_1 u + n_2 v}{\lambda\alpha}} dudv, \quad (2)$$

which represents the inverse Fourier transform of the circular aperture of the objective $P(u, v)$. The parameter α is the focal length of the objective and is inversely related to the numerical aperture (NA) of the microscope. A detailed discussion of this model can be found in [19].

As with other imaging techniques, images are acquired under non-ideal conditions that generate undesirable distortions or aberrations. In microscopy imaging, detection of light emissions is severely limited by the properties of photo sensors. Light detection is effectively down to counting the number of photos incident upon a photodetector. At such level, the photon emission is described by a Poisson random variable which generates noise contaminated images [19, 21, 22, 23]. Background shading is also a common distortion and can be caused by several factors: non uniform illumination, inhomogeneous detector sensitivity, dirt particles in the optics, nonspecific sample staining, and even auto-fluorescence. The background shading process can be modeled as

$$b(n_1, n_2) = I(n_1, n_2)a(n_1, n_2), \quad (3)$$

where $b(n_1, n_2)$ represents the product of the illumination $I(n_1, n_2)$ and the original sample $a(n_1, n_2)$ [19, 24]. Now, if we model the detector's gain and offset we have

$$c(n_1, n_2) = g(n_1, n_2)b(n_1, n_2) + o(n_1, n_2), \quad (4)$$

where $g(n_1, n_2)$ is some gain and $o(n_1, n_2)$ some offset. Using (3) to substitute in (4) we have

$$c(n_1, n_2) = g(n_1, n_2)I(n_1, n_2)a(n_1, n_2) + o(n_1, n_2). \quad (5)$$

These defects need to be corrected before analysis is performed. Digital enhancement and restoration techniques have been extensively studied [3, 9, 10, 25, 26]. In [27] we introduced a restoration method that recovers $a(n_1, n_2)$ given the noisy observation $c(n_1, n_2)$ described in (5). The proposed method is simple and performed better than traditional ones. Essentially the method consists of a local median filter (e.g. 3×3 followed by background subtraction. For an image consisting of red (R) and green (G) channels, a restored image $\hat{a}(n_1, n_2)$ is generated by

$$\hat{a}(n_1, n_2) = \begin{cases} \varrho_{[3 \times 3]} \{c(n_1, n_2)_G\} - \hat{o}(n_1, n_2)_G \\ \varrho_{[3 \times 3]} \{c(n_1, n_2)_R\} - \hat{o}(n_1, n_2)_R \end{cases}, \quad (6)$$

where $\varrho_{[3 \times 3]} \{\cdot\}$ is a 3×3 median filter is applied to each channel and $\hat{o}(n_1, n_2)$ is the offset estimate obtained as the average of a set of representative images from the database.

To assess restoration filter performance is computed using different metrics over synthetic and real life images. Consider the synthetic images shown

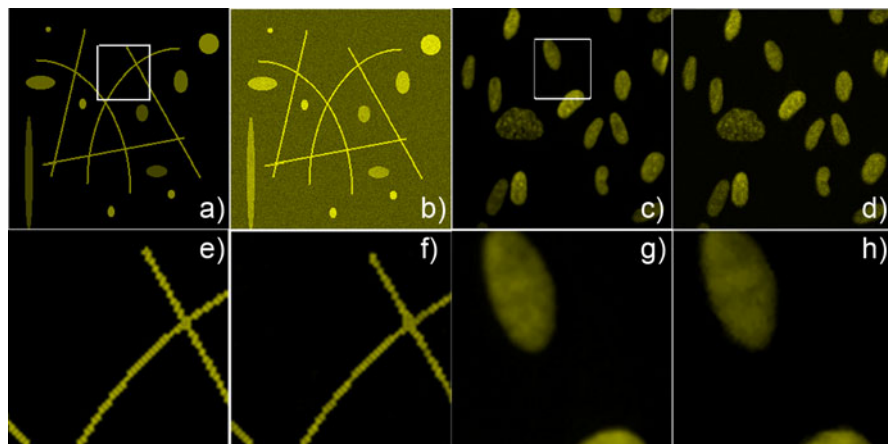


Fig. 2. In (a) is shown the “circles and lines” image, and its degraded version in (b). The “cells” image is shown in (c), and its degraded version in (d). In (e) is presented the original amplified section of circles and lines, and its restored image in (f). The original amplified cells in (g), and its restored image in (h). The results show very good performance over Poisson noise and offset, visually and quantitatively.

in Figure 2 (a) and (c). Synthetic images were degraded with a constant offset $o(n_1, n_2) = \eta$ added to each pixel. Then, images were contaminated with random Poisson noise. Figure 2 (a) is a typical case of study in image processing [9, 26] used to address the ability of a filter to recover edges, lines, and shape of the original image given a noisy observation as in (b). Likewise, Figure 2 (c) is representative of a microscopy imaging scenario, and the goal is to test the filter ability to recover texture, edges, and illumination given a degraded image as in (d). In the test of Figure 2 (b) we used $\eta = 89$ and in Figure 2 (d) $\eta = 11$. Clearly, the results over synthetic images shown in Figure 2 (f) and (h) are very good when compared to the true images in Figure 2 (e) and (g). Therefore, we proceeded to test restoration results over real life data.

For the case of real biological images, we evaluated the restoration filter with two databases. First, we used the yeast database [6] which contains 547 two channel image samples. It is a very good alternative to test the restoration methods for colocalization analysis. In Figure 3 (a) is shown an example of the content of the yeast database as well as its restored version using M33GM in Figure 3 (a).

The second database was reported in [9, 26] and consist of visible fluorescent fusion-proteins expressed in A431 cells and a fluorescent label. The erbB family of receptor tyrosine kinases includes the epidermal growth factor receptor (erbB1) and erbB3. These membrane proteins regulate cell growth and differentiation through binding of ligands to their extracellular domain,

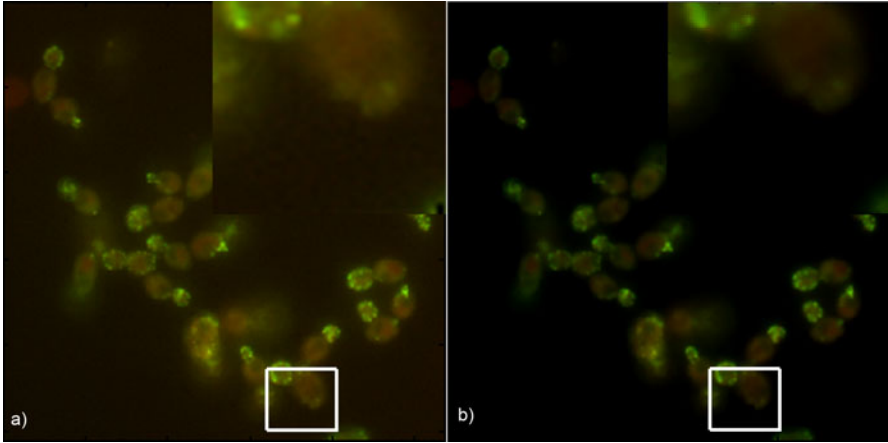


Fig. 3. In (a) is shown the noisy raw image from the Yeast database and in (b) its restored image

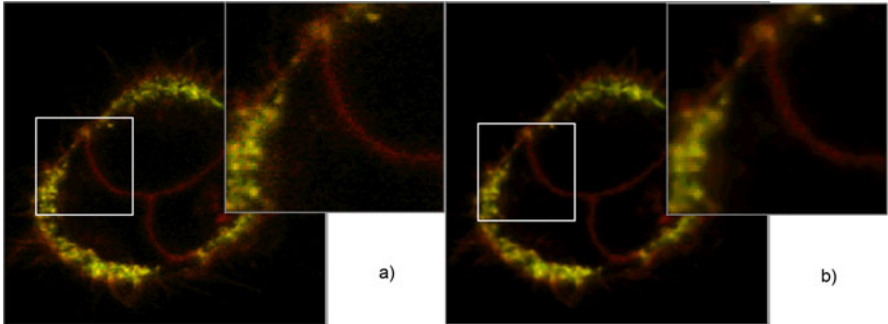


Fig. 4. Visible fluorescent fusion-proteins expressed in A431 cells. In (a) the noisy image, and in (b) the restored image

which activates the protein and initiates signalling. We identify this group of images as the A431 cells database. An example image is shown in Figure 4 (a), and the results of restoration are shown in Figure 4 (b). The filter effectively reduces Poisson noise, preserves the texture, and subtracts the offset.

3 Quantitative Colocalization Feature Extraction

The quantitative analysis of colocalization consists of computing the 2D signal spatial overlap across multiple channels. This high precision analysis allows researchers to understand the mechanisms of protein-to-protein

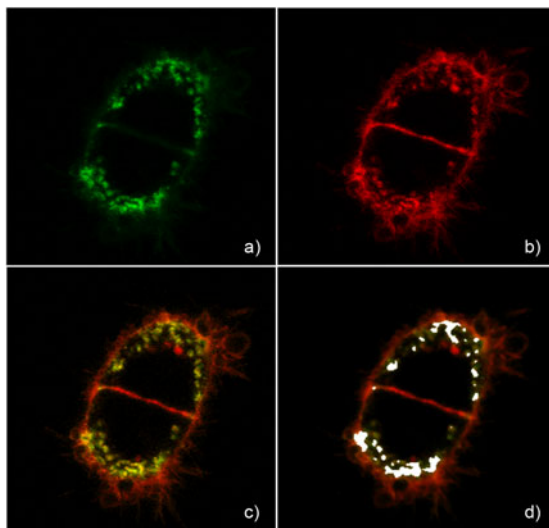


Fig. 5. An example of colocalization. In (a) is shown the green channel $G(n_1, n_2)$; in (b) is shown the red channel $R(n_1, n_2)$; in (c) are shown the superimposed channels; and in (d) are shown in white, the regions of more colocalization.

interactions. [3, 7, 8, 11, 12, 13, 19, 28]. As an example, consider the cellular image acquired by the fluorescence microscopy method; its green channel $G(n_1, n_2)$ is shown in Figure 5 (a), and its red channel $R(n_1, n_2)$ is shown in Figure 5 (b). We can study the cellular protein-to-protein interaction by observing and quantifying the superposition of Figure 5 (a) and Figure 5 (b) as in Figure 5 (c). The regions of more colocalization are highlighted in white on Figure 5 (d). Ultimately, visual assessment of colocalization is a subjective process highly dependent on the experience of the analyst. Attempts to reduce the subjectivity of this process have focused on the computation of statistical coefficients or parameters [7, 8, 10, 11, 25] and algorithms that exploit them [1, 19, 20, 28, 29]. There are specific parameters (i.e., features) used to estimate the degree of colocalization between two channels. We have identified 14 colocalization features in literature. In this paper, we will describe and use the five most popular features: Pearson's correlation coefficient, Overlap coefficient, Fraction of colocalizing regions, Mander's colocalization coefficient, and Intensity correlation coefficient.

Table 1 summarizes the properties of the coefficients that will be explained in detail next.

3.1 Pearson's Correlation Coefficient r_P

Also known as Correlation Coefficient, the Pearson's Correlation Coefficient is widely used and accepted, specially in regression applications. It provides

Table 1. Summary of features (coefficients) for quantitative colocalization

Name	Variables	Range	Ref.
Pearson's Corr. C.	r_P	$[-1, 1]$	[9, 10, 25]
Overlap C. (Mult. Meth.)	r	$[0, 1]$	[7, 12, 13]
Frac. Coloc. Reg. (O. C.)	k_1, k_2	vary	[7, 12, 13]
Mander's Coloc. C. - Gen.	M_1, M_2	$[0, 1]$	[7, 12, 13]
Intensity Corr. Quot.	ICQ	$[-0.5, 0.5]$	[7]

information about the relationship between the region of intensities and their distribution [7, 8, 9, 10, 11, 12, 13]. The Pearson's Correlation Coefficient r_P is defined as

$$r_P = \frac{\sum (R(n_1, n_2) - E[R]) (G(n_1, n_2) - E[G])}{\sqrt{\sum (R(n_1, n_2) - E[R])^2 \sum (G(n_1, n_2) - E[G])^2}} \tag{7}$$

Its domain is between $[-1, 1]$. It will be -1 if the data has negative linear relationship, a particular meaning is hard to explain. It will be 0 (or close to zero) if the data has perfect exclusion or no linear relationship and the pixels are distributed along the entire scatter diagram. If the value is 1 it means perfect correlation; the data has a linear relationship. For the case shown in Figure 5 the value is $r_P = 0.7267$.

3.2 Overlap Coefficient r (The Multiply Method)

This coefficient dictates the overlap between two channels; it shows a degree of colocalization [12]. In contrast with Pearson's, this coefficient does not return negative values, does not average any pixel intensity, and it is not sensitive to intensity variations [11, 13]. However, the authors of [7] recommend its usage under the following condition: $\frac{\sum_{n_1, n_2} G(n_1, n_2)}{\sum_{n_1, n_2} R(n_1, n_2)} \approx 1$. The Overlap coefficient is formally defined as

$$r = \frac{\sum_{n_1, n_2} G(n_1, n_2)R(n_1, n_2)}{\sqrt{\sum_{n_1, n_2} G(n_1, n_2)^2 \sum_{n_1, n_2} R(n_1, n_2)^2}}, \tag{8}$$

where its domain is between $[0, 1]$. A value of 0 means that no pixels overlap; while 1 means that all the pixels overlap. The case when $r = 0.5$ implies that 50% of the pixels overlap with each other. For the case shown in Figure 5 $r = 0.8316$. It is clearly high because of the non-zero background content.

3.3 Fraction of Colocalizing Regions k_1, k_2 (Overlap Coefficients)

These coefficients represent the differences between each channel intensities [11, 12, 13]. This two coefficients overcome the problems generated from a restriction in the overlap coefficient in (8). However, k_1, k_2 are very sensitive to

the absolute fluorescent intensity. If one channel has been treated differently from the other, such that the total intensity vary (bleaching for instance), this will affect the coefficients [7]. We can denote the coefficients as follows

$$k_1 = \frac{\sum_{n_1, n_2} G(n_1, n_2)R(n_1, n_2)}{\sum_{n_1, n_2} G(n_1, n_2)^2}, \tag{9}$$

$$k_2 = \frac{\sum_{n_1, n_2} R(n_1, n_2)G(n_1, n_2)}{\sum_{n_1, n_2} R(n_1, n_2)^2}. \tag{10}$$

The individual range of values may vary. Results can be interpreted as some indicator of each antigen’s contribution to colocalization areas.

3.4 Mander’s Colocalization Coefficients M_1, M_2

The coefficients M_i , describe i th channel’s contribution to colocalization using its intensity values. These coefficients are proportional to i th channel’s fluorescence amount, relative to the i th channel total fluorescence. Such relationship is described in [7, 11, 12, 13] as

$$M_1 = \frac{\sum_{n_1, n_2} (G(n_1, n_2)|R(n_1, n_2) > 0)}{\sum_{n_1, n_2} G(n_1, n_2)}, \tag{11}$$

$$M_2 = \frac{\sum_{n_1, n_2} (R(n_1, n_2)|G(n_1, n_2) > 0)}{\sum_{n_1, n_2} R(n_1, n_2)}, \tag{12}$$

where M_1, M_2 are within the range $[0, 1]$. The meaning can be explained with the following example: if $M_1 = 1.0$ and $M_2 = 0.3$, it means that green channel pixels colocalize with red’s, but only 30% of pixels in red channel colocalize with green’s.

3.5 Intensity Correlation Quotient ICQ

Two images vary around their respective mean if their intensities vary in synchronous [7]. Therefore the *product of the differences from the mean* (PDM) will be positive for such images, $PDM > 0$. However, if the intensities vary asynchronously, the PDM will be negative, $PDM < 0$. The PDM analyzes the relationship between intensities, and is denoted as

$$PDM(n_1, n_2) = (G(n_1, n_2) - E[G]) (R(n_1, n_2) - E[R]), \tag{13}$$

where $E[G]$ and $E[R]$ denote the expected value for each channel respectively.

The ICQ value is based on the sign of the *PDM* [7]. The ICQ is defined as the PDM positive occurrences number ζ , divided by the negative occurrences ξ . Thus, the ICQ is the quotient denoted as

$$ICQ = \left(\frac{\zeta}{\xi} \right) - 0.5. \tag{14}$$

The range of *ICQ* falls between $[-0.5, 0.5]$. The coefficient value can be interpreted as follows: if the *ICQ* ≈ 0 , means random decoloration; if $-0.5 \leq ICQ < 0$, means segregated decoloration; and if $0 < ICQ \leq 0.5$ means a dependent decoloration. The latter case is an indicator of good colocalization.

4 Colocalization Subjective Quantification with Fuzzy Logic Theory

The colocalization study performer often would like to obtain a linguistic result from a coefficient rather than just a rough number. Consider Pearson’s correlation coefficient $r_P = 0.865$ after some colocalization experiment. Then, the observer of the study would probably say, “Pearson’s correlation coefficient is *high*.” However, this is a subjective assessment that may not correspond to another observer whose definition of *high* may be a coefficient valued at 0.99. Hence, there is paucity of reference levels to map coefficient values to a human like quantification. The process of giving such subjective quantification is called *fuzzification* [18]. Prior knowledge regarding the variables to fuzzify is desired, but not mandatory. Information about the variable range and distribution is useful. In this section we address the problem of the subjective quantification by using fuzzy logic theory.

4.1 Linguistic Variables in Fuzzy Logic

A linguistic variable is a word represented by a fuzzy set [18]. For example, a linguistic variable can take an attribute like “*negative, positive, zero, high, low, more or less,*” etc. Each of this words has its own membership function.

Following the notation in [18], let us define a linguistic variable by the following quintuple

$$\begin{aligned} &(x, T(x), U, G, M) \tag{15} \\ &x : \text{name of variable} \\ &T(x) : \text{set of possible linguistic terms for } x \\ &U : \text{set of universe of discourse} \\ &G : \text{syntactic grammar that produces terms in } T(x) \\ &M : \text{semantic rules which map terms in } T(x) \text{ to} \\ &\quad \text{fuzzy sets in } U. \end{aligned}$$

Given the previous definitions, we introduce the translation of coefficients in Section 3 to linguistic variables. To distinguish between a crisp and a linguistic (fuzzy) variable we use the symbol $\hat{\cdot}$. For instance, the crisp Pearson’s correlation coefficient is denoted as r_P , while the linguistic Pearson’s correlation coefficient is defined as \hat{r}_P .

4.2 Fuzzy Pearson’s Correlation Coefficient

From [12] we know that r_P ’s universe of discourse is $[-1, 1]$. Now, we must find the possible words and their membership functions. In [12] it is found that -1 corresponds to “negative” correlation; so, let $\mu_{neg}^{\hat{r}_P}(u)$ denote this possible term for \hat{r}_P . From [7] and [12] we also can find $\mu_{lit}^{\hat{r}_P}(u)$ as a “little” correlation, with an average value close to 0; as well as “reasonable” $\mu_{rea}^{\hat{r}_P}(u)$, with an average value close to 0.346. Finally, we found “strong” correlation $\mu_{str}^{\hat{r}_P}(u)$, centered at 0.920. In [26] can be found another two possible terms. The first is “high” $\mu_{high}^{\hat{r}_P}(u)$, in the range $[0.685, 0.877]$, with an average value of 0.761. The second term is “low” $\mu_{low}^{\hat{r}_P}(u)$, in the range $[0.434, 0.615]$ with average 0.516.

At universe of discourse upper and lower limits, “s membership functions” (*SMF*) and “z membership functions” (*ZMF*) are commonly used since they consider the inclusion of such limits. In comparison, a Gaussian membership function does not fully cover the limits of the universe of discourse. Formally a “z membership function” is defined as

$$\mu(u) = \begin{cases} 1, & u \leq a \\ 1 - 2 \left(\frac{u-a}{b-a} \right)^2, & a \leq u \leq \frac{a+b}{2} \\ 2 \left(b - \frac{u}{b-a} \right)^2, & \frac{a+b}{2} \leq u \leq b \\ 0, & u \geq b \end{cases} \tag{16}$$

where parameters a and b define the start and end of the function respectively.

Similarly, an “s membership function” is formally defined as

$$\mu(u) = \begin{cases} 0, & u \leq a \\ 2 \left(\frac{u-a}{b-a} \right)^2, & a \leq u \leq \frac{a+b}{2} \\ 1 - 2 \left(b - \frac{u}{b-a} \right)^2, & \frac{a+b}{2} \leq u \leq b \\ 1, & u \geq b \end{cases} \tag{17}$$

where the parameters a and b define the start and end of the function respectively. A Gaussian-shaped membership function (*GMF*) falls in the family of the exponential and radial basis functions; and it is defined as

$$\mu(u) = e^{-\frac{(u-c)^2}{2\sigma^2}}, \tag{18}$$

where c is the centering parameter, and σ is the spread parameter. A similar function is the π -shaped membership function (*PIMF*) defined as

$$\mu(u) = \frac{1}{1 + \left| \frac{u-c}{a} \right|^{2b}}, \tag{19}$$

with parameters a and b defining the start and end of the function respectively; and c as the centering parameter. Now, following the definition in (15), we can define \widehat{r}_P as follows

$$\begin{aligned} \widehat{r}_P = & \tag{20} \\ & x : \text{Pearson's Correlation} \\ T(x) : & \{\text{negative, little, reasonable, strong, high, low}\} \\ U : & [-1, 1] \\ G(x) : & T^{i+1} = \{\text{strong}\} \vee \{\text{very } T^i\} \\ M : & \left\{ \begin{array}{l} (u, \mu_{neg}^{\widehat{r}_P}(u)), (u, \mu_{lit}^{\widehat{r}_P}(u)), \\ (u, \mu_{rea}^{\widehat{r}_P}(u)), (u, \mu_{str}^{\widehat{r}_P}(u)), \\ (u, \mu_{high}^{\widehat{r}_P}(u)), (u, \mu_{pos}^{\widehat{r}_P}(u)), \\ (u, \mu_{low}^{\widehat{r}_P}(u)) \mid u \in U \end{array} \right\} \end{aligned}$$

where the operator \vee denotes the fuzzy operator max; $\mu_{neg}^{\widehat{r}_P}(u)$ uses a function of the type defined in (16) with parameters $a = -1$, and $b = 0$; $\mu_{lit}^{\widehat{r}_P}(u)$ uses a function of the type defined in (18) with parameters $c = 0$, and $\sigma = 0.2$; $\mu_{rea}^{\widehat{r}_P}(u)$ uses a function of the type defined in (18) with parameters $c = 0.346$, $\sigma = 0.2$; $\mu_{str}^{\widehat{r}_P}(u)$ uses a function of the type defined in (18) with parameters $c = 0.92$, $\sigma = 0.2$; $\mu_{high}^{\widehat{r}_P}(u)$ uses a function of the type defined in (18) with parameters $c = 0.761$, $\sigma = 0.2$; $\mu_{low}^{\widehat{r}_P}(u)$ uses a function of the type defined in (18) with parameters $c = 0.516$, $\sigma = 0.2$; and finally $\mu_{pos}^{\widehat{r}_P}(u)$ uses a function of the type defined in (17) with parameters $a = 0.434$, and $b = 1$.

4.3 Fuzzy Overlap Coefficient

In [7, 12] we can find that the universe of discourse for the linguistic variable \widehat{r} is $[0, 1]$. From [7] we obtain $\mu_{ran}^{\widehat{r}}(u)$ as a “*random colocalization*” overlap, centered near to 0.5; as well as “*high colocalization*” $\mu_{high}^{\widehat{r}}(u)$, when it is 1. Finally, we found “*low colocalization*” $\mu_{low}^{\widehat{r}}(u)$, when the value reaches 0. So, following the notation in (15), we can define \widehat{r} as

$$\begin{aligned}
 \widehat{r} &= & (21) \\
 x &: \text{Overlap Coefficient} \\
 T(x) &: \left\{ \begin{array}{l} \text{random colocalization,} \\ \text{high colocalization,} \\ \text{low colocalization} \end{array} \right\} \\
 U &: [0, 1] \\
 G(x) &: T^{i+1} = \{\text{low}\} \vee \{\text{very } T^i\} \\
 M &: \left\{ \begin{array}{l} (u, \mu_{\widehat{r}an}(u)), \\ (u, \mu_{\widehat{r}ig}(u)), \\ (u, \mu_{\widehat{r}ow}(u)) \mid u \in U \end{array} \right\}
 \end{aligned}$$

where $\mu_{\widehat{r}an}(u)$ uses (19) for $a = 0.25$, $b = 3$, and $c = 0.5$; $\mu_{\widehat{r}ig}(u)$ uses (17) for $a = 0.5$, and $b = 1$; and finally, $\mu_{\widehat{r}ow}(u)$ uses (16) for $a = 0$, and $b = 0.5$.

4.4 Fuzzy Fraction of Colocalizing Regions (Overlap Coefficients)

For this coefficient, the most used is the k_1 coefficient since it shows the ratio between channel red versus green [7]. As specified previously this variable has a variable universe of discourse; however, the most common values lie close to one. From this finding we can define the linguistic variable \widehat{k}_1 over the range $[-\infty, \infty]$. The variable \widehat{k}_1 has one membership function centered at one and is denoted as $\mu_{\widehat{k}_1}^{val}(u)$ which is a “valid” ratio between channels. Following the notation in (15), we can define \widehat{k}_1 as

$$\begin{aligned}
 \widehat{k}_1 &= & (22) \\
 x &: \text{Ch.Red/Ch.Green ratio} \\
 T(x) &: \{\text{valid, not valid}\} \\
 U &: [-\infty, \infty] \\
 G(x) &: \overline{T}(x) = 1 - T(x) \\
 M &: \left\{ \begin{array}{l} (u, \mu_{\widehat{k}_1}^{val}(u)), \\ (u, \mu_{\widehat{k}_1}^{nov}(u)) \mid u \in U \end{array} \right\}
 \end{aligned}$$

where $\mu_{\widehat{k}_1}^{val}(u)$ uses (18), for $c = 1$, and $\sigma = \frac{1}{3}$; while $\mu_{\widehat{k}_1}^{nov}(u)$ is just $1 - \mu_{\widehat{k}_1}^{val}(u)$.

4.5 Fuzzy Mander’s Colocalization Coefficients - General

The colocalization coefficients M_1, M_2 from Mander’s are defined over the universe of discourse $[0, 1]$ as explained previously, thus the linguistic variables $\widehat{M}_1, \widehat{M}_2$ will be defined in this universe as well. In [7] is described the word “significant” $\mu_{\widehat{M}_1}^{sig}(u)$ for a value of 0.746. Also we can find the word “not much” $\mu_{\widehat{M}_1}^{nom}(u)$ for 0.155, as well as “high” $\mu_{\widehat{M}_1}^{high}(u)$ for the range

of $[0.879, 0.993]$ (with mean value 0.936). Similarly, we can find the word “strong” $\mu_{str}^{\widehat{M}_1}(u)$ defined over the range $[0.970, 0.998]$ (with mean value 0.984). We also find $\mu_{noc}^{\widehat{M}_1}(u)$ which is “not coincident” correlation for values less than 0.008. Finally the word “hard” $\mu_{har}^{\widehat{M}_1}(u)$ is found for a value of 0.590. Following the notation in (15), we define \widehat{M}_1 as

$$\begin{aligned} \widehat{M}_1 = & \tag{23} \\ & x : \text{Colocalization Coefficient for Red Channel} \\ T(x) : & \left\{ \begin{array}{l} \text{not coincident, not much, hard,} \\ \text{significant, high, strong} \end{array} \right\} \\ U : & [0, 1] \\ G(x) : & T^{i+1} = \{\text{strong}\} \vee \{\text{very } T^i\} \end{aligned}$$

$$M : \left\{ \begin{array}{l} (u, \mu_{noc}^{\widehat{M}_1}(u)), \\ (u, \mu_{nom}^{\widehat{M}_1}(u)), (u, \mu_{har}^{\widehat{M}_1}(u)), \\ (u, \mu_{sig}^{\widehat{M}_1}(u)), (u, \mu_{high}^{\widehat{M}_1}(u)), \\ (u, \mu_{str}^{\widehat{M}_1}(u)) \mid u \in U \end{array} \right\}$$

where $\mu_{noc}^{\widehat{M}_1}(u)$ uses (16) for $a = 0.008$ and $b = 0.155$; $\mu_{nom}^{\widehat{M}_1}(u)$ uses (18) for $c = 0.155$ and $\sigma = 0.2$; $\mu_{har}^{\widehat{M}_1}(u)$ uses (18) for $c = 0.590$ and $\sigma = 0.2$; $\mu_{sig}^{\widehat{M}_1}(u)$ uses (18) for $c = 0.746$ and $\sigma = 0.2$; $\mu_{high}^{\widehat{M}_1}(u)$ uses (18) for $c = 0.936$ and $\sigma = 0.2$; finally $\mu_{str}^{\widehat{M}_1}(u)$ uses (17) for $a = 0.746$ and $b = 0.984$. The linguistic variable \widehat{M}_2 is defined in the same way as \widehat{M}_1 , but the only difference is the value for x which is “Colocalization Coefficient for Green Channel.”

4.6 Fuzzy Intensity Correlation Quotient Coefficient

Since the *ICQ* coefficient [7, 30] is defined over the universe of discourse $[-0.5, 0.5]$, the linguistic variable \widehat{ICQ} will be defined in this universe as well. In [7] is described the word “segregated” $\mu_{seg}^{\widehat{ICQ}}(u)$ for a values in the range $[-0.5, 0]$. It is found as well the word “random” $\mu_{ran}^{\widehat{ICQ}}(u)$ for values near to zero. Finally the word “dependent” $\mu_{dep}^{\widehat{ICQ}}(u)$ is found for values on the range $[0, 0.5]$. Following the notation in (15), we define \widehat{ICQ} as

$$\begin{aligned}
 \widehat{ICQ} = & \tag{24} \\
 x : & \text{Intensity Correlation Quotient} \\
 T(x) : & \{\text{segregated, random, dependent}\} \\
 U : & [-0.5, 0.5] \\
 G(x) : & T^{i+1} = \{\text{segregated}\} \vee \{\text{very } T^i\} \\
 M : & \left\{ \begin{array}{l} \left(u, \mu_{seg}^{\widehat{ICQ}}(u) \right), \left(u, \mu_{ran}^{\widehat{ICQ}}(u) \right), \\ \left(u, \mu_{dep}^{\widehat{ICQ}}(u) \right) \mid u \in U \end{array} \right\}
 \end{aligned}$$

where $\mu_{seg}^{\widehat{ICQ}}(u)$ uses (16) for $a = -0.5$ and $b = 0.5$; $\mu_{ran}^{\widehat{ICQ}}(u)$ uses (18) for $c = 0$ and $\sigma = \frac{1}{8}$; and finally $\mu_{dep}^{\widehat{ICQ}}(u)$ uses (17) for $a = -0.5$ and $b = 0.5$.

4.7 Fuzzy Predicates from Linguistic Variables

Fuzzy predicates are a semantic representation of a sentence. For instance, “It is cold outside.” Here we identify “cold” as the linguistic variable. In colocalization analysis, we defined fuzzy predicates. As an example, consider the fuzzy variables and the fuzzy predicates shown in Table 2. The syntax is flexible so that the colocalization study performer can pick the combination that best represent the human-like reasoning or help the best in human-like decision making. Human-like knowledge representation can be achieved using ordinary fuzzy predicates. Such fuzzy predicates depend on linguistic variables. For the particular case of quantitative colocalization analysis through fluorescence microscopy, fuzzy membership functions were created according to the cases found in literature. In Figure 6 we present a summary of the modeled membership functions. The relevance of this research can be summarized in two parts: first, we addressed the inaccurate subjective human interpretation of colocalization, providing a formal review of quantitative colocalization coefficients; second, using fuzzy logic, we designed linguistic variables

Table 2. Definition of linguistic variables and fuzzy predicate examples

Fuzzy Predicate Definition	Fuzzy Predicate Example
“ $\widehat{r}_P[x]$ is $\widehat{r}_P[T(x)]$ ”	Pearson’s Correlation is strong
“ $\widehat{r}[x]$ is showing $\widehat{r}[T(x)]$ ”	“Overlap Coefficient is showing high colocalization”
“ $\widehat{k}_1[x]$ is $\widehat{k}_1[T(x)]$ ”	“Ch.Red/Ch.Green ratio is not valid”
“ $\widehat{M}_1[x]$ shows $\widehat{M}_1[T(x)]$ colocalization”	“Colocalization Coefficient for Red Channel is showing significant colocalization”
“ $\widehat{ICQ}[x]$ shows $\widehat{ICQ}[T(x)]$ staining”	“Intensity Correlation Quotient Coefficient shows random staining”

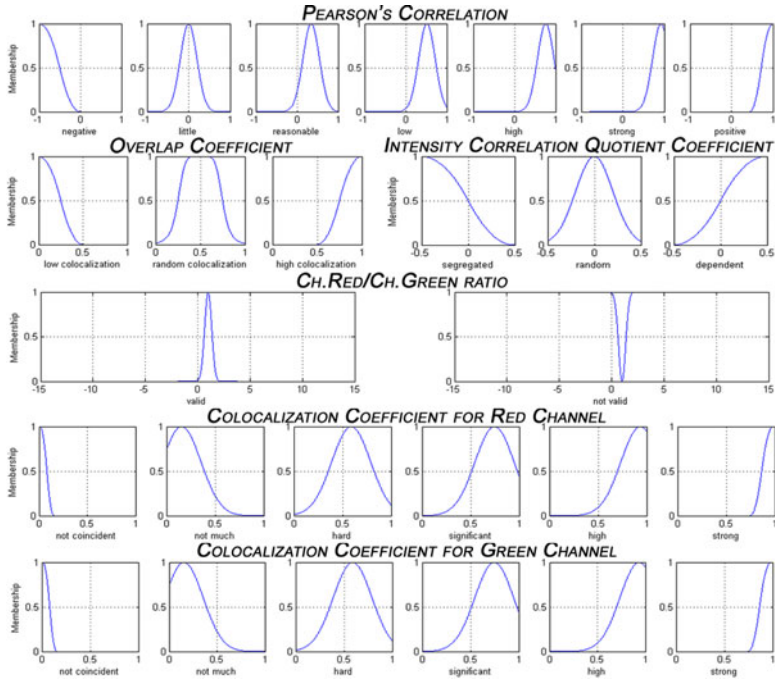


Fig. 6. Graphical representation of the linguistic variables, and its associated membership functions modeled from a literature review

based on literature review and provided an accurate human-like colocalization quantification.

4.8 Experimental Results and Discussion

We performed several experiments in different databases such as Yeast and A431 cells. The experiments consist of computing the previously described coefficients and using them as inputs to a fuzzy system. The fuzzy system consists of five linguistic variables that take its associated coefficient numerical value and produce a set of human like colocalization quantification. The linguistic variables x are constructed in the form of (15). The output of each linguistic variable is computed by taking the maximum valued membership function associated to that linguistic variable. Then, we use the linguistic term $T(x)$ associated to the maximum valued membership function to construct a human-like sentence (fuzzy predicate). This procedure can be formalized in four steps:

Step 1. Image restoration. Channels $R(n_1, n_2)$ and $G(n_1, n_2)$ are restored using the filter discussed in Section 2 to obtain $\hat{c}(n_1, n_2)$.

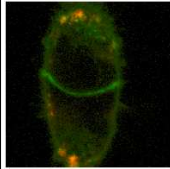
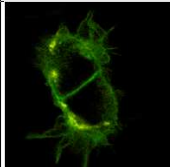
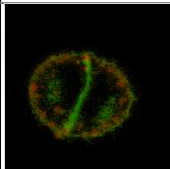
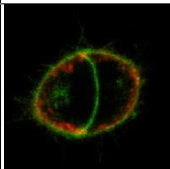
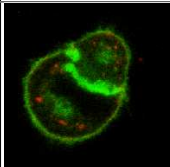
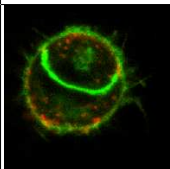
Table 3. Fuzzy predicates obtained from samples in Figures 1, 3, 4, and 5

Figure #	Fuzzy Predicates.
Figure 1	Pearson's Correlation is strong. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.
Figure 3	Pearson's Correlation is strong. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.
Figure 4.	Pearson's Correlation is strong. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.
Figure 5.	Pearson's Correlation is high. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.

Step 2. Feature extraction. Features proposed in Section 3 are extracted from $\widehat{c}(n_1, n_2)$ to form a row vector $F = [r_P, r, [k_1, k_2], [M_1, M_2], ICQ]^T$.

Step 3. Fuzzification. Features vector F is fuzzified according to the linguistic variable associated to each feature. Follows to construct a row vector $\widehat{F} = [\widehat{r}_P, \widehat{r}, [\widehat{k}_1, \widehat{k}_2], [\widehat{M}_1, \widehat{M}_2], \widehat{ICQ}]^T$.

Table 4. Fuzzy predicates obtained from the A341 cells database

	<p>Pearson's Correlation is high. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.</p>
	<p>Pearson's Correlation is strong. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.</p>
	<p>Pearson's Correlation is high. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.</p>
	<p>Pearson's Correlation is high. Overlap Coefficient is showing high colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.</p>
	<p>Pearson's Correlation is low. Overlap Coefficient is showing random colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.</p>
	<p>Pearson's Correlation is low. Overlap Coefficient is showing random colocalization. Ch.Red/Ch.Green ratio is not valid. Colocalization Coefficient for Red Channel shows strong colocalization. Colocalization Coefficient for Green Channel shows strong colocalization. Intensity Correlation Quotient Coefficient shows segregated staining.</p>

Step 4. Fuzzy predicates. For each element of the fuzzyfied features vector \hat{F} the linguistic term $T(x)$ is extracted according to the semantic rules M associated to each linguistic variable. Then, fuzzy predicates are constructed according to the skeleton shown in Table 2.

Let us consider the case of the samples shown in Figures 1, 3, 4, and 5. The fuzzy predicates obtained from these cases are presented in Table 3.

Clearly, fuzzy predicates alleviate the need for quantitative results in natural language. Consider the experiments with the A341 cells database shown in Table 4. These cases are also successful experiments in the construction of colocalization fuzzy predicates.

The results shown represent a paradigm change in the quantitative colocalization analysis since current problems are solved visually and by human analysis. In this paper is proposed the usage of features and fuzzy linguistic variables to provide an invariant human-independent result of colocalization. Results provide a descriptive set of colocalization coefficient interpretations that could lead researchers to a uniform and consistent interpretation colocalization.

5 Conclusion

In this document we addressed the subjectivity problem in quantitative colocalization analysis through fluorescence microscopy. We presented the most common quantitative colocalization coefficients reported in literature. These coefficients are utilized to model a fuzzy system. Linguistic variables are modeled using fuzzy logic theory from the reported colocalization coefficients. Fuzzy membership functions were constructed based on literature review. The proposed fuzzy model provides natural language quantitative colocalization results through the evaluation of membership functions associated with linguistic variables. Our model demonstrated successful assessment of quantitative colocalization with natural language results. Results over the Yeast and A431 cells database show consistency in the computations and in the quantitative colocalization assessment. The proposed model can be utilized in the field of biological sciences where confocal fluorescence microscopy techniques are used in the analysis of protein-to-protein interactions. This will alleviate the paucity of formal quantitative colocalization analysis using novel computational intelligence methods. The proposed model provides a human-like ensemble of colocalization coefficient interpretations that could lead researchers to a uniform and consistent interpretation colocalization.

Acknowledgement

The authors want to thank to the National Council of Science and Technology, CONACyT Mexico, for partially supporting this research under grant 193324. The authors acknowledge Dr. Filip Rooms for providing the A431 cell images.

References

1. Savio, E., Goldhaber, J.I., Bridge, J.H.B., Sachse, F.B.: A framework for analyzing confocal images of transversal tubules in cardiomyocytes. In: Sachse, F.B., Seemann, G. (eds.) FIHM 2007. LNCS, vol. 4466, pp. 110–119. Springer, Heidelberg (2007)

2. Matula, P., Kozubek, M., Matula, P.: Applications of image registration in human genome research. In: ECCV Workshops CVAMIA and MMBIA, pp. 376–384 (2004)
3. Lukas Landmann, P.M.: Colocalization analysis yields superior results after image restoration. *Microsc. Res. Tech.* 64, 103–112 (2004)
4. Huh, W., Falvo, J.V., Gerke, L.C., Carroll, A.S., Howson, R.W., Weissman, J.S., O’Shea, E.K.: Global analysis of protein localization in budding yeast. *Nature* 425, 686–691 (2003)
5. Ghaemmaghani, S., Huh, W., Bower, K., Howson, R.W., Belle, A., Dephoure, N., O’Shea, E.K., Weissman, J.S.: Global analysis of protein expression in yeast. *Nature* 425, 737–741 (2003)
6. Howson, R., Huh, W.K., Ghaemmaghani, S., Falvo, J.V., Bower, K., Belle, A., Dephoure, N., Wykoff, D.D., Weissman, J.S., O’Shea, E.K.: Construction, verification and experimental use of two epitope-tagged collections of budding yeast strains. *Comp. Funct. Genomics* 6, 2–16 (2005)
7. McMaster Biophotonics Facility (30/11/2006). Colocalization, p. 20 (November 1, 2008)
http://www.macbiophotonics.ca/PDF/MBF_colocalisation.pdf
8. Indiana Center for Biological Microscopy (10/05/2007); Analysis of colocalization using metamorph, p. 7 (November/01, 2008),
<http://www.nephrology.iupui.edu/imaging/tutorials/AnalysisofColocalizationusingMetamorph.pdf>
9. Rooms, F.: Nonlinear Techniques in Image Restoration applied to Confocal Microscopy. In: Faculty of Engineering, Department TELIN (June 10, 2005)
10. Adler, J., Parmryd, I.: Letter to the Editor. *Microsc.* 227, 83 (2007)
11. MediaCybernetics (22/03/2002), Application note 1: Colocalization of fluorescence probes, pp. 20 (November/01, 2008),
<http://www.spectraservices.com/Merchant2/pdf/AppInfo-CoLocFluorProbes.pdf>
12. Zinchuk, V., Zinchuk, O., Okada, T.: Quantitative colocalization analysis of multicolor confocal immunofluorescence microscopy images: pushing pixels to explore biological phenomena. *Acta Histochem. Cytochem.* 40, 101–111 (2007)
13. CoLocalization Research Software (02/11/2008), CoLocalizer pro user guide. ver. 2.5., pp. 67 (November/01, 2008), <http://homepage.mac.com/colocalizerpro/Resources/CoLocalizerProUserGuide.pdf>
14. Ding, Y.: A New Obfuscation Scheme in Constructing Fuzzy Predicates. In: WRI World Congress on Software Engineering, WCSE 2009, May 19–21, vol. 4, pp. 379–382 (2009)
15. Drobics, M., Adlassnig, K.-P.: Extending the Medical Concept of Reference Intervals Using Fuzzy Predicates. In: International Conference on Computational Intelligence for Modelling Control and Automation, December 10–12, pp. 603–608 (2008)
16. Prade, H., Serrurier, M.: Getting adaptability or expressivity in inductive logic programming by using fuzzy predicates. In: Proceedings of 2004 IEEE International Conference on Fuzzy Systems, July 25–29, vol. 1, pp. 73–77 (2004)
17. Bosc, P., Pivert, O.: An approach for a hierarchical aggregation of fuzzy predicates. In: Second IEEE International Conference on Fuzzy Systems, vol. 2, pp. 1231–1236 (1993)
18. Tsoukalas, L.H., Uhrig, R.E.: *Fuzzy and Neural Approaches in Engineering*, p. 587. Wiley, New York (1997)

19. Wu, Q., Merchant, F., Castleman, K.R.: *Microscope Image Processing*, p. 548. Elsevier/Academic Press, Amsterdam (2008)
20. Agnati, L.F., Fuxe, K., Torvinen, M., Genedani, S., Franco, R., Watson, S., Nussdorfer, G.G., Leo, G., Guidolin, D.: New Methods to Evaluate Colocalization of Fluorophores in Immunocytochemical Preparations as Exemplified by a Study on A2A and D2 Receptors in Chinese Hamster Ovary Cells. *J. Histochem. Cytochem.* 53, 941–953 (2005)
21. Stark, H., Woods, J.W.: *Probability and Random Processes with Applications to Signal Processing*, 3rd edn. Prentice Hall, Englewood Cliffs (2001)
22. Krishnan, V.: *Probability and Random Processes*. John Wiley, Chichester (2006)
23. Gubner, J.A.: *Probability and Random Processes for Electrical and Computer Engineers*. Cambridge University Press, Cambridge (2006)
24. Gonzalez, R.C., Woods, R.E.: *Digital Image Processing*, 2nd edn. Prentice Hall, Englewood Cliffs (2002)
25. Adler, J., Bergholm, F., Pagakis, S., Parmryd, I.: Noise and Colocalization in Fluorescence Microscopy: Solving a Problem. *Microsc. and Analysis* 22 (2008)
26. Rooms, F., Philips, W., Lidke, D.S.: Simultaneous degradation estimation and restoration of confocal images and performance evaluation by colocalization analysis. *J. Microscopy* 218(1), 22–36 (2005)
27. Rivas-Perea, P., Rosiles, J.G., Qian, W.: Self organizing maps for class discovery in the quantitative colocalization analysis feature space. In: *Proc. of IEEE International Joint Conference on Neural Networks*, pp. 2678–2684 (2009)
28. Bolte, S., Cordelieres, F.P.: A guided tour into subcellular colocalization analysis in light microscopy. *Journal of Microscopy* 224(3), 213–232
29. Costes, S.V., Daelemans, D., Cho, E.H., Dobbin, Z., Pavlakis, G., Lockett, S.: Automatic and Quantitative Measurement of Protein-Protein Colocalization in Live Cells. *Journal of Biophysics* 86(6), 3993–4003
30. Wu, Y., Eghbali, M., Ou, J., Li, M., Toro, L., Stefani, E.: Quantitative Determination of Spatial Protein-protein Proximity in Fluorescence Confocal Microscopy. *Biological Physics* (2009)

Iterated Local Search Algorithm for the Linear Ordering Problem with Cumulative Costs (LOPCC)

Jesús David Terán Villanueva^{1,*}, Héctor Joaquín Fraire Huacuja²,
Rodolfo Pazos Rangel², Juan Martín Carpio Valadez¹,
Héctor José Puga Soberanes¹, and Juan Javier González Barbosa²

¹ Instituto Tecnológico de León (ITL)

² Instituto Tecnológico de Ciudad Madero (ITCM)

Abstract. In this article we approach the linear ordering problem with cumulative costs (LOPCC). Bertacco developed this problem [2] and propose two exact algorithms, due to the complexity of the problem Duarte propose a Tabu algorithm for LOPCC [3] and until now that algorithm is the state of the art. In this ongoing research we propose a set of iterated local search algorithms (ILS) to solve the LOPCC. The experimental evidence shows that the performance of the iterated local search algorithms evaluated have similar quality to the best solution reported and get better efficiency than the reference solution. Also with the local search algorithms we improve the best known solution for 32 instances. Now we are working in developing new algorithms with population metaheuristics.

1 Introduction

In wireless communication systems it is required that the cell phones communicate constantly with some base station. In order for the base station to distinguish between different signals from the cell phones, the Universal Mobile Telecommunication Standard (UMTS) adopted the technique of Code Division for Multiple Access (CDMA) in this technique each cell phone have a different code. Due to the induced distortion because of radio waves propagation, each cell phone interferes with each other, so in order to preserve the connectivity it is necessary to keep the interference below an acceptable level.

In UMTS exists a technique to reduce the interference called, Successive Interference Cancellation (SIC). In this technique each cell phone emits a

* Authors thank the support received from Consejo Nacional de Ciencia y Tecnología (CONACYT) and Consejo Tamulipeco de Ciencia y Tecnología (COTACYT), for the research reported in this paper. Also we thank to Manuel Laguna and Abraham Duarte for their support with the LOPCC source code and the set of instances.

signal, on a sequential order previously established, and that signal is detected by the base station. After each detection, the interference is eliminated for the other cell phones, improving the detection for the other users.

One of the main problems of the SIC technique is the order of detection for each cell phone. Usually the users are ordered given their emission level in decrease order, however we could have a better performance considering the interference between users. A second problem is the selection of the levels of energy α_i at which the cell phone i have to transmit its data. A cell phone that transmit its data using a high level of energy to transmit its data would be easily detected and a cell phone that uses a low level of energy would have a longer duration of its battery. Besides, there exists physical an regulation restrictions that limit the energy of transmission of any cell phone to a value U .

Then one faces the problem of finding the order of detection for the users on a SIC technique and find the minimal power levels of transmission for each cell phone, that assures a proper reception for all the users. This problem is called Joint Power Control and Receiver Optimization (JOPCO) and was formulated by Benvenuto [1]. Using this problem Bertacco formulates the Linear Ordering Problem With Cumulative Costs (LOPCC) [2].

2 Linear Ordering Problem with Cumulative Costs(LOPCC)

Given a complete digraph $G = (V, A)$ with no negative weighted nodes c_{uv} , the Linear Ordering Problem With cumulative Costs (LOPCC) consists in finding a permutation $P = \{p_1, p_2, \dots, p_{n-1}, p_n\}$ of nodes, which minimize the next objective function.

$$LOPCC(P) = \sum_{i=n}^1 \alpha_{p_i}$$

Where:

$$\alpha_{p_i} = d_{p_i} + \sum_{j=i+1}^n (c_{p_i p_j} \alpha_{p_j})$$

Under this context α_{p_i} represents the energy level of the signal coming from the cell phone p_i .

3 Related Work

3.1 Linear Ordering Problem (LOP)

The linear ordering problem (LOP) is close related with the linear ordering problem with cumulative costs, in this section we review some recent works related to LOP.

Laguna in [4] propose a Tabu search metaheuristic for the linear ordering problem. This approach uses a insertion neighborhood along with a local search based on a *first* strategy. Laguna implements intensification and diversification besides the ones inherent to tabu search. The additional intensification is produced with a pathrelinking algorithm that is applied to a set of elite solutions. The additional diversification uses long term memory to insert certain elements into a complementary position related to the previously selected positions. Laguna propose a balance in the intensification and diversification methods so that an algorithm should not waste time in methods that does not produce results, this observation could be applied to any algorithm that have more than one method.

Schiavinotto in [6] approach the linear ordering problem (LOP) using the *insertion* neighborhood. In his work he propose *insertion* movements by using consecutive *swap* movements. Schiavinotto uses two metaheuristics algorithms to solve the LOP: an Iterated Local Search (ILS) and a Memetic Algorithm (MA). The local search to both methods is LS_F which is a method that stops some where between *first* and *best* improvements. The memetic algorithm does not uses mutation due to its inherent diversity. His reported experiments shows that the Memetic Algorithm have a better performance, in time and deviation from the optimum, than the Iterated Local Search (ILS). This results were validated using a Wilcoxon test with $\alpha = 0.01$. However there were not enough evidence that the MA had better performance than ILS for instances of size 250. A very important element of Schiavinotto's work is the economical way to recalculate the objective function value, and that is possible because of the order in which the neighborhood is visited.

3.2 Linear Ordering Problem with Cumulative Costs (LOPCC)

Now we review the works directly related to the Linear Ordering Problem with Cumulative Costs (LOPCC). Bertacco in [2] propose a problem formulation and two exact algorithms to solve LOPCC. One of them is based on Mixed Integer-Linear programming, while the other is an enumerative ad-hoc algorithm that produce the best results.

Righini in [5] propose a polynomially computable lower bound algorithm that can be embedded in a branch-and-bound algorithm, also he presents a constructive heuristic based on the truncation of the resulting branch-and-bound algorithm. Righini reports the branch-and-bound algorithm with lower bounds on instances of 16 elements, as a result he shows an average time of 0.96 and a error deviation from the optimal of 52%.

Duarte in [3] propose a tabu search algorithm to solve the LOPCC, similar to the algorithm proposed by Laguna on [4]. In this work it is an initial construction based on a greedy algorithm and a *first* local search on an *insertion* neighborhood. Duarte reports results on instances of size 150.

The intensification phase uses a candidate list based on a *score* calculated as follows, view Eq. (1). In this way it is given priority to those elements

that have a lesser *score* and doing that it is expected that the intensification produces lower values of the objective function.

$$score(i) = \sum_j c_{ij}d_j + \sum_j c_{ji}d_i. \quad (1)$$

The diversification method uses a long term memory, which allows the algorithm to select an element to be inserted in other position. The probability of selecting an element is inversely proportional to the quantity of times that the item was selected in the phase of intensification. The reported experiments were done with 25 UMTS instances of each scenario; where the scenarios are instances synchronous and asynchronous, with and without scrambling. Also there were 49 LOLIB instances and 75 Random instances with a uniform distribution between 0 and 100 of different sizes, 25 instances of size 35, 25 of size 100, and 25 of size 150. A comparative study was done against the enumerative method proposed by Bertacco [2], a Random Greedy method proposed by Benvenuto [1] and an adaptation of the Tabu search proposed by Laguna [4].

We begin our research with a preliminary evaluation of the algorithm proposed by Duarte to identify possible improving areas. Then an Iterated Local Search approach was used to evaluate strategies to improve the reported results.

4 Solution Proposal

4.1 Iterated Local Search (ILS)

In this section we describe the Iterated Local Search algorithms proposed. The main methods used on the algorithms are the initial solution construction, the neighborhood, the local search and the solution perturbation.

The initial solution construction is made by a greedy algorithm *Greedy-HeuristicConstruction()*. It generates 10 solutions which are built in inverse order. For each one, the last element of the permutation is randomly selected and from there the elements are selected using a roulette that gives preference to those elements that produces the lower cost if they are placed at the next position of the permutation. Once the new element is placed somewhere, the algorithm verifies if that element could produce a better solution if it is inserted on another position that was already taken by other selected element. This construction implies checking the costs of the objective function for each element that could be on the permutation at the next position. This is an expensive process but permits that the algorithm produces good quality solutions.

Other initial constructions used consists of selecting in a deterministic way an initial solution. This construction is based on the *score* proposed by Duarte, where the elements with a lower *score* would be placed on the last positions of the permutation, this algorithm is named as ILS_C.

All the ILS algorithms uses the insertion neighborhood. This neighborhood takes an element and it is inserted into another position of the permutation, preferably on a position that produces a better objective function value.

The local search tries to place every element of the permutation somewhere else that gives a better objective function value. Given an element the position where it must be inserted is determined reviewing the neighborhood from the element position i to the left and continuing from element position i to the right, making swap moves.

In the solution perturbation phase it is selected a random element and is placed in some other position. It is intended that the new position improves the actual solution, however if that was not possible then it is selected the best none-improving position.

The local search method is used to move the actual solution to an optimal local solution. It is applied after the initial solution construction and the solution perturbations. The algorithm that uses the *GreedyHeuristicConstruction()* and a random perturbation perturbation is named ILS_B. The general structure of the ILS algorithms is the next, see fig.(1):

```

p = GreedyHeuristicConstruction()
p = LocalSearch(p)
Pbetter = p
While globalIterations without improve < maxIterations
{
    p = Perturbation(p)
    p = LocalSearch(p)
    if (p < Pbetter)
    {
        Pbetter = p
    }
    else
    {
        if (p > Pbetter * β)
        {
            p = Pbetter
        }
    }
}

```

Where:

β is a numerical value between 1 and 2.

Fig. 1. General Structure of the ILS algorithm

5 Experimental Results

All the algorithms were implemented in C and all the experiments were made on a computer with a Xenon 3.06 GHz processor and 4 GB of RAM. The instances used on the experiments were the same as those used by Duarte in [3].

Because the reference algorithm is the one proposed by Duarte and it uses one seed for the random numbers and a time limit of 60 seconds, we did our experiments on the ILS algorithms with the same parameters.

The tables 1, 2, 3 and 4 show the results obtained by the UMTS instances for which are known the optimal values. This results shows that the ILS_B algorithm is slightly worse than the TS_Duarte algorithm in quality. But in efficiency the ILS_B is better than the reference algorithm. The ILS_B algorithm outperforms the average efficiency of the TS_Duarte by 73%.

Table 1. 25 UMTS instances synchronous and without scramble of size 16

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	6.24	6.24	6.28
Avg. Error	0%	0%	0.76%
Standard Dev.	0%	0%	2.13%
Number of opt.	25	25	18
CPU Seconds	0.06	0.01	0.01

Table 2. 25 UMTS instances synchronous and with scramble of size 16

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	14	14.04	14.02
Avg. Error	0.03%	0.34%	0.12%
Standard Dev.	0.2%	1.02%	0.35%
Number of Opt.	24	22	22
CPU seconds	0.03	0.01	0.01

Table 3. 25 UMTS instances asynchronous and without scramble of size 16

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	12.43	12.43	12.79
Avg. Error	0.03%	0.39%	2.57%
Standard Dev.	0.1%	1.87%	5.36%
Number of Opt.	24	21	16
CPU seconds	0.03	0.01	0.01

Table 4. 25 UMTS instances asynchronous and with scramble of size 16

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	19.05	19.31	19.20
Avg. Error	0.06%	0.82%	1.46%
Standard Dev.	0.3%	2.14%	3.93%
Number of Opt.	24	19	19
CPU seconds	0.03	0.01	0.01

The table 5 shows the experimental results on the LOLIB instances for which no optimal solutions are known. This table is shows that the ILS_B algorithm finds new best known solutions for 29 instances and outperforms the average Error of the TS_Duarte by 38.42% and the efficiency by 48.92%.

Table 5. 49 LOLIB instances of sizes 44(28), 50(2), 56(11) y 60(3)

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	19650.88	19949.41	373356.87
Avg. Error	56.38%	17.96%	139.75%
Standard Dev.	262.03%	59.76%	399.46%
Best known	30	29	19
CPU seconds	2.31	1.18	1.57

The tables 6, 7 and 8 show the results of the random instances for 35, 100 and 150 elements for which no optimal solutions are known. The ILS algorithms were able to find new best known solutions for 20 random instances, but at the same time the average error grows with the size of the instances.

Table 6. 25 random instances of size 35

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	0.34132	0.34444	0.37812
Avg. Error	0.36%	1.76%	4.8%
Standard Dev.	0.99%	5.84%	7.79%
Best known	20	15	11
CPU seconds	0.39	0.32	0.35

On the random instances of size 35 the ILS_B algorithm finds 3 new best known solutions, while the ILS_C algorithm finds 4 new best known solutions. The average efficiency of the ILS algorithms improves by 14% the average efficiency of the reference algorithm.

On the random instances of size 100 the ILS_B algorithm finds 7 new best known solutions and the ILS_C algorithm finds 1 new best known solutions.

Table 7. 25 random instances of size 100

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	1197.2316	1321.82744	1676.55224
Avg. Error	2.667%	9.528%	25.683%
Standard Dev.	7.093%	9.064%	33.559%
Best known	17	7	1
CPU seconds	30.75	32.78	26.56

Table 8. 25 random instances of size 150

	TS_Duarte	ILS_B	ILS_C
Objective Fun.	2250558.454	3020078.359	490485.4494
Avg. Error	3.784%	14.638%	44.956%
Standard Dev.	7.737%	15.638%	30.307%
Best known	18	6	1
CPU seconds	180.43	73.25	81.84

Table 9. The best known solutions updated for the LOLIB instances

Instance	Value	Instance	Value	Instance	Value
be75eec	5.09	t65n11xx	2.09	t75i11xx	4455.05
be75np	16543623.61	t65w11xx	19.26	t75k11xx	1.32
be75oi	3.07	t69r11xx	14.04	t75n11xx	9.9
be75tot	297138.99	t70b11xx	93.67	t75u11xxa	0.068427
stabu70	14.86	t70d11xx	79.74	tiw56n54	2.64
stabu74	14.03	t70d11xxb	4.44	tiw56n58	3.62
stabu75	9.42	t70f11xx	1.27	tiw56n62	3.02
t59b11xx	76262.11	t70i11xx	121146.03	tiw56n66	2.69
t59d11xx	4086.33	t70k11xx	0.5	tiw56n67	1.88
t59f11xx	24.14	t70l11xx	798.92	tiw56n72	1.57
t59i11xx	621682.25	t70n11xx	0.05	tiw56r54	2.63
t59n11xx	1618.9	t70u11xx	35490330260	tiw56r58	3.6
t65b11xx	28230.62	t70w11xx	0.04	tiw56r66	2.19
t65d11xx	3898.61	t70x11xx	0.23	tiw56r67	1.54
t65f11xx	1.25	t74d11xx	4.76	tiw56r72	1.35
t65i11xx	826127.4	t75d11xx	5.06		
t65l11xx	2657.72	t75e11xx	2063.41		

Table 10. The best known solutions updated for the aleatory instances

Instance	Value	Instance	Value	Instance	Value
t1d35.1	0.931	t1d100.1	263.066	t1d150.1	9652.803
t1d35.2	0.167	t1d100.2	297.27	t1d150.2	216751.9
t1d35.3	0.154	t1d100.3	1431.21	t1d150.3	772872.52
t1d35.4	0.196	t1d100.4	8864.08	t1d150.4	88416.64
t1d35.5	1.394	t1d100.5	172.11	t1d150.5	87690.014
t1d35.6	0.2	t1d100.6	479.3	t1d150.6	58579.73
t1d35.7	0.12	t1d100.7	6638.95	t1d150.7	172627.38
t1d35.8	0.226	t1d100.8	3095.19	t1d150.8	353334.04
t1d35.9	0.436	t1d100.9	75.193	t1d150.9	403646.64
t1d35.10	0.205	t1d100.10	174.44	t1d150.10	146736.66
t1d35.11	0.369	t1d100.11	256.74	t1d150.11	13900.04
t1d35.12	0.235	t1d100.12	246.92	t1d150.12	82560.127
t1d35.13	0.196	t1d100.13	629.248	t1d150.13	102720.712
t1d35.14	0.138	t1d100.14	279.38	t1d150.14	95393.684
t1d35.15	1.376	t1d100.15	458.02	t1d150.15	352880.29
t1d35.16	0.287	t1d100.16	771.94	t1d150.16	24497326.54
t1d35.17	0.2	t1d100.17	783.01	t1d150.17	98564.17
t1d35.18	0.384	t1d100.18	704.937	t1d150.18	767813.83
t1d35.19	0.236	t1d100.19	249.29	t1d150.19	96627.16
t1d35.20	0.068	t1d100.20	272.82	t1d150.20	2279010.02
t1d35.21	0.202	t1d100.21	232.05	t1d150.21	48116.464
t1d35.22	0.177	t1d100.22	176.64	t1d150.22	866394.89
t1d35.23	0.346	t1d100.23	1724.49	t1d150.23	23972543.85
t1d35.24	0.132	t1d100.24	535.28	t1d150.24	119426.11
t1d35.25	0.143	t1d100.25	698.71	t1d150.25	462316.51

The average efficiency of the ILS algorithms improves by 4% the average efficiency of the reference algorithm.

On the instances of size 150 even though the TS_Duarte is fixed to 60 seconds it takes 180 seconds to solve the instances, that is one of the reasons for its better quality results. Nevertheless the ILS_B algorithm finds 6 new best known solutions and the ILS_C algorithm finds 1 new best known solutions. The average efficiency of the ILS algorithms for random instances of size 150 improves by 57% the TS_Duarte algorithm.

6 Conclusions

This ongoing investigation, studies the lineal ordering problem with cumulative costs and uses Iterative Local Search (ILS) algorithms. Besides that the ILS algorithms proposed in this work are still under development and need tuning, they tend to work as well for small instances as the state of

art algorithms and get even better efficiency. Meanwhile for bigger instances, their quality is reduced but at the same time some algorithms find new best known solutions than the ones reported by Duarte. We show an update to the best known solutions reported by Duarte in [3], we remark in bold the new best known solutions found in this work, see tables 9 and 10.

References

1. Benvenuto, N., Carnevale, G., Tomasin, S.: Optimum Power Control and Ordering in SIC Receivers for Uplink CDMA Systems. In: *IEEE-ICC 2005* (2005)
2. Bertacco, L., Brunetta, L., Fischetti, M.: The Linear Ordering Problem With Cumulative Costs. *Eur. J. Oper. Res.* 189(3), 1345–1357 (2008)
3. Duarte, A., Laguna, M., Martí, R.: Tabu Search for the Linear Ordering Problem with Cumulative Costs. Springer Science+Business Media, Heidelberg (2009)
4. Laguna, M., Martí, R., Campos, V.: Intensification and Diversification with Elite Tabu Search Solutions for the Linear Ordering Problem (1998)
5. Righini, G.: A branch-and-bound Algorithm for the Linear Ordering Problem with Cumulative Costs. *European Journal of Operational Research* (2008)
6. Schiavinotto, T., Stützle, T.: *The Linear Ordering Problem: Instances, Search Space Analysis and Algorithms* (2004)

An Observer for the Type-1 Fuzzy Control of a Servomechanism with Backlash Using Only Motor Measurements

Nohe R. Cazarez-Castro¹, Luis T. Aguilar², and Oscar Castillo³

¹ Facultad de Ciencias Químicas e Ingeniería, Universidad Autónoma de Baja California, Tijuana, BC, 22414, Mexico

Ph.: +52 664 6827681

nohe@ieee.org

<http://www.nohe.mx>

² Instituto Politécnico Nacional, Mexico

luis.aguilar@ieee.org

³ Instituto Tecnológico de Tijuana, Mexico

ocastillo@hafsamx.org

Abstract. This paper addresses the analysis and design of an observer in order to ease the difficulty of working with various variables for the design of type-1 fuzzy controllers. In this paper Fuzzy Lyapunov Synthesis, based on the observed system, is extended to the design of type-1 fuzzy logic controllers for nonsmooth mechanical systems. The output regulation problem for a servomechanism with nonlinear backlash is proposed as a case of study. The problem at hand is to design a feedback controller so as to obtain the closed-loop system in which all trajectories are bounded and the load of the driver is regulated to a desired position while also attenuating the influence of external disturbances. The servomotor position is the only measurement available for feedback; the proposed extension is far from trivial because of the nonminimum phase properties of the system.

1 Introduction

A major problem in control engineering is a robust feedback design that asymptotically stabilizes a plant while also attenuating the influence of parameter variations and external disturbances. In the last decade, this problem was heavily studied and considerable research efforts have resulted in the development of systematic design methodologies for nonlinear feedback systems. A survey of these methods, fundamental in this respect is given in [13]. A novel approach to the modeling and identification of elastic robot joints with hysteresis and backlash is reported in [23].

The design of fuzzy logic systems is a heavy task that fuzzy logic practitioners face every time that they try to use fuzzy logic as a solution to some problem, the design of fuzzy logic systems implies at least two stages: design of the rule-base and design of the membership functions (MFs).

There has been some publications in the design of type-1 fuzzy logic systems (T1FLS), for example [11] presents Genetic Algorithms (GA) as an optimization method of control parameters, the authors optimize parameters of the closed-loop system but not of the T1FLS. In [16] GAs are used to optimize all the parameters of a T1FLS. In [18] a hybridizing of Neural Networks and GAs is presented to optimize a T1FLSs. A Hierarchical Genetic Algorithms is proposed in [4] to optimize rules and MFs parameters of a T1FLS.

The use of fuzzy systems in the control of mechanisms with backlash is a topic that has been treated in recent years, for example in [10] an energy-compensated fuzzy swing-up and balance control is investigated for the planetary-gear-type inverted pendulum. In the present paper the output regulation problem is studied for an electrical actuator consisting of a motor part driven by a DC motor and a reducer part (load) operating under uncertainty conditions in the presence of nonlinear backlash effects. The objective is to drive the load to a desired position while providing the roundedness of the system motion and attenuating external disturbances. Because of practical requirements (see e.g., [15]), the motor's angular position is assumed to be the only information available for feedback.

This problem was first reported in [1], where the problem of controlling nonminimum phase systems was solved by using nonlinear H_∞ control, which not does provides robustness evidence. In [9], authors report a solution to the regulation problem using T1FLS as the controller. In [6] and [5], authors report solutions using T2FLS controllers, and making a genetic optimization of the membership function's parameters, but do not specify the criteria used in the optimization process and GA design. In [7], a comparison of the use of GAs to optimize type-1 and type-2 FLS controllers is reported, but a method to achieve this optimization is not provided, but in [8] the authors provides a method to achieve this optimization.

The solution that is proposed in this paper is to design an observer for the mathematical model of our case of study in order to minimize the number of variables involved in the system and to make the design of type-1 FLSs (Fuzzy Logic Controllers -FLCs-) extending the Fuzzy Lyapunov Synthesis [17], a concept that is based on the Computing with Words [19][24] approach of the Lyapunov Synthesis [14], this approach was reported in [3] and [2] for the design of stable FLCs.

The paper is organized as follows. The dynamic model of the nonminimum phase servomechanism with nonlinear backlash is presented in Section 2. Section 3 addresses the problem statement. Section 4 discusses the observer design. The design of fuzzy logic controllers using the Fuzzy Lyapunov Synthesis is presented in Section 5. Experimental results for the designed FLSs are presented in Section 6. Concluding remarks are collected in Section 7.

2 Dynamic Model

The dynamic model of the angular position $q_i(t)$ of the DC motor and the $q_o(t)$ of the load are given according to

$$\begin{aligned} J_0 N^{-1} \ddot{q}_0 + f_0 N^{-1} \dot{q}_0 &= T + w_0 \\ J_i \ddot{q}_i + f_i \dot{q}_i + T &= \tau_m + w_i, \end{aligned} \tag{1}$$

hereafter, J_0 , f_0 , \ddot{q}_0 and \dot{q}_0 are, respectively, the inertia of the load and the reducer, the viscous output friction, the output acceleration, and the output velocity. The inertia of the motor, the viscous motor friction, the motor acceleration, and the motor velocity are denoted by J_i , f_i , \ddot{q}_i and \dot{q}_i , respectively. The input torque τ_m serves as a control action, and T stands for the transmitted torque. The external disturbances $w_i(t)$, $w_0(t)$ have been introduced into the driver equation (1) to account for destabilizing model discrepancies due to hard-to-model nonlinear phenomena, such as friction and backlash.

The transmitted torque T through a backlash with an amplitude j is typically modeled by a dead-zone characteristic [21, p. 7]:

$$T(\Delta q) = \begin{cases} 0 & |\Delta q| \leq j \\ K\Delta q - Kj \operatorname{sign}(\Delta q) & \text{otherwise} \end{cases} \tag{2}$$

with

$$\Delta q = q_i - Nq_o, \tag{3}$$

where K is the stiffness, and N is the reducer ratio. Such a model is depicted in Fig. 1. Provided the servomotor position $q_i(t)$ is the only available measurement on the system, the above model (1)-(3) appears to be non-minimum phase because along with the origin the unforced system possesses a multivalued set of equilibria (q_i, q_o) with $q_i = 0$ and $q_o \in [-j, j]$.

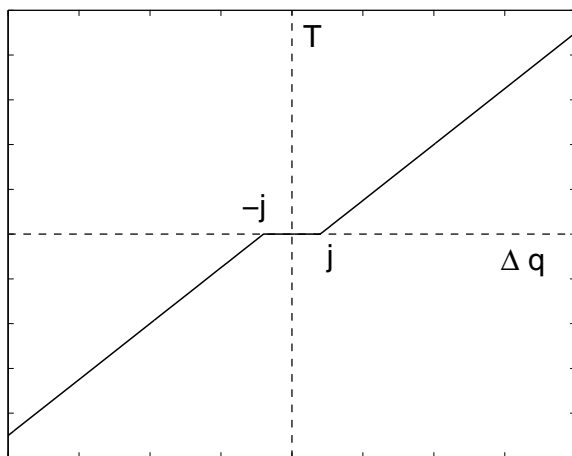
To avoid dealing with a non-minimum phase system, we replace the backlash model (2) with its monotonic approximation:

$$T = K\Delta q - K\eta(\Delta q) \tag{4}$$

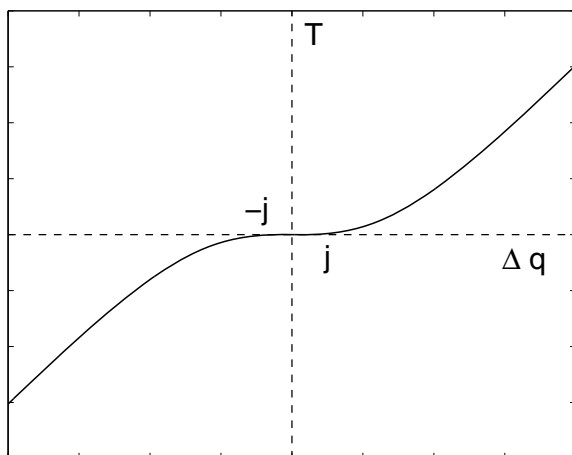
where

$$\eta = -2j \frac{1 - \exp\left\{-\frac{\Delta q}{j}\right\}}{1 + \exp\left\{-\frac{\Delta q}{j}\right\}}. \tag{5}$$

The present backlash approximation is inspired from [20]. Coupled to the drive system (1) subject to motor position measurements, it is subsequently shown to continue a minimum phase approximation of the underlying servomotor, operating under uncertainties $w_i(t)$, $w_0(t)$ to be attenuated. As a matter of fact, these uncertainties involve discrepancies between the physical backlash model (2) and its approximation (4) and (5).



(a)



(b)

Fig. 1. a) The dead-zone model of backlash; b) The monotonic approximation of the dead-zone model

3 Problem Statement

To formally state the problem, let us introduce the state deviation vector $x = [x_1, x_2, x_3, x_4]^T$ with

$$\begin{aligned} x_1 &= q_0 - q_d, \\ x_2 &= \dot{q}_0, \\ x_3 &= q_i - Nq_d, \\ x_4 &= \dot{q}_i, \end{aligned}$$

where x_1 is the load position error, x_2 is the load velocity, x_3 is the motor position deviation from its nominal value, and x_4 is the motor velocity. The nominal motor position Nq_d has been pre-specified in such a way to guarantee that $\Delta q = \Delta x$, where

$$\Delta x = x_3 - Nx_1.$$

Then, system (1)-(5), represented in terms of the deviation vector x , takes the form

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= J_0^{-1}[KNx_3 - KN^2x_1 - f_0x_2 + KN\eta(\Delta q) + w_o], \\ \dot{x}_3 &= x_4, \\ \dot{x}_4 &= J_i^{-1}[\tau_m + KNx_1 - Kx_3 - f_ix_4 + K\eta(\Delta q) + w_i]. \end{aligned} \tag{6}$$

The zero dynamics

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= J_0^{-1}[-KN^2x_1 - f_0x_2 + KN\eta(-Nx_1)], \end{aligned} \tag{7}$$

of the undisturbed version of system (6) with respect to the output

$$y = x_3 \tag{8}$$

is formally obtained (see [12] for details) by specifying the control law that maintains the output identically to zero. The following result, extracted from [22], guarantees that the error system (6) and (8) is globally minimum phase.

Theorem 1. *Let the nonlinearity η be governed by (5). Then system (7) is globally asymptotically stable.*

Proof. To begin with, let us consider a Lyapunov function of the form

$$\begin{aligned} V(x_1, x_2) &= \frac{1}{2}x_2^2 + \frac{1}{2}KN^2J_0^{-1}x_1^2 \\ &\quad + 2j^2KJ_0^{-1} \left(2\ln \frac{e^{Nx_1/j+1} + 1}{2} - \frac{Nx_1}{j} \right). \end{aligned} \tag{9}$$

Since

$$2\ln \frac{e^{Nx_1/j+1} + 1}{2} \geq \frac{Nx_1}{j}$$

for all $x_1 \in \mathbf{R}$, by inspection, the function V , governed by (9), appears to be positive definite.

Then, let us compute the time derivative of the Lyapunov function on the trajectories of (7)

$$\begin{aligned} \dot{V} &= -J_0^{-1} \left(KN^2 x_1 + f_0 X_2 + 2jKN \frac{e^{Nx_1/j} - 1}{e^{Nx_1/j+1}} \right) x_2 \\ &\quad + KN^2 J_0^{-1} x_1 x_2 \\ &\quad + 2j^2 K J_0^{-1} \left(\frac{4Ne^{Nx_1/j}}{2j(e^{Nx_1/j+1})} x_2 - \frac{Nx_2}{j} \right) \\ &= -J_0^{-1} f_0 X_2^2 \leq 0. \end{aligned} \tag{10}$$

Now let us observe that the zero-dynamics system (7) has no trivial solutions on the manifold $x_2 = 0$, where the time derivative of the Lyapunov function equals to zero. Indeed, if $x_2 = 0$, then due to (7)

$$\frac{Nx_1}{j} + 2 \frac{1 - e^{Nx_1/j}}{1 + e^{Nx_1/j}} = 0 \tag{11}$$

thus concluding that $x_1 = 0$. To reproduce the latter conclusion, it suffices to represent (11) in terms of $v = Nx_1/j$

$$v + 2 \frac{1 - e^v}{1 + e^v} = 0 \tag{12}$$

and note that the left-hand side of (12) is a strictly increasing function of v because its derivative is positive definite by inspection

$$1 - \frac{4e^v}{(1 + e^v)^2} = \frac{(1 - e^v)^2}{(1 + e^v)^2} > 0 \tag{13}$$

for all $v \neq 0$.

In order to complete the proof, it remains to apply the LaSalle-Krasovski invariance principle to the system in question.

The objective of the Fuzzy Control output regulation of the nonlinear driver system (1) with backlash (4) and (5), is thus to design a Fuzzy Controller so as to obtain the closed-loop system in which all these trajectories are bounded and the output $q_0(t)$ asymptotically decays to a desired position q_d as $t \rightarrow \infty$ while also attenuating the influence of the external disturbances $w_i(t)$ and $w_0(t)$.

4 Observer Design

Whereas the design of fuzzy controllers is a task that increases the degree of difficulty as the number of variables increases, to reduce the number of system variables (6), we propose the following observer:

$$\begin{aligned} \dot{\tilde{x}}_1 &= \tilde{x}_2, \\ \dot{\tilde{x}}_2 &= J_0^{-1} [KNx_3 - KN^2\tilde{x}_1 - f_0\tilde{x}_2 + KN\eta(\tilde{\Delta}q) + w_o] \end{aligned} \quad (14)$$

where

$$\tilde{\Delta}q = \tilde{\Delta}x = x_3 - N\tilde{x}_1. \quad (15)$$

The observer (14)-(15) reduces to two the original four variables of (6). To verify that (14) behaves as (6) we define the error variables:

$$\begin{aligned} \dot{e}_1 &= e_2 \\ \dot{e}_2 &= J_0^{-1} [-KN^2e_1 - f_0e_2 + KN\eta(\tilde{\Delta}q)] \end{aligned} \quad (16)$$

Then, the following candidate Lyapunov function is proposed

$$\begin{aligned} V(e_1, e_2) &= \frac{1}{2}e_2^2 + \frac{1}{2}KN^2J_0^{-1}e_1^2 \\ &\quad + 2j^2KJ_0^{-1} \left(2\ln \frac{e^{Ne_1/j+1}}{2} - \frac{Ne_1}{j} \right), \end{aligned} \quad (17)$$

whose time derivative is

$$\begin{aligned} \dot{V} &= -J_0^{-1} \left(KN^2e_1 + f_0e_2 + 2jKN \frac{e^{Ne_1/j-1}}{e^{Ne_1/j+1}} \right) e_2 \\ &\quad + KN^2J_0^{-1}e_1e_2 \\ &\quad + 2j^2KJ_0^{-1} \left(\frac{4Ne^{Ne_1/j}}{2j(e^{Ne_1/j+1})} e_2 - \frac{Ne_2}{j} \right) \\ &= -J_0^{-1}f_0e_2^2 \leq 0. \end{aligned} \quad (18)$$

In order to complete the proof, it remains to apply the LaSalle-Krasovski invariance principle to the system in question, concluding asymptotic stability.

5 Design of the Fuzzy Logic Controllers

To apply the Fuzzy Lyapunov Synthesis, we assume the following:

1. The system has really two degrees of freedom referred to as \tilde{x}_1 and \tilde{x}_2 , respectively. Hence by (14), $\dot{\tilde{x}}_1 = \tilde{x}_2$.
2. The states \tilde{x}_1 and \tilde{x}_2 are the observed variables.
3. The exact equations (1)-(5) are not necessarily known.
4. The angular acceleration $\dot{\tilde{x}}_2$ is proportional to τ_m , that is, when τ_m increases (decreases) $\dot{\tilde{x}}_2$ increases (decreases).
5. The initial conditions $\tilde{x}(0) = (\tilde{x}_1(0), \tilde{x}_2(0))^T$ belong to the set $\mathfrak{X} = \{\tilde{x} \in \mathbf{R}^2 : \|\tilde{x} - \tilde{x}^* \| \leq \varepsilon\}$ where \tilde{x}^* is the equilibrium point and $\tilde{x} = [\tilde{x}_1, \tilde{x}_2]^T$.

The *control objective* is to design the rule-base as a fuzzy controller $\tau_m = \tau_m(\tilde{x}_1, \tilde{x}_2)$ to stabilize the system (1)-(5).

Theorem 2 that follows establish conditions that help in the design of the fuzzy controller to ensure asymptotic stability. The proof can be found in [14].

Theorem 2 (Asymptotic stability [14]). Consider the nonlinear system (1)-(5) with an equilibrium point at the origin, i.e., $f(0)=(0)$, and let $x \in \mathfrak{X}$, then the origin is asymptotically stable if there exists a scalar Lyapunov function $V(\tilde{x})$, with continuous partial derivatives such that

- $V(\tilde{x})$ is positive definite
- $\dot{V}(\tilde{x})$ is negative definite.

The fuzzy controller design proceeds as follows. Let us introduce the Lyapunov function candidate

$$V(\tilde{x}_1, \tilde{x}_2) = \frac{1}{2} (\tilde{x}_1^2 + \tilde{x}_2^2), \quad (19)$$

which is positive-definite and radially unbounded function. The time derivative of $V(\tilde{x}_1, \tilde{x}_2)$ results in:

$$\dot{V}(\tilde{x}_1, \tilde{x}_2) = \tilde{x}_1 \dot{\tilde{x}}_1 + \tilde{x}_2 \dot{\tilde{x}}_2 = \tilde{x}_1 \tilde{x}_2 + \tilde{x}_2 \dot{\tilde{x}}_2, \quad (20)$$

To guarantee stability of the equilibrium point $(\tilde{x}_1^*, \tilde{x}_2^*)^T = (0, 0)^T$ we wish to have:

$$\tilde{x}_1 \tilde{x}_2 + \tilde{x}_2 \dot{\tilde{x}}_2 \leq 0. \quad (21)$$

We can now derive sufficient conditions so that inequality (21) holds: If \tilde{x}_1 and \tilde{x}_2 have opposite signs, then $\tilde{x}_1 \tilde{x}_2 < 0$ and (21) will hold if $\dot{\tilde{x}}_2 = 0$; if \tilde{x}_1 and \tilde{x}_2 are both positive, then (21) will hold if $\dot{\tilde{x}}_2 < -\tilde{x}_1$; if \tilde{x}_1 and \tilde{x}_2 are both negative, then (21) will hold if $\dot{\tilde{x}}_2 > -\tilde{x}_1$.

We can translate these conditions into the following fuzzy rules:

- If \tilde{x}_1 is *positive* and \tilde{x}_2 is *positive* then $\dot{\tilde{x}}_2$ must be *negative big*.
- If \tilde{x}_1 is *negative* and \tilde{x}_2 is *negative* then $\dot{\tilde{x}}_2$ must be *positive big*.
- If \tilde{x}_1 is *positive* and \tilde{x}_2 is *negative* then $\dot{\tilde{x}}_2$ must be *zero*.
- If \tilde{x}_1 is *negative* and \tilde{x}_2 is *positive* then $\dot{\tilde{x}}_2$ must be *zero*.

However, using our knowledge that $\dot{\tilde{x}}_2$ is proportional to u , we can replace each $\dot{\tilde{x}}_2$ with u to obtain the following fuzzy rule-base for the stabilizing controller:

- If \tilde{x}_1 is *positive* and \tilde{x}_2 is *positive* then u must be *negative big*.
- If \tilde{x}_1 is *negative* and \tilde{x}_2 is *negative* then u must be *positive big*.
- If \tilde{x}_1 is *positive* and \tilde{x}_2 is *negative* then u must be *zero*.
- If \tilde{x}_1 is *negative* and \tilde{x}_2 is *positive* then u must be *zero*.

This fuzzy rule-base can be represented as in Table 1.

It is interesting to note that the fuzzy partitions for \tilde{x}_1 , \tilde{x}_2 , and u follow elegantly from expression (20). Because $\dot{V} = \tilde{x}_2 (\tilde{x}_1 + \dot{\tilde{x}}_2)$, and since we require that \dot{V} be negative, it is natural to examine the signs of \tilde{x}_1 and \tilde{x}_2 ; hence, the obvious fuzzy partition is *positive*, *negative*. The partition for $\dot{\tilde{x}}_2$, namely *negative big*, *zero*, *positive big* is obtained similarly when we plug the linguistic values *positive*, *negative* for \tilde{x}_1 and \tilde{x}_2 in (20). To ensure that $\dot{\tilde{x}}_2 < -\tilde{x}_1$ ($\dot{\tilde{x}}_2 > -\tilde{x}_1$) is satisfied even though we do not know \tilde{x}_1 's exact magnitude, only that it is *positive* (*negative*), we must set

Table 1. Fuzzy IF-THEN rules

No.	error	change of error	control
1	positive	positive	negative big
2	negative	negative	positive big
3	positive	negative	zero
4	negative	positive	zero

\dot{x}_2 to *negative big* (*positive big*). Obviously, it is also possible to start with a given, pre-defined, partition for the variables and then plug each value in the expression for \dot{V} to find the rules. Nevertheless, regardless of what comes first, we see that fuzzy Lyapunov synthesis transforms classical Lyapunov synthesis from the world of exact mathematical quantities to the world of computing with words [24], [19].

To complete the controller’s design, we must model the linguistic terms in the rule-base using fuzzy membership functions and determine an inference method. Following [3], we characterize the linguistic terms *positive*, *negative*, *negative big*, *zero* and *positive big*. The type-1 MFs are depicted in Fig. 2 for a type-1 fuzzy logic controller. To this end, we had systematically designed a FLC following the Lyapunov stability criterion.

6 Results

In this section we present results for the closed-loop systems for the type-1 fuzzy logic controllers. To perform experiments we use the dynamical model (1)-(5), which involves a DC motor linked to a mechanical load through an imperfect contact gear train [1], Fig. 3 show the systems testbed. The parameters of the dynamical model (1)-(5) are in Table 2, while $N = 3$, $j = 0.2$ [rad], and $K = 5$ [N-m/rad].

Table 2. Nominal parameters

Description	Notation	Value	Units
Motor inertia	J_i	2.8×10^{-6}	Kg-m ²
Load inertia	J_o	1.07	Kg-m ²
Motor viscous friction	f_i	7.6×10^{-7}	N-m-s/rad
Load viscous friction	f_o	1.73	N-m-s/rad

The experimental setup involves a DC motor linked to a mechanical load through an imperfect contact gear train. Fig. 4 shows the control diagrams, showing the locations of sensors, actuators and load.

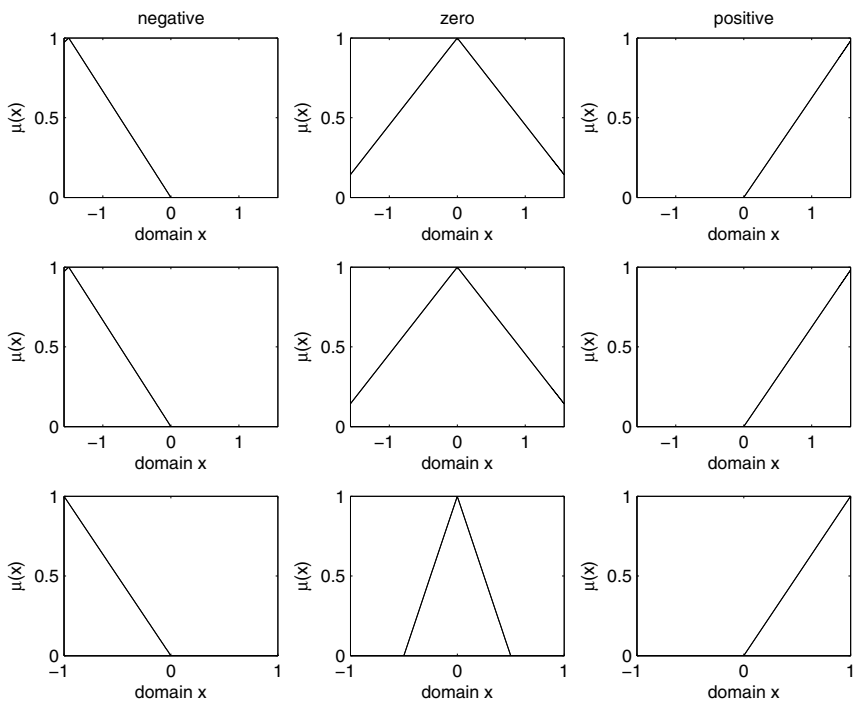


Fig. 2. Set of type-1 membership functions (First row for x_1 , second row for x_2 and third row for u)

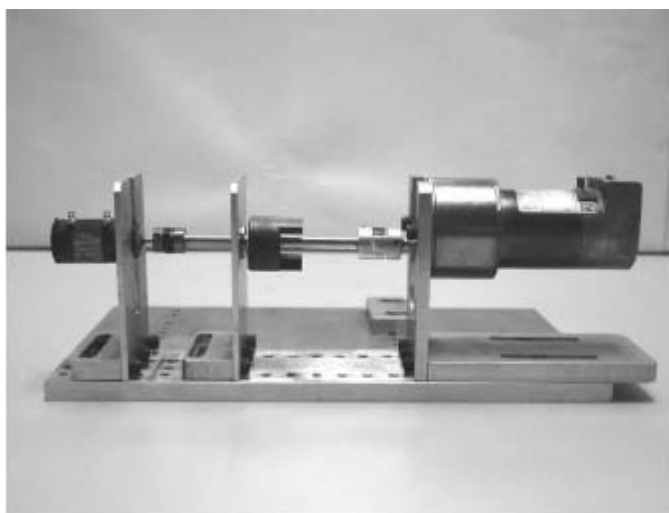


Fig. 3. Physical testbed

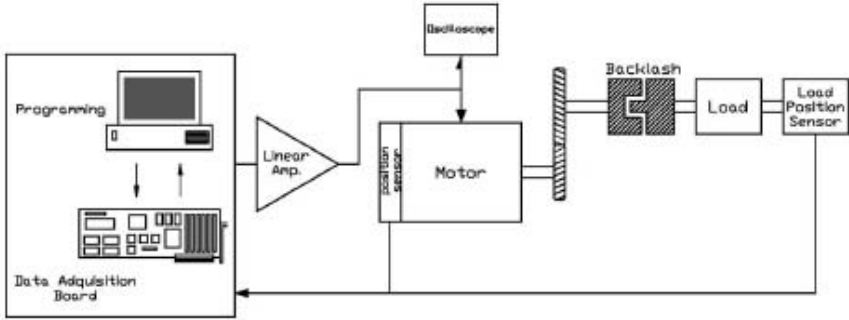


Fig. 4. Control diagram

With the fuzzy rule-base of Table 1 and the MFs depicted in Fig. 2, we obtain the Control Surface of Fig. 5.

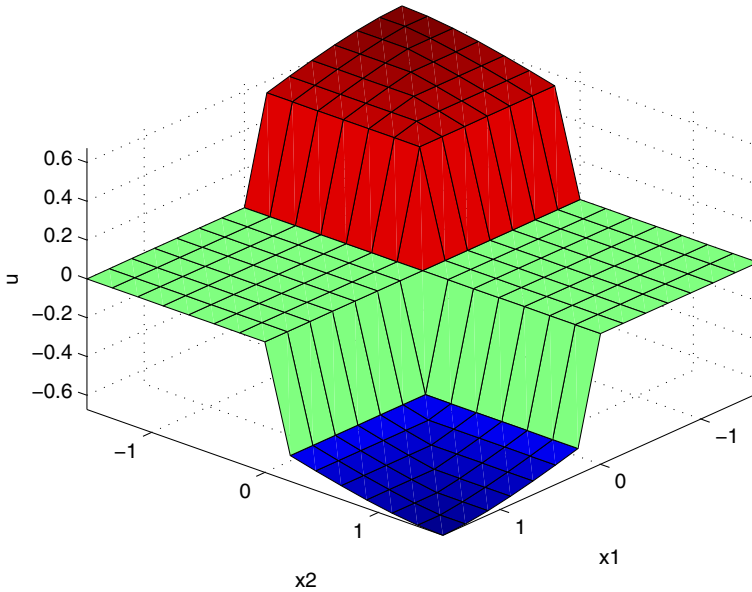


Fig. 5. Control surface for the T1FLC

Applying this T1FLC to the proposed problem, we obtain the system's response of Fig. 6, that is, $q_0 \rightarrow q_d$ while $x_1 \rightarrow 0$, this convergence is due to the behavior of \dot{x}_1 and \dot{x}_2 , this convergence is shown in Fig. 7. To emphasize the fact that the system (14) observes (6), Fig. 8 shows the behavior of the variables e_1 and e_2 . Fig. 9 shows the behavior of the Lyapunov Function V (19) and its derivative \dot{V} (20).

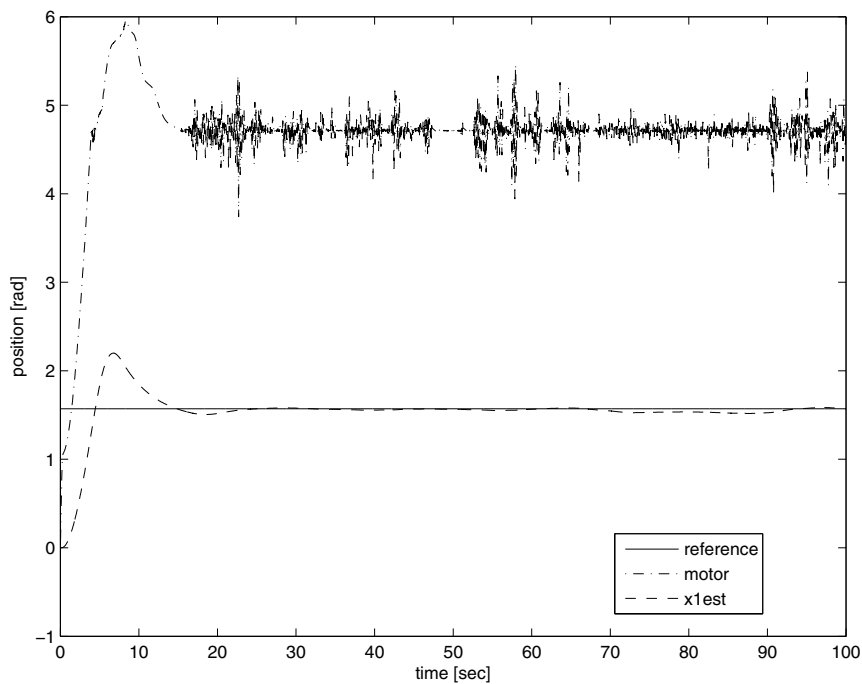


Fig. 6. System's response for the T1FLC

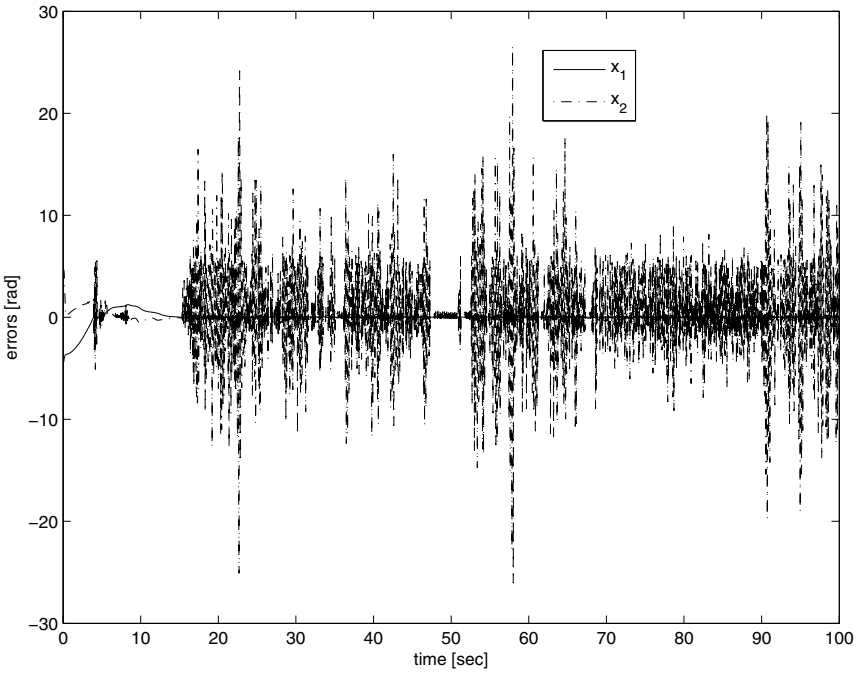


Fig. 7. \dot{x}_1 and \dot{x}_2

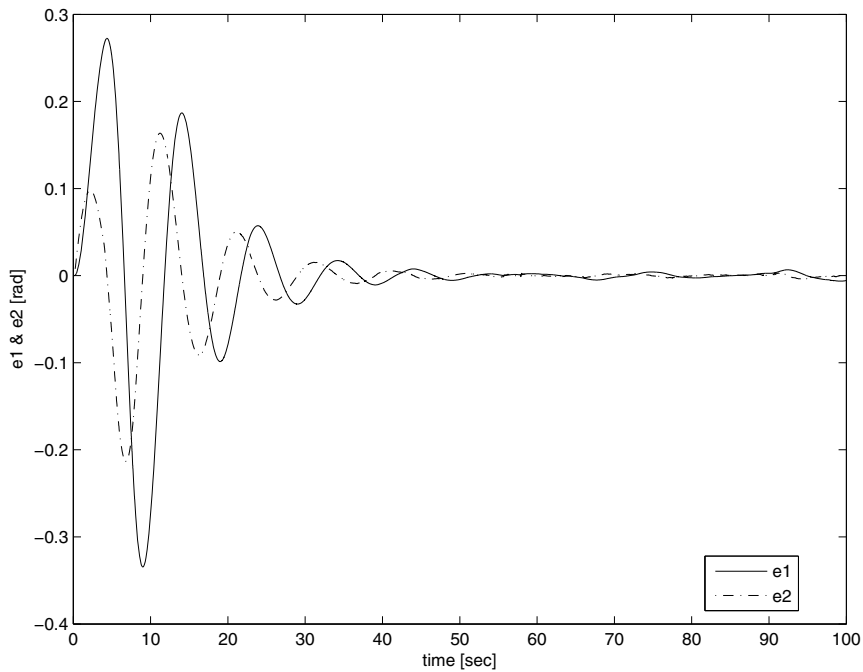


Fig. 8. e_1 and e_2

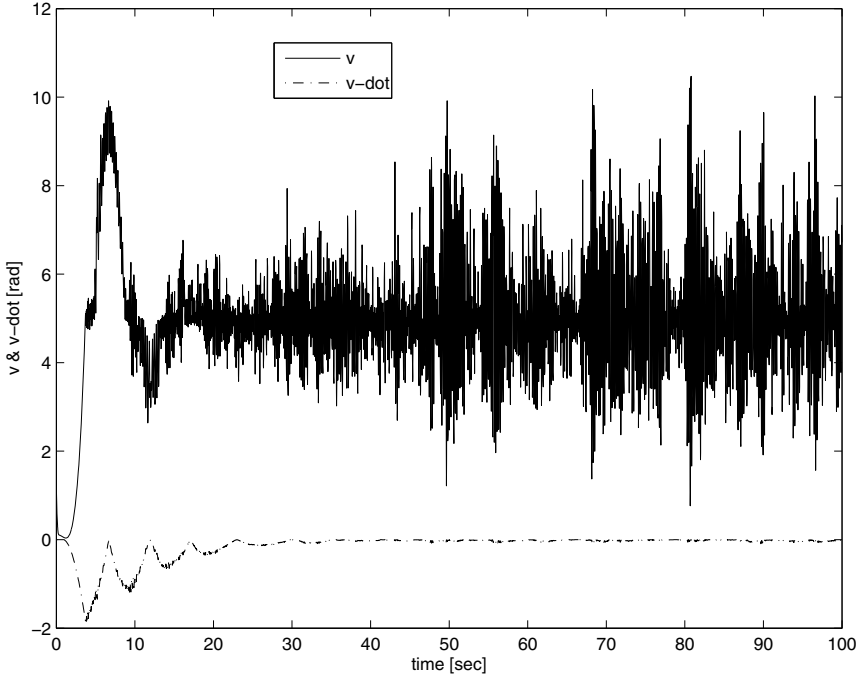


Fig. 9. V and \dot{V}

Table 3 concentrates some measures that can help us to understand the results presented in this paper.

Table 3. T1FLC results

Settling time	Overshoot	ISE	IAE	ITSE	ITAE
15 s	2.319	63233	19697	2639600	827260

7 Conclusion

The main goal of this paper was to design an observer for the mathematical model of the case of study in order to propose a systematic methodology to design type-1 fuzzy logic controllers to solve the output regulation problem of a servomechanism with nonlinear backlash.

We proposed an observer for the system and we did a demonstration of its effectiveness through a Lyapunov stability test.

From the observed variables we design, using fuzzy Lyapunov synthesis, the rule base of a controller that ensures stability for the closed-loop system, and based on the design of the rule base, we identify the granulation necessary to know the number and characteristics of the membership functions for each variable involved in the fuzzy controller.

The proposed design strategy results in two controllers that guarantee that the load reaches the desired position q_d , solving the regulation problem as was predicted, this affirmation is supported with experiments where both, the type-1 FLC designed by following the Fuzzy Lyapunov Synthesis achieve solving the regulation problem.

References

1. Aguilar, L.T., Orlov, Y., Cadiou, J.C., Merzouki, R.: Nonlinear H_∞ -output regulation of a nonminimum phase servomechanism with backlash. *Journal of Dynamic Systems, Measurement, and Control* 129(4), 544–549 (2007), <http://link.aip.org/link/?JDS/129/544/1>, doi:10.1115/1.2719774
2. Castillo, O., Aguilar, L., Cazarez-Castro, N., Rico, D.: Intelligent control of dynamic systems using type-2 fuzzy logic and stability issues. *International Mathematical Forum* 1(28), 1371–1382 (2006)
3. Castillo, O., Aguilar, L.T., Cazarez, N., Cardenas, S.: Systematic design of a stable type-2 fuzzy logic controller. *Journal of Applied Soft Computing* 8(3), 1274–1279 (2008)
4. Castillo, O., Lozano, A., Melin, P.: Hierarchical genetic algorithms for fuzzy system optimization in intelligent control. *IEEE Annual Meeting of the Fuzzy Information. Processing NAFIPS 2004* 1, 292–297 (2004), doi:10.1109/NAFIPS.2004.1336294
5. Cazarez-Castro, N.R., Aguilar, L.T., Castillo, O.: Genetic optimization of a type-2 fuzzy controller for output regulation of a servomechanism with backlash. In: *5th International Conference on Electrical Engineering, Computing Science and Automatic Control. CCE 2008.*, pp. 268–273 (2008), doi:10.1109/ICEEE.2008.4723381
6. Cazarez-Castro, N., Aguilar, L., Castillo, O.: Hybrid genetic-fuzzy optimization of a type-2 fuzzy logic controller. In: *Eighth International Conference on Hybrid Intelligent Systems. HIS 2008*, pp. 216–221 (2008), doi:10.1109/HIS.2008.170
7. Cazarez-Castro, N., Aguilar, L., Castillo, O., Rodriguez, A.: Optimizing type-1 and type-2 fuzzy logic systems with genetic algorithms. *Research in Computing Science* 39, 131–153 (2008)
8. Cazarez-Castro, N.R., Aguilar, L.T., Castillo, O.: Fuzzy logic control with genetic membership function parameters optimization for the output regulation of a servomechanism with nonlinear backlash. *Expert Systems with Applications* 37(6), 4368 (2010), <http://www.sciencedirect.com/science/article/B6V03-4XVBP4G-3/2/cb7495057c9a8efb1becf7f86384685e>, doi:10.1016/j.eswa.2009.11.091
9. Cazarez-Castro, N.R., Aguilar, L.T., Castillo, O., Cardenas, S.: Fuzzy Control for Output Regulation of a Servomechanism with Backlash. In: *Soft Computing for Hybrid Intelligent Systems, 1st edn. Studies in Computational Intelligence*, vol. 154, pp. 19–28. Springer, Berlin (2008)
10. Chang, Y.-H., Chang, C.-W., Taur, J.-S., Tao, C.-W.: Fuzzy swing-up and fuzzy sliding-mode balance control for a planetary-gear-type inverted pendulum. *IEEE Transactions on Industrial Electronics* 56(9), 3751–3761 (2009), doi:10.1109/TIE.2009.2025292
11. Grefenstette, J.: Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics* 16(1), 122–128 (1986), doi:10.1109/TSMC.1986.289288
12. Isidori, A.: *Nonlinear Control Systems*. Springer, New York (1995)

13. Isidori, A.: A tool for semi-global stabilization of uncertain non-minimum-phase nonlinear systems via output feedback. *IEEE Transactions on Automatic Control* 45(10), 1817–1827 (2000), doi:10.1109/TAC.2000.880972
14. Khalil, H.K.: *Nonlinear Systems*, 3rd edn. Prentice Hall, Englewood Cliffs (2002)
15. Lagerberg, A., Egardt, B.: Estimation of backlash with application to automotive powertrains. In: *Proc. of the 42th Conf. on Decision and Control*, pp. 135–147 (1999)
16. Lee, M.A., Takagi, H.: Integrating design stage of fuzzy systems using genetic algorithms. In: *Second IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 612–617 (1993), doi:10.1109/FUZZY.1993.327418
17. Margaliot, M., Langholz, G.: Adaptive fuzzy controller design via fuzzy lyapunov synthesis. In: *The 1998 IEEE International Conference on Fuzzy Systems Proceedings. IEEE World Congress on Computational Intelligence*, vol. 1, pp. 354–359 (1998), doi:10.1109/FUZZY.1998.687511
18. Melin, P., Castillo, O.: Intelligent control of complex electrochemical systems with a neuro-fuzzy-genetic approach. *IEEE Transactions on Industrial Electronics* 48(5), 951–955 (2001), doi:10.1109/41.954559
19. Mendel, J.M.: Computing with words: Zadeh, turing, popper and occam. *Computational Intelligence Magazine* 2(4), 10–17 (2007), doi:10.1109/MCI.2007.9066897
20. Merzouki, R., Cadiou, J.C., M'Sirdi, N.K.: Compensation of friction and backlash effects in an electrical actuator. *J. System and Control Engineering* 218(12), 75–84 (2004)
21. Nordin, M., Bodin, P., Gutman, P.: *New Models and Identification Methods for Backlash and Gear Play*. In: *Adaptive Control of Nonsmooth Dynamic Systems*, pp. 1–30. Springer, Berlin (2001)
22. Orlov, Y.: Finite time stability and quasihomogeneous control synthesis of uncertain switched systems with application to underactuated manipulators. In: *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference, CDC-ECC 2005*, pp. 4566–4571 (2005)
23. Ruderman, M., Hoffmann, F., Bertram, T.: Modeling and identification of elastic robot joints with hysteresis and backlash. *IEEE Transactions on Industrial Electronics* 56(10), 3840–3847 (2009), doi:10.1109/TIE.2009.2015752
24. Zadeh, L.: Fuzzy logic = computing with words. *IEEE Transactions on Fuzzy Systems* 4(2), 103–111 (1996), doi:10.1109/91.493904

Neuro-Fuzzy Based Output Feedback Controller Design for Biped Robot Walking

Selene L. Cardenas-Maciel¹, Oscar Castillo², and Luis T. Aguilar³

¹ Universidad Autonoma de Baja California, Tijuana, BC, Mexico
lilettecardenas@ieee.org

² Instituto Tecnologico de Tijuana, Mexico
ocastillo@hafsamx.org

³ Instituto Politecnico Nacional, Mexico
luis.aguilar@ieee.org

Abstract. A neuro-fuzzy learning algorithm is applied to design Takagi-Sugeno Fuzzy Controllers for a biped robot walking problem. The appropriate control signal is provided with a rule to switch between the controllers. A simulation of generation of walking motions is presented to illustrate the effectiveness of this approach.

1 Introduction

Fuzzy logic control as an application area of Fuzzy Logic has attracted attention from scientist and engineers due that through if-then rules can express knowledge of how to control a system incorporating uncertainty. The existence of a big number of variables in the measurable variables space (system o external variables) and the partition needed in these variables, are factors that have influence in the size of the rule base, which implies a nontrivial task into the modeling. In order to overcome this difficulties, Clustering, Adaptive Neuronal Networks and Neuro-fuzzy approaches have been proposed and applied to identification and control tasks where the model of the system to control are not available as in [2]; for observer design proposed in [6] to control of robot manipulators. In biped robot locomotion control, adaptive neural networks and neuro-fuzzy methods are implemented where uses the Zero Moment Point (ZMP) [12] information. For example, in [10] several Adaptive-Network-based Fuzzy Inference System (ANFIS)[7] are proposed to design a locomotion controller and another one to determine the model of the system. A learning algorithm and a grid partition method are proposed in [3] to obtain a walking controller. Neuro-fuzzy approaches applied in ZMP control trajectories are proposed in [4].

This paper is related to the application of ANFIS a design tool of an output feedback controller such that it can induce walking motions in a biped robot whose model consider the impact with the environment. The control loop uses an output definition, which include the movement task, if the output is carried to zero the biped robot perform the desired motion. Thus, the controller design is based in the off-line learning of a nonlinear relationship between the state space, the outputs and the control inputs information which are distributed in multiple instances of the learning algorithm which produce a set of Takagi-Sugeno type Fuzzy Logic Controllers specialized in a subspace. The final control input is defined as a switching function dependant of the time.

The rest of this paper is organized as follows: Section 2 introduces the mathematical model of the biped robot. In Section 3, describes some aspects of the neuro-fuzzy approach. Section 4 explains the control strategy used for controller design. Section 5, illustrates simulation results of the closed-loop system. Section 6 provides some conclusions.

2 Biped Robot Model

In this section the dynamical model of a planar biped robot is introduced. The biped robot consists of a torso, hips, and two legs of equal length without ankles and knees. Two torques are applied between the legs and torso. The definition of the angular coordinates and the disposition of the masses of the torso, hips and legs of the biped robot are shown in Fig. 1, the positive angles are computed in a clockwise manner with respect to the indicated vertical lines and all masses of the links are lumped.

The walking cycle takes place in the sagittal plane and on a surface level, and it is assumed as successive phases where only one leg (stance leg) touching the walking surface (swing phase), with the transition from one leg to another taking place in a smaller length of time. During the swing phase, the stance leg is modeled like a pivot and the swing leg is assumed to move into the frontal plane [9] and to reenter the plane of the motion when the angle of the stance leg attains a given desired value. The assumptions concerning the walking cycle, define the biped robot model in two parts: the model that describes the swing phase and another one that describes the contact event of the swing leg with the walking surface; which are presented below.

2.1 Swing Phase Model

The dynamical model of the robot during the swing phase is a second order system derived from the Lagrange method [8]:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) = Bu, \quad (1)$$

where $\theta = [\theta_1, \theta_2, \theta_3]^T$ are the generalized coordinates, θ_1 parameterizes the stance leg, θ_2 the swing leg and θ_3 the torso; $u = [u_1, u_2]^T$ are the input; $M(\theta)$ is a positive-definite inertia matrix, $C(\theta, \dot{\theta})\dot{\theta}$ is the vector of the centripetal and Coriolis forces; and $G(\theta)$ is the vector of the conservative forces and B is the input matrix.

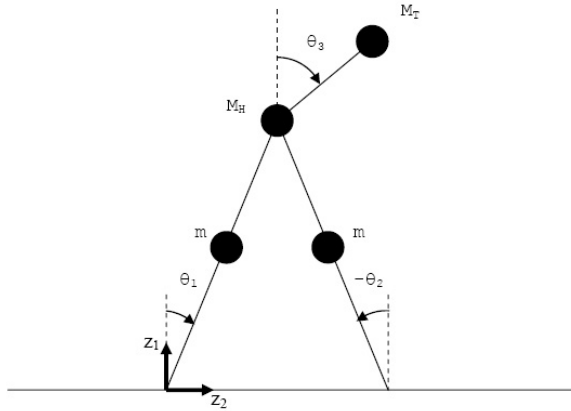


Fig. 1. 3-link biped robot

the state-space form of the second order equation (1) is defined as:

$$\begin{aligned} \dot{x} &:= \frac{d}{dt} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ M^{-1}(\theta)[-C(\theta, \dot{\theta})\dot{\theta} - G(\theta) + Bu] \end{bmatrix} \\ &:= f(x) + g(x)u. \end{aligned} \tag{2}$$

2.2 Impact Model

The impact between the swing leg and the ground is modeled as the contact between two rigid bodies. The main objective is to obtain the velocity of the generalized coordinates after the impact of the swing leg with the walking surface in terms of the velocity and position before the impact. The impact model for the biped robot is based on the rigid impact model of [5] and assuming the case where the contact of the swing leg with the walking surface produce either no rebound nor slipping of the swing leg, and the stance leg lifting the walking surface without interaction. the impact model is expressed with equation [13]:

$$\begin{bmatrix} M_e & -E^T \\ E & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_e^+ \\ F \end{bmatrix} = \begin{bmatrix} M_e \dot{\theta}_e^- \\ 0 \end{bmatrix}, \tag{3}$$

where $\theta_e = [\theta_1, \theta_2, \theta_3, z_1, z_2]^T$ are the generalized coordinates resulting of the add the cartesian coordinates $[z_1, z_2]^T$ of the end of the stance leg as shown in Figure 1; M_e is the positive-definite inertia matrix of the biped model with the new generalized coordinates; $F = [F_T, F_N]^T$ represent the tangential and normal forces applied at the

end of the swing leg; $\dot{\theta}_e^+$ is the velocity after the impact; $\dot{\theta}_e^-$ is the velocity before the impact and

$$E := \frac{\partial Y}{\partial \theta_e} = \begin{bmatrix} r \cos(\theta_1) & -r \cos(\theta_2) & 0 & 1 & 0 \\ -r \sin(\theta_1) & r \sin(\theta_2) & 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

solving for $\dot{\theta}_e^+$ of equation (3) and a change of coordinates must be done and also a re-initialization of the model (2). The change of coordinates is an expression for $x^+ := (\theta^+, \dot{\theta}^+)$ in terms of $x^- := (\theta^-, \dot{\theta}^-)$, which is given for the mapping function [13]:

$$x^+ := \Delta(x^-) := \begin{bmatrix} \theta_2^- \\ \theta_1^- \\ \theta_3^- \\ \dot{\theta}_2^+(x^-) \\ \dot{\theta}_1^+(x^-) \\ \dot{\theta}_3^+(x^-) \end{bmatrix}. \quad (5)$$

Thus, the overall model of biped walking is the combination of the swing phase model and impact model, as follows:

$$\Sigma_{cl} : \begin{cases} \dot{x} = f(x) + g(x)u(x) & x^- \notin S \\ x^+ = \Delta(x^-) & x^- \in S \end{cases} \quad (6)$$

where

$$S := \{(\theta, \dot{\theta}) | z_1 > 0, z_2 = 0, \theta_1 = \theta_1^d\} \quad (7)$$

3 ANFIS and Fuzzy Models

ANFIS [7] is a class of adaptive neural network whose functionally is equivalent to the T-S fuzzy reasoning mechanism [11]. The ANFIS architecture consist on five layers and the node function for each node of same layer corresponds to the same family functions. The architecture is detailed as follows:

3.1 Layer 1

Every node i in this layer is an adaptive node with a node function

$$O_i^1 = \mu_{A_i}(x) \quad (8)$$

where x is the input to node i and A_i is the linguistic label associated with this node, O_i^1 is the membership function of the fuzzy set A_i , it specifies the grade that the given value x satisfies the quantifier A . The membership function $\mu_{A_i}(x)$ can be parameterized for

$$\mu_{A_i}(x) = \exp\left\{-\frac{1}{2}\left(\frac{x-c_i}{a_i}\right)^2\right\} \quad (9)$$

where a_i and c_i are referred as premise parameters.

3.2 Layer 2

Every node i in this layer is a fixed node labeled Π , whose output is the product of all the incoming signals. For example, consider two linguistic variables A and B characterized by two fuzzy sets each. Then the output of a node i of this layer is obtained by:

$$O_i^2 = w_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1, 2. \tag{10}$$

each node represent the firing strength of a rule.

3.3 Layer 3

Every node in this layer is a fixed node labeled N . The i th node calculate the ratio of the i th rule's firing strength to the sum of the all rules' firing strengths. The outputs of this layer are called normalized firing strengths (\bar{w}_1).

$$O_i^3 = \bar{w}_i = \frac{w_i}{\sum_i w_i}, \quad i = 1, 2. \tag{11}$$

3.4 Layer 4

Every node i in this layer is an adaptive node with node function

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i), \quad i = 1, 2. \tag{12}$$

where p_i, q_i, r_i is the parameter set of this node, this are the consequent parameters.

3.5 Layer 5

The single node in this layer is a fixed node labeled Σ , which computes the overall output as the summation of all incoming signals:

$$O_i^5 = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{13}$$

The learning rule combines gradient method and least squares estimate to identify and update the parameters of the network, these parameters are the premise and consequent parameters which describe a Fuzzy Model.

Thus, ANFIS can serve as basis for constructing a set of T-S type fuzzy model [11] with appropriate membership functions starting from a set of input-output data pairs. Consider a T-S fuzzy model with input variables x_1, x_2, \dots, x_j and output variables u_1, u_2, \dots, u_k , the l th rule takes the form:

$$\begin{aligned} R^l: & \text{ IF } \quad x_1 \text{ is } A_{1,i} \text{ and } x_2 \text{ is } A_{2,i} \text{ and } \dots x_j \text{ is } A_{j,i} \\ & \text{ THEN} \\ & \quad u_1 = G(x_1, x_2, \dots, x_j; p_{1,1}, p_{1,2}, \dots, p_{1,j}, p_{1,j+1}) \\ & \quad \text{and } \dots \text{ and} \\ & \quad u_k = G(x_1, x_2, \dots, x_j; p_{k,1}, p_{k,2}, \dots, p_{k,j}, p_{k,j+1}) \end{aligned} \tag{14}$$

where $A_{j,i}$ with $i = 1, 2, \dots, m_1$ and $j = 1, 2, \dots, m_2$, are the i th fuzzy set of the j th input variable; u_k with $k = 1, 2, \dots, m_3$, each $G(\cdot)$ is an crisp continuous function in terms of the inputs with $j + 1$ parameters p .

4 Control Strategy

This paper uses the output form proposed in [13], which encode a geometric task for the robot, identifies the zero dynamics of the system model including the presence of impacts and design a feedback controller that generate exponential stable walking motions. The output is defined as follows:

$$y := \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} := \begin{bmatrix} h_1(\theta, a) \\ h_2(\theta, a) \end{bmatrix} := \begin{bmatrix} \theta_3 - h_{d,1}(\theta_1, a) \\ \theta_2 - h_{d,2}(\theta_1, a) \end{bmatrix} \quad (15)$$

where

$$\begin{aligned} h_{d,1}(\theta_1, a) &:= a_1^0 + \dots + a_1^3(\theta_1)^3 \\ h_{d,2}(\theta_1, a) &:= -\theta_1 + (a_2^0 + \dots + a_2^3(\theta_1)^3) \times (\theta_1 + \theta_1^d) \times (\theta_1 - \theta_1^d). \end{aligned} \quad (16)$$

The control objective is to find a intelligent control law taking the trajectories of the closed-loop system during a walking cycle obtained in [13] such that the output is driving to zero.

4.1 Controller Design

This paper uses ANFIS for the T-S Fuzzy Logic Controller (T-S FLC) design. In order to accomplish the control objective, the state trajectories $x = [\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3]^T$ are considered, the output trajectories $y = [y_1, y_2]^T$ and the control inputs $u = [u_1, u_2]^T$ corresponding to closed-loop system simulation of the work proposed in [13] during a walking cycle, to built the data set (input-output data pairs).

The data set is partitioned to diminish the computational load and the time to the algorithm, and then distributed to different instances of the learning algorithm as follows: 2 ANFIS take each q th partition and use as input data the state and output trajectories; as output data, one of the control inputs. The l th rule of each T-S FLC has the form as:

$$\begin{aligned} R^l: \text{ IF } & \theta_1 \text{ is } A_{1,i} \text{ and } \theta_2 \text{ is } A_{2,i} \text{ and } \theta_3 \text{ is } A_{3,i} \text{ and} \\ & \dot{\theta}_1 \text{ is } A_{4,i} \text{ and } \dot{\theta}_2 \text{ is } A_{5,i} \text{ and } \dot{\theta}_3 \text{ is } A_{6,i} \text{ and} \\ & y_1 \text{ is } A_{7,i} \text{ and } y_2 \text{ is } A_{8,i} \\ \text{ THEN} & \\ & u_k = p_{1,1}\theta_1 + p_{1,2}\theta_2 + p_{1,3}\theta_3 + p_{1,4}\dot{\theta}_1 + \\ & p_{1,5}\dot{\theta}_2 + p_{1,6}\dot{\theta}_3 + p_{1,7}y_1 + p_{1,8}y_2 + p_{1,9} \end{aligned} \quad (17)$$

where $A_{j,i}$ with $i = 1, 2$ are fuzzy sets for each input variable: $\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, y_1$ and y_2 respectively; u_k with $k = 1, 2$ is one control input. The fuzzy sets are parameterized as in (9).

Thus, the overall controller consist of four controllers, each one consists of two first order T-S FLC, and each T-S FLC provides one input control. The overall scheme of proposed controller is presented in Fig. 2. The closed-loop system is shown in Fig. 3, the signal defined as $h_0(\theta)$ are the quantities to control, which represent to the first term of each element of the vector defined in (15).

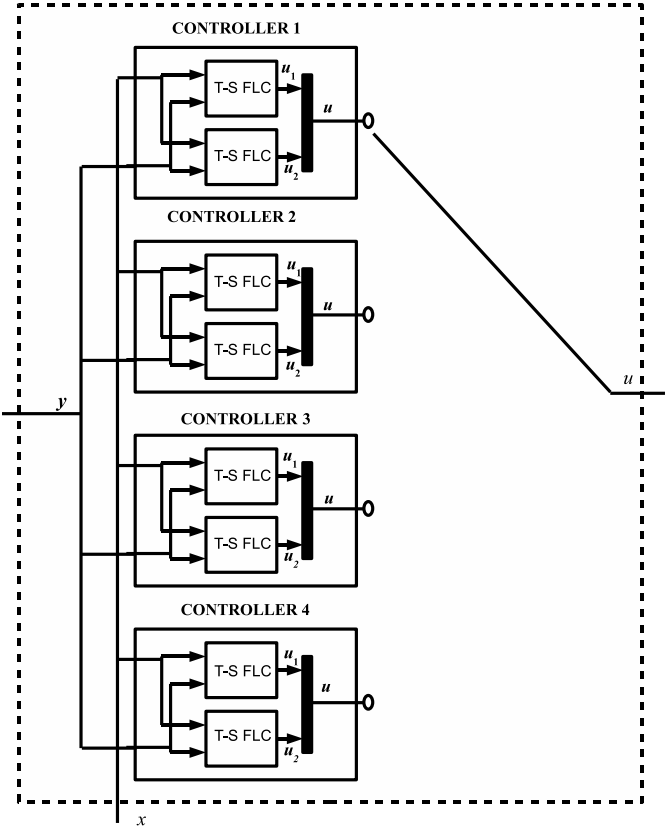


Fig. 2. Controller Structure

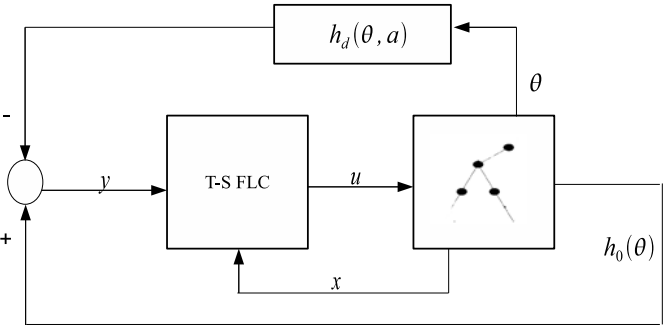


Fig. 3. Closed-loop system block diagram

The control input $u = [u_1^c, u_2^c]^T$ is defined through this function:

$$u(x, y, t) := \begin{cases} u_1^1 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & , 0 \leq t < t_1 \\ u_2^1 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & \\ u_1^2 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & \\ u_2^2 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & , t_1 \leq t < t_2 \\ u_1^3 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & \\ u_2^3 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & , t_2 \leq t < t_3 \\ u_1^4 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & \\ u_2^4 = \frac{\sum_i w_i G_i(x, y; p)}{\sum_i w_i} & , t_3 \leq t \leq t_4 \end{cases} \quad (18)$$

where $x = [\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3]$, $y = [y_1, y_2]$, $i = 1, 2, \dots, l_c$, l_c is the number of rules in each controller; u_k^c denotes the k th control input provided by c th controller and represents the weight average of rules activation within antecedents projected on the consequents through a time interval; w_i is the firing strength of each rule as in (10); $G(x, y, p)$ is the consequent of the each rule which takes the form as in (12) and p are the consequents parameters of each rule provided by the learning algorithm. The total rule base of the controller comprises 2048 if-then rules, considering that the universe of discourse of each input variable is represented through two membership functions. The parameters which describe each T-S FLC are provided in Section 6.

5 Simulation Results

The numerical verification of the closed-loop system was performed in MatLab, in a Pentium IV computer with 512 MB of RAM, using the software implementation of the biped robot available in [1]. The physical parameters of the biped model are shown in Table 1, and the impact occurs when $\theta_1^d = \pi/8$. The time values that define the interval in (18) are shown in Table 3. The parameters of the output (15)-(16) are given in Table 2. The matrices of the equations (2)-(3) which describe the mathematical model of the biped robot are given at Section 6.

In Fig. 4 to 7 show the state trajectories, output trajectories and input control of closed-loop simulation where the biped robot to perform one step.

Table 1. Biped model parameters

Parameter	Value
Mass of Hips (M_H)	15 kg
Mass of Torso (M_T)	10 kg
Mass of Legs (m)	5 kg
Length of legs (r)	1 m
Length of torso (l)	0.5 m

Table 2. Parameters of the output definition y in (16)

a_1^0	a_1^1	a_1^2	a_1^3	a_2^0	a_2^1	a_2^2	a_2^3
0.512	0.0730	0.0350	-0.8190	-2.2700	3.2600	3.1100	1.8900

Table 3. Time values of the control law definition in (18)

t_1	t_2	t_3	t_4
0.098	0.383	0.980	1.121

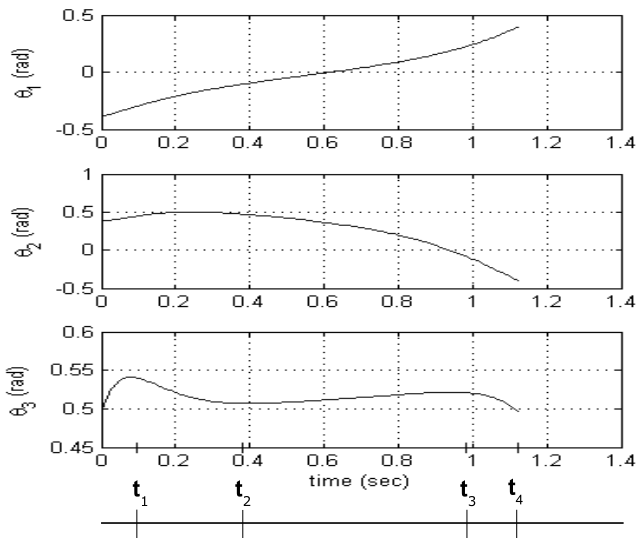


Fig. 4. Joint position trajectories

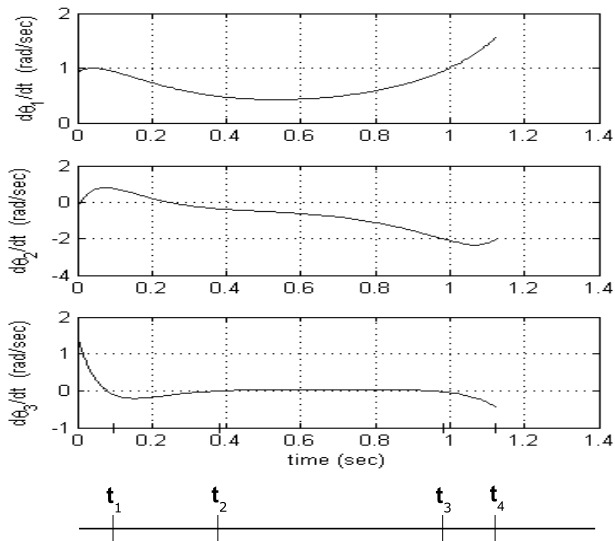


Fig. 5. Joint velocities trajectories

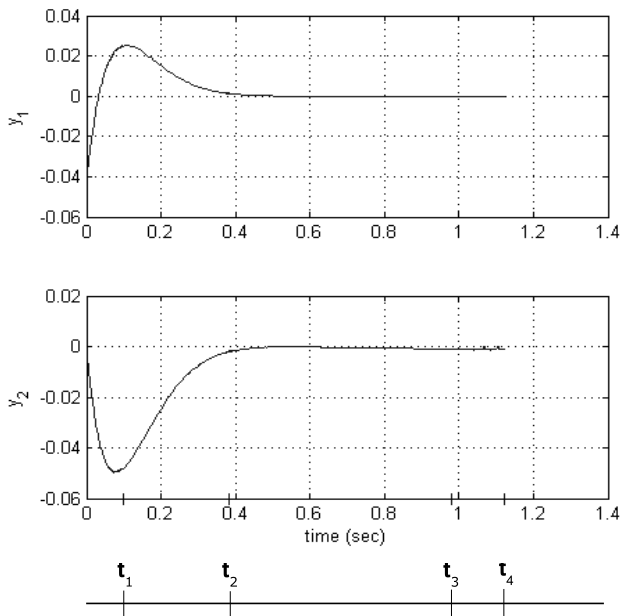


Fig. 6. Output trajectories

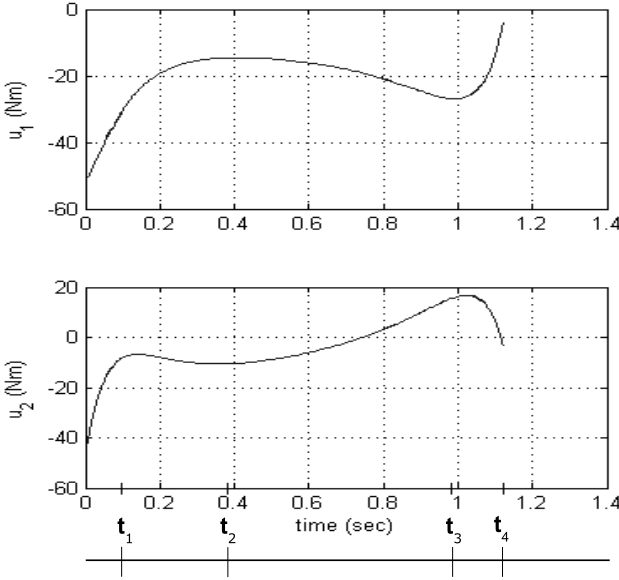


Fig. 7. Control signals

6 Conclusions

In this paper a T-S Fuzzy Logic Controller design to generate walking motions is presented. The control strategy consider an output function imposed in the feedback and several ANFIS are used to adapt the parameters of T-S FLCs such that fit a set of input-output data pairs which represent the evolution of the state variables and the output function during a walking cycle. An intelligent control law as switching function is proposed whose objective is to drive the quantities to control $h_0(\theta)$ to track the output function. The simulations results demonstrate that this method can be applied to solve the problem, its effectiveness is related to the capabilities to perform the data fitting and at this time, the controller is ad hoc for the output function and biped mathematical model parameters presented within the paper.

Appendix 1

Model Details

This section completes the equations of the biped model (2) - (3). In the following, this is the notation used:

$$s_{1j} = \sin(\theta_1 - \theta_j), j \in \{2, 3\}$$

$$c_{1j} = \cos(\theta_1 - \theta_j), j \in \{2, 3\}$$

Swing phase Model

$$M = \begin{bmatrix} (\frac{5}{4}m + M_H + M_T)r^2 & -\frac{1}{2}mr^2c_{12} & M_Trlc_{13} \\ -\frac{1}{2}mr^2c_{12} & \frac{1}{4}mr^2 & 0 \\ M_Trlc_{13} & 0 & M_Tl^2 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & -\frac{1}{2}mr^2s_{12}\dot{\theta}_2 & M_Trls_{13}\dot{\theta}_3 \\ \frac{1}{2}mr^2s_{12}\dot{\theta}_1 & 0 & 0 \\ -M_Trls_{13}\dot{\theta}_1 & 0 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} -\frac{1}{2}g(2M_H + 3m + 2M_T)r \sin(\theta_1) \\ \frac{1}{2}gmr \sin(\theta_2) \\ -gM_Tl \sin(\theta_3) \end{bmatrix}$$

$$B = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{bmatrix}$$

Impact Model

The inertia matrix M_e has the entries:

$$M_e^{11} = \left(\frac{5}{4}m + M_H + M_T\right)r^2$$

$$M_e^{12} = -\frac{1}{2}mr^2c_{12}$$

$$M_e^{13} = M_Trlc_{13}$$

$$M_e^{14} = \left(\frac{3}{2}m + M_H + M_T\right)r\cos(\theta_1)$$

$$M_e^{15} = -\left(\frac{3}{2}m + M_H + M_T\right)r\sin(\theta_1)$$

$$M_e^{22} = \frac{1}{4}mr^2$$

$$M_e^{24} = -\frac{1}{2}mr\cos(\theta_2)$$

$$M_e^{25} = \frac{1}{2}mr\sin(\theta_2)$$

$$M_e^{33} = M_Tl^2$$

$$M_e^{34} = M_Tl\cos(\theta_3)$$

$$M_e^{35} = -M_Tl\sin(\theta_3)$$

$$M_e^{44} = 2m + M_H + M_T$$

$$M_e^{55} = 2m + M_H + M_T$$

Appendix 2

T-S Fuzzy Controllers Parameters

Antecedent Parameters

This subsection Tables 4 to 7 show the membership functions parameters of each of the controller explained in the section 4.

Table 4. Membership function parameters of first Controller

Input	Fuzzy Set	T-S FLC 1 Parameters		T-S FLC 2 Parameters	
		a	c	a	c
θ_1	$A_{1,1}$	0.04179	-0.38862	0.04136	-0.39037
	$A_{1,2}$	0.01850	-0.28397	0.01845	-0.28763
θ_2	$A_{2,1}$	0.02619	0.39265	0.01917	0.39171
	$A_{2,2}$	0.00797	0.45614	0.00766	0.45421
θ_3	$A_{3,1}$	0.00331	0.47294	0.02037	0.49516
	$A_{3,2}$	0.03138	0.53158	-0.00288	0.55305
θ_1	$A_{1,1}$	0.02612	0.90943	0.02432	0.92174
	$A_{1,2}$	0.04841	0.97931	0.04458	0.98509
θ_2	$A_{2,1}$	0.43232	-0.24231	0.43764	-0.23986
	$A_{2,2}$	0.43512	0.79588	0.43810	0.79484
θ_3	$A_{3,1}$	0.67073	-0.10894	0.67329	-0.10776
	$A_{3,2}$	0.67316	1.48427	0.67532	1.48346
y_1	$A_{4,1}$	0.01973	-0.04125	0.03504	-0.03530
	$A_{4,2}$	0.01592	0.03541	0.01244	0.04119
y_2	$A_{5,1}$	0.03453	-0.03573	-0.00024	-0.06683
	$A_{5,2}$	0.01242	0.02543	0.02626	-0.00515

Table 5. Membership function parameters of second Controller

Input	Fuzzy Set	T-S FLC 1 Parameters		T-S FLC 2 Parameters	
		a	c	a	c
θ_1	$A_{1,1}$	0.11912	-0.26805	0.04926	-0.31383
	$A_{1,2}$	0.04469	-0.05091	0.07628	-0.10665
θ_2	$A_{2,1}$	0.00209	0.43179	0.02466	0.43144
	$A_{2,2}$	0.03183	0.49398	0.02314	0.49563
θ_3	$A_{3,1}$	0.01710	0.51069	0.01704	0.50994
	$A_{3,2}$	0.00148	0.55181	-0.00160	0.54978
θ_1	$A_{1,1}$	0.15245	0.45236	0.19035	0.47829
	$A_{1,2}$	0.19932	0.92833	0.18665	0.94280
θ_2	$A_{2,1}$	0.46471	-0.36901	0.47436	-0.36670
	$A_{2,2}$	0.46829	0.76003	0.47100	0.75723
θ_3	$A_{3,1}$	0.04614	-0.23211	0.09284	-0.19973
	$A_{3,2}$	0.08043	-0.01404	0.07787	0.00310
y_1	$A_{4,1}$	-0.00075	-0.00595	0.01248	0.00329
	$A_{4,2}$	0.01384	0.02182	-0.00078	0.03306
y_2	$A_{5,1}$	0.01796	-0.08025	0.01307	-0.06672
	$A_{5,2}$	0.00909	-0.02957	0.01867	-0.00895

Table 6. Membership function parameters of third Controller

Input	Fuzzy Set	T-S FLC 1 Parameters		T-S FLC 2 Parameters	
		a	c	a	c
θ_1	$A_{1,1}$	0.13746	-0.10592	0.13768	-0.10662
	$A_{1,2}$	0.13887	0.22189	0.14065	0.22035
θ_2	$A_{2,1}$	0.23348	-0.07858	0.23411	-0.07802
	$A_{2,2}$	0.23356	0.47241	0.23311	0.47289
θ_3	$A_{3,1}$	0.00812	0.50028	0.00461	0.49942
	$A_{3,2}$	0.01100	0.51772	0.01351	0.51900
$\dot{\theta}_1$	$A_{1,1}$	0.22045	0.42482	0.21962	0.42427
	$A_{1,2}$	0.21908	0.94333	0.21921	0.94308
$\dot{\theta}_2$	$A_{2,1}$	0.69502	-2.00380	0.69503	-2.00372
	$A_{2,2}$	0.69512	-0.36665	0.69490	-0.36648
$\dot{\theta}_3$	$A_{3,1}$	0.02408	-0.03254	0.02631	-0.03178
	$A_{3,2}$	0.03000	0.03532	0.02974	0.03535
y_1	$A_{4,1}$	0.00061	0.00626	-0.00204	0.01551
	$A_{4,2}$	-0.00696	0.00520	-0.00870	0.00277
y_2	$A_{5,1}$	0.00253	-0.01839	-0.00363	-0.00373
	$A_{5,2}$	0.00774	0.00191	-6.53×10^{-5}	0.00076

Table 7. Membership function parameters of fourth Controller

Input	Fuzzy Set	T-S FLC 1 Parameters		T-S FLC 2 Parameters	
		a	c	a	c
θ_1	$A_{1,1}$	0.07507	0.22460	0.08552	0.23442
	$A_{1,2}$	0.07136	0.39446	0.03315	0.41950
θ_2	$A_{2,1}$	0.13168	-0.39218	0.12745	-0.39340
	$A_{2,2}$	0.12964	-0.07554	0.12509	-0.07315
θ_3	$A_{3,1}$	-0.01008	0.48029	0.00837	0.48193
	$A_{3,2}$	0.01417	0.51326	0.00808	0.50781
$\dot{\theta}_1$	$A_{1,1}$	0.25863	0.94355	0.25958	0.94432
	$A_{1,2}$	0.25689	1.55087	0.25120	1.55362
$\dot{\theta}_2$	$A_{2,1}$	0.14937	-2.34638	0.15849	-2.34355
	$A_{2,2}$	0.14704	-2.00387	0.14260	-2.00340
$\dot{\theta}_3$	$A_{3,1}$	0.16569	-0.43464	0.14283	-0.44816
	$A_{3,2}$	0.17196	-0.03286	0.17654	-0.03871
y_1	$A_{4,1}$	-0.00826	-0.00474	-0.00815	-0.00371
	$A_{4,2}$	-0.00034	-0.00333	-0.00145	-0.00814
y_2	$A_{5,1}$	-0.00932	0.00284	0.00288	0.00011
	$A_{5,2}$	-0.00032	0.00309	0.00189	0.01339

Consequent Parameters

This subsection presents the consequent parameters of the controller explained in the section 4. In Tables 8 to 13 provide the consequent parameters of 151 rules of the first controller.

Table 8. Consequent parameters of the Controller 1 - T-S FLC 1 (1 of 3)

R^l	$P_{1,1}$	$P_{1,2}$	$P_{1,3}$	$P_{1,4}$	$P_{1,5}$
R^1	-2.36×10^{-10}	2.36×10^{-10}	2.98×10^{-10}	5.55×10^{-10}	-1.48×10^{-10}
R^2	-5.26×10^{-11}	5.26×10^{-11}	6.65×10^{-11}	1.24×10^{-10}	-3.29×10^{-11}
R^3	-1.55×10^{-15}	1.55×10^{-15}	1.95×10^{-15}	3.64×10^{-15}	-9.93×10^{-16}
R^4	-3.60×10^{-16}	3.60×10^{-16}	4.54×10^{-16}	8.47×10^{-16}	-2.28×10^{-16}
R^5	-4.06×10^{-09}	4.07×10^{-09}	5.13×10^{-09}	9.57×10^{-09}	-2.54×10^{-09}
R^6	-9.00×10^{-10}	9.00×10^{-10}	1.14×10^{-09}	2.12×10^{-09}	-5.61×10^{-10}
R^7	-2.71×10^{-14}	2.71×10^{-14}	3.42×10^{-14}	6.38×10^{-14}	-1.73×10^{-14}
R^8	-6.23×10^{-15}	6.23×10^{-15}	7.87×10^{-15}	1.47×10^{-14}	-3.93×10^{-15}
R^9	-1.33×10^{-11}	1.33×10^{-11}	1.68×10^{-11}	3.13×10^{-11}	-8.40×10^{-12}
R^{10}	-2.30×10^{-12}	3.00×10^{-12}	3.79×10^{-12}	7.07×10^{-12}	-1.88×10^{-12}
R^{11}	-8.42×10^{-17}	8.42×10^{-17}	1.06×10^{-16}	1.98×10^{-16}	-5.47×10^{-17}
R^{12}	-2.00×10^{-17}	2.00×10^{-17}	2.53×10^{-17}	4.72×10^{-17}	-1.28×10^{-17}
R^{13}	-2.30×10^{-10}	2.30×10^{-10}	2.90×10^{-10}	5.41×10^{-10}	-1.45×10^{-10}
R^{14}	-5.15×10^{-11}	5.15×10^{-11}	6.51×10^{-11}	1.21×10^{-10}	-3.22×10^{-11}
R^{15}	-1.49×10^{-15}	1.49×10^{-15}	1.88×10^{-15}	3.51×10^{-15}	-9.61×10^{-16}
R^{16}	-3.49×10^{-16}	3.49×10^{-16}	4.41×10^{-16}	8.23×10^{-16}	-2.22×10^{-16}
R^{17}	-1.57×10^{-10}	1.57×10^{-10}	1.98×10^{-10}	3.69×10^{-10}	-9.91×10^{-11}
R^{18}	-3.54×10^{-11}	3.54×10^{-11}	4.47×10^{-11}	8.34×10^{-11}	-2.22×10^{-11}
R^{19}	-9.98×10^{-16}	9.97×10^{-16}	1.26×10^{-15}	2.35×10^{-15}	-6.48×10^{-16}
R^{20}	-2.37×10^{-16}	2.37×10^{-16}	2.99×10^{-16}	5.58×10^{-16}	-1.51×10^{-16}
R^{21}	-2.71×10^{-09}	2.71×10^{-09}	3.42×10^{-09}	6.39×10^{-09}	-1.71×10^{-09}
R^{22}	-6.07×10^{-10}	6.07×10^{-10}	7.67×10^{-10}	1.43×10^{-09}	-3.79×10^{-10}
R^{23}	-1.76×10^{-14}	1.76×10^{-14}	2.23×10^{-14}	4.15×10^{-14}	-1.14×10^{-14}
R^{24}	-4.13×10^{-15}	4.12×10^{-15}	5.21×10^{-15}	9.72×10^{-15}	-2.62×10^{-15}
R^{25}	-8.75×10^{-12}	8.74×10^{-12}	1.10×10^{-11}	2.06×10^{-11}	-5.57×10^{-12}
R^{26}	-2.00×10^{-12}	2.00×10^{-12}	2.53×10^{-12}	4.72×10^{-12}	-1.26×10^{-12}
R^{27}	-5.35×10^{-17}	5.34×10^{-17}	6.74×10^{-17}	1.26×10^{-16}	-3.52×10^{-17}
R^{28}	-1.30×10^{-17}	1.30×10^{-17}	1.65×10^{-17}	3.07×10^{-17}	-8.40×10^{-18}
R^{29}	-1.52×10^{-10}	1.52×10^{-10}	1.92×10^{-10}	3.59×10^{-10}	-9.65×10^{-11}
R^{30}	-3.45×10^{-11}	3.45×10^{-11}	4.36×10^{-11}	8.14×10^{-11}	-2.17×10^{-11}
R^{31}	-9.57×10^{-16}	9.56×10^{-16}	1.21×10^{-15}	2.25×10^{-15}	-6.24×10^{-16}
R^{32}	-2.29×10^{-16}	2.29×10^{-16}	2.89×10^{-16}	5.40×10^{-16}	-1.47×10^{-16}
R^{33}	3.01×10^{-01}	-3.04×10^{-01}	-3.89×10^{-01}	-7.27×10^{-01}	9.32×10^{-02}
R^{34}	1.65×10^{-01}	-1.63×10^{-01}	-2.02×10^{-01}	-3.75×10^{-01}	1.82×10^{-01}
R^{35}	7.07×10^{-02}	-9.53×10^{-02}	-1.20×10^{-01}	-2.14×10^{-01}	-1.65×10^{-01}
R^{36}	9.21×10^{-07}	-8.66×10^{-07}	-1.02×10^{-06}	-1.88×10^{-06}	2.24×10^{-06}
R^{37}	4.66×10^{-00}	-4.66×10^{-00}	-5.80×10^{-00}	-10.8×10^{-00}	4.25×10^{-00}
R^{38}	4.42×10^{-00}	-4.40×10^{-00}	-5.52×10^{-00}	-10.3×10^{-00}	3.57×10^{-00}
R^{39}	1.38×10^{-02}	-1.66×10^{-02}	-2.14×10^{-02}	-3.92×10^{-02}	-2.38×10^{-02}
R^{40}	1.89×10^{-05}	-1.91×10^{-05}	-2.50×10^{-05}	-4.70×10^{-05}	-2.90×10^{-06}
R^{41}	9.66×10^{-02}	-1.06×10^{-01}	-1.39×10^{-01}	-2.60×10^{-01}	-1.11×10^{-01}
R^{42}	8.36×10^{-04}	-8.38×10^{-04}	-9.49×10^{-04}	-1.71×10^{-03}	2.20×10^{-03}
R^{43}	9.81×10^{-01}	-1.32×10^{-00}	-1.66×10^{-00}	-2.98×10^{-00}	-2.31×10^{-00}
R^{44}	1.55×10^{-07}	-2.21×10^{-07}	-3.18×10^{-07}	-5.99×10^{-07}	-1.15×10^{-06}
R^{45}	3.71×10^{-01}	-3.77×10^{-01}	-4.81×10^{-01}	-8.99×10^{-01}	9.85×10^{-01}
R^{46}	9.93×10^{-02}	-9.79×10^{-02}	-1.19×10^{-01}	-2.21×10^{-01}	1.42×10^{-01}
R^{47}	1.27×10^{-01}	-1.64×10^{-01}	-2.09×10^{-01}	-3.79×10^{-01}	-2.87×10^{-01}
R^{48}	-1.17×10^{-06}	1.24×10^{-06}	1.77×10^{-06}	3.41×10^{-06}	3.24×10^{-06}
R^{49}	2.46×10^{-00}	-2.62×10^{-00}	-3.46×10^{-00}	-6.49×10^{-00}	-2.28×10^{-00}
R^{50}	3.98×10^{-02}	-3.98×10^{-02}	-4.93×10^{-02}	-9.14×10^{-02}	4.05×10^{-02}
R^{51}	2.25×10^{-00}	-2.65×10^{-00}	-3.45×10^{-00}	-6.36×10^{-00}	-3.84×10^{-00}

Table 9. Consequent parameters of the Controller 1 - T-S FLC 1 (cont. 1 of 3)

R^l	$p_{1,6}$	$p_{1,7}$	$p_{1,8}$	$p_{1,9}$
R^1	8.95×10^{-10}	-2.53×10^{-11}	2.17×10^{-13}	6.00×10^{-10}
R^2	2.00×10^{-10}	-5.65×10^{-12}	3.65×10^{-14}	1.34×10^{-10}
R^3	5.89×10^{-15}	-1.67×10^{-16}	2.40×10^{-18}	3.93×10^{-15}
R^4	1.37×10^{-15}	-3.88×10^{-17}	4.06×10^{-19}	9.15×10^{-16}
R^5	1.54×10^{-08}	-4.36×10^{-10}	3.35×10^{-12}	1.03×10^{-08}
R^6	3.41×10^{-09}	-9.66×10^{-11}	5.62×10^{-13}	2.29×10^{-09}
R^7	1.03×10^{-13}	-2.93×10^{-15}	3.70×10^{-17}	6.90×10^{-14}
R^8	2.37×10^{-14}	-6.71×10^{-16}	6.26×10^{-18}	1.59×10^{-14}
R^9	5.04×10^{-11}	-1.43×10^{-12}	1.46×10^{-14}	3.38×10^{-11}
R^{10}	1.14×10^{-11}	-3.22×10^{-13}	2.47×10^{-15}	7.63×10^{-12}
R^{11}	3.22×10^{-16}	-9.15×10^{-18}	1.60×10^{-19}	2.14×10^{-16}
R^{12}	7.63×10^{-17}	-2.16×10^{-18}	2.73×10^{-20}	5.10×10^{-17}
R^{13}	8.72×10^{-10}	-2.47×10^{-11}	2.25×10^{-13}	5.85×10^{-10}
R^{14}	1.95×10^{-10}	-5.53×10^{-12}	3.79×10^{-14}	1.31×10^{-10}
R^{15}	5.68×10^{-15}	-1.62×10^{-16}	2.49×10^{-18}	3.79×10^{-15}
R^{16}	1.33×10^{-15}	-3.77×10^{-17}	4.21×10^{-19}	8.89×10^{-16}
R^{17}	5.95×10^{-10}	-1.69×10^{-11}	1.70×10^{-13}	3.99×10^{-10}
R^{18}	1.34×10^{-10}	-3.80×10^{-12}	2.86×10^{-14}	9.00×10^{-11}
R^{19}	3.81×10^{-15}	-1.08×10^{-16}	1.86×10^{-18}	2.54×10^{-15}
R^{20}	9.01×10^{-16}	-2.58×10^{-17}	3.17×10^{-19}	6.03×10^{-16}
R^{21}	1.03×10^{-08}	-2.92×10^{-10}	2.61×10^{-12}	6.90×10^{-09}
R^{22}	2.30×10^{-09}	-6.52×10^{-11}	4.40×10^{-13}	1.55×10^{-09}
R^{23}	6.72×10^{-14}	-1.91×10^{-15}	2.88×10^{-17}	4.49×10^{-14}
R^{24}	1.57×10^{-14}	-4.45×10^{-16}	4.89×10^{-18}	1.05×10^{-14}
R^{25}	3.33×10^{-11}	-9.44×10^{-13}	1.14×10^{-14}	2.23×10^{-11}
R^{26}	7.61×10^{-12}	-2.16×10^{-13}	1.93×10^{-15}	5.10×10^{-12}
R^{27}	2.05×10^{-16}	-5.83×10^{-18}	1.24×10^{-19}	1.36×10^{-16}
R^{28}	4.97×10^{-17}	-1.41×10^{-18}	2.12×10^{-20}	3.32×10^{-17}
R^{29}	5.79×10^{-10}	-1.64×10^{-11}	1.76×10^{-13}	3.88×10^{-10}
R^{30}	1.31×10^{-10}	-3.71×10^{-12}	2.97×10^{-14}	8.79×10^{-11}
R^{31}	3.66×10^{-15}	-1.04×10^{-16}	1.92×10^{-18}	2.44×10^{-15}
R^{32}	8.73×10^{-16}	-2.48×10^{-17}	3.28×10^{-19}	5.83×10^{-16}
R^{33}	-1.05×10^{-00}	2.76×10^{-02}	4.29×10^{-03}	-7.75×10^{-01}
R^{34}	-6.98×10^{-01}	2.15×10^{-02}	-3.83×10^{-03}	-4.14×10^{-01}
R^{35}	-1.65×10^{-02}	-4.40×10^{-03}	1.05×10^{-02}	-2.22×10^{-01}
R^{36}	-5.10×10^{-06}	1.82×10^{-07}	-8.01×10^{-08}	-2.21×10^{-06}
R^{37}	-18.8×10^{-00}	5.64×10^{-01}	-6.52×10^{-02}	-11.8×10^{-00}
R^{38}	-17.5×10^{-00}	5.15×10^{-01}	-4.14×10^{-02}	-11.2×10^{-00}
R^{39}	-1.60×10^{-02}	-2.96×10^{-04}	1.59×10^{-03}	-4.01×10^{-02}
R^{40}	-5.86×10^{-05}	1.32×10^{-06}	6.74×10^{-07}	-4.90×10^{-05}
R^{41}	-1.90×10^{-01}	1.46×10^{-03}	8.20×10^{-03}	-2.66×10^{-01}
R^{42}	-4.61×10^{-03}	1.69×10^{-04}	-7.82×10^{-05}	-2.06×10^{-03}
R^{43}	-1.98×10^{-01}	-6.28×10^{-02}	1.47×10^{-01}	-3.08×10^{-00}
R^{44}	6.88×10^{-07}	-4.82×10^{-08}	6.03×10^{-08}	-5.31×10^{-07}
R^{45}	-1.28×10^{-00}	3.31×10^{-02}	6.10×10^{-03}	-9.57×10^{-01}
R^{46}	-4.50×10^{-01}	1.46×10^{-02}	-3.82×10^{-03}	-2.48×10^{-01}
R^{47}	-5.58×10^{-02}	-6.95×10^{-03}	1.83×10^{-02}	-3.88×10^{-01}
R^{48}	8.12×10^{-07}	6.76×10^{-08}	-1.86×10^{-07}	3.23×10^{-06}
R^{49}	-5.57×10^{-00}	7.03×10^{-02}	1.81×10^{-01}	-6.63×10^{-00}
R^{50}	-1.65×10^{-01}	5.01×10^{-03}	-7.52×10^{-04}	-1.01×10^{-01}
R^{51}	-2.81×10^{-00}	-4.16×10^{-02}	2.56×10^{-01}	-6.48×10^{-00}

Table 10. Consequent parameters of the Controller 1 - T-S FLC 1 (2 of 3)

R^l	$p_{1,1}$	$p_{1,2}$	$p_{1,3}$	$p_{1,4}$	$p_{1,5}$
R^{52}	2.05×10^{-05}	-2.16×10^{-05}	-2.89×10^{-05}	-5.45×10^{-05}	-2.30×10^{-05}
R^{53}	5.85×10^{-00}	-6.08×10^{-00}	-7.87×10^{-00}	-14.7×10^{-00}	-1.60×10^{-00}
R^{54}	1.44×10^{-00}	-1.43×10^{-00}	-1.76×10^{-00}	-3.27×10^{-00}	1.62×10^{-00}
R^{55}	1.05×10^{-00}	-1.20×10^{-00}	-1.58×10^{-00}	-2.95×10^{-00}	-1.83×10^{-00}
R^{56}	-2.44×10^{-06}	3.36×10^{-07}	-2.68×10^{-08}	2.09×10^{-07}	-2.93×10^{-05}
R^{57}	5.32×10^{-00}	-5.98×10^{-00}	-7.88×10^{-00}	-14.7×10^{-00}	-8.07×10^{-00}
R^{58}	3.09×10^{-03}	-3.07×10^{-03}	-3.91×10^{-03}	-7.31×10^{-03}	1.66×10^{-03}
R^{59}	4.43×10^{-00}	-7.27×10^{-00}	-7.81×10^{-00}	-12.49×10^{-00}	1.33×10^{-00}
R^{60}	1.48×10^{-05}	-1.60×10^{-05}	-2.06×10^{-05}	-3.83×10^{-05}	-7.89×10^{-06}
R^{61}	6.12×10^{-00}	-6.52×10^{-00}	-8.58×10^{-00}	-16.1×10^{-00}	-5.52×10^{-00}
R^{62}	3.38×10^{-02}	-3.41×10^{-02}	-4.34×10^{-02}	-8.10×10^{-02}	1.42×10^{-02}
R^{63}	6.65×10^{-00}	-7.32×10^{-00}	-9.77×10^{-00}	-18.3×10^{-00}	-10.4×10^{-00}
R^{64}	5.77×10^{-05}	-5.97×10^{-05}	-7.90×10^{-05}	-1.49×10^{-04}	-3.97×10^{-05}
R^{65}	-4.19×10^{-24}	4.18×10^{-24}	5.29×10^{-24}	9.87×10^{-24}	-2.62×10^{-24}
R^{66}	-9.24×10^{-25}	9.24×10^{-25}	1.17×10^{-24}	2.18×10^{-24}	-5.74×10^{-25}
R^{67}	-2.83×10^{-29}	2.83×10^{-29}	3.57×10^{-29}	6.67×10^{-29}	-1.80×10^{-29}
R^{68}	-6.45×10^{-30}	6.45×10^{-30}	8.15×10^{-30}	1.52×10^{-29}	-4.06×10^{-30}
R^{69}	-7.17×10^{-23}	7.17×10^{-23}	9.06×10^{-23}	1.69×10^{-22}	-4.47×10^{-23}
R^{70}	-1.58×10^{-23}	1.58×10^{-23}	1.99×10^{-23}	3.72×10^{-23}	-9.78×10^{-24}
R^{71}	-4.92×10^{-28}	4.92×10^{-28}	6.21×10^{-28}	1.16×10^{-27}	-3.11×10^{-28}
R^{72}	-1.11×10^{-28}	1.11×10^{-28}	1.40×10^{-28}	2.62×10^{-28}	-6.96×10^{-29}
R^{73}	-2.38×10^{-25}	2.38×10^{-25}	3.00×10^{-25}	5.60×10^{-25}	-1.49×10^{-25}
R^{74}	-5.30×10^{-26}	5.30×10^{-26}	6.70×10^{-26}	1.23×10^{-25}	-3.31×10^{-26}
R^{75}	-1.57×10^{-30}	1.57×10^{-30}	1.98×10^{-30}	3.69×10^{-30}	-1.01×10^{-30}
R^{76}	-3.64×10^{-31}	3.63×10^{-31}	4.59×10^{-31}	8.57×10^{-31}	-2.30×10^{-31}
R^{77}	-4.09×10^{-24}	4.09×10^{-24}	5.17×10^{-24}	9.64×10^{-24}	-2.56×10^{-24}
R^{78}	-9.06×10^{-25}	9.06×10^{-25}	1.14×10^{-24}	2.14×10^{-24}	-5.64×10^{-25}
R^{79}	-2.74×10^{-29}	2.74×10^{-29}	3.46×10^{-29}	6.46×10^{-29}	-1.75×10^{-29}
R^{80}	-6.29×10^{-30}	6.29×10^{-30}	7.94×10^{-30}	1.48×10^{-29}	-3.96×10^{-30}
R^{81}	-2.80×10^{-24}	2.80×10^{-24}	3.54×10^{-24}	6.61×10^{-24}	-1.76×10^{-24}
R^{82}	-6.25×10^{-25}	6.24×10^{-25}	7.89×10^{-25}	1.47×10^{-24}	-3.89×10^{-25}
R^{83}	-1.85×10^{-29}	1.85×10^{-29}	2.34×10^{-29}	4.37×10^{-29}	-1.19×10^{-29}
R^{84}	-4.29×10^{-30}	4.29×10^{-30}	5.42×10^{-30}	1.01×10^{-29}	-2.71×10^{-30}
R^{85}	-4.82×10^{-23}	4.82×10^{-23}	6.09×10^{-23}	1.14×10^{-22}	-3.02×10^{-23}
R^{86}	-1.07×10^{-23}	1.07×10^{-23}	1.35×10^{-23}	2.52×10^{-23}	-6.64×10^{-24}
R^{87}	-3.24×10^{-28}	3.24×10^{-28}	4.09×10^{-28}	7.64×10^{-28}	-2.07×10^{-28}
R^{88}	-7.42×10^{-29}	7.42×10^{-29}	9.37×10^{-29}	1.75×10^{-28}	-4.67×10^{-29}
R^{89}	-1.58×10^{-25}	1.58×10^{-25}	2.00×10^{-25}	3.73×10^{-25}	-10.00×10^{-26}
R^{90}	-3.56×10^{-26}	3.56×10^{-26}	4.50×10^{-26}	8.40×10^{-26}	-2.23×10^{-26}
R^{91}	-1.02×10^{-30}	1.01×10^{-30}	1.28×10^{-30}	2.39×10^{-30}	-6.57×10^{-31}
R^{92}	-2.40×10^{-31}	2.40×10^{-31}	3.03×10^{-31}	5.65×10^{-31}	-1.53×10^{-31}
R^{93}	-2.74×10^{-24}	2.73×10^{-24}	3.45×10^{-24}	6.45×10^{-24}	-1.72×10^{-24}
R^{94}	-6.11×10^{-25}	6.11×10^{-25}	7.72×10^{-25}	1.44×10^{-24}	-3.82×10^{-25}
R^{95}	-1.79×10^{-29}	1.79×10^{-29}	2.26×10^{-29}	4.22×10^{-29}	-1.15×10^{-29}
R^{96}	-4.17×10^{-30}	4.17×10^{-30}	5.27×10^{-30}	9.83×10^{-30}	-2.64×10^{-30}
R^{97}	1.61×10^{-04}	-2.33×10^{-04}	-2.86×10^{-04}	-5.03×10^{-04}	-4.08×10^{-04}
R^{98}	1.99×10^{-12}	-2.66×10^{-12}	-3.37×10^{-12}	-6.11×10^{-12}	-5.21×10^{-12}
R^{99}	3.16×10^{-02}	-4.69×10^{-02}	-5.69×10^{-02}	-9.94×10^{-02}	-8.04×10^{-02}
R^{100}	3.72×10^{-10}	-4.89×10^{-10}	-6.23×10^{-10}	-1.14×10^{-09}	-9.88×10^{-10}
R^{101}	1.21×10^{-05}	-1.72×10^{-05}	-2.12×10^{-05}	-3.75×10^{-05}	-3.04×10^{-05}

Table 11. Consequent parameters of the Controller 1 - T-S FLC 1 (cont. 2 of 3)

R^l	$P_{1,6}$	$P_{1,7}$	$P_{1,8}$	$P_{1,9}$
R^{52}	-4.34×10^{-05}	4.09×10^{-07}	1.69×10^{-06}	-5.51×10^{-05}
R^{53}	-16.9×10^{-00}	3.59×10^{-01}	2.48×10^{-01}	-15.4×10^{-00}
R^{54}	-6.11×10^{-00}	1.89×10^{-01}	-3.49×10^{-02}	-3.61×10^{-00}
R^{55}	-1.40×10^{-00}	-1.77×10^{-02}	1.20×10^{-01}	-2.96×10^{-00}
R^{56}	3.96×10^{-05}	-1.76×10^{-06}	1.37×10^{-06}	2.41×10^{-06}
R^{57}	-8.41×10^{-00}	-2.37×10^{-02}	5.48×10^{-01}	-14.9×10^{-00}
R^{58}	-1.15×10^{-02}	3.20×10^{-04}	9.54×10^{-06}	-7.85×10^{-03}
R^{59}	-7.79×10^{-00}	1.72×10^{-01}	1.57×10^{-01}	-15.4×10^{-00}
R^{60}	-3.81×10^{-05}	6.84×10^{-07}	8.25×10^{-07}	-4.00×10^{-05}
R^{61}	-13.9×10^{-00}	1.81×10^{-01}	4.43×10^{-01}	-16.5×10^{-00}
R^{62}	-1.21×10^{-01}	3.27×10^{-03}	3.10×10^{-04}	-8.68×10^{-02}
R^{63}	-10.7×10^{-00}	-3.12×10^{-02}	6.95×10^{-01}	-18.4×10^{-00}
R^{64}	-1.48×10^{-04}	2.45×10^{-06}	3.54×10^{-06}	-1.53×10^{-04}
R^{65}	1.59×10^{-23}	-4.49×10^{-25}	3.15×10^{-27}	1.07×10^{-23}
R^{66}	3.50×10^{-24}	-9.91×10^{-26}	5.28×10^{-28}	2.35×10^{-24}
R^{67}	1.08×10^{-28}	-3.05×10^{-30}	3.49×10^{-32}	7.20×10^{-29}
R^{68}	2.45×10^{-29}	-6.94×10^{-31}	5.90×10^{-33}	1.64×10^{-29}
R^{69}	2.72×10^{-22}	-7.70×10^{-24}	4.84×10^{-26}	1.83×10^{-22}
R^{70}	5.97×10^{-23}	-1.69×10^{-24}	8.09×10^{-27}	4.01×10^{-23}
R^{71}	1.87×10^{-27}	-5.30×10^{-29}	5.38×10^{-31}	1.25×10^{-27}
R^{72}	4.22×10^{-28}	-1.19×10^{-29}	9.08×10^{-32}	2.83×10^{-28}
R^{73}	9.03×10^{-25}	-2.56×10^{-26}	2.12×10^{-28}	6.05×10^{-25}
R^{74}	2.01×10^{-25}	-5.69×10^{-27}	3.57×10^{-29}	1.35×10^{-25}
R^{75}	5.97×10^{-30}	-1.70×10^{-31}	2.34×10^{-33}	3.99×10^{-30}
R^{76}	1.38×10^{-30}	-3.92×10^{-32}	3.97×10^{-34}	9.25×10^{-31}
R^{77}	1.55×10^{-23}	-4.39×10^{-25}	3.27×10^{-27}	1.04×10^{-23}
R^{78}	3.44×10^{-24}	-9.72×10^{-26}	5.48×10^{-28}	2.31×10^{-24}
R^{79}	1.04×10^{-28}	-2.96×10^{-30}	3.61×10^{-32}	6.98×10^{-29}
R^{80}	2.39×10^{-29}	-6.77×10^{-31}	6.12×10^{-33}	1.60×10^{-29}
R^{81}	1.06×10^{-23}	-3.01×10^{-25}	2.46×10^{-27}	7.14×10^{-24}
R^{82}	2.37×10^{-24}	-6.70×10^{-26}	4.14×10^{-28}	1.59×10^{-24}
R^{83}	7.06×10^{-29}	-2.01×10^{-30}	2.72×10^{-32}	4.72×10^{-29}
R^{84}	1.63×10^{-29}	-4.62×10^{-31}	4.61×10^{-33}	1.09×10^{-29}
R^{85}	1.83×10^{-22}	-5.18×10^{-24}	3.79×10^{-26}	1.23×10^{-22}
R^{86}	4.05×10^{-23}	-1.14×10^{-24}	6.35×10^{-27}	2.72×10^{-23}
R^{87}	1.23×10^{-27}	-3.50×10^{-29}	4.20×10^{-31}	8.25×10^{-28}
R^{88}	2.82×10^{-28}	-7.99×10^{-30}	7.10×10^{-32}	1.89×10^{-28}
R^{89}	6.01×10^{-25}	-1.70×10^{-26}	1.66×10^{-28}	4.03×10^{-25}
R^{90}	1.35×10^{-25}	-3.83×10^{-27}	2.79×10^{-29}	9.07×10^{-26}
R^{91}	3.87×10^{-30}	-1.10×10^{-31}	1.82×10^{-33}	2.58×10^{-30}
R^{92}	9.12×10^{-31}	-2.59×10^{-32}	3.09×10^{-34}	6.10×10^{-31}
R^{93}	1.04×10^{-23}	-2.94×10^{-25}	2.55×10^{-27}	6.96×10^{-24}
R^{94}	2.33×10^{-24}	-6.56×10^{-26}	4.29×10^{-28}	1.56×10^{-24}
R^{95}	6.82×10^{-29}	-1.94×10^{-30}	2.81×10^{-32}	4.56×10^{-29}
R^{96}	1.59×10^{-29}	-4.50×10^{-31}	4.77×10^{-33}	1.06×10^{-29}
R^{97}	3.23×10^{-05}	-1.27×10^{-05}	2.58×10^{-05}	-5.29×10^{-04}
R^{98}	-3.25×10^{-14}	-1.48×10^{-13}	3.20×10^{-13}	-6.21×10^{-12}
R^{99}	9.58×10^{-03}	-2.61×10^{-03}	5.12×10^{-03}	-1.05×10^{-01}
R^{100}	-1.01×10^{-11}	-2.78×10^{-11}	6.02×10^{-11}	-1.15×10^{-09}
R^{101}	1.72×10^{-06}	-9.30×10^{-07}	1.92×10^{-06}	-3.92×10^{-05}

Table 12. Consequent parameters of the Controller 1 - T-S FLC 1 (3 of 3)

R^j	$p_{1,1}$	$p_{1,2}$	$p_{1,3}$	$p_{1,4}$	$p_{1,5}$
R^{102}	2.35×10^{-13}	-2.98×10^{-13}	-3.74×10^{-13}	-6.76×10^{-13}	-4.01×10^{-13}
R^{103}	2.13×10^{-03}	-3.12×10^{-03}	-3.80×10^{-03}	-6.65×10^{-03}	-5.36×10^{-03}
R^{104}	3.11×10^{-11}	-4.27×10^{-11}	-5.35×10^{-11}	-9.60×10^{-11}	-8.11×10^{-11}
R^{105}	2.47×10^{-04}	-3.58×10^{-03}	-4.38×10^{-04}	-7.71×10^{-03}	-6.21×10^{-03}
R^{106}	3.96×10^{-11}	-5.53×10^{-11}	-6.88×10^{-11}	-1.27×10^{-10}	-1.02×10^{-10}
R^{107}	4.53×10^{-01}	-6.78×10^{-01}	-8.19×10^{-01}	-1.43×10^{-00}	-1.14×10^{-00}
R^{108}	7.78×10^{-09}	-1.09×10^{-08}	-1.35×10^{-08}	-2.41×10^{-08}	-2.02×10^{-08}
R^{109}	1.93×10^{-04}	-2.73×10^{-04}	-3.37×10^{-04}	-5.97×10^{-04}	-4.83×10^{-04}
R^{110}	3.08×10^{-12}	-4.34×10^{-12}	-5.37×10^{-12}	-9.55×10^{-12}	-7.88×10^{-12}
R^{111}	3.10×10^{-02}	-4.53×10^{-02}	-5.52×10^{-02}	-9.68×10^{-02}	-7.73×10^{-02}
R^{112}	5.79×10^{-10}	-8.20×10^{-10}	-1.01×10^{-09}	-1.80×10^{-09}	-1.49×10^{-09}
R^{113}	1.15×10^{-03}	-1.61×10^{-03}	-2.00×10^{-03}	-3.56×10^{-03}	-2.96×10^{-03}
R^{114}	1.35×10^{-11}	-1.91×10^{-11}	-2.36×10^{-11}	-4.18×10^{-11}	-3.45×10^{-11}
R^{115}	1.96×10^{-01}	-2.69×10^{-01}	-3.36×10^{-01}	-6.02×10^{-01}	-4.98×10^{-01}
R^{116}	2.40×10^{-09}	-3.38×10^{-09}	-4.18×10^{-09}	-7.43×10^{-09}	-6.12×10^{-09}
R^{117}	8.52×10^{-05}	-1.22×10^{-04}	-1.50×10^{-04}	-2.65×10^{-04}	-2.21×10^{-04}
R^{118}	1.05×10^{-12}	-1.47×10^{-12}	-1.82×10^{-12}	-3.22×10^{-12}	-2.55×10^{-12}
R^{119}	1.67×10^{-02}	-2.32×10^{-02}	-2.88×10^{-02}	-5.15×10^{-02}	-4.28×10^{-02}
R^{120}	1.92×10^{-10}	-2.70×10^{-10}	-3.34×10^{-10}	-5.94×10^{-10}	-4.91×10^{-10}
R^{121}	2.12×10^{-02}	-2.97×10^{-02}	-3.69×10^{-02}	-6.57×10^{-02}	-5.47×10^{-02}
R^{122}	2.38×10^{-10}	-3.36×10^{-10}	-4.16×10^{-10}	-7.38×10^{-10}	-6.11×10^{-10}
R^{123}	3.66×10^{-00}	-5.00×10^{-00}	-6.26×10^{-00}	-11.2×10^{-00}	-9.31×10^{-00}
R^{124}	4.24×10^{-08}	-5.92×10^{-08}	-7.36×10^{-08}	-1.31×10^{-07}	-1.08×10^{-07}
R^{125}	1.55×10^{-03}	-2.22×10^{-03}	-2.73×10^{-03}	-4.83×10^{-03}	-4.03×10^{-03}
R^{126}	1.75×10^{-11}	-2.50×10^{-11}	-3.08×10^{-11}	-5.45×10^{-11}	-4.52×10^{-11}
R^{127}	3.12×10^{-01}	-4.33×10^{-01}	-5.39×10^{-01}	-9.63×10^{-01}	-8.01×10^{-01}
R^{128}	3.42×10^{-09}	-4.80×10^{-09}	-5.95×10^{-09}	-1.06×10^{-08}	-8.76×10^{-09}
R^{129}	-6.95×10^{-18}	6.95×10^{-18}	8.77×10^{-18}	1.64×10^{-17}	-4.41×10^{-18}
R^{130}	-1.58×10^{-18}	1.58×10^{-18}	2.00×10^{-18}	3.73×10^{-18}	-9.94×10^{-19}
R^{131}	-4.32×10^{-23}	4.32×10^{-23}	5.45×10^{-23}	1.02×10^{-22}	-2.83×10^{-23}
R^{132}	-1.04×10^{-23}	1.04×10^{-23}	1.32×10^{-23}	2.46×10^{-23}	-6.69×10^{-24}
R^{133}	-1.21×10^{-16}	1.21×10^{-16}	1.52×10^{-16}	2.84×10^{-16}	-7.63×10^{-17}
R^{134}	-2.72×10^{-17}	2.72×10^{-17}	3.44×10^{-17}	6.42×10^{-17}	-1.71×10^{-17}
R^{135}	-7.70×10^{-22}	7.69×10^{-22}	9.71×10^{-22}	1.81×10^{-21}	-4.99×10^{-22}
R^{136}	-1.83×10^{-22}	1.82×10^{-22}	2.30×10^{-22}	4.30×10^{-22}	-1.16×10^{-22}
R^{137}	-3.86×10^{-19}	3.85×10^{-19}	4.87×10^{-19}	9.08×10^{-19}	-2.47×10^{-19}
R^{138}	-8.92×10^{-20}	8.92×10^{-20}	1.13×10^{-19}	2.10×10^{-19}	-5.64×10^{-20}
R^{139}	-2.28×10^{-24}	2.28×10^{-24}	2.88×10^{-24}	5.37×10^{-24}	-1.52×10^{-24}
R^{140}	-5.69×10^{-25}	5.69×10^{-25}	7.18×10^{-25}	1.34×10^{-24}	-3.69×10^{-25}
R^{141}	-6.74×10^{-18}	6.74×10^{-18}	8.51×10^{-18}	1.59×10^{-17}	-4.29×10^{-18}
R^{142}	-1.54×10^{-18}	1.54×10^{-18}	1.95×10^{-18}	3.64×10^{-18}	-9.72×10^{-19}
R^{143}	-4.13×10^{-23}	4.12×10^{-23}	5.20×10^{-23}	9.71×10^{-23}	-2.72×10^{-23}
R^{144}	-1.01×10^{-23}	1.01×10^{-23}	1.27×10^{-23}	2.37×10^{-23}	-6.48×10^{-24}
R^{145}	-4.56×10^{-18}	4.56×10^{-18}	5.75×10^{-18}	1.07×10^{-17}	-2.92×10^{-18}
R^{146}	-1.05×10^{-18}	1.05×10^{-18}	1.33×10^{-18}	2.48×10^{-18}	-6.65×10^{-19}
R^{147}	-2.71×10^{-23}	2.71×10^{-23}	3.42×10^{-23}	6.38×10^{-23}	-1.80×10^{-23}
R^{148}	-6.74×10^{-24}	6.74×10^{-24}	8.51×10^{-24}	1.59×10^{-23}	-4.37×10^{-24}
R^{149}	-7.97×10^{-17}	7.96×10^{-17}	1.01×10^{-16}	1.88×10^{-16}	-5.07×10^{-17}
R^{150}	-1.82×10^{-17}	1.82×10^{-17}	2.30×10^{-17}	4.29×10^{-17}	-1.15×10^{-17}
R^{151}	-4.90×10^{-22}	4.89×10^{-22}	6.17×10^{-22}	1.15×10^{-21}	-3.22×10^{-22}

Table 13. Consequent parameters of the Controller 1 - T-S FLC 1 (cont. 3 of 3)

R^l	$P_{1,6}$	$P_{1,7}$	$P_{1,8}$	$P_{1,9}$
R^{102}	-2.35×10^{-13}	-5.98×10^{-15}	2.73×10^{-14}	-7.04×10^{-13}
R^{103}	5.05×10^{-04}	-1.70×10^{-04}	3.42×10^{-04}	-7.03×10^{-03}
R^{104}	2.34×10^{-12}	-2.39×10^{-12}	5.02×10^{-12}	-9.86×10^{-11}
R^{105}	4.37×10^{-04}	-1.93×10^{-04}	3.94×10^{-04}	-8.11×10^{-03}
R^{106}	4.52×10^{-12}	-3.07×10^{-12}	6.38×10^{-12}	-1.27×10^{-10}
R^{107}	1.43×10^{-01}	-3.72×10^{-02}	7.30×10^{-02}	-1.52×10^{-00}
R^{108}	1.14×10^{-09}	-6.13×10^{-10}	1.26×10^{-09}	-2.50×10^{-08}
R^{109}	1.96×10^{-05}	-1.46×10^{-05}	3.05×10^{-05}	-6.24×10^{-04}
R^{110}	3.68×10^{-13}	-2.38×10^{-13}	4.93×10^{-13}	-9.93×10^{-12}
R^{111}	6.43×10^{-03}	-2.43×10^{-03}	4.93×10^{-03}	-1.02×10^{-01}
R^{112}	9.48×10^{-11}	-4.56×10^{-11}	9.33×10^{-11}	-1.87×10^{-09}
R^{113}	1.36×10^{-04}	-8.89×10^{-05}	1.85×10^{-04}	-3.69×10^{-03}
R^{114}	1.89×10^{-12}	-1.05×10^{-12}	2.16×10^{-12}	-4.35×10^{-11}
R^{115}	7.18×10^{-04}	-1.44×10^{-02}	3.10×10^{-02}	-6.22×10^{-02}
R^{116}	2.06×10^{-10}	-1.82×10^{-10}	3.83×10^{-10}	-7.72×10^{-09}
R^{117}	1.74×10^{-05}	-6.85×10^{-06}	1.38×10^{-05}	-2.77×10^{-04}
R^{118}	1.05×10^{-14}	-7.44×10^{-14}	1.61×10^{-13}	-3.37×10^{-12}
R^{119}	1.16×10^{-03}	-1.26×10^{-03}	2.66×10^{-03}	-5.33×10^{-03}
R^{120}	2.19×10^{-11}	-1.48×10^{-11}	3.07×10^{-11}	-6.18×10^{-10}
R^{121}	2.63×10^{-03}	-1.65×10^{-03}	3.41×10^{-03}	-6.82×10^{-02}
R^{122}	3.41×10^{-11}	-1.86×10^{-11}	3.82×10^{-11}	-7.69×10^{-10}
R^{123}	-2.26×10^{-02}	-2.67×10^{-01}	5.78×10^{-01}	-11.57×10^{-00}
R^{124}	2.98×10^{-09}	-3.20×10^{-09}	6.75×10^{-09}	-1.36×10^{-07}
R^{125}	3.47×10^{-04}	-1.26×10^{-04}	2.52×10^{-04}	-5.04×10^{-04}
R^{126}	3.47×10^{-12}	-1.40×10^{-12}	2.83×10^{-12}	-5.70×10^{-11}
R^{127}	2.22×10^{-02}	-2.37×10^{-02}	4.99×10^{-02}	-9.96×10^{-01}
R^{128}	3.74×10^{-10}	-2.63×10^{-10}	5.47×10^{-10}	-1.10×10^{-08}
R^{129}	2.64×10^{-17}	-7.50×10^{-19}	8.42×10^{-21}	1.77×10^{-17}
R^{130}	6.01×10^{-18}	-1.70×10^{-19}	1.42×10^{-21}	4.03×10^{-18}
R^{131}	1.65×10^{-22}	-4.70×10^{-24}	9.20×10^{-26}	1.10×10^{-22}
R^{132}	3.97×10^{-23}	-1.13×10^{-24}	1.57×10^{-26}	2.65×10^{-23}
R^{133}	4.59×10^{-16}	-1.30×10^{-17}	1.30×10^{-19}	3.07×10^{-16}
R^{134}	1.03×10^{-16}	-2.93×10^{-18}	2.19×10^{-20}	6.93×10^{-17}
R^{135}	2.94×10^{-21}	-8.35×10^{-23}	1.42×10^{-24}	1.96×10^{-21}
R^{136}	6.94×10^{-22}	-1.97×10^{-23}	2.42×10^{-25}	4.64×10^{-22}
R^{137}	1.47×10^{-18}	-4.17×10^{-20}	5.65×10^{-22}	9.81×10^{-19}
R^{138}	3.39×10^{-19}	-9.61×10^{-21}	9.57×10^{-23}	2.27×10^{-19}
R^{139}	8.75×10^{-24}	-2.50×10^{-25}	6.12×10^{-27}	5.80×10^{-24}
R^{140}	2.17×10^{-24}	-6.18×10^{-26}	1.05×10^{-27}	1.45×10^{-24}
R^{141}	2.56×10^{-17}	-7.28×10^{-19}	8.72×10^{-21}	1.72×10^{-17}
R^{142}	5.86×10^{-18}	-1.66×10^{-19}	1.47×10^{-21}	3.93×10^{-18}
R^{143}	1.58×10^{-22}	-4.50×10^{-24}	9.50×10^{-26}	1.05×10^{-22}
R^{144}	3.83×10^{-23}	-1.09×10^{-24}	1.62×10^{-26}	2.56×10^{-23}
R^{145}	1.74×10^{-17}	-4.93×10^{-19}	6.55×10^{-21}	1.16×10^{-17}
R^{146}	4.00×10^{-18}	-1.13×10^{-19}	1.11×10^{-21}	2.68×10^{-18}
R^{147}	1.04×10^{-22}	-2.97×10^{-24}	7.10×10^{-26}	6.90×10^{-23}
R^{148}	2.57×10^{-23}	-7.32×10^{-25}	1.22×10^{-26}	1.72×10^{-23}
R^{149}	3.03×10^{-16}	-8.60×10^{-18}	1.01×10^{-19}	2.03×10^{-16}
R^{150}	6.91×10^{-17}	-1.96×10^{-18}	1.71×10^{-20}	4.63×10^{-17}
R^{151}	1.87×10^{-21}	-5.34×10^{-23}	1.10×10^{-24}	1.25×10^{-21}

References

1. Feedback control of dynamic bipedal robot locomotion book webpage (2007), <http://www.dynamicbipedcontrol.org>
2. Ahtiwas, O.M., Abdulmuin, M.Z., Siraj, S.-F.: A neural-fuzzy logic approach for modeling and control of nonlinear systems. In: IEEE International Symposium on Intelligent Control, pp. 270–275 (2002)
3. Ferreira, J., Amaral, T., Pires, V., Crisostomo, M., Coimbra, A.: A neural-fuzzy walking control of an autonomous biped robot. In: Proceedings of World Automation Congress, vol. 15, pp. 253–258 (2004), doi:10.1109/WAC.2004.185229
4. Ferreira, J., Crisóstomo, M., Coimbra, A.P.: Neuro-fuzzy zmp control of a biped robot. In: SMO 2006: Proceedings of the 6th WSEAS International Conference on Simulation, Modelling and Optimization, pp. 331–337. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point (2006)
5. Hurmuzlu, Y., Marghitu, D.B.: Rigid body collisions of planar kinematic chains with multiple contact points. *Int. J. Robot Res.* 13(1), 82–92 (1994)
6. Islam, S., Liu, P.X.: Adaptive fuzzy output feedback control for robot manipulators. In: IEEE International Conference on Systems, Man and Cybernetics, SMC 2009, pp. 2630–2635 (2009)
7. Jang, J.-S.R.: Anfis: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics* 23(3), 665–685 (1993), doi:10.1109/21.256541
8. Kelly, R., Santibáñez, V., Loría, A.: Control of Robot Manipulators in Joint Space. In: Advanced Textbooks in Control and Signal Processing, Springer, Heidelberg (2005)
9. McGeer, T.: Passive dynamic walking. *Int. J. Robot Res.* 9(2), 62–82 (1990)
10. Shieh, M.-Y., Chang, K.-H., Chuang, C.-Y., Chiou, J.-S., Li, J.-H.: Anfis based controller design for biped robots. In: IEEE International Conference on Mechatronics, pp. 1–6 (2007), doi:10.1109/ICMECH.2007.4280017
11. Sugeno, M., Kang, T.: Structure identification of fuzzy model. *Fuzzy Sets and Systems* 28, 15–33 (1988)
12. Vukobratovic, M., Juricic, D.: Contribution to the synthesis of biped gait. *IEEE Transactions on Biomedical Engineering* 16(1), 1–6 (1969)
13. Westervelt, E.R., Grizzle, J.W., Chevallereau, C., Choi, J.-H., Morris, B.: Systematic Design of Within-Stride Feedback Controllers for Walking. In: Feedback Control of Dynamic Bipedal Robot Locomotion, pp. 137–190. Taylor & Francis/CRC (2007)

Fuzzy System to Control the Movement of a Wheeled Mobile Robot

Oscar Montiel Ross¹, Jesús Camacho², Roberto Sepúlveda¹, and Oscar Castillo²

¹ Centro de Investigación y Desarrollo de Tecnología Digital del Instituto Politécnico Nacional (CITEDI-IPN), Av. del Parque No.1310, Mesa de Otay, 22510, Tijuana, B.C., México

`o.montiel@ieee.org`, `r.sepulveda@ieee.org`

² M.S. Student at CITEDI-IPN

`camacho@citedi.mx`

³ Division of Graduate Studies and Research,
Calzada Tecnológico S/N, Tijuana, B.C., México

`ocastillo@hafsamx.org`

Abstract. This paper describes the design of a locomotion controller to regulate the pose of a nonholonomic mobile robot with frontal differential driving wheels, and a swivel castor wheel in rear. Differently to common proposals base on dynamic models, this controller uses the kinematics model and the mobile robot architecture to generate the adequate linear and angular speeds to reach the desired pose. Two independent and coordinated fuzzy inference systems are the core of this development that is a step further in autonomy in the sense of independence. Experiments and results are shown.

1 Introduction

The field of autonomous navigation of mobile robot (MR) has been a focus of research interest during many years for several reasons: usually, because they are very useful when humans are not able to reach certain target because of terrain conditions, or they could be in danger; particularly, for scientist because of its interdisciplinary nature entails many complex challenges that can be tackled from different points of view.

The autonomy adjective glued to the word navigation confers an intelligent behavior of the MR, and it is related with its ability to sense the situation and act on it appropriately, the gained benefit comes along with the necessity to design the robot in a way that it is able to respond to a list of complex situations, that includes at least

the ability to navigate autonomously, avoiding modeled and unmodeled obstacles especially in crowded and unpredictably changing environment. The intelligence of a robot in the sense of independence will depend on how much and quickly it can sense its environment and itself, as well as the capacity to create its own control law; thus, an MR with such capacities must have at least sensors, effectors/actuators, a locomotion system, the on-board computer system, and a set of controllers to coordinate all the components to achieve the desired task in the desired way [18].

This work falls in the robot control area that refers to the way in which the sensing and action of a robot are coordinated. There are several approaches to control a MR, being the best known [11]:

- **Reactive Control.** It is a technique that produces timely robotic response in dynamic and non-structured worlds. The idea can be summarized as “Do not think, just react”. This idea tightly couples sensing and actions. Learning is not an objective of this technique [15] [17] [19].
- **Deliberative Control.** This is a model based on sensing, planning and action. The planning stage slows down the process [16] [21].
- **Hybrid Control.** Combines the two extremes of reactive and deliberative systems [4] [10].
- **Behavior-Based Control.** The basic idea is to subdivide the navigation task into small and easy to manage programs behaviors that focus on concurrently execution of specific subtasks. It uses a “divide and conquer” strategy that makes the system modular [14] [18] [20].

Being more specific, we present a controller for the locomotion system that can be used in any of the aforementioned approaches to control a nonholonomic differential MR with frontal traction and swivel castor wheel in rear. Differently to common methods that require an accurate dynamic model that involves the interacting forces, and the design of algorithms to directly control the motors torque that difficult implementation since they are based on unrealistic models, this proposal uses the kinematics model and the mobile robot architecture to generate a fuzzy knowledge base, in order to obtain the adequate actions to achieve the desired pose without the need to know a complex unrealistic mathematical model.

A detailed analytical study of the structure of the kinematic and dynamic models of wheeled mobile robots can be found in [6]. Wheeled mobile robots constitute a class of mechanical systems called nonholonomic mechanical systems [8] characterized by kinematic constraints that are not integrable and cannot, therefore, be eliminated from the model equations. Nonholonomic behavior in robotic systems is, particularly interesting because it implies that the mechanism can be completely controlled with reduced number of actuators.

Fuzzy logic approaches to mobile robot navigation and obstacle avoidance have been investigated by several researchers, because they have the ability to treat uncertain and imprecise information using linguistic rules, thus, they offer possible implementation of human knowledge and experience. Li and Yang [7], proposed an

obstacle avoidance approach using fuzzy logic where the input sensors are separately inferred. Lee and Wang [13] proposed a collision-avoidance approach using fuzzy logic, where different modules, such as avoiding-static-obstacle module, avoiding-moving-obstacle module, and directing-toward-target module, are created for the robot navigation. However, these modules are separately inferred and are not as coordinated as human reasoning. Xu and Tso [22], proposed a reactive behavior-based fuzzy logical controller, which utilizes the rules to define the robot reaction to unknown environments and applies fuzzy inference to coordinate different reactive behaviors. Saffiotti reviewed some of the proposals in the literature about the uses of fuzzy logic in autonomous robot navigation and discussed the pros and cons of fuzzy logic solutions in [3]. He pointed out that if only weak knowledge is available, then fuzzy logic may provide a more adequate tool, otherwise, the stronger techniques, such as classical control or probability theory, should probably be used [2]. Saffiotti also proposed some fuzzy logic methods for robot navigation; however, these methods cannot guarantee that the robot will not be trapped on local minima or infinite loops.

This work has five main sections, the first section corresponds to the present introduction. Section two briefly outlines the kinematics of a differential-drive non-holonomic MR to obtain the equations used in this development. In section three the locomotion system for controlling the pose of the MR is explained. Two representative experiments and results are given in section four. Finally, the conclusions are given in section five.

2 Kinematics of a Differential-Drive MR

Fig. 1 shows the structure of an MR with two connected traction wheels (differential) in the front, and one unpowered swivel castor wheel in rear. This MR belongs to a large class of mechanical nonholonomic systems described by the dynamic equation given in (1) based on the Euler Lagrange Formulation [1] [5].

$$M(q)\ddot{q} + V(q, \dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d = B(q)\tau - A^T(q)\lambda \tag{1}$$

where q is the n dimensional vector of configuration variables, $M(q) \in \mathbb{R}^{n \times n}$ is a symmetric positive definite inertia matrix, $V(q, \dot{q}) \in \mathbb{R}^{n \times n}$ is the centripetal and coriolis matrix, $F(\dot{q}) \in \mathbb{R}^{n \times 1}$ denotes the surface friction, $G(q) \in \mathbb{R}^{n \times 1}$ is the gravitational vector, τ_d denotes bounded unknown disturbances including unstructured unmodeled dynamics, $B(q) \in \mathbb{R}^{n \times r}$ is the input transformation matrix, $\tau \in \mathbb{R}^{n \times 1}$ is the input vector, $A(q) \in \mathbb{R}^{m \times n}$ is the matrix associated with the constraints, and $\lambda \in \mathbb{R}^{m \times 1}$ is the vector of constraints forces.

The nonholonomic constraint is given by,

$$A(q)\dot{q} = 0 \tag{2}$$

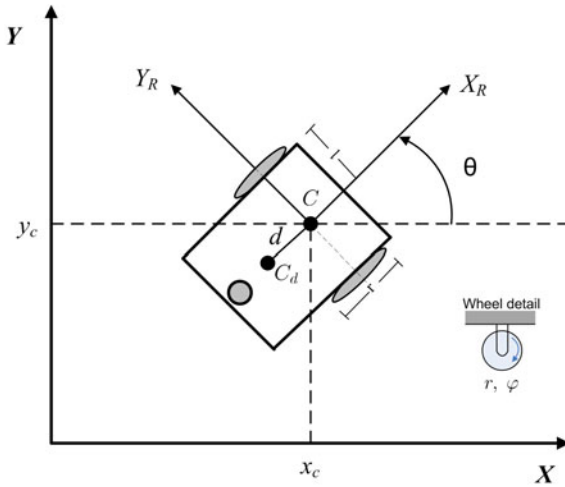


Fig. 1. A nonholonomic mobile robot. The reference point C can be shifted a distance d , i.e., the point C_d ; in our case, $d = 0$.

Defining $S(q)$ as a Jacobian matrix that transforms velocities \mathbf{v} in mobile base coordinates, to velocities \dot{q} in Cartesian coordinates, where $S(q)$ is as a full rank matrix $(n - m)$ formed by a set of smooth and linearly independent vector fields spanning the null space of $A(q)$,

$$S^T(q)A^T(q) = 0 \tag{3}$$

According to expressions (2) and (4), it is possible to find an auxiliary vector time function $\mathbf{v}(t) \in \mathbb{R}^{(n-m)}$ such that, for all t ,

$$\dot{q} = S(q)\mathbf{v}(t) \tag{4}$$

This MR is analyzed as a rigid body on wheels operating on a horizontal plane; by a rigid body we refer to the robot chassis ignoring the degrees of freedom (DOF) and flexibility due to the wheel axes, wheel steering joints, and the swivel caster wheel. The total of DOF of this robot chassis on the plane is three, two for position in the plane (X, Y) , and one for the orientation θ . however, only two DOF are controllable, it can go to the front or to the rear applying equal speed to the driving motors, the angle is obtained by the application of differential speed, hence we have only two controllable DOF. This MR is nonholonomic because the number of controllable DOF is smaller than the existing. The nonholonomic constraint states that the robot can only move in the direction normal to the axis of the driving wheels, i.e., the mobile base satisfies the conditions of pure rolling and nonslipping [5] [9],

$$\dot{y}_c \cos\theta - \dot{x}_c \sin\theta - d\dot{\theta} = 0 \tag{5}$$

It is necessary to use two different reference frames: A global reference coordinate system for the position and a robot local reference frame. Hence, to specify the position of the MR on the plane it is indispensable to establish a relationship between the global reference frame of the plane and the local frame of the robot, as it is illustrated in Fig. 1; for this purpose, the axes X and Y define an arbitrary inertial basis on the plane as the global reference frame from some origin $O : \{X, Y\}$. To specify the position of the MR, choose a point C on its chassis as a reference point that can be shifted a distance d from the driving wheels axis which is marked as C_d . The robot's local reference frame is defined using the basis $\{X_R, Y_R\}$ that defines two axes relative to C on the robot chassis [19]. In this way the position of C in the global reference frame is given by coordinates x_c and y_c , and the angular difference between the global and local reference frames is given by θ ; using these three values the pose of the robot is defined by,

$$q = (x_c, y_c, \theta)^T \tag{6}$$

Using (4) we can write the next equations

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix} \mathbf{v} \tag{7}$$

where,

$$\mathbf{v} = \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{8}$$

hence,

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{9}$$

because $d = 0$ in Fig. 1

$$\dot{q} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{10}$$

To describe robot motion in terms of component motion, it is necessary to map motion along the axes of the global reference using a transformation matrix and the speed components. Note that speed components are the linear and angular speeds, and the transforming matrix $S(q)$ for $d = 0$ is given by,

$$S(q) = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \tag{11}$$

In this proposal of pose controller, it is necessary to know the actual pose q , and the desired pose q_d to calculate the pose error; hence, we need to obtain q from (10), that is easily achieved by integrating \dot{q} .

For the robot of Fig. 1, we assume that the diameter of each wheel is r , the point C is centered between the two drive wheels, each wheel is a distance l from C to the wheel; so, given r , l , θ , and the spinning speed of each wheel, $\dot{\phi}_1$ and $\dot{\phi}_2$, a forward kinematic model must be designed to predict the robot's overall speed in the global reference frame; so, we can write expression (12)

$$\dot{q} = \begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \theta \end{bmatrix} = f(l, r, \theta, \dot{\phi}_1, \dot{\phi}_2). \quad (12)$$

The robot of Fig. 1 is aligned along $+X_R$ and the robot will move along this axis, C is halfway between the two wheels, each wheel is contributing with a half of the total robot's speed \dot{x}_R , therefore we have $\dot{x}_R = \dot{x}_{r1} + \dot{x}_{r2}$, where $\dot{x}_{r1} = \frac{1}{2}r\dot{\phi}_1$ (right wheel) and $\dot{x}_{r2} = \frac{1}{2}r\dot{\phi}_2$ (left wheel) are the speed of each wheel. For a differential robot in which each wheel spins with equal speed but in opposite directions the result is a stationary spinning movement and the \dot{x}_R will be zero, and because neither wheel can contribute to make sideways motion in the robot's reference frame, \dot{y}_R is always zero. Finally, it is necessary to calculate the rotational component $\dot{\theta}_R$ of \dot{q}_R . Since each wheel contributes additively for the final displacement, forward spin of the right wheel combined with the opposite spinning of the left wheel results in *counterclockwise* rotation at point C , similarly a clockwise rotation can be obtained using the opposite spinning of the wheels. If right wheel spin alone, the robot pivot around left wheel, and viceversa. For the *counterclockwise* case, the rotation velocity ω_1 at C can be calculated because the wheel is instantaneously moving along the arc of circle of radius $2l$,

$$\omega_1 = \frac{r\dot{\phi}_1}{2l} \quad (13)$$

For the left wheel, to obtain a *clockwise* rotation at point C , we have,

$$\omega_2 = \frac{r\dot{\phi}_2}{2l} \quad (14)$$

The kinematic model provides information about the motion of a robot using its component wheel speeds for simple cases. However, to determine the space of possible motions for a determined chassis design, it is necessary to describe formally the constrains of the robot that depends on type of each wheel. In order to simplify the model several assumption must be considered; so there are two important constraints: the first one enforces the concept of rolling contact, this means that the wheel must roll when motion takes place in the appropriated direction; the second enforces the concept of no lateral slippage which means that the wheel must no slide orthogonal to the wheel plane. It was demonstrated in [19] that for an MR with M wheels, only the fixed standard wheels and steerable wheels have impact on the robot chassis kinematics requiring consideration when computing the robot's kinematics, on the other hand, the castor wheels impose no kinematics constrains on the robot chassis.

3 Locomotion System Controller

In this section, the locomotion system controller to regulate the posture of the differential MR of Fig. 1 is explained. The controller’s goal is to make that the difference between the desired pose and the actual pose trend to zero, i.e.,

$$\lim_{t \rightarrow \infty} \|q_d - q(t)\| = 0 \tag{15}$$

Fig. 2 shows the controller block diagram. Basically, it consists of two kind of feedback controllers: The Proportional Integral (PI) “DC motor controllers” of the two drive wheels; and the “fuzzy pose controller” associated with the kinematic model, which sends, as the reference for the PI controller, an estimate value of the adequate angular speed for each wheel to achieve the desired goal.

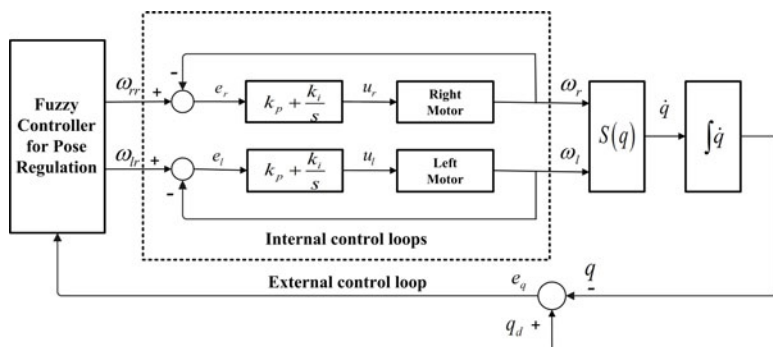


Fig. 2. Fuzzy controller to regulate the Pose of a differential mobile robot

Each drive wheel of the MR uses a Pittman GM9236S025 DC Motor equipped with a shaft optical encoder to implement the feedback loop. Fig. 3 illustrates the main electro-mechanical components used to drive the DC Motor: For a desired speed, and rotation way (CW or CCW), the controller board calculates the duty cycle to generate the Pulse Width Modulation (PWM) signal in order to send it to the H-bridge together with its configuration to obtain the desired rotation way; then, each H-bridge associated with its corresponding motor provides polarization and gives enough power to drive the motor. As it was mentioned, the motor speed is regulated using a PI controller, to calculate it, a “black-box” mathematical model [12], that involves the applied power, H-bridge, and DC gearmotor was obtained. For this task, a system identification process was achieved, we used a sampling time of $T_s = 100ms$ and 500 samples, the linearized model (transfer function) is shown in (16), a 78.19% of model accuracy was obtained, which from the practical point of view,

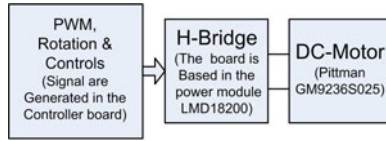


Fig. 3. Power stage to drive a DC gearmotor using PWM and H-Bridge

it was enough to achieve the PI controller design and simulations. Fig. 4 shows a comparison between the real system and the obtained model. The PI control law is given by (17), where $K_p = 0.008$ and $K_i = 0.06$.

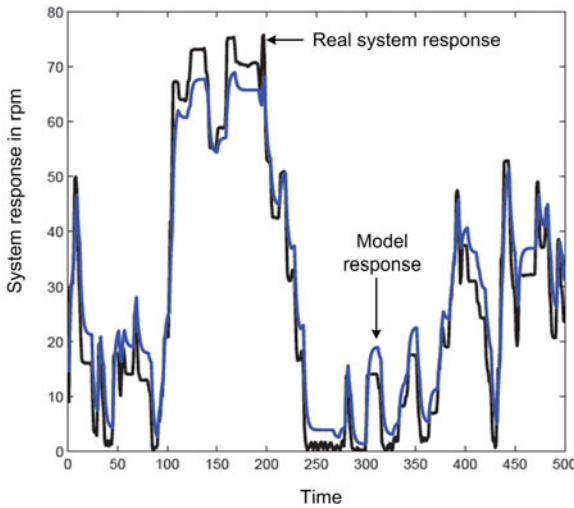


Fig. 4. A pseudorandom sequence to achieve the system identification process was used

$$G(s) = \frac{351.9}{s + 4.933} \tag{16}$$

$$K(s) = K_p + \frac{K_i(s)}{s} \tag{17}$$

In Fig. 2, the external control loop contains the pose fuzzy controller. This block handles three inputs and two outputs; to simplify, the block uses two fuzzy inference systems (FIS): one FIS for handling position in the (X, Y) plane (FIS1), and one to control the angle of orientation θ of the MR (FIS2), see Fig. 5.

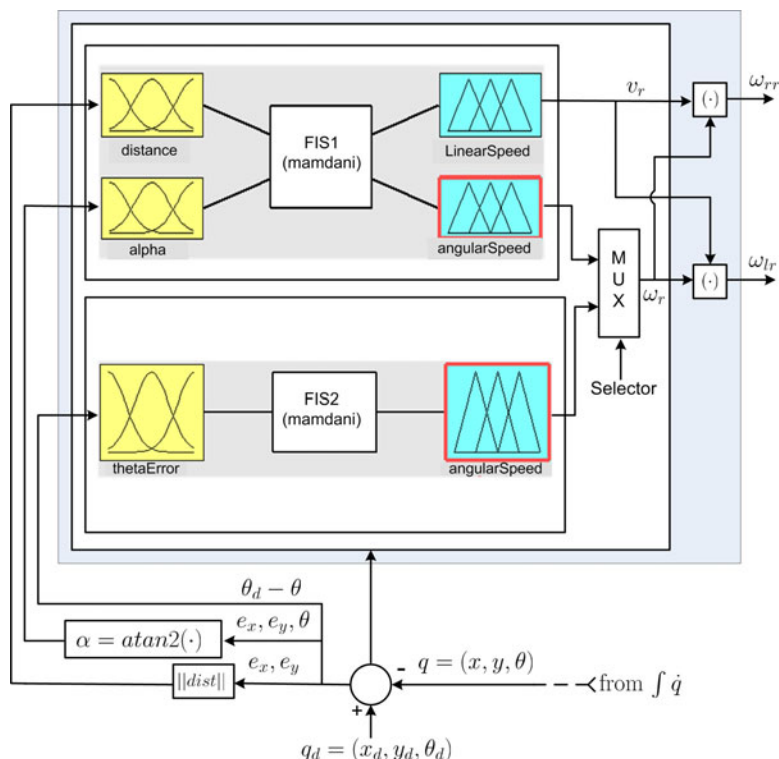


Fig. 5. Fuzzy systems of the locomotion pose controller. The two FIS infers the reference angular speed for right and left wheels.

3.1 Fuzzy Controller for Pose Regulation

The objective of this controller is to provide the appropriated reference speeds to take the MR from the actual position and orientation to the target pose, Fig. 5 shows with more detail the block named “Fuzzy Controller for Pose Regulation”, where there are two FIS sub-blocks, at the top is the FIS1 for position control, at the bottom is the FIS2 for orientation control. The output of the two FIS are combined and synchronized to obtained the required angular speeds of each drive wheel.

3.1.1 FIS1 for Position Control

This FIS has two linguistic variables as inputs: the distance “*dist*” from the actual position to the target position, and the alpha angle “ α ” situated in the reference frame, which represents the angle of deviation of the actual position to the target position, see Fig. 6. FIS1 has two linguistic variables as outputs: linear speed “ v_r ”,

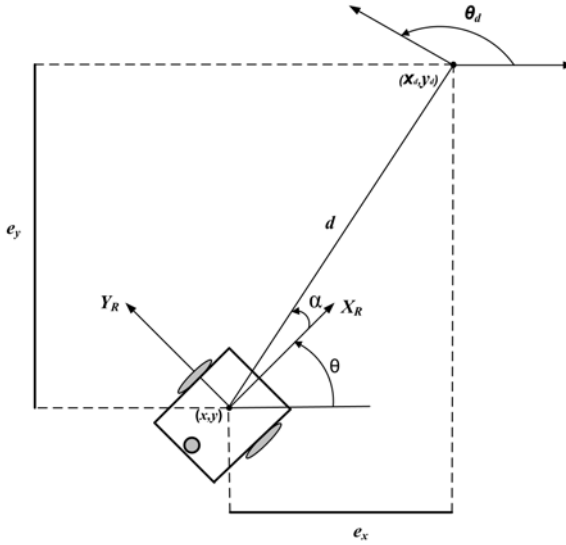


Fig. 6. The MR is disoriented the angle α . First, the MR needs to orientate to the desired position, (i.e., $\alpha = 0$), then it will reach the target position. Finally, it will get the right orientation.

and angular speed “ ω_r ”, the inference is achieved using the Mamdani approach, Table 1 shows the rule base.

The linguistic variable “distance” has two linguistic terms: Small (S) and Large (L) with triangular and trapezoidal shape, respectively; they were defined in the universe of discourse $[0, 4]$, see Fig. 7a. The numeric value at this input is the euclidian distance $dist$ obtained by (18).

$$\|dist\| = \sqrt{e_x^2 + e_y^2} = \sqrt{(x_d - x)^2 + (y_d - y)^2} \tag{18}$$

Table 1. Fuzzy rule base of FIS1

		<i>alpha</i>					
		NB	NM	Z	MP	LP	
<i>distance</i>	S	v_r	VNM	VPM	VZ	VPM	VNM
		ω_r	WNB	WNM	WZ	WPM	WPB
	L	v_r	VNB	VPB	VPB	VPB	VNB
		ω_r	WNB	WNM	WZ	WPM	WPB

The linguistic variable “alpha” is the angle of orientation that the MR forms between the straight line of actual and target position, the input numeric value “ α ” is calculated using (19),

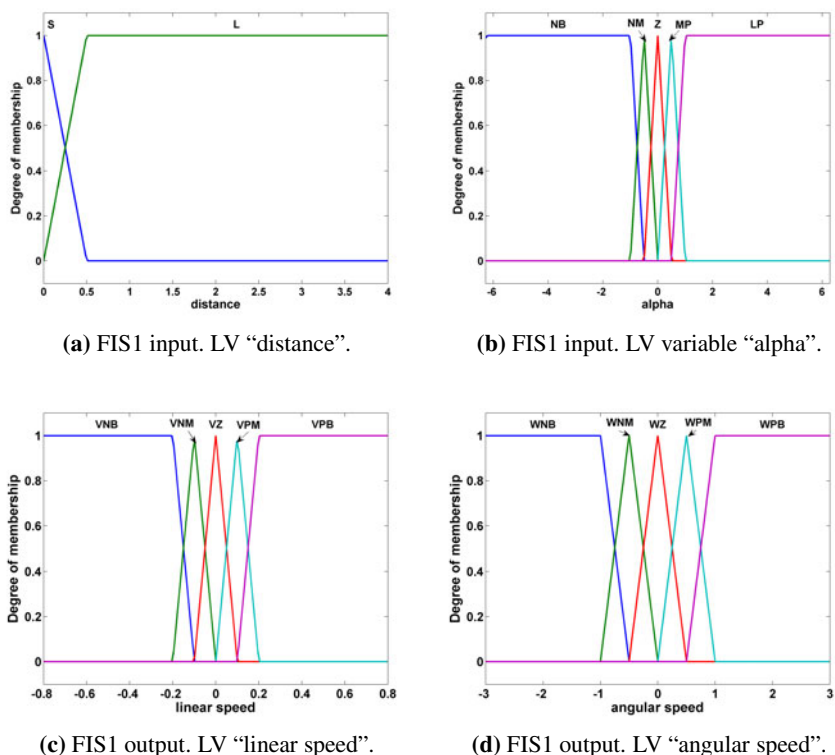


Fig. 7. Position controller. Linguistic variables (LVs) and terms for inputs and outputs of FIS1.

$$\alpha = atan2(e_y, e_x) - \theta \tag{19}$$

The variable "alpha" has five linguistic terms: Negative Big(NB), Negative Medium (NM), Zero (Z), Medium Positive (MP), and Large Positive (LP). They are represented by three triangular membership function (MF), and two trapezoidal MFs defined in the interval $[-2\pi, 2\pi]$ radians to represent a whole counterclockwise (CCW) turn, or a clockwise (CW) turn. Fig. 7b shows the MFs of each term.

FIS1 has two outputs, v_r and ω_r . The linguistic variables are "linearSpeed" and "angularSpeed", that provide an estimate of linear and angular speed, their MFs and terms are shown in Fig. 7c and Fig. 7d, respectively.

3.1.2 FIS2 for Orientation Control

The goal of this Mamdani FIS is to face the MR in the desired pose, it uses one linguistic variable as input, and one as output. Fig. 8a shows the input linguistic

variable “theta error” with its linguistic terms: Negative Big (NB), Negative Medium (NM), Zero (Z), Positive Medium (PM), and Positive Big (PB). Fig. 8b shows the output linguistic variable and terms, note that term names and ranges are the same of Fig.7d for FIS1. Table 2 shows the rule base. The “theta error” at time t is given by (20).

$$e_{\theta}(t) = \theta_d(t) - \theta(t) \tag{20}$$

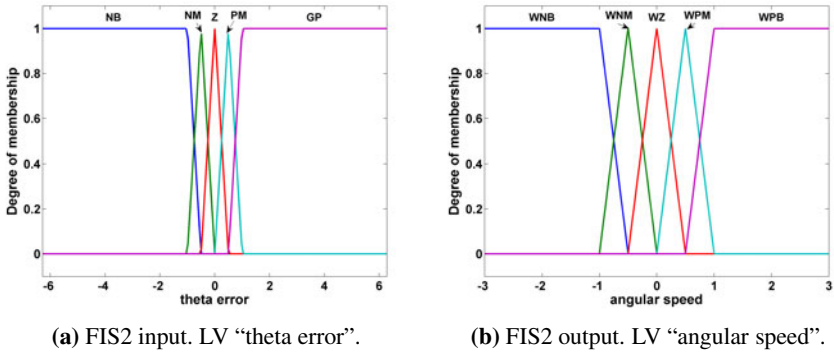


Fig. 8. FIS2. Orientation control.

Table 2. Fuzzy rule base of FIS2

<i>error theta</i>	ω_r
NB	<i>WPB</i>
NM	<i>WPM</i>
Z	<i>WZ</i>
PM	<i>WNM</i>
PB	<i>WNB</i>

3.1.3 Output Conversion and FIS Synchronization

Fig. 2 shows that the outputs of the fuzzy controller for pose regulation” are reference values of angular speeds for the right wheel “ ω_{rr} ” and left wheel “ ω_{lr} ” that the PI controllers will use as input reference to control the angular speed of each wheel. Note that FIS1 nor FIS2 provide directly these speeds, so we used (21) and (22) in order to obtain the desired angular speeds.

$$\omega_{rr} = \frac{v_r + (b/2)\omega_r}{r} \tag{21}$$

$$\omega_{lr} = \frac{v_r - (b/2)\omega_r}{r} \quad (22)$$

The algorithm implemented in the Fuzzy controller block for pose regulation has two steps synchronized with the help of the multiplexer (MUX) that is shown in Fig. 2, they are explained using Fig.6. The steps are:

1. **Step 1.** Using FIS1, the MR is oriented from the actual position and orientation to the target position regardless final orientation, in this step the MR rotates the α angle showed in Fig. 6, and it moves to the target position.
2. **Step 2.** Once the MR has reached the target position, using FIS2 the MR is oriented according the target pose, this is illustrated in Fig.6 as the angle θ_d .

4 Experimental Results

We present two representative experiments that were performed. The basic idea is to propose the initial and target poses. The general characteristics for both experiments are the next:

1. The drive wheel of the MR has a radius $r = 0.08$ meters.
2. The distance between the drive wheels is $b = 0.45$ meters.
3. The MR platform has a reference point C displaced a distance $d = 0$ from the driving wheels.
4. The initial pose is $q_0 = (0.1, 0.1, 0)^T$.

Experiment 1.

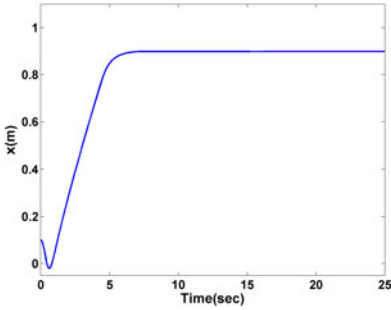
The MR wants to reach the pose given by

$$q_d = (0.9, 2, 0.5)^T \quad (23)$$

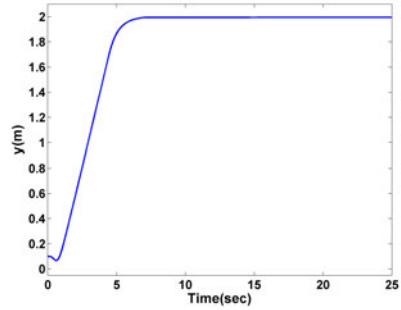
The steps explained in section 3.1.3 are followed. Fig. 9a and Fig.9b show the time in seconds that the MR lasted to orientate and arrive to the desired position (target), this process was achieved using FIS1. Fig. 9c shows the time and maximal error that the MR lasted to achieve the desired orientation. Fig. 10a, Fig. 10b, and 10c show the error evolution in time for positioning the MR in the desired position (x_d, y_d) and orientation θ_d .

Experiment 2. The MR wants to reach the pose given by

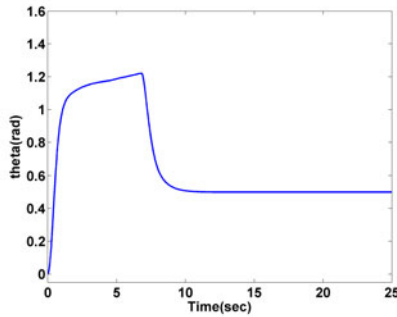
$$q_d = (2, 0.5, \frac{\pi}{4})^T \quad (24)$$



(a) Final position in x .



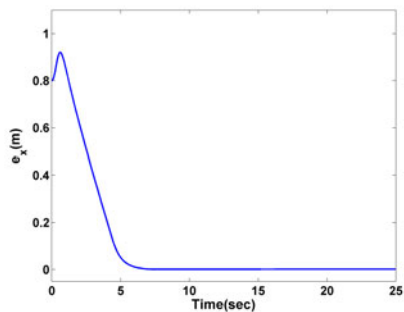
(b) Final position in y .



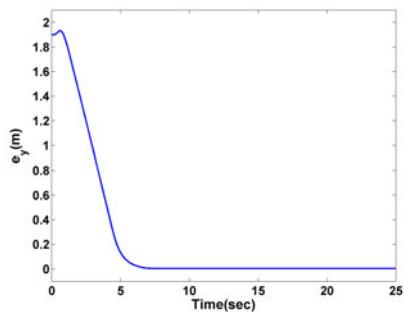
(c) Final orientation in θ .

Fig. 9. Experiment 1. Evolution in time of the MR to achieve the target pose $q_d = (0.9, 2, 0.5)^T$.

Similarly to experiment 1, we followed the algorithm explained in section 3.1.3. Fig. 11a and Fig.11b show the time that the MR lasted to orientate and arrive to the desired position (target), this process was achieved using FIS1. Fig. 11c shows the time and maximal error that the MR lasted to achieve the desired orientation. Fig. 12a, Fig. 12b, and 12c show the error evolution in time for positioning the MR in the desired position (x_d, y_d) and orientation θ_d .



(a) Position error in x.



(b) Position error in y.

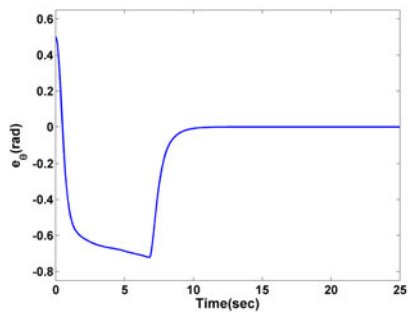
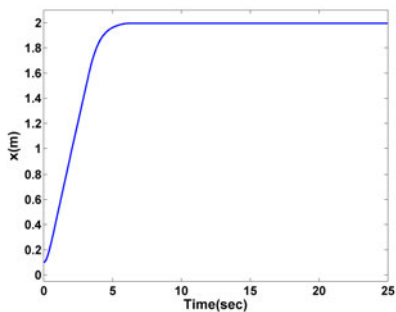
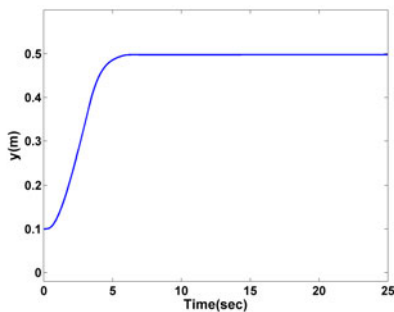
(c) Position error in θ .

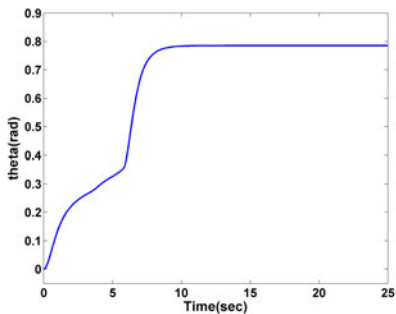
Fig. 10. Experiment 1. Initial pose $q_0 = (0.1, 0.1, 0)^T$, target pose $q_d = (0.9, 2, 0.5)^T$.



(a) Final position in x.



(b) Final position in y.



(c) Position error in θ .

Fig. 11. Experiment 2. Evolution in time of the MR to achieve the target pose $q_d = (2, 0.5, \frac{\pi}{4})^T$.

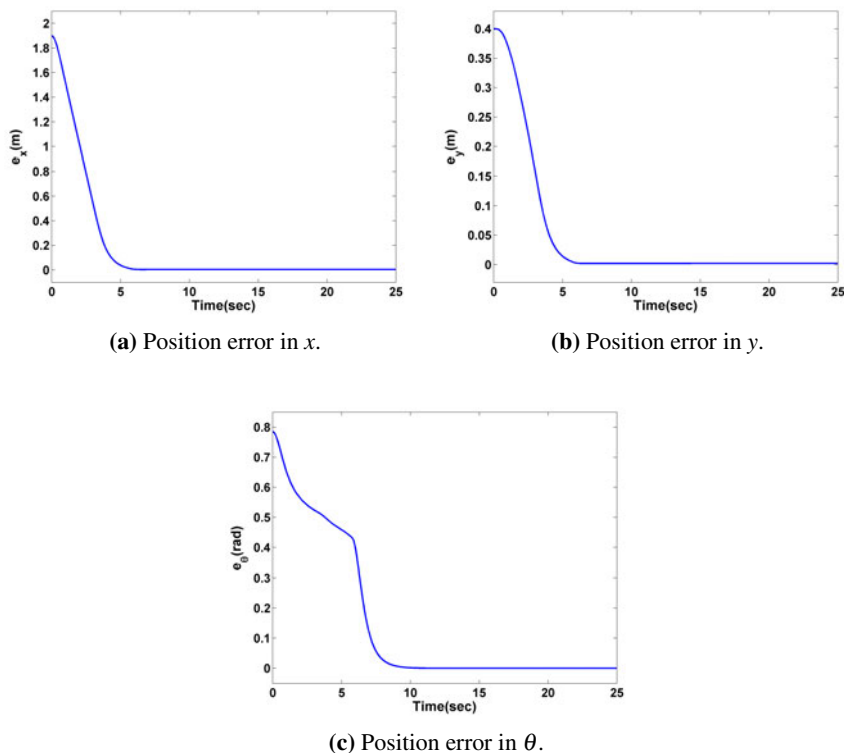


Fig. 12. Experiment 2. Evolution in time of positioning errors of the MR to achieve the target pose $q_d = (2, 0.5, \frac{\pi}{4})^T$.

5 Conclusions

The development of a locomotion controller to regulate the pose of a MR that uses a Fuzzy Control block as the core of the controller was presented. The proposed controller works with the kinematic model and the MR architecture, this characteristic allows to implement the controller without the need of using the system dynamic model. A controller requirement is to know the actual position and orientation (pose), and the desired pose in order to reach it. It was supposed that the MR has the adequate onboard equipment to provide the controller model with the different positions and orientation when it is required. Computer simulations confirms that the performance of the locomotion controller satisfied the controller’s goal imposed by $\lim_{t \rightarrow \infty} \|q_d - q(t)\| = 0$, with a no significant error with a fast time response for the selected motors. This controller is a step further in autonomy in the sense of independence since its core is a fuzzy system based on rules derived from experience, instead of using analytical mathematical models to generate the control law.

Portability is an important characteristic of this proposal since modifications to the MR architecture impose only small program changes.

Acknowledgement. The authors thank Comisión de Operación y Fomento de Actividades Académicas del Instituto Politécnico Nacional, Instituto Tecnológico de Tijuana, and CONACYT for supporting our research activities.

References

1. Bloch, A.M., Reyhanoglu, M., McClamorch, H.: Control and stabilization of nonholonomic dynamic systems. *IEEE Tran. on Automatic Control* 37(11), 1746–1757 (1992)
2. Saffiotti, A.: Fuzzy logic in autonomous navigation. In: Driankov, D., Saffiotti, A. (eds.) *Fuzzy Logic Techniques for Autonomous Vehicle*. Studies in Fuzziness and Soft Computing, pp. 3–24. Springer-Physica Verlag (2001)
3. Saffiotti, A.: The uses of the fuzzy logic in autonomous robot navigation. *Soft Computing* 1, 180–197 (1997)
4. Gat, E.: On Three-Layer Architectures, Artificial Intelligence and Mobile Robotics. In: Kortenkamp, D., Bonnasso, R.P., Murphy, R. (eds.), pp. 195–210. MIT, AAAI Press, Cambridge (1998)
5. Fierro, R., Lewis, F.L.: Control of a nonholonomic mobile robot Using Neural Networks. *IEEE Transactions on Neural Networks* 9(4) (1998)
6. Campion, G., Bastin, G., D'AndreaNovel, B.: Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE Trans. Robot Autom.* 12(1), 47–62 (1996)
7. Li, H., Yang, S.X.: A behavior-based mobile robot with a visual landmark recognition system. *IEEE/ASME Transactions on Mechatronics* 10(5), 695–708 (2003)
8. Kolmanovsky, I., McClamroch, N.H.: Developments in nonholonomic control problems. *IEEE Control Syst. Mag.* 15, 20–36 (1995)
9. Barraquand, J., Latombe, J.-C.: Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. In: *Proc. IEEE Int. Conf. Robot. Automat.*, Sacramento, CA, pp. 2328–2335 (1991)
10. Connell, J.: SSS: A Hybrid Architecture Applied to Robot Navigation. In: *Proceedings of IEEE Conference on Robotics and Automation*, Nice, France, pp. 2719–2724 (1992)
11. Mataric, M.J.: Behavior-Based Control: Examples from Navigation, Learning, and Group Behavior. In: Hexmoor, H., Horswill, I., Kortenkamp, D. (eds.) *Journal of Experimental and Theoretical Artificial Intelligence*, special issue on Software Architectures for Physical Agents, vol. 9(2/3), pp. 323–336 (1997)
12. Montiel, O., Castillo, O., Melin, P., Sepúlveda, R.: Black box evolutionary mathematical modeling applied to linear systems. *International Journal of Intelligent Systems* 20(2), 293–311 (2005)
13. Lee, P.S., Wang, L.L.: Collision avoidance by fuzzy logic control for automated guided vehicle navigation. *Journal of Robotic Systems* 11(8), 743–760 (1994)
14. Arkin, R.: *Behavior-Based Robotics*. MIT Press, Cambridge (1998)
15. Brooks, R., Connell, J.: Asynchronous Distributed Control Systems for a Mobile Robot. In: *Proceedings of SPIE's Cambridge Symposium on Optical and Optoelectronic Engineering*, Cambridge, MA, pp. 77–84 (1986)
16. Brooks, R.: *Elephants Don't Play Chess*, Designing Autonomous Agents, vol. 6, pp. 3–15. MIT Press, Cambridge (1990)

17. Brooks, R.: Intelligence without Representation. *Artificial Intelligence* 47, 139–160 (1991)
18. Arkin, R.C.: *Towards Cosmopolitan Robots: Intelligent Navigation in Extended Man-made Environments*, PhD Thesis, University of Massachusetts, Department of Computer and Information Science (1987)
19. Siegwart, R., Nourbakhsh, I.R.: *Introduction to Autonomous Mobile Robots*. A Bradford Book The MIT Press, Cambridge (2004)
20. Parasuraman, S., Ganapathy, V., Shirinzadeh, B.: Behaviour based Mobile Robot Navigation Technique using AI System: Experimental Investigations. In: *ARAS 2005 Conference*, Cairo, Egypt (2005)
21. Rosenchein, S., Kaelbling, L.: A Situated View of Representation and Control, Special Issue on Computational Research on Interaction and Agency, pp. 515–540. Elsevier Science, Amsterdam (1995)
22. Xu, W.L., Tso, S.K.: Real-Time self-reaction of a mobile robot in unstructured environments using fuzzy reasoning. *Engineering Applications of Artificial Intelligence* 9(5), 475–485 (1996)

Embedding a Fuzzy Locomotion Pose Controller for a Wheeled Mobile Robot into an FPGA

Oscar Montiel¹, Jesús Camacho², Roberto Sepúlveda¹, and Oscar Castillo³

¹ Centro de Investigación y Desarrollo de Tecnología Digital del Instituto Politécnico Nacional (CITEDI-IPN), Av. del Parque No.1310, Mesa de Otay, 22510, Tijuana, B.C., México

o.montiel@ieee.org, r.sepulveda@ieee.org

² M.S. Student at CITEDI-IPN

camacho@citedi.mx

³ Division of Graduate Studies and Research, Calzada Tecnológico S/N, Tijuana, B.C., México

ocastillo@hafsamx.org

Abstract. This work deals with the embedding architecture into an FPGA of a fuzzy locomotion controller for pose regulation of a differential nonholonomic mobile robot. It is presented an standardized design based on the actual and target pose, it does not need any dynamic model to work, the design provides the estimate angular speeds, then using the kinematic model, a feedback of the actual position is provided, hence the same system can be used by different mobile robots considering that the speed control of driving wheels is a subsystem and a configuration is provided. These features makes this proposal viable to be used by the automotive industry in the automatic steering system for self-parking for different car models with differential tracking. The controller was developed using VHDL code, its functionality is simulated by merging the code into the Simulink environment. The experimental framework, experiments and results are explained.

1 Introduction

During several years, the problem of motion planning of nonholonomic robots have been studied from different perspectives because they offer several interesting problems to researchers [1] [8]. In most of the cases, a nonholonomic mobile robot (MR) has a limited steering angle, rectangular shape, frontal or rear driving wheels; in general, it has similar features than a car. Recently, automotive industry also has put its attention to this kind of MR, such is the example of new cars with automatic steering for self parking. This work is focused in the development of a locomotion controller for pose regulation embedded into an FPGA, that works with the kinematic model, it

does not need any complicated mathematical dynamic model since its functionality is based on the system architecture. This controller is a step further in autonomy in the sense of independence because its core is a fuzzy system based on expertise knowledge. The controller only needs to know the actual pose (position in x, y and the angle of orientation) and the target pose. Embedding a controller with such characteristics offers to designers the possibility of developing an standard dedicated computer for controlling cars movement; i.e., nonholonomic mobile robots [7].

There are several previous works that deal with this problem using different approaches, in [2] was presented a partial listing of C-code for solving the problem known as “truck backer-upper control” for implementation into a reconfigurable FPGA system. In [3], a new version of the Xfuzzy design environment is presented using as an application example, the self parking of a car problem. In [10], the design and synthesis of a mobile robot controller using fuzzy logic for reactive control was presented. Other works that use fuzzy logic in this field are [14] [15]. The proposal presented in this work is fairly different to previous works in several aspects.

This work has five main sections, the first section corresponds to the present introduction. In section two the locomotion system is described, this section begins explaining the kinematic model, following with a description about the instances used to build the Fuzzy Pose Controller (FPC) for FPGA implementation. In section three, the experimental platform to test the proposal is explained. In section four, the experiments and results are commented. Finally in section five are the conclusions.

2 Locomotion Controller of a Nonholonomic Wheeled MR

In Fig. 1 the structure of a differential nonholonomic MR with two connected traction wheels in the front and one unpowered swivel castor wheel in rear is shown. In this figure, there are two different reference frames: the global reference frame $\{X, Y\}$ to position the MR in the world from some origin $O : \{X, Y\}$, and the robot's local reference frame $\{X_R, Y_R\}$ that maps a reference point C in the robot's chassis considering its orientation. In this way, the point C in the global frame is given by coordinates x_c and y_c , and the angular difference between global and local frames is given by θ , using these three values the pose of a robot is defined by,

$$q = (x_c, y_c, \theta)^T \quad (1)$$

The robot of Fig. 1 has three degrees of freedom in the plane, x_c and y_c for position, and θ for orientation; however, only two are controllable, it can go to the front or to the rear applying equal speed to the driving motors, by the application of differential speeds the angle is obtained. The nonholonomic constrains states that the robot can only move in direction normal to the axle of the driving wheels, the slipping condition is not allowed. In Fig. 2 the locomotion controller to regulate the posture of an MR such as the shown in Fig 1, is illustrated. Basically, it has two kind of feedback controllers: The Proportional Integral (PI) “DC gearmotor” controllers of the two drive wheels, and the Fuzzy Pose Controller (FPC) associated with the

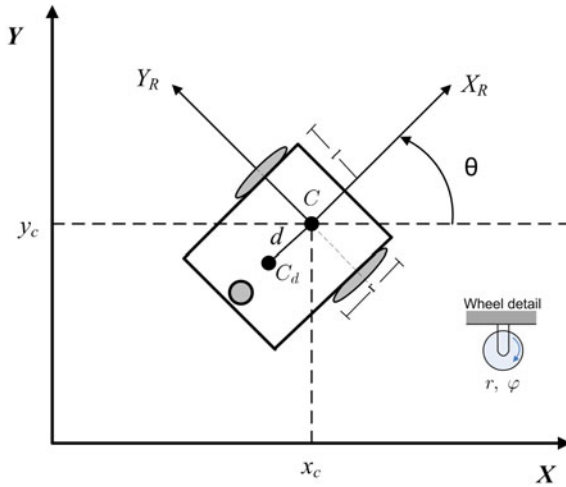


Fig. 1. A nonholonomic mobile robot. The reference point C can be shifted a distance d , i.e. the point C_d ; in our case, $d = 0$.

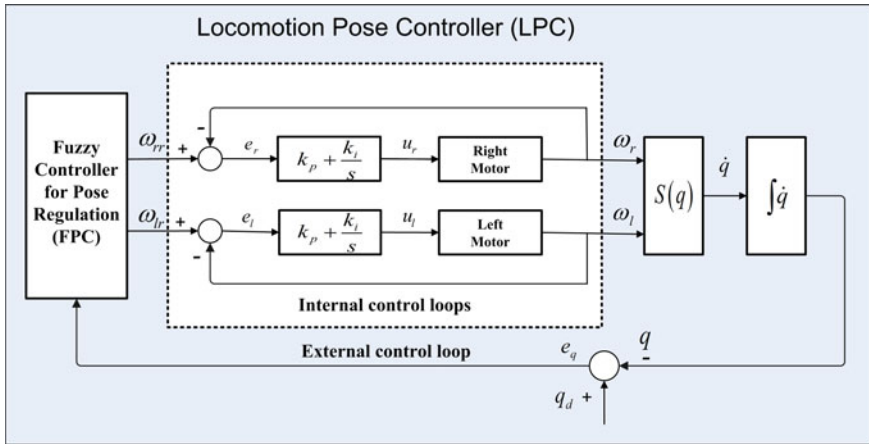


Fig. 2. Fuzzy locomotion system to regulate the Pose of a differential mobile robot

kinematic model. The objective of the FPC is to estimate the angular speed values for each drive wheel that will be used by the PI controllers as the reference input. The goal of the locomotion controller is given by,

$$\lim_{t \rightarrow \infty} \|q_d - q(t)\| = 0 \tag{2}$$

Each wheel of the MR uses a Pittman GM9236S025 DC gearmotor equipped with a shaft quadrature optical encoder to implement the feedback loop, the speed is regulated using Pulse Width Modulation (PWM) and the power stage is handled

by an H-bridge. The desired speed and rotation way, clockwise (CW) or counterclockwise (CCW), is calculated by the onboard controller that sends the appropriated PWM signal and H-bridge configuration. To calculate the PI controller was necessary to obtain a “black-box” mathematical model [11], that involves the applied power (produced by the PWM signal), H-bridge, and DC gearmotor. To achieve the system identification, a sampling time of $T_s = 100ms$ and 500 samples were used, the linearized model (transfer function) is shown in (3), a 78.19% of model accuracy was obtained, which from the practical point of view, it was enough to achieve the PI controller design and simulations. The PI controller law is given by (4),

$$G(s) = \frac{351.9}{s + 4.933} \tag{3}$$

$$u(t) = K_p e(t) + \frac{K_i}{s} e(t) \tag{4}$$

where $K_p = 0.008$, and $K_i = 0.06$. The output of each PI controller is the duty cycle of the PWM signal. The actual angular speed of each drive wheel is calculated using the optical encoder, it serves as the controller feedback as well as input to the matrix $S(q)$ that transforms velocities \mathbf{v} in the MR base coordinates, to velocities \dot{q} in Cartesian coordinates. The speed is calculated using [5] [6] [13],

$$\dot{q} = S(q)\mathbf{v}(t) \tag{5}$$

where the matrix $S(q)$ for a nonholonomic MR with a reference point C positioned in the center of the drive wheel axle (i.e., $d = 0$) is given by,

$$S(q) = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \tag{6}$$

therefore,

$$\dot{q} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \mathbf{v}, \quad \text{where } \mathbf{v} = \begin{bmatrix} v \\ \omega \end{bmatrix}, \quad \text{and } \dot{q} = \begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} \tag{7}$$

The locomotion pose controller uses the “pose error” signal denoted by e_q ,

$$e_q(t) = q_d - q(t) \tag{8}$$

where, q is obtained integrating equation (7). Fig. 3 shows the FPC architecture with more detail. It consists of two Fuzzy Inference Systems (FIS1 and FIS2), one multiplexer (MUX), and two functions, equations (9) and (10), that converts linear and angular speeds to angular reference speeds for right ω_{rr} , and left ω_{lr} , wheels. The MUX and its selector input, coordinates the two FIS actions to reach the desired pose; hence, the selector input allows to achieve the next two coordination steps:

- **Step 1.** Using FIS1, the MR is oriented from the actual position and orientation to the target position regardless final orientation, in this step the MR rotates the α angle showed in Fig. 4, and it moves to the target position.
- **Step 2.** Once the MR has reached the target position, using FIS2 the MR is oriented according to the target pose, this is illustrated in Fig. 4 as the angle θ_d .

$$\omega_{rr} = \frac{v_r + (b/2)\omega_r}{r} \tag{9}$$

$$\omega_{lr} = \frac{v_r - (b/2)\omega_r}{r} \tag{10}$$

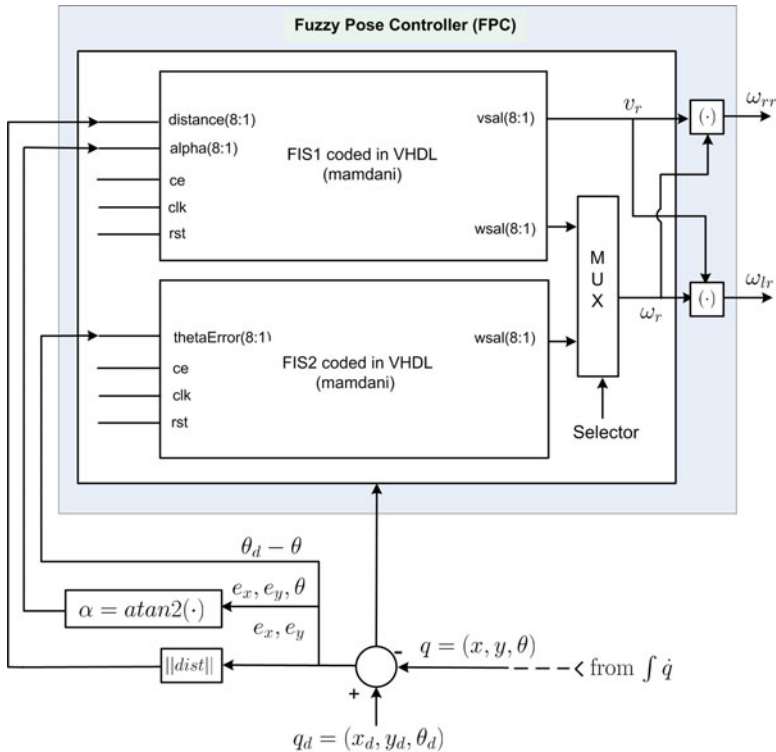


Fig. 3. Fuzzy systems of the locomotion pose controller. The two FIS infers the reference angular speed for right and left wheels. Signal ce , clk and rst appear not connected, however they need to be connected. Signal ce enables or disables the instances of FIS1 and FIS2. Signal rst provides initial conditions to the instances (hardware) in the FPGA. Signal clk synchronizes the different stages of the FIS: fuzzification [12], inference [9] and defuzzification [4].

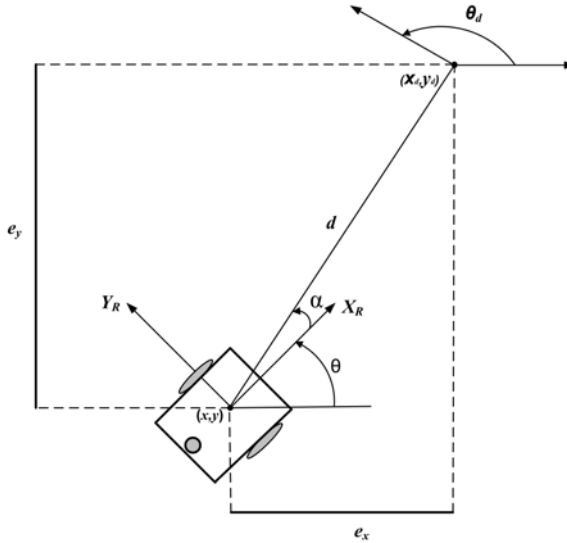


Fig. 4. The MR is disoriented the angle α . First, the MR needs to orientate to the desired position, (i.e., $\alpha = 0$), after that, it will reach the target position. Finally, it will get the right orientation.

Table 1. Position controller (FIS1). It has two inputs and two outputs

Linguistic variable		Linguistic terms		
FIS input	FIS output	Name	Type	Parameters
distance		Small (S)	Triangular	[0, 0, 0.5]
		Large (L)	Trapezoidal	[0, 0.5, 4, 4]
alpha		Negative Big (NB)	Trapezoidal	$[-\pi, -\pi, -1, -0.5]$
		Negative Medium (NM)	Triangular	$[-1, -0.5, 0]$
		Zero (Z)	Triangular	$[-0.5, 0, 0.5]$
		Positive Medium (PM)	Triangular	[0, 0.5, 1]
	linear speed, v_r	Negative Big (NB)	Trapezoidal	$[-0.8, -0.8, -0.2, -0.1]$
		Negative Medium (NM)	Triangular	$[-0.2, -0.1, 0]$
		Zero (Z)	Triangular	$[-0.1, 0, 0.1]$
		Positive Medium (PM)	Triangular	[0, 0.1, 0.2]
		Positive Big (PB)	Trapezoidal	[0.1, 0.2, 0.8, 0.8]
	angular speed, ω_r	Negative Big (NB)	Trapezoidal	$[-3, -3, -1, -0.5]$
		Negative Medium (NM)	Triangular	$[-1, -0.5, 0]$
		Zero (Z)	Triangular	$[-0.5, 0, 0.5]$
		Positive Medium (PM)	Triangular	[0, 0.5, 1]
		Positive Big (PB)	Trapezoidal	[0.5, 1, 3, 3]

FIS1 for Position Control

This FIS has two linguistic variables as inputs: the distance *dist* from the actual position to the target position, and the alpha angle α situated in the reference frame,

which represents the angle of deviation of the actual position to the target position, see Fig. 4. FIS1 has two linguistic variables as outputs: linear speed v_r , and angular speed ω_r , the inference is achieved using the Mamdani approach, in [9] [4] [12] is explained how to code in VHDL the different stages of a FIS, i.e.; fuzzification, inference, and defuzzification. The parameters of linguistic variables and terms are described in Table 1, in Table 2 the rule base is shown.

The numeric value of the input linguistic variable $dist$ is the euclidian distance obtained by (11).

$$\|dist\| = \sqrt{e_x^2 + e_y^2} = \sqrt{(x_d - x)^2 + (y_d - y)^2} \tag{11}$$

The linguistic variable α represents the angle of orientation that the MR forms between the straight line of actual and target position, the input numeric value α is calculated using (12). The universe of discourse is the interval $[-2\pi, 2\pi]$ radians to represent a whole counterclockwise (CCW) turn, or a clockwise (CW) turn.

$$\alpha = atan2(e_y, e_x) - \theta \tag{12}$$

FIS2 for Target Orientation

Once the MR has reached the target position in (X, Y) , the goal of FIS2 is to face the MR in the desired pose. It has one input and one output, the associated linguistic variable are “theta error” and “angular speed”, respectively. In Table 3 are the input and output linguistic variables, their terms name, shapes, and parameters are also shown. In Table 4 the FIS2 rule base is given. The “theta error” at time t is the difference between desired angle and the actual angle, see equation (13).

$$e_\theta(t) = \theta_d(t) - \theta(t) \tag{13}$$

3 Experimental Platform

In this section, the methodology that was carried out to test and validate the locomotion controller of Fig. 2 and its corresponding VHDL implementation is explained. In all the experiments we consider the Spartan 3 FPGA as the target system, so we used as the main development software tools: the Xilinx Integrated Software Environment (Xilinx ISE) to develop the VHDL code of the FPC, the Xilinx System Generator (XSG) to migrate the VHDL code to the Simulink/Matlab environment, and the Simulink. The idea is to design first the system in Simulink, then in VHDL and compare both systems. Next we comment the main steps.

1. Design the FPC block of Fig. 2 using Simulink/Matlab tools. We called this design “FPC-smk”. It contains the fuzzy systems for position (FIS1), and for orientation (FIS2). Fig. 5a shows how to test FIS1, Fig. 5b shows how to test FIS2; in Figures 6a, 6b, and 6c are the surface control obtained with these two last models.

Table 2. Fuzzy rule base of FIS1

		alpha					
		NB	NM	Z	MP	LP	
distance	S	v_r	VNM	VPM	VZ	VPM	VNM
		ω_r	WNB	WNM	WZ	WPM	WPB
	L	v_r	VNB	VPB	VPB	VPB	VNB
		ω_r	WNB	WNM	WZ	WPM	WPB

Table 3. Orientation controller (FIS2). It has two inputs and two outputs, whose linguistic terms have “W” as the first letter to indicate “angular speed”.

Linguistic variable		Linguistic terms		
FIS input	FIS output	Name	Type	Parameters
theta error		Negative Big (NB)	Trapezoidal	$[-\pi, -\pi, -1, -0.5]$
		Negative Medium (NM)	Triangular	$[-1, -0.5, 0]$
		Zero (Z)	Triangular	$[-0.5, 0, 0.5,]$
		Positive Medium (PM)	Triangular	$[0, 0.5, 1]$
		Positive Big (PB)	Trapezoidal	$[0.5, 1, \pi, \pi]$
	angular speed, ω_r	Negative Big (WNB)	Trapezoidal	$[-3, -3, -1, -0.5]$
		Negative Medium (WNM)	Triangular	$[-1, -0.5, 0]$
		Zero (WZ)	Triangular	$[-0.5, 0, 0.5]$
		Positive Medium (WPM)	Triangular	$[0, 0.5, 1]$
		Positive Big (WPB)	Trapezoidal	$[0.5, 1, 3, 3]$

Table 4. Fuzzy rule base of FIS2

<i>theta error</i>	ω_r
NB	WPB
NM	WPM
Z	WZ
PM	WNM
PB	WNB

2. Design the Locomotion Pose Controller (LPC) of Fig. 2 using Simulink/Matlab tools, include the “FPC-smk” module of step 1. Fig. 7 shows this design. We called this module “LPC-smk”. Test the module and save results for future comparisons.
3. Codify in VHDL the FPC block of Fig. 2. Import the design to Simulink through the XSG, this module was called “FPC-vhdl”. See Fig. 8 for testing the VHDL module. Figures 9a, 9b, and 9c show the surface control for this codification.
4. In the LPC Model (designed in step 2) of Fig. 7, include the module “FPC-vhdl”, instead of the “FPC-smk” module. We called this module “LPC-vhdl”. Test the module, and save results to achieve comparisons against results obtained in Step 2.

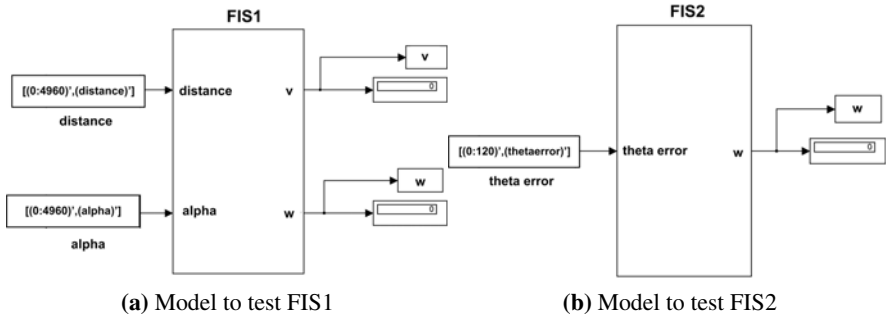


Fig. 5. Simulink models of the FPC block, FIS1 is for positioning the MR, and FIS2 for orientation. The outputs of the “FPC-smk” module are linear and angular speeds. These models were designed using only native Simulink blocks.

4 Experiments and Results

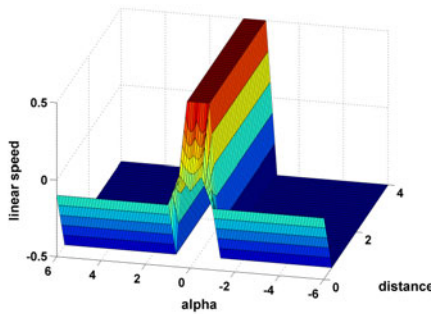
Several comparative experiments were achieved. They were be divided in two groups:

1. Validation of the “FPC-vhdl” module. We compare results of the models “FPC-smk” vs. “FPC-vhdl”, i.e., Figures 6 and 9 were used to analyze differences in the control surfaces, the obtained results are shown in Fig. 10. We used a lenght word of eight bits. Error surfaces are shown in Fig.10, in general the error is small, it can be reduced increasing the lenght word.
2. Validation of the “LPC-vhdl” module. Similarly, we used results obtained using models constructed using only native Simulink blocks “LPC-smk” model, to contrast results against those obtained using the “LPC-vhdl” model.

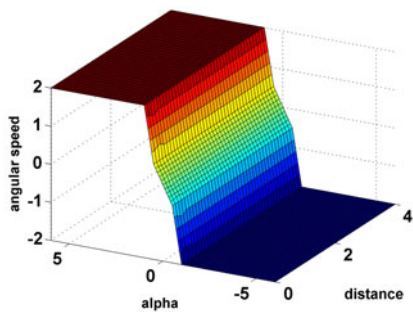
Validating the “LPC-vhdl” Module

To validate the LPC system coded in VHDL (“LPC-vhdl”) several experiments were performed. Here, we show one representative experiment, the results are compared with the equivalent model that was developed with native blocks of Simulink (“LPC-smk”). The initial conditions of the MR are $q_0 = (0.1, 0.1, 0)^T$, the wheel’s radio is $r = 0.08m$, and the axle of driving wheels is $0.45m$

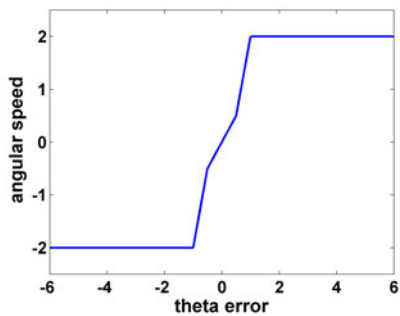
The goal of the experiment is to reach the pose $q_d = (1.5, 0.75, \frac{\pi}{4})^T$. We used the model “LPC-vhdl” that includes the FPC coded in VHDL, Fig. 11 shows with a dashed line the system response given by the module in VHDL “LPC-vhdl”; here Figures 11a and 11b show the evolution in time to achieve the desired (x, y) position, in Fig. 11c the evolution in time to achieve the final orientation is plotted. Using a solid line, in these figures also are shown the results obtained with the model “LPC-smk”. The position and orientation errors are shown in Fig 12. Differences are mainly due to length word, they can be reduced increasing it, however for most of practical applications they are not significant.



(a) Linear speed (FIS1)



(b) Angular speed (FIS1)



(c) Angular speed (FIS2)

Fig. 6. Control surfaces obtained with the Simulink model of Fig. 5

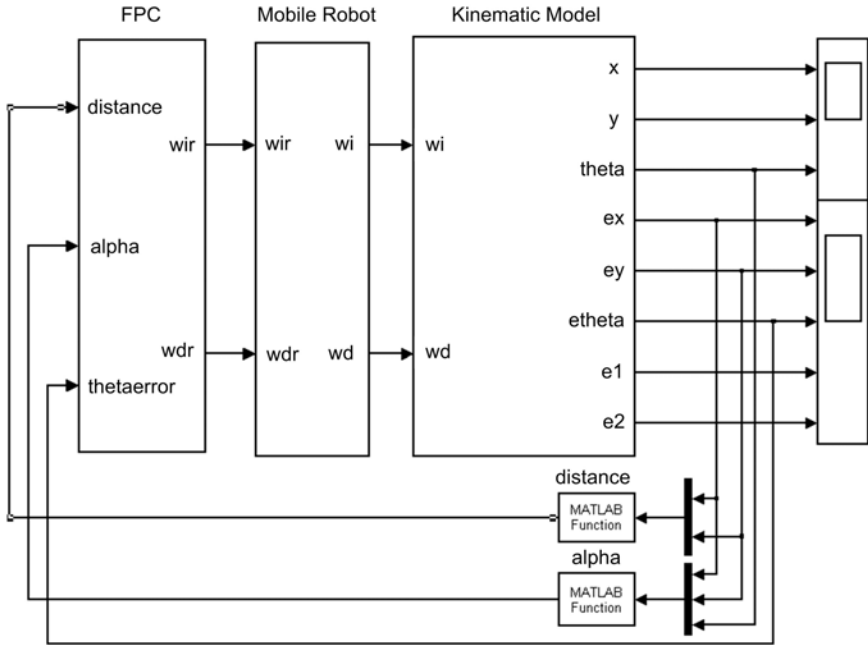
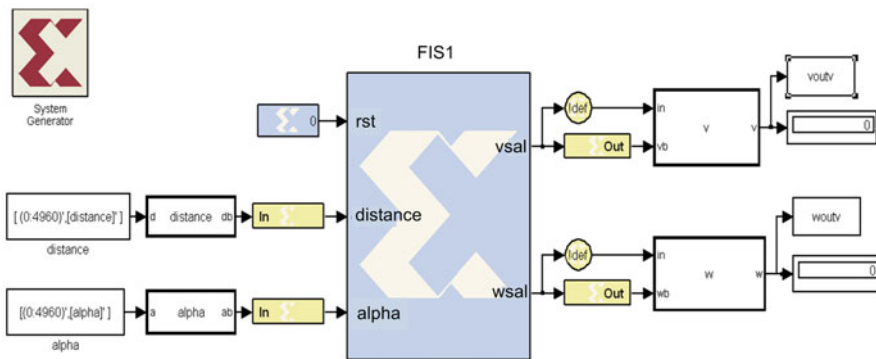
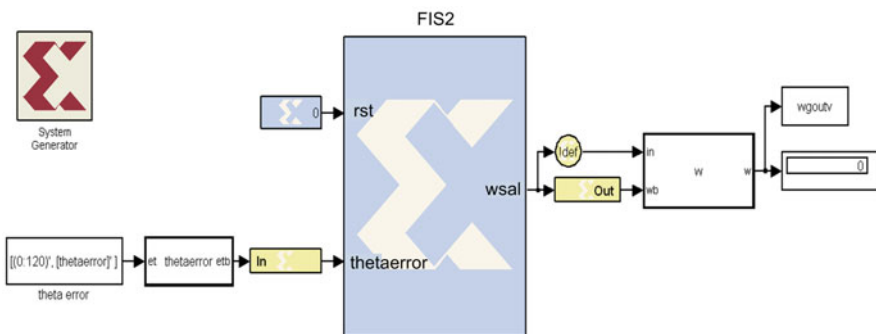


Fig. 7. Simulink model of the LPC. The structure of this model is the same for testing the “FPC-smk” and the “FPC-vhd” modules. Note that in essence, this is the Simulink implementation of the block diagram showed in Fig. 2. Therefore, in this figure, the first block labeled as “FPC” is the the Fuzzy Pose Controller module, here we can test the “FPC-smk” or the “FPC-vhd” modules. In the second block labeled as “Mobile Robot” are the internal control loops illustrated in Fig. 2. The third block labeled as “Kinematic Model” contains the transformation matrix $S(q)$ and the integral operation $\int \dot{q}$ to obtain the actual pose q . Below, there are two Matlab function blocks, *distance* and *alpha* to achieve operations given by (11) and (12). The other blocks are just viewers to observe results.



(a) FIS1 in VHDL



(b) FIS2 in VHDL

Fig. 8. Simulink models of FIS1 and FIS2 coded in VHDL. Both models, according to Fig. 3 conform the “FPC-vhdl” model.

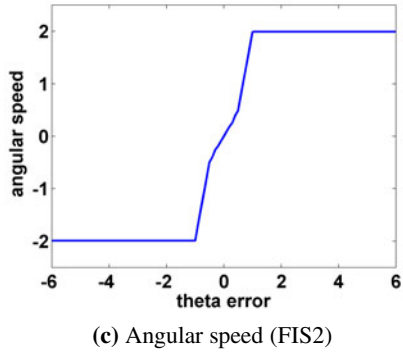
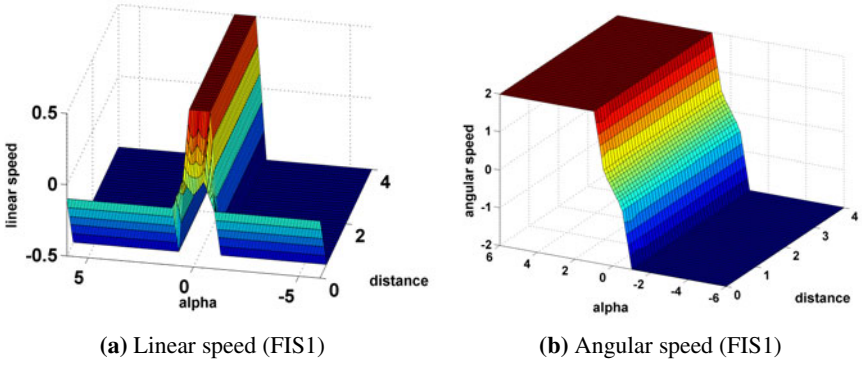
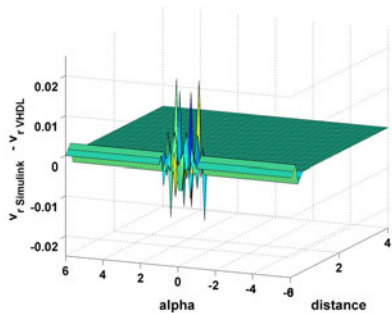
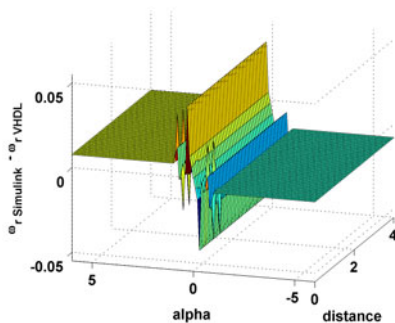


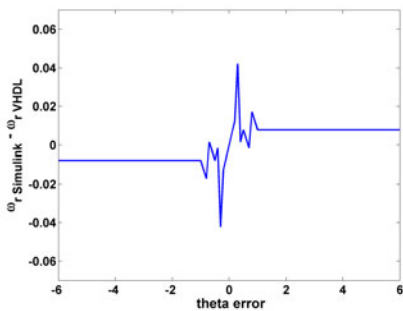
Fig. 9. Control surfaces of the “FPC-vhd” module. The two FIS coded in VHDL were imported to Simulink in order to evaluate them. To obtain the surfaces the modules showed in Fig. 8 were used.



(a) Linear speed error (FIS1)

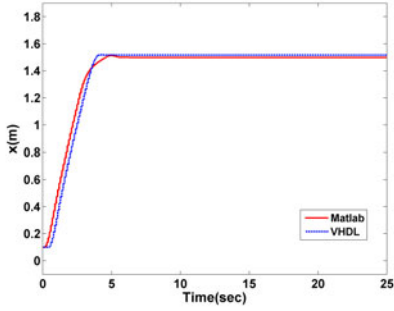


(b) Angular speed error (FIS1)

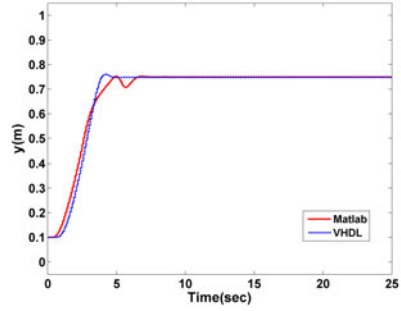


(c) Angular speed error (FIS2)

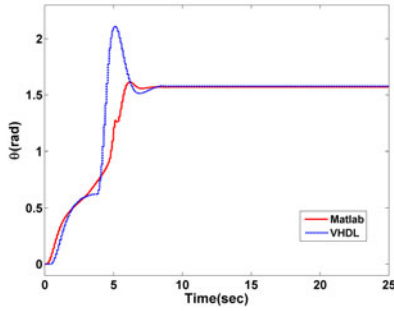
Fig. 10. Error plots, they were obtained subtracting control surfaces of models “FPC-smk” and “FPC-vhdl”. The FPC-vhdl module was coded in VHDL using a length word of 8 bits. The plots show that the error between the surfaces of the module in VHDL against the module in Simulink that uses floating point numeric format is small. By increasing the length word in VHDL codification the differences between surfaces will be reduced.



(a) Time to achieve final position in x .



(b) Time to achieve final position in y .



(c) Time to achieve the desired position in θ

Fig. 11. VHDL code was imported to the Simulink environment. Although an inference into an FPGA can last some nanoseconds, the experimental plant (MR) has a slow time response, hence the times showed in the plots. The code into an FPGA is highly parallelized, when it is migrated to Simulink, a typical PC cannot reach the same granularity. Also differences in time to reach the goal (position x,y), and precision are due to modeling errors and length word. The important thing is that the “LPC-vhdl” model showed consistency in all the experiments.

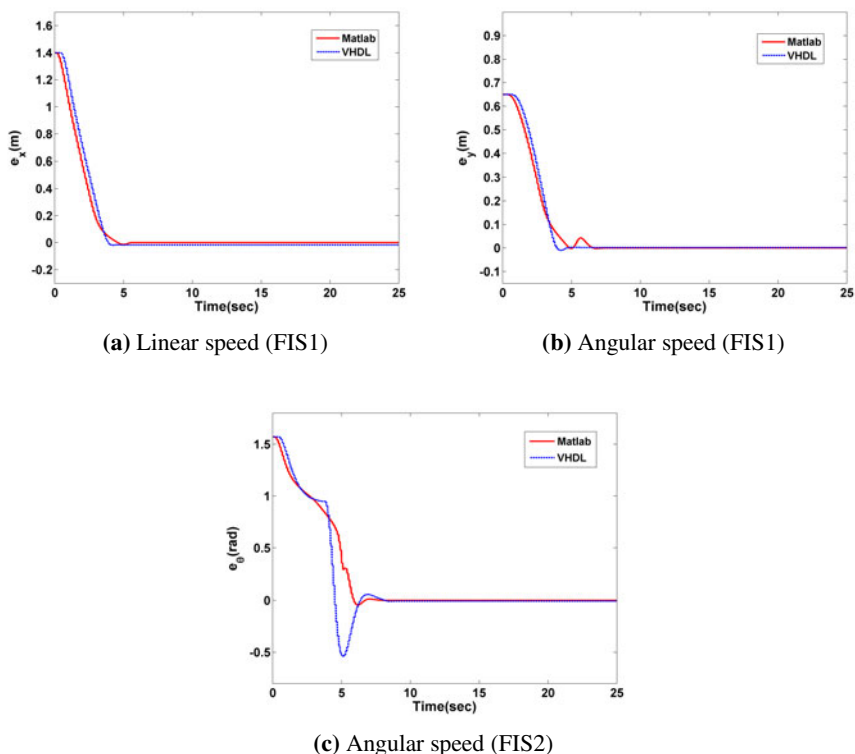


Fig. 12. Performance comparison of modules “LPC-smk” and “LPC-vhdl”. Note that the module coded in VHDL introduces a small error in orientation; however, it can be reduced increasing the length words to represent numbers.

5 Conclusions

This work dealt with the development of a VHDL module of a fuzzy pose controller (FPC-vhdl), that can be a subsystem of a locomotion controller embedded into an FPGA. Based on the FPC, the complete locomotion controller for pose regulation (LPC) was developed, it has the characteristic that does not need any complicated dynamic model to work; hence providing the model with the adequate speed motor controller it can work for several applications. This feature makes the system viable for the development of an application specific integrated circuit (ASIC), that can be used in diverse application; for example in automotive industry for automatic steering for self-parking for cars with differential tracking. In general, the differences (errors) between the proposal coded in VHDL and the results obtained with Simulink were small, and expected since we used a length word of eight-bits, whereas Simulink uses floating point.

Acknowledgement. The authors thank Comisión de Operación y Fomento de Actividades Académicas del Instituto Politécnico Nacional, Instituto Tecnológico de Tijuana, and CONACYT for supporting our research activities.

References

1. Bloch, A.M., Reyhanoglu, M., McClamorch, H.: Control and stabilization of nonholonomic dynamic systems. *IEEE Tran. on Automatic Control* 37(11), 1746–1757 (1992)
2. Kim, D.: An Implementation of Fuzzy Logic Controller on the Reconfigurable FPGA System. *IEEE Transaction on Industrial Electronics* 47(3) (2000)
3. Moreno-Velo, F.J., Baturone, I., Sánchez-Solano, S., Barriga, A.: Rapid Design of Fuzzy Systems with XFUZZY. In: *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2003* (2003)
4. Lizárraga, G., Sepúlveda, R., Montiel, O., Castillo, O.: Modeling and Simulation of the Defuzzification Stage Using Xilinx System Generator and Simulink. *Soft Computing for Hybrid Intelligent Systems*, pp. 333–343. Springer, Berlin (2008)
5. Fierro, R., Lewis, F.L.: Control of a nonholonomic mobile robot Using Neural Networks. *IEEE Transactions on Neural Networks* 9(4) (1998)
6. Campion, G., Bastin, G., D’AndreaNovel, B.: Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE Trans. Robot Autom.* 12(1), 47–62 (1996)
7. Kolmanovsky, I., McClamroch, N.H.: Developments in nonholonomic control problems. *IEEE Control Syst. Mag.* 15, 20–36 (1995)
8. Barraquand, J., Latombe, J.-C.: Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. In: *Proc. IEEE Int. Conf. Robot. Automat.*, Sacramento, CA, pp. 2328–2335 (1991)
9. Olivas, J.A., Sepúlveda, R., Montiel, O., Castillo, O.: Methodology to Test and Validate a VHDL Inference Engine through the Xilinx System Generator. *Soft Computing for Hybrid Intelligent Systems*, pp. 325–331. Springer, Heidelberg (2008)
10. Shabiul Islam, M., Anwarul Azim, M., Saukat Jahan, M., Othman, M.: Design and Synthesis of Mobile Robot Controller using Fuzzy. In: *28th International Conference on Software Engineering (ICSE 2006)*, Shangai, China (2006)
11. Montiel, O., Castillo, O., Melin, P., Sepúlveda, R.: Black box evolutionary mathematical modeling applied to linear systems. *International Journal of Intelligent Systems* 20(2), 293–311 (2005)
12. Maldonado, Y., Montiel, O., Sepúlveda, R., Castillo, O.: Design and Simulation of the Fuzzification Stage through the Xilinx System Generator. In: *Soft Computing for Hybrid Intelligent Systems*, pp. 297–305. Springer, Berlin (2008)
13. Siegwart, R., Nourbakhsh, I.R.: *Introduction to Autonomous Mobile Robots*. Bradford Book The MIT Press, Cambridge (2004)
14. Saffiotti, A.: Fuzzy logic in autonomous navigation. In: Driankov, D., Saffiotti, A. (eds.) *Fuzzy Logic Techniques for Autonomous Vehicle*. Studies in Fuzziness and Soft Computing, pp. 3–24. Springer-Physica Verlag (2001)
15. Saffiotti, A.: The uses of the fuzzy logic in autonomous robot navigation. *Soft Computing* 1, 180–197 (1997)

Author Index

- Aguilar, Leocundo 125
Aguilar, Luis T. 181, 405, 423
Álvarez, Carlos 23
Arce, Francisco 67
- Barbosa, Juan Javier González 395
Barrera-Cortes, Josefina 81
Barrón-Estrada, Lucia 3
Baruch, Ieroham 81
- Camacho, Jesús 445, 465
Cardenas-Maciel, Selene L. 423
Castañón-Puga, Manuel 315
Castillo, Oscar 81, 103, 125, 143,
161, 181, 195, 213, 225, 233,
269, 287, 329, 405, 423,
445, 465
Castro, Juan R. 23, 269
Cazarez-Castro, Nohe R. 405
Cervantes, Leticia 213
- Espinoza-Hernández, Iván 315
- Flores, Dora-Luz 315
- García, Yazmani 57
García-Macías, J. Antonio 125
García-Pedrero, Angel 253
Garza, Arnulfo Alanis 143, 225, 233
Gaxiola, Carelia 315
Gomez-Gil, Pilar 253
Gómez-Ramírez, Eduardo 103
González B., Juan J. 303
- Hernández-Aguirre, Arturo 355
Huacuja, Héctor Joaquín Fraire 395
Hugo, Terashima-Marin 43
- Jorge A., Soria-Alcaraz 43
- Leal-Ramirez, Cecilia 329
Licea, Guillermo 23, 125
- Maldonado, Yazmín 195
Mariaca-Gaspar, Carlos-Roman 81
Martin, Carpio-Valadez J. 43
Martínez, Luis G. 23
Martinez, Ricardo 181
Medellín-Martínez, Fabián 303
Meléndez, Abraham 225
Melin, Patricia 103, 161, 195, 269
Mendoza, Olivia 269
Montiel, Oscar 465
Morales-Rodríguez, María Lucila 303
- Ochoa, Alberto 57
- Qian, Wei 373
- Ramirez-Cortes, Juan Manuel 253
Rangel, Rodolfo Pazos 395
Reyes-Galaviz, Orion Fausto 3
Reyes-García, Carlos Alberto 3
Rivas-Perea, Pablo 373
Rivera-Meraz, Mariano J.J. 355
Rodríguez, Antonio 181
Rodríguez-Díaz, Antonio 23, 269,
315

- Rodriguez-Diaz, Antonio 329
Rosiles, Jose Gerardo 373
Ross, Oscar Montiel 445
- Salinas-Gutiérrez, Rogelio 355
Sepúlveda, Roberto 445, 465
Soberanes, Héctor José Puga 395
Soria, José 287
Soto, Jesus 287
- Valadez, Juan Martín Carpio 395
Valdez, Fevrier 161
Valdez, José Mario García 67, 143,
233
Villa-Diharce, Enrique R. 355
Villanueva, Jesús David Terán 395
- Yañez, Javier 57
- Zatarain, Ramón 3