Volume 2 ▪ Issue 4 ▪ October 2011

Editor-in-Chief Dr. Bekir Karlik

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE)

ISSN : 2180-124X

Publication Frequency: 6 Issues / Year

CSC PUBLISHERS
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE)

VOLUME 2, ISSUE 4, 2011

EDITED BY
DR. NABEEL TAHIR

ISSN (Online): 2180-124X

International Journal of Artificial Intelligence and Expert Systems (IJAE) is published both in traditional paper form and in Internet. This journal is published at the website http://www.cscjournals.org, maintained by Computer Science Journals (CSC Journals), Malaysia.

IJAE Journal is a part of CSC Publishers
Computer Science Journals
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS (IJAE)

Book: Volume 2, Issue 4, October 2011
Publishing Date: 31-10-2011
ISSN (Online): 2180-124X

IJAE Journal is a part of CSC Publishers
http://www.cscjournals.org

Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers, 2011

EDITORIAL PREFACE

The International Journal of Artificial Intelligence and Expert Systems (IJAE) is an effective medium for interchange of high quality theoretical and applied research in Artificial Intelligence and Expert Systems domain from theoretical research to application development. This is the third issue of volume two of IJAE. The Journal is published bi-monthly, with papers being peer reviewed to high international standards. IJAE emphasizes on efficient and effective Artificial Intelligence, and provides a central for a deeper understanding in the discipline by encouraging the quantitative comparison and performance evaluation of the emerging components of Expert Systems. IJAE comprehensively cover the system, processing and application aspects of Artificial Intelligence. Some of the important topics are AI for Service Engineering and Automated Reasoning, Evolutionary and Swarm Algorithms and Expert System Development Stages, Fuzzy Sets and logic and Knowledge-Based Systems, Problem solving Methods Self-Healing and Autonomous Systems etc.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Starting with volume 2, 2011, IJAE appears in more focused issues. Besides normal publications, IJAE intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

IJAE give an opportunity to scientists, researchers, and vendors from different disciplines of Artificial Intelligence to share the ideas, identify problems, investigate relevant issues, share common interests, explore new approaches, and initiate possible collaborative research and system development. This journal is helpful for the researchers and R&D engineers, scientists all those persons who are involve in Artificial Intelligence and Expert Systems in any shape.

Highly professional scholars give their efforts, valuable time, expertise and motivation to IJAE as Editorial board members. All submissions are evaluated by the International Editorial Board. The International Editorial Board ensures that significant developments in image processing from around the world are reflected in the IJAE publications.

IJAE editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJAE. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJAE provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members
International Journal of Artificial Intelligence and Expert Systems (IJAE)

EDITORIAL BOARD

TABLE OF CONTENTS

Volume 2, Issue 4, October 2011

Pages

An Active Elitism Mechanism for Multi-objective Evolutionary Algorithms

Engin Ufuk ERGUL *engin.ergul@amasya.edu.tr*
Vocational School, Amasya University,
Amasya, Turkey

Abstract

Classical (or passive) elitism mechanisms in the MOEA (Multi-objective Evolutionary Algorithm) literature have a holding/sending back structure. In this paper, an active elitism mechanism for multi-objective evolutionary algorithms is proposed. In the active elitism mechanism, a set of elite (or non-dominated) individuals is excited by genetic operators (crossover/mutation) in the archive in the hope of generating better and more diverse individuals than themselves. If a set of excited elites are any better than originals, then archive can be viewed as a place of active solution provider rather than a static storage place. The main motivation behind this approach is that elite individuals are inherently the closest individuals to the solution (of any optimization problem on hand) and exciting those individuals can likely generate more significant outcomes than a far away one. The proposed active elitism mechanism is embedded into well-known multi-objective SPEA and SPEA2 methods (named ACE_SPEA and ACE_SPEA2 respectively) and compared to the original methods using four unconstrained test problems. The active elitist versions of SPEA and SPEA2 maintain better spread and convergence properties than the original methods on all test problems. The proposed active elitism mechanism can easily be integrated into existing multi-objective evolutionary algorithms to improve their performance.

Keywords: Active Elitism, Evolutionary Algorithms, Multi-objective Optimization, SPEA, SPEA2

1. INTRODUCTION

Multi-objective optimization is with no doubt a very important research topic both for scientists and engineers, not only because of the multi-objective nature of most real world problems but also because there are still many open questions in this area [1]. MOEAs are playing a dominant role in solving problems with multiple conflicting objectives and obtaining a set of non-dominated solutions which are close to the Pareto optimal front. They have a number of advantages such as, obtaining a set of non-dominated solutions in a single run, easy handling of problems with local Pareto fronts and discrete nature due to their population approach and flexible recombination operators [2].

In the literature, there have been a lot of research activities on multi-objective optimization using evolutionary algorithms and a number of well performing MOEAs have been published. MOEAs differ from each other with some mechanisms: fitness assignment, elitism (or archive management) and hybridization to other algorithms. Elitism mechanism has critical role on the performance of a MOEA. Because, elite (or non-dominated) individuals are the best individuals found by an algorithm in a single run. The presence of elitism could improve the performance of MOEAs, but care must be taken to apply it efficiently [2].

Classical sense of elitism in single-objective evolutionary algorithms (SOEAs), the best solution is always copied into the next population. Although the incorporation of elitism in MOEAs is more complex than SOEAs, fundamental practice is similar. There are various ways to incorporate elitism into MOEAs. Most algorithms make use of a second population (or archive) of elite individuals. Though there are great varieties in the implementation of elitism, it can be summarized some important features as follows: First, they require preserving the best solutions in the population, an archive or sub-population. Second, they require consideration of several strategies, including the elitism strategy, or how the elitist population is updated; the re-insertion strategy, or how elite individuals take part in the production of offspring; and the control flow, or when archiving and re-insertion take place. Unfortunately, particular implementations leave the determination of many parameters to the decision makers. They also inevitably increase the space and time complexity. Nevertheless, the evidence shows that elitism is an important and indispensable factor in EMOO (Evolutionary Multi-objective Optimization) [3].

It has been reported that the elitist versions of two different MOEAs perform equally well [4],[5]. Therefore, elitism makes MOEAs more capable than those which do not employ it. Rudolph has proved that GAs (Genetic Algorithms) converge to the global optimal solution of some test functions in the presence of elitism [6],[7]. Furthermore, the presence of elite individuals enhances the probability of creating better offsprings.

Two major phases in classical elitism mechanism are:

i) **Holding phase:** *Which individuals are going to be kept in archive?* In this phase, non-dominated solutions (or elites) in parent (primary) population are stored in the archive.

ii) **Sending back phase:** *Which individuals are going back into the primary population?* One strategy is to copy all elite individuals from current population to next population. Another strategy is to copy only a number of elite individuals to next population [4].

Elitism procedures in the literature are generally similar. Generally, elite individuals are stored in an extra population (or archive). And then, archive is updated by some criterions. Adding new elite individuals to archive and removing dominated elites from archive are named as archive truncation. There are many archive truncation methods used in MOEAs [4],[8],[9],[10],[11]. As a result, common point of all classical (or passive) elitism mechanisms is storing elites in an archive *without any excitation* and passing them into the main population.

In this paper, an active elitism mechanism is proposed. In the active elitism mechanism, a set of elite individuals is excited by genetic operators (crossover/mutation) in archive in the hope of generating better or more diverse individuals than themselves. If a set of excited elites is better than originals, then the archive can be viewed as a place of active solution provider rather than a static storage place.

In this paper, well-known multi-objective SPEA (Strength Pareto Evolutionary Algorithm) and SPEA2 (Strength Pareto Evolutionary Algorithm 2) methods [2],[4],[12] with passive elitism and modified version of them (ACE_SPEA, ACE_SPEA2 respectively) with active elitism mechanism are simulated and compared.

The remainder of this paper is organized in five sections. Section 2 describes the basic concepts of MOPs (Multi-objective Problems). Section 3 explains SPEA and SPEA2 algorithms, respectively. Section 4 gives detailed descriptions of the proposed active elitism mechanism. Experimental problem sets and simulation results are given in Section 5. Finally, Section 6 contains the discussion, concluding remarks and future directions.

2. MULTI-OBJECTIVE OPTIMIZATION

A multi-objective optimization problem can be stated as follows:

$$\min\{f_1(x), f_2(x), \cdots, f_m(x)\} \quad , \quad x \in \Omega \tag{1}$$

where $x = (x_1, x_2, \cdots, x_n)$ is the decision variable vector and $f_i(x)$ are the objective functions. Ω is the decision space. A solution *x* is said to dominate solution *y* if and only if $f_i(x) \le f_i(y)$ for every

$i \in \{1,2,\cdots,m\}$ and $f_i(x) < f_i(y)$ for at least one index $j \in \{1,2,\cdots,m\}$. A point $x^* \in \Omega$ is Pareto optimal to (1) if there is no point $x \in \Omega$ such that $f(x)$ dominates $f(x^*)$. $f(x^*)$ is Pareto-optimal objective vector. The set of all the Pareto-optimal points is called the Pareto Set (PS). The set of all the Pareto-optimal objective vectors is called the Pareto Front (PF).

3. SPEA and SPEA2

3.1. SPEA (Strength Pareto Evolutionary Algorithm)

SPEA is an elitist evolutionary algorithm method based on Pareto definition [4],[5]. This method was introduced by Zitzler and Thiele. Evolutionary processes of SPEA are managed by two different populations. One of these populations, called regular population is utilized to generate offspring. Other population, called an archive with pre-defined size individuals is employed to preserve the evolutionary information of Pareto front [13]. At each generation, non-dominated individuals are copied to the archive. For each individual in the archive, a strength value is computed.

In SPEA, the fitness of each member of the current population is computed according to the strengths of all external non-dominated solutions that dominate it. Additionally, a clustering technique called “average linkage method” is used to keep diversity.

3.1.1. Fitness Assignment Strategy of SPEA

Let *n* be the number of individuals dominated by a non-dominated solution and let *N* be the total number of dominated individuals (all the individuals except the first Pareto front). Then the fitness (or strength) of the non-dominated solution is defined as:

$$f_i = s_i = \frac{n}{N+1} \tag{2}$$

The fitness of a dominated solution is calculated by adding the fitness of all the non-dominated individuals that dominate it, plus one:

$$f_j = 1 + \sum s_i \tag{3}$$

For the sake of clarity, the fitness of a non-dominated solution (C) and a dominated solution (E) are numerically given in Figure 1. There are four dominated individuals (E,F,G and H) in the population, so $N = 4$. Solution C dominates two individuals (E and H), so the fitness/strength of C is 2/5. Solution E is only dominated by non-dominated C. For this reason the fitness of E is (1+(2/5)=7/5, so E(7/5) is assigned.

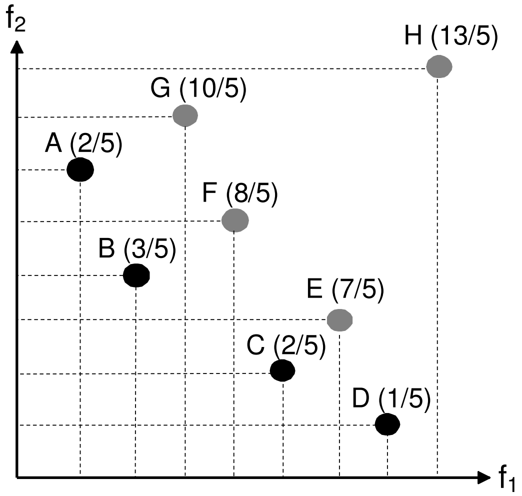

FIGURE 1: Fitness assignment example of SPEA (Non-dominated individuals are shown by black circles, dominated individuals are shown by grey circles).

3.1.2. Elitism Mechanism of SPEA

SPEA uses a regular population and an archive. Starting with an initial population and an empty archive, the following steps are performed per iteration. First, all non-dominated population members are copied to the archive; any dominated individuals or duplicates (regarding the objective values) are removed from the archive during this update operation. If the size of the updated archive exceeds a predefined limit, further archive members are deleted by a clustering technique which preserves the characteristics of the non-dominated front. After the mating selection, recombination and mutation phases, the old population is replaced by the resulting offspring population.

3.2. Strength Pareto Evolutionary Algorithm 2 (SPEA2)

SPEA2 has three main differences with respect to its predecessor [12]: (i) it incorporates a fine-grained fitness assignment strategy which takes into account for each individual the number of individuals that dominates it and the number of individuals by which it is dominated; (ii) it uses a nearest neighbor density estimation technique which guides the search more efficiently, and (iii) it has an enhanced archive truncation method that guarantees the preservation of boundary solutions.

3.2.1. Fitness Assignment Strategy of SPEA2

Each individual ***i*** in the archive $\overline{P_t}$ and the main population P_t is assigned a strength value $S(i)$, representing the number of individuals it dominates:

$$S(i) = \left|\{ j \mid j \in P_t + \overline{P_t} \wedge i \succ j\}\right| \tag{4}$$

where + stands for multiset union and the symbol $\succ$ corresponds to the Pareto dominance relation. On the basis of the *S* values, the raw fitness *R*(***i***) of an individual ***i*** is calculated:

$$R(i) = \sum_{j \in P_t + \overline{P_t}, j \succ i} S(j) \tag{5}$$

That is the raw fitness determined by the strengths of its dominators in both archive and population, as opposed to SPEA where only archive members are considered in this context. The density estimation technique used in SPEA2 is an adaptation of the *k*-th nearest neighbor method, where the density at any point is a (decreasing) function of the distance to the *k*-th nearest data point. To be more precise, for each individual ***i*** the distances (in objective space) to all individuals ***j*** in archive and population are calculated and stored in a list. After sorting the list in increasing order, the *k*-th element gives the distance sought, denoted as σ_i^k. As a common setting, we use *k* equal to the square root of the sample size, thus, $k = \sqrt{N + \overline{N}}$. ($N$: population size, $\overline{\mathrm{N}}$: archive size). Afterwards, the density $D(i)$ corresponding to ***i*** is defined by:

$$D(i) = \frac{1}{\sigma_i^k + 2} \tag{6}$$

Finally, adding $D(i)$ to the raw fitness value $R(i)$ of an individual ***i*** yields its final fitness:

$$F(i) = R(i) + D(i) \tag{7}$$

A fitness assignment example of SPEA2 is shown in Figure 2.

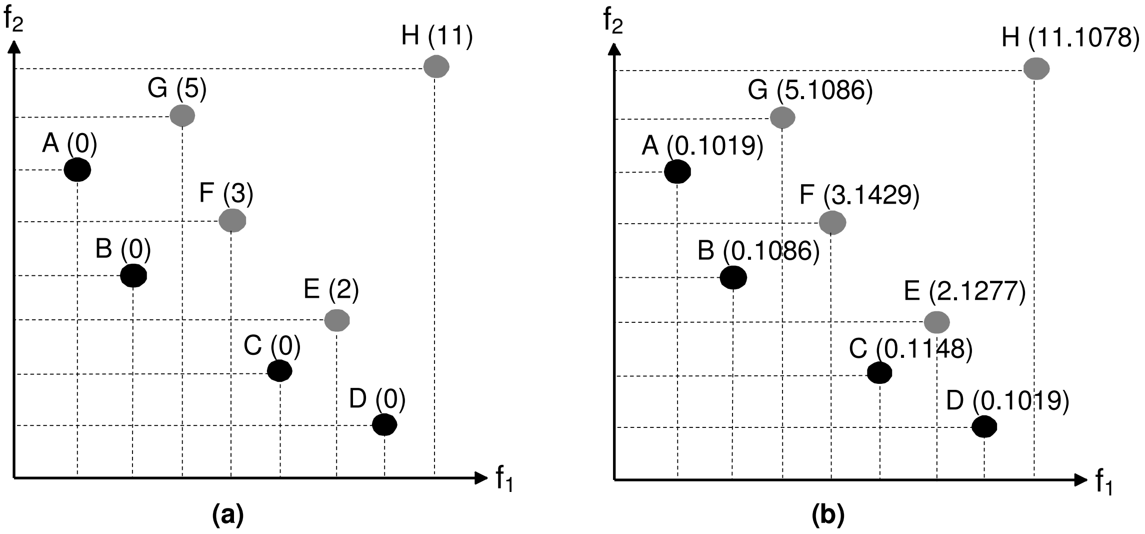

FIGURE 2. (a) Raw fitness values assigned by SPEA2. (b) Final fitness values assigned by SPEA2 (k=3.1623).

The first step is the determination of strength of an individual. Strength of an individual is defined as the total number of individuals dominated by this individual. For example, A dominates two individuals {G,H}, strength of A or $S(A) = 2$, and H dominates no individual, thus $S(H) = 0$. The strength of an individual indicates relative domination ability of an individual (in terms of number of dominated individuals).

The second step is the determination of raw fitness. The summation of strength values of a set of individuals that dominates an individual is defined as raw fitness value of the individual. Due to this definition, non-dominated individuals {A,B,C and D} get '0' raw fitness values. For example, raw fitness of A or $R(A) = 0$, because A is not dominated by any individual and similarly $R(H)$ is 11 because all individuals dominate H.

In the third step, density information is added to the raw fitness value according to Eq.7 and the final fitness value is obtained for each individual. Figure 2 depicts the raw and final fitness values of an example population assigned by SPEA2.

3.2.2. Elitism Mechanism of SPEA2

The archive update operation in SPEA2 differs from the one in SPEA in two respects: (i) the number of individuals contained in the archive is constant over time, and (ii) the truncation method prevents boundary solutions being removed. The first step is to copy all non-dominated individuals, i.e., those which have a fitness value lower than one, from archive and population to the archive of the next generation (P - main population, $\overline{P}$ - archive, N - population size, $\overline{N}$ - archive size):

$$\overline{P}_{t+1} = \left\{ \; i \mid i \in P_t + \overline{P}_t \wedge F(i) < 1 \right\} \tag{8}$$

If the non-dominated front fits exactly into the archive ($\left|\overline{P}_{t+1}\right| = \overline{N}$) the environmental selection step is completed. Otherwise, there can be two situations: either the archive is too small ($\left|\overline{P}_{t+1}\right| < \overline{N}$) or too large ($\left|\overline{P}_{t+1}\right| > \overline{N}$). In the first case, the best $\overline{N} - \left|\overline{P}_{t+1}\right|$ dominated individuals in the previous archive and population are copied to the new archive. This can be implemented by sorting the multiset $P_t + \overline{P_t}$ according to the fitness values and copy the first $\overline{N} - \left|\overline{P}_{t+1}\right|$ individuals ***i*** with $F(i) \geq 1$ from the resulting ordered list to $\overline{P}_{t+1}$. In the second case, when the size of the current non-dominated set exceeds $\overline{N}$, an archive truncation procedure is invoked which iteratively removes individuals from $\overline{P}_{t+1}$ until $\left|\overline{P}_{t+1}\right| = \overline{N}$. Here, at each iteration, the individual which has the minimum distance to another individual is chosen for removal; if there are several individuals with minimum distance the tie is broken by considering the second smallest distances and so forth.

4. ACTIVE ELITISM MECHANISM

In the elitism mechanism as stated in [14], a set of elite individuals is excited by genetic operators (crossover/mutation) in archive in the hope of generating better or more diverse individuals than themselves. If a set of excited elites is any better than originals, then the secondary population can be viewed as a place of active solution provider rather than a static storage place. The main motivation behind this approach is that elite individuals are inherently the closest individuals to the solution (of any optimization problem on hand) and exciting those individuals can likely generate more significant outcomes than a far away one. This structure is named as the active elitism mechanism.

It is not always possible to select all elite individuals and to place them into reproduction process by means of selection mechanism. Unselected elite individuals can’t be excited by genetic operators and they can’t generate new solutions. However, the active elitism mechanism gives a chance to all elite individuals for generating new solutions. Block diagram of an active elitist MOEA is given in Figure 3.

Active elitism mechanism is applied to MOEAs in two forms:

i) Excitation of the elite individuals by mutation operator,

ii) The elite individuals in the current archive are recombined with the elite individuals in the *past* archive (**n** step before).

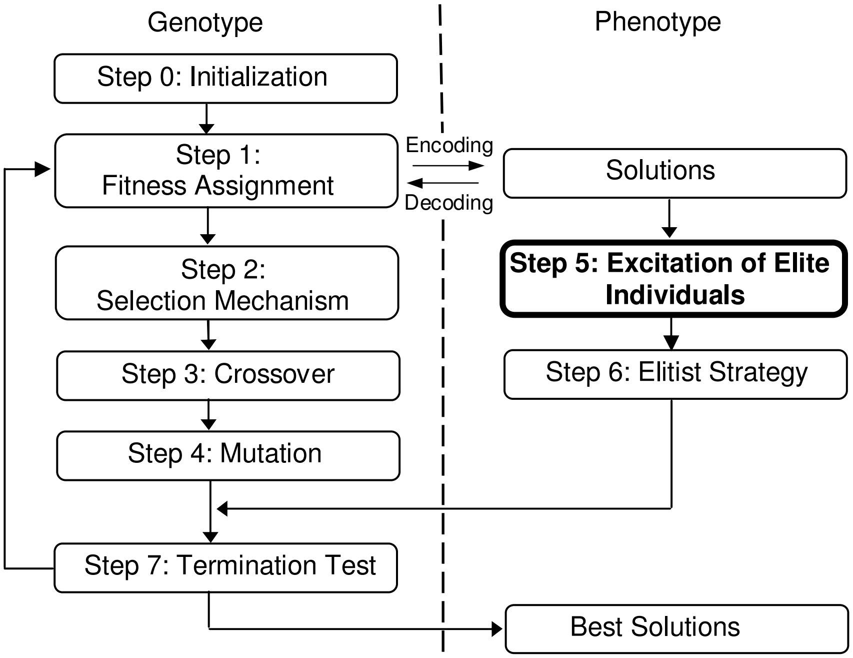

FIGURE 3. Block Diagram of an Active Elitist MOEA.

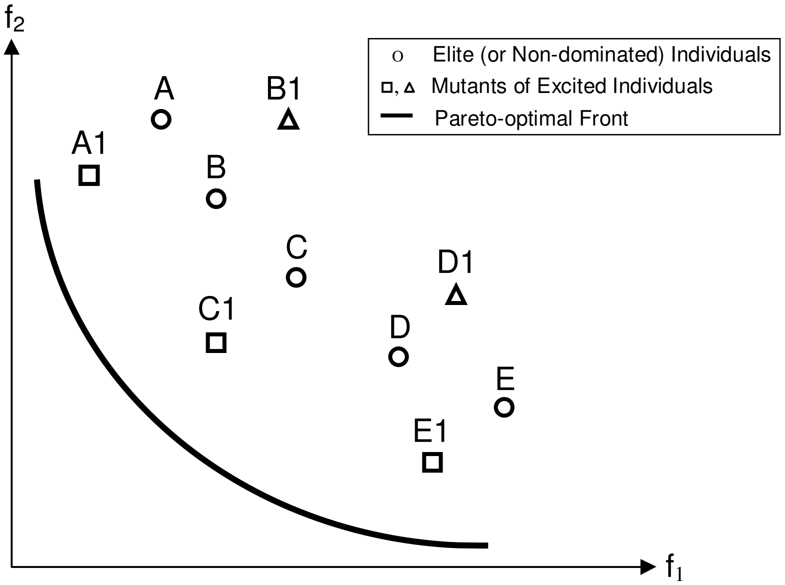

FIGURE 4. Excitation of elite (or non-dominated) individuals by mutation operator.

An example of excitation of elite individuals by the mutation operator is shown in Figure 4. Each individual in the archive (or secondary population) is mutated once. A set of mutants depicted by squares (A1,C1,E1) are closer to the Pareto-optimal front (in terms of two objectives) than the original ones depicted by circles (A,C,E). Thus; A, C and E is replaced by A1, C1 and E1 in the archive. Other mutants depicted by triangles (B1 and D1) are farther from the Pareto-optimal front than the original ones (B and D). Therefore; B and D remain unchanged in the archive. Euclidian distance is used to calculation of the distance of the individuals to the Pareto-optimal front.

An example of excitation of elite individuals by the crossover operator is shown in Figure 5. In this phase, elite individuals in *k*th iteration are recombined to elite individuals in *(k-n)*th iteration. Obtained individuals symbolically depicted by triangles in Figure 5 are better (or closer) than original elites in *k*th iteration. So, they are substituted in the archive. Generally speaking, the archive remains unchanged if offspring (generated by excitation of crossover or mutation) is not any better than the original one.

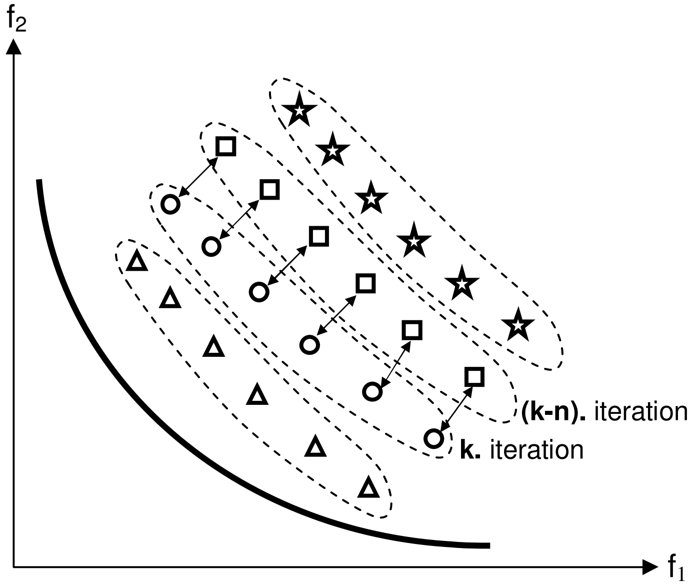

FIGURE 5. The current and past elites are recombined with crossover operation**.** Circles are the elite individuals in *k*th iteration, squares are the elite Individuals in *(k-n)*th iteration, triangles and stars are offsprings.

5. SIMULATION RESULTS

5.1. Test Problems

Four continuous test functions (ZDT1, ZDT2, ZDT3 and ZDT6) which have been used in various MOEA studies are experimented. ZDTs are state-of-art test problems for MOEA comparisons [4],[5]. All test problems with their properties are presented in Table 1. All these problems in Table 1 are bi-objective minimization problems. None of these problems have any inequality or equality constraints. Table 1 also shows the number of variables, their bounds, the Pareto-optimal solutions, and the nature of Pareto-optimal front for each problem.

Problem	n	Variable Bounds	Objectives Functions	Comments
ZDT1	30	$[0,1]^n$	$f_1(x)=x_1$, $f_2(x)=g(x)\left[1-\sqrt{x_1/g(x)}\right]$, $g(x)=1+9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	Convex
ZDT2	30	$[0,1]^n$	$f_1(x)=x_1$, $f_2(x)=g(x)\left[1-(x_1/g(x))^2\right]$ *g(x)* is the same as those of ZDT1	Non-convex
ZDT3	30	$[0,1]^n$	$f_1(x)=x_1$, $f_2(x)=g(x)\left[1-\sqrt{x_1/g(x)}-\frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ *g(x)* is the same as those of ZDT1	Convex, disconnected
ZDT6	30	$[0,1]^n$	$f_1(x)=1-\exp(-4x_1)\sin^6(6\pi x_1)$, $f_2(x)=g(x)\left[1-(f_1(x)/g(x))^2\right]$ $g(x)=1+9\left[\left(\sum_{i=2}^{n} x_i\right)/(n-1)\right]^{0.25}$	Non-convex, non-uniformly spaced

TABLE 1: Test problems used in this paper, *n* is the number of decision variables.

5.2. Parameter Settings

The parameter settings in the experimental studies are given in Table 2. For ZDT1 and ZDT3, decision variables are coded with 30 bits. For ZDT2 and ZDT6, decision variables are coded with 10 bits. The population size of 100 is selected for making comparisons on the same conditions. For SPEA, a population of size 100 and an archive of size 25 are employed (this 4:1 ratio is suggested by the developers of SPEA to maintain an adequate selection pressure for the elite solutions).

	SPEA / ACE_SPEA	SPEA2 / ACE_SPEA2
The number of decision variables (n)	30	30
The mutation rate (p_m)	$1/l$ (where l is the string length)	$1/l$
The crossover rate (p_c)	0.9	0.9
The active mutation rate (p_{m_active})	0.05	0.05
The population size (N)	100	100
The archive size (or the secondary population size) ($\overline{N}$)	25	100
Maximal number of generations	100	100
Number of runs	20	20

TABLE 2: Parameter Settings for all MOEAs.

5.3. Performance Metrics

There are some metrics (or indicators) used for MOEA comparison in the literature. Some of them are C-measure “[4],[5]”, IGD (Inverted generational distance [11], HV (Hypervolume) indicator (S measure in [4], epsilon indicator [15], GD (Generational Distance) and Δ “[2],[9]”. GD and Δ metrics are used in assessing the performance of the algorithms in the experimental studies. The first metric (GD) measures the extent of convergence to a known set of Pareto-optimal solutions. First, a set of uniformly spaced solutions (500 is used in this paper) from the true Pareto-optimal front in the objective space is found. For each solution obtained with a MOEA, we compute the minimum Euclidean distance between it and the chosen solutions on the Pareto-optimal front. The average of these distances is used as the convergence metric (GD) [2],[9]. Figure 6(a) shows the calculation procedure of this metric. The smaller the value of this metric, the better the convergence toward the Pareto-optimal front is.

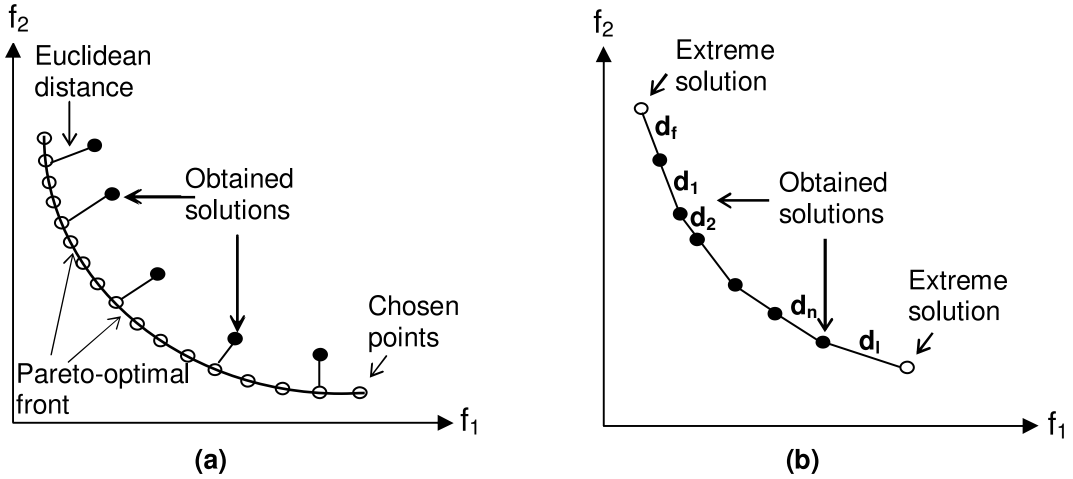

FIGURE 6. (a) Convergence Metric (GD). (b) Diversity Metric (Δ).

The second metric (Δ) measures the extent of spread achieved among the obtained solutions. The Euclidean distance d_i between consecutive solutions in the obtained non-dominated set of solutions is calculated. The average $\overline{d}$ of these distances is then calculated. Thereafter, from the obtained set of non-dominated solutions, we first calculate the extreme solutions (in the objective space) by fitting a curve parallel to that of the true Pareto-optimal front. Then, the following metric is used to calculate the non-uniformity in the distribution [2],[9]:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \overline{d} \right|}{d_f + d_l + (N-1)\overline{d}} \tag{9}$$

Here, the parameters d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set, as depicted in Figure 6(b). The values of these two metrics are desired to be small.

5.4. Discussion of the Simulation Results

In this paper, binary tournament is employed as a selection procedure. The single-point crossover and bit-wise mutation are applied. All methods have been run twenty-times on four benchmarks from the literature. Afterwards, mean and variance values of convergence (GD) and diversity (Δ) metrics are calculated. These two metrics are desired to be small.

In ACE_SPEA (ACtive Elitist SPEA) and ACE_SPEA2 (ACtive Elitist SPEA 2), the elite individuals are excited by mutation operator up to 31 iterations (it is decided by experimentation), after 31 iterations mutation is being ceased. If the mutants of excited elites better than original elites (Euclidian distance is used), they are substituted in the archive, otherwise original elites remain unchanged. At the same time, the elite individuals in *k*th iteration are recombined to the elite individuals in *(k-n)*th iteration. For example, the elites in iteration-20 are recombined sequentially to the elites in iteration-10 as depicted in Figure 3. If the excited individuals are better than the elites in iteration-20, then they are substituted in the archive, otherwise original elites remain unchanged.

Table 3 presents the mean and variance of the GD - metric values of the 20 final populations. Table 4 presents the mean and variance of the Δ - metric values of the 20 final populations. To make an easy reading, the best results in Table 3 and Table 4 are shown in **bold**.

It is evident from Tables 3 and 4 that ACE_SPEA significantly outperforms SPEA in terms of both the convergence (GD) and diversity (Δ) metrics on all test problems. Only on ZDT6 test functions, SPEA performs better than ACE_SPEA in terms of diversity metric.

ACE_SPEA2 significantly outperforms SPEA2 in terms of both the convergence (GD) and diversity (Δ) metrics on all test problems. Only on ZDT2 test functions, SPEA2 performs better than ACE_SPEA2 in terms of diversity metric.

	ZDT1	**ZDT2**	**ZDT3**	**ZDT6**
SPEA	0.0627	0.0133	0.0517	1.2824
	2.1822e-4	2.3891e-4	2.5627e-4	0.2391
ACE_SPEA	**0.0549**	**0.0086**	**0.0488**	**1.1638**
	1.3940e-4	3.3388e-5	8.4167e-5	0.1942
SPEA2	0.0327	0.0044	0.0441	0.7672
	2.5225e-5	5.1226e-6	2.5372e-4	0.0232
ACE_SPEA2	**0.0293**	**0.0031**	**0.0345**	**0.7238**
	1.7667e-5	2.8834e-6	6.5541e-5	0.0410

TABLE 3: Mean (First Rows) and Variance (Second Rows) of the Convergence Metric (GD).

	ZDT1	**ZDT2**	**ZDT3**	**ZDT6**
SPEA	0.6170	0.6189	0.6320	**0.8425**
	0.0047	0.0032	0.0073	0.0079
ACE_SPEA	**0.5945**	**0.6062**	**0.6148**	0.8761
	0.0055	0.0056	0.0059	0.0070
SPEA2	0.5895	**0.6115**	0.6080	0.8285
	0.0041	0.0033	0.0112	0.0060
ACE_SPEA2	**0.5761**	0.6322	***0.5874***	**0.8147**
	0.0038	0.0031	0.0095	0.0089

TABLE 4: Mean (First Rows) and Variance (Second Rows) of the Diversity Metric (Δ).

To demonstrate some of the cases graphically, SPEA with ACE_SPEA and SPEA2 with ACE_SPEA2 are compared on several problems. Figure 7 to 14 shows the non-dominated solutions of all algorithms for arbitrarily chosen single runs of test problems ZDT1 and ZDT2.

The plots of the final solutions in the objective space in the Figure 7,8,9 and 10 also clearly show that the approximations generated by ACE_SPEA are better than those by SPEA on all test problems. Also, ACE_SPEA has a better spread than SPEA.

The plots of the final solutions in the objective space in the Figure 11,12,13 and 14 also clearly show that the approximations generated by ACE_SPEA2 are better than those by SPEA2 on all test problems. Also, ACE_SPEA2 has a better spread than SPEA2.

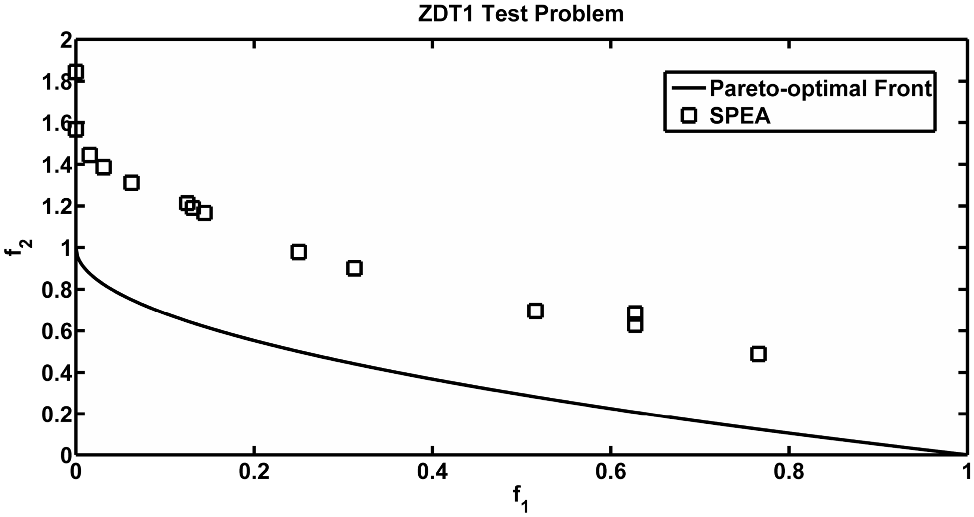

FIGURE 7. Non-dominated solutions of SPEA for a single run of ZDT1.

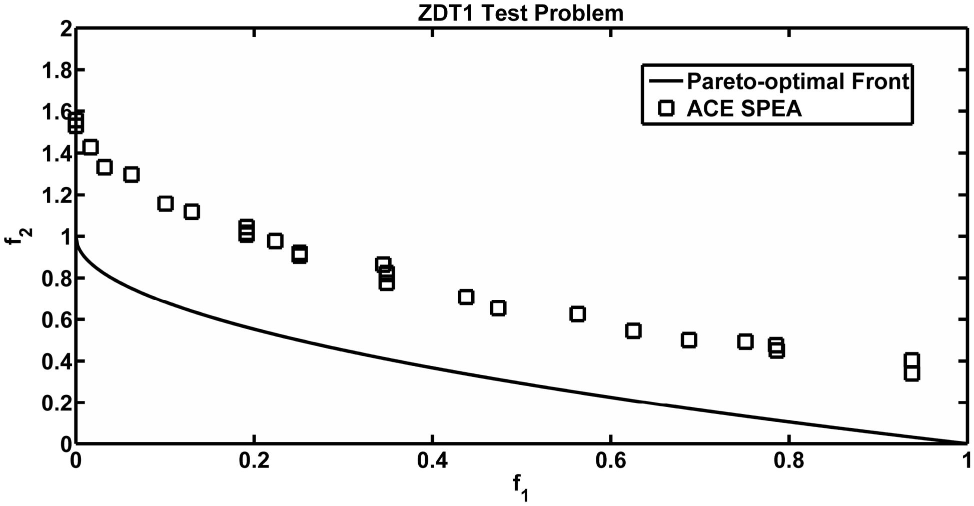

FIGURE 8. Non-dominated solutions of ACE_SPEA for a single run of ZDT1.

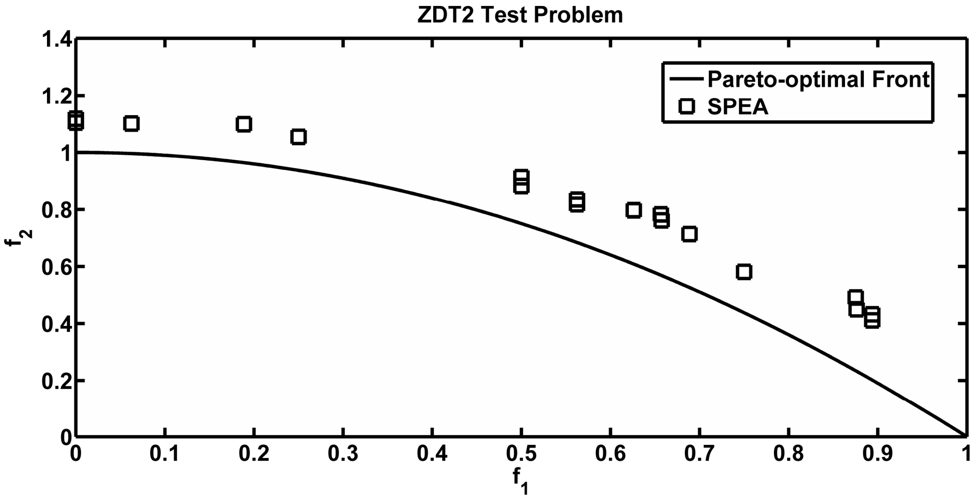

FIGURE 9. Non-dominated solutions of SPEA for a single run of ZDT2.

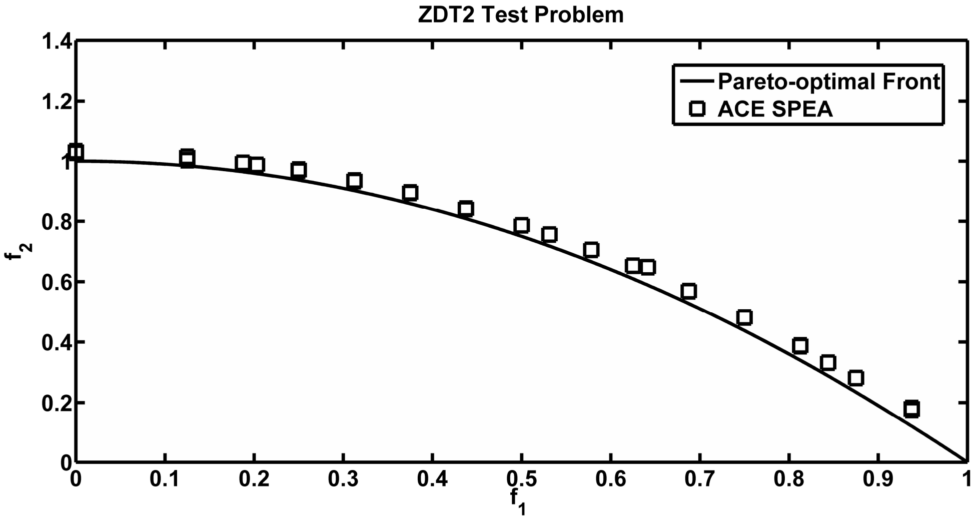

FIGURE 10. Non-dominated solutions of ACE_SPEA for a single run of ZDT2.

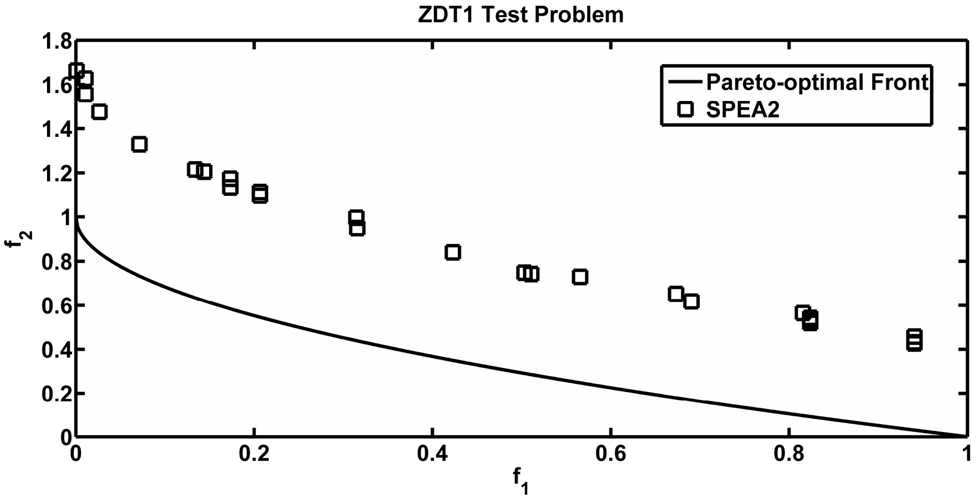

FIGURE 11. Non-dominated solutions of SPEA2 for a single run of ZDT1.

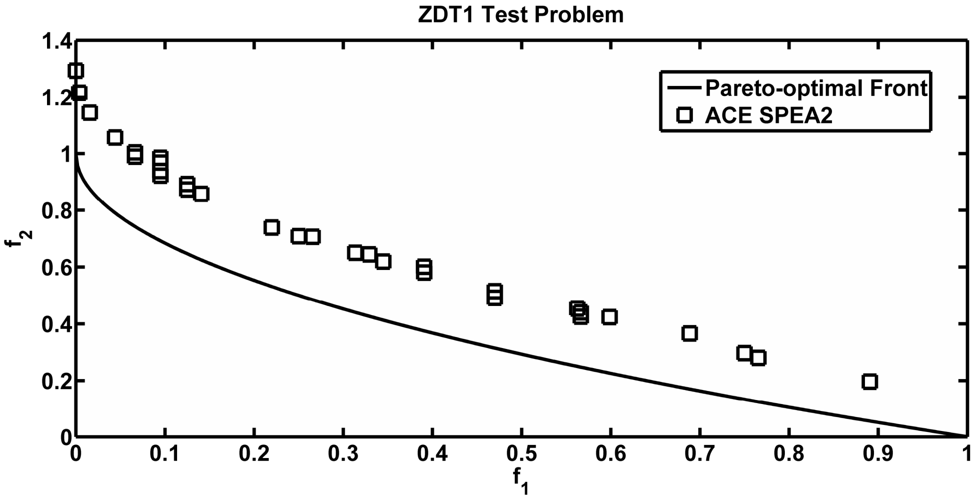

FIGURE 12. Non-dominated solutions of ACE_SPEA2 for a single run of ZDT1.

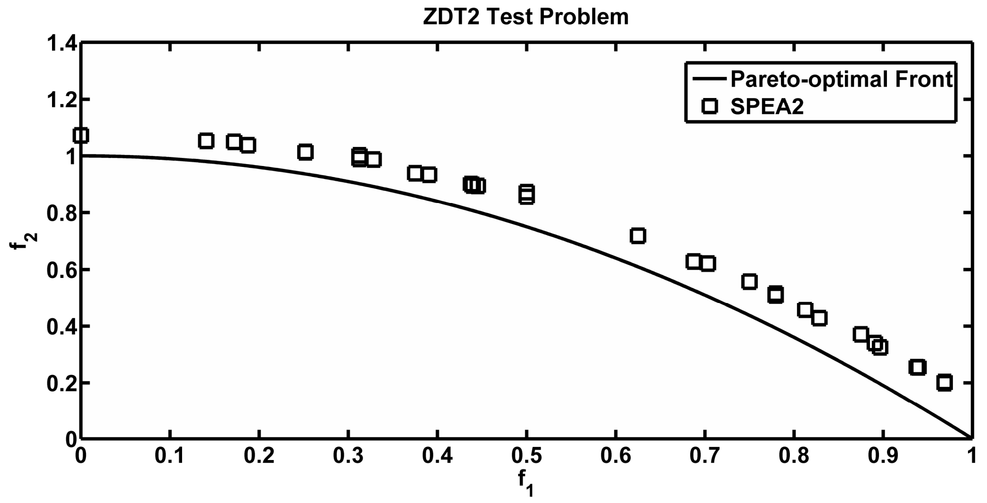

FIGURE 13. Non-dominated solutions of SPEA2 for a single run of ZDT2.

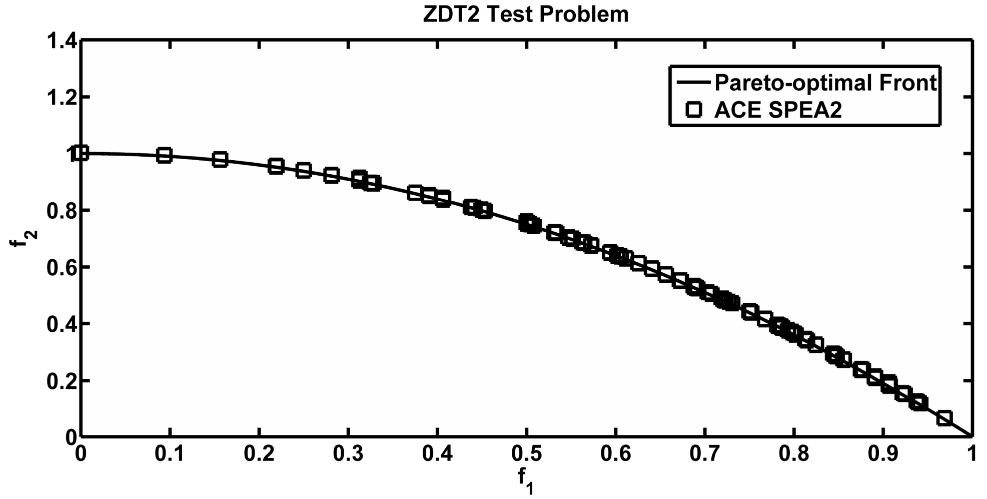

FIGURE 14. Non-dominated solutions of ACE_SPEA2 for a single run of ZDT2.

6. CONCLUSION

In this paper, an active elitism mechanism for multi-objective evolutionary algorithms is proposed. Passive elitism mechanism has been replaced by the proposed active elitism mechanism. The elite individuals are the best individuals and also the nearest individuals to the solution in the current population. Therefore, excitement of a set of elites in the archive with crossover/mutation operator forces them to generate likely better and more diverse than the original elites. In this regards, classical *holding/sending back* passive elitism structure has been improved and evolved into an active elitism structure of *holding/exciting* and *sending back*. It means that, primary and also secondary populations (or archives) can generate solutions in the active elitism.

Well-known SPEA and SPEA2 methods equipped with passive elitism and proposed ACE_SPEA and ACE_SPEA2 methods equipped with active elitism have been simulated and compared on four continuous problems taken from the literature (including ZDT1, ZDT2, ZDT3 and ZDT6). It is found that ACE_SPEA significantly outperforms SPEA on all test problems. Only exception of this is that SPEA generates a slightly better spread on ZDT6 than ACE_SPEA. ACE_SPEA2 is superior to SPEA2 in terms of both the convergence (GD) and diversity (Δ) metrics on all test problems. Only exception of this is that SPEA2 generates a slightly better spread on ZDT2 than ACE_SPEA2.

Simulation results show that the active elitism is especially beneficial to improve the performance of MOEAs and that the use of the active elitism mechanism speeds up convergence to the Pareto-optimal front. In addition to that, proposed active elitism mechanism can easily be integrated into existing MOEAs in the literature to improve their performance without changing their entire computational structures.

In the future work, the active elitism will be embedded to the other MOEAs in the literature. Furthermore, it is planned to speed up the active elitism algorithm.

7. REFERENCES

[1] C.A. Coello Coello. "A Comprehensive Survey of Evolutionary-Based Multi-objective Optimization Techniques". Knowledge and Information Systems, Vol.1, Issue.3, pp. 269-308, 1999.

[2] K. Deb. Multiobjective Optimization Using Evolutionary Algorithms. Chichester, U.K: Wiley, 2001.

[3] Y. Zhenyu, K. Lishan, B. Mckay, F. Penghui. "SEEA for Multi-Objective Optimization: Reinforcing Elitist MOEA through Multi-Parent Crossover, Steady Elimination and Swarm Hill Climbing". Proceedings of the 4th Asia-Pasific Conference on Simulated Evolution and Learning, 2002, pp. 21-26.

[4] E. Zitzler. "Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications". Ph.D Thesis, Swiss Federal Institute of Technology, Zurich, Switzerland, 1999.

[5] E. Zitzler, L. Thiele. "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach". IEEE Trans. on Evolutionary Computation, Vol.3, No. 4, pp. 257-271, 1999.

[6] G. Rudolph. "Evolutionary Search Under Partially Ordered Fitness Sets". Proceedings of the International Symposium on Information Science Innovations in Engineering of Natural and Artificial Intelligent Systems, 2001, pp. 818-822.

[7] G. Rudolph. "Convergence of Evolutionary Algorithms in General Search Spaces". Proceedings of the Third IEEE Conference on Evolutionary Computation, 1996, pp. 50-54.

[8] E. Zitzler, M. Laumanns, L. Thiele. "SPEA2: Improving the Strength Pareto Evolutionary Algorithm".TIK-Report 103, Swiss Federal Institute of Technology, Zurich, Switzerland, May 2001.

[9] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II". IEEE Transactions on Evolutionary Computation, Vol.6, No.2, April 2002.

[10] A. Osyczka, S. Kundu. "A New Method to Solve Generalized Multicriteria Optimization Problems Using The Simple Genetic Algorithm". Structural Optimization,10(2), pp. 94-99,1995.

[11] D.V. Veldhuizen. "Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovaitons". Ph.D. Thesis, Dayton, OH: Air Force Institute of Technology, 1999.

[12] E. Zitzler, M. Laumanns, L. Thiele. "SPEA2: Improving the Strength Pareto Evolutionary Algorithm". TIK-Report 103, Swiss Federal Institute of Technology, Zurich, May 2001.

[13] T. Talaslioglu. “Multi-objective Design Optimization of Grillage Systems According to LRFD-AISC”. Advances in Civil Engineering, 2011. (In Press).

[14] H. Usta. “Effects Of Mutation, Crossover And Improvements in Elitism Mechanism In Genetic Algorithms”. MSc Thesis (in Turkish), Ondokuz Mayis University, 2007.

[15] E. Zitzler , L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. Fonseca. “Performance Assesments of Multiobjective Optimizers: An Analysis and Review”, IEEE Transacitons on Evolutionary Computation, Vol.7, No. 2, pp. 117-132, 2003.

Design of an Analog CMOS Based Interval Type-2 Fuzzy Logic Controller Chip

Mamta Khosla khoslam@nitj.ac.in
Associate professor
Department of Electronics and Communication Engineering
Dr B R Ambekdar National Institute of Technology
Jalandhar – 144011. India

Rakesh Kumar Sarin sarinrk@nitj.ac.in
Professor
Department of Electronics and Communication Engineering
Dr B R Ambekdar National Institute of Technology
Jalandhar – 144011. India

Moin Uddin prof_moin@yahoo.com
Pro Vice Chancellor
Delhi Technological University
Delhi – 110042. India

Abstract

We propose the design of an analog Interval Type-2 (IT2) fuzzy logic controller chip that is based on the realization approach of averaging of two Type-1 Fuzzy Logic Systems (T1 FLSs). The fuzzifier is realized using transconductance mode membership function generator circuits. The membership functions are made tunable by setting some reference voltages on the IC pins. The inference is realized using current mode MIN circuits. The consequents are also tunable by providing five reference current sources on chip. Defuzzification of both the T1 FLSs is based on weighted average method realized through scalar and multiplier-divider circuits. An analog current-mode averager circuit is used for obtaining the defuzzified output of the IT2 fuzzy logic controller chip. The chip is designed for two inputs, one output and nine tunable fuzzy rules and is realized in 0.18 µm technology. Cadence Virtuoso Schematic/Layout Editor has been used for the chip design and the performances of all the circuits are confirmed through the simulations carried out using Cadence Spectre tool. The proposed architecture has an operation speed of 20 MFLIPS and a power consumption of 20mW. The whole chip occupies an area of 0.32 mm^2. As compared to the previous designs, the proposed design has achieved a considerable high speed along with a significant reduction in power and area.

Keywords: Type-2 Fuzzy logic Systems, Interval Type-2 Fuzzy Logic Systems, Footprint of Uncertainty, CMOS, Current Mirror.

1. INTRODUCTION

Type-1 fuzzy logic has been the most popular form of fuzzy logic, and has been successfully used in various domains. However, there are various sources of uncertainties facing T1 FLSs, which are usually present in most of the real world applications. T1 FLSs cannot fully model and handle these uncertainties since they use precise and crisp Type-1 Fuzzy Sets (T1 FSs). However, Type-2 Fuzzy Logic Systems (T2 FLSs), which use Type-2 FSs (T2 FSs) characterized by fuzzy membership functions (MFs), have an additional third dimension. This third dimension and Footprint of Uncertainty (FOU) provide additional design degrees of freedom for T2 FLSs to directly model and handle uncertainties [1]. Thus, T2 FLSs are expected to perform better than their traditional counter parts.

Although T2 FLSs have been used successfully in a number of applications [2-7], their design and implementation is comparatively more difficult, time consuming and slower than T1 FLSs. This is attributed to their much higher computational complexities, difficulty in visualization and use, and non availability of suitable software tools. Thus, the designers cannot reap the benefits of T2 FLSs. Whereas, T1 FLSs are much simpler to design, simulate and realize, and their popularity has been greatly aided by the Graphical User Interface (GUI) based software tools like Fuzzy Logic Toolbox for MATLAB.

Hardware implementation of T1 FLSs is a well-known area [8]. The approaches for implementing these systems cover technologies like microcontrollers, FPGAs, digital and analog VLSI among others [8]-[16]. On the other hand, the hardware realization of T2 FLSs is a relatively nascent research area and a few digital implementations reported in literature have been around microcontrollers, FPGAs etc. [17]-[20]. Digital VLSI implementation was presented by Huang and Chen [21] where the T2 FLS was designed at the transistor level on a single chip based on 0.35 μm technology. Particularly, these implementations have focused on Interval Type-2 Fuzzy Logic Systems (IT2 FLSs), which are a special case of the T2 FLSs and are computationally much simpler than general T2 FLSs. Furthermore, many researchers have validated that IT2 FLS outperforms T1 FLS [2, 22-24].

In this paper, we have designed an analog IT2 fuzzy chip, which is based on the realization methodology of averaging of two T1 FLSs. This methodology has been validated though two case studies by the authors [25] and has also been adopted for the implementation of IT2 FLSs on FPGAs [26, 27]. To the best of our knowledge, there is no report of an analog CMOS based hardware realization of an IT2 FLS in the literature. Analog implementation is superior to digital implementation in terms of processing speed, power dissipation and chip size. The main drawback of analog circuits is their comparatively low accuracy than the digital circuits, which however, is not a severe limitation in view of the typical demands of most fuzzy applications. The main processing stages of the IT2 FLS viz. fuzzification, rule inference, defuzzification all are realized using analog circuits designed in UMC 180 MMRF CMOS (180nm 1P/6M 3.3V) technology. The workings of all the modules are verified through the simulations carried out in Cadence Spectre tool. The synthesis of the modules as a two input, one output, nine rules FLS is simulated and the results demonstrate an inference speed of 20MFLIPS and power consumption of 20mW.

The paper is organized in five sections. Section 2 briefly describes the IT2 FSs and the working of IT2 FLSs. In Section 3, we discuss the design of the IT2 processor in detail; the realization methodology followed for designing IT2 FLS using T1 FLSs is discussed; the circuits of all the analog modules used in the design and their simulation results are presented under this section. In Section 4, the design and performance of analog IT2 fuzzy chip is presented, that has been obtained by combining the various modules presented in Section 3. Finally, Section 5 concludes the paper.

2. OVERVIEW OF IT2 FSS AND IT2 FLSS

2.1 Generalized T2 FSs and Interval T2 FSs

A T2 FS can be informally defined as a fuzzy set that is characterized by a fuzzy or non-crisp membership function. This means there is uncertainty in the primary membership grades of a T2 MF, which introduces a third dimension to the MF, defined by the secondary membership grades [28, 29].

Such a T2 FS, denoted by $\tilde{A}$ can be expressed mathematically as in (1)

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

Where, $\mu_{\tilde{A}}(x,u)$ is the T2 MF, and $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$; *x*, the *primary variable*, has domain *X*; $u \in U$, the *secondary variable*, has domain J_x at each $x \in X$; J_x is called the *primary membership* of *x* and $u \in J_x \subseteq [0,1]$

Uncertainty in the primary memberships of a T2 FS consists of a bounded region which is called the Footprint of Uncertainty (FOU). All the embedded FSs of FOU are T1 FSs and their union covers the entire FOU, [1] as in (2)

$$FOU(\tilde{A}) = \bigcup_{x \in X} J_x \tag{2}$$

IT2 is a special case of a T2 FS where all the secondary membership grades equal one. IT2 FS is completely characterized by its 2-D FOU that is bound by a Lower MF (LMF) and an Upper MF (UMF), $\underline{\mu}_{\tilde{A}}(x)$ and $\overline{\mu}_{\tilde{A}}(x)$, respectively, both of which are T1 MFs. The FOU of an IT2 FS is described in terms of these MFs, as in (3).

$$FOU(\tilde{A}) = \bigcup_{x \in X} [\underline{\mu}_{\tilde{A}}(x), \overline{\mu}_{\tilde{A}}(x)] \tag{3}$$

IT2 FSs are the most widely used T2 FSs to date, used in almost all applications because all calculations are easy to perform. Because of the computational complexity of using a general T2 FLS, most designers only use IT2 FSs in a T2 FLS, the result being an IT2 FLS. LMF and UMF together are popularly used in most of research papers to represent IT2 FLSs [28].

2.2 Working of IT2 FLS

A general block diagram for a T2 FLS is depicted in Fig. 1 [28]. It is very similar to a T1 FLS, the major structural difference being that the defuzzifier block of a T1 FLS is replaced by the Output Processing block in a T2 FLS. This block consists of a Type-Reduction sub-module followed by a Defuzzifier.

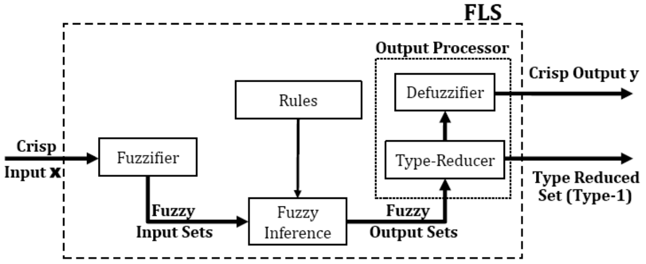

FIGURE 1: A T2 FLS block diagram.

An IT2 FLS is an FLS, where all of the consequent and antecedent T2 FSs are IT2 FSs. Hence, the working of an IT2 FLS is also similar to that of a general T2 FLS, as depicted in Fig.1. The IT2 FLS works as follows: the crisp inputs are first fuzzified into IT2 FSs, which then activate the inference engine and the rule base to produce output IT2 FSs. These IT2 FSs are then processed by a type-reducer. Type-reduction basically represents mapping of T2 FS into a T1 FS that is called a type-reduced set. A defuzzifier then defuzzifies the type-reduced set to produce crisp outputs [29].

3. DESIGN OF ANALOG MODULES FOR IT2 FLS

3.1 Realization Methodology for IT2 FLS with T1 FLSs

As mentioned in Section II, an IT2 FS can be completely characterized by its 2-D FOU, which in turn can be represented in terms of two T1 FSs. There are two approaches for obtaining these T1 FSs and the corresponding T1 FLSs as shown in Fig 2.

a) In the first approach, one T1 FLS can be formed with the LMFs of all the input and output IT2 FSs and the second T1 FLS with their corresponding UMFs. UMF and LMF are the outer and inner envelopes of the FOU respectively as shown in Fig 2.
b) In the second approach, one T1 FLS can be obtained with the Left FSs of all the input and output IT2 FSs and the second T1 FLS with their corresponding Right FSs. These Left and Right FSs are represented with bold red and blue lines respectively in Fig 2.

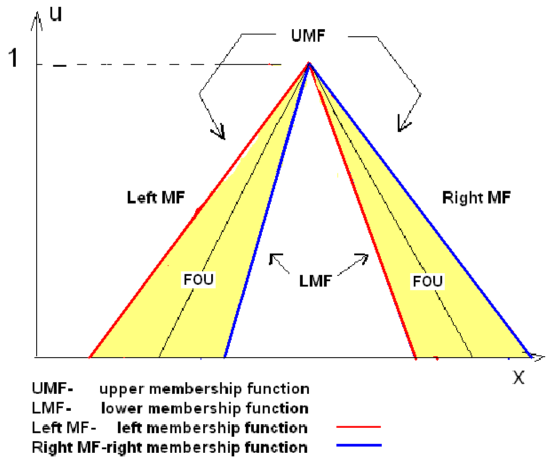
u
1
UMF
Left MF
Right MF
FOU
LMF
FOU
x
UMF- upper membership function
LMF- lower membership function
Left MF- left membership function 🟥
Right MF-right membership function 🟦

FIGURE 2: FOU of an IT2 FS.

Authors have proposed and validated [25] that IT2 FLS can be realized with the average of two T1 FLSs, where two T1 FLSs were formed based on the first approach as described above. For validation, this methodology was applied on (i) an arbitrary system of two inputs, one output and nine rules, and (ii) the Mackey-Glass time-series forecasting. In the second case study, T1 FLS was evolved using Particle Swarm Optimization (PSO) algorithm for the Mackey-Glass time-series data with added noise, and was then upgraded to IT2 FLS by adding FOU. Further, four experiments were conducted in the second case study for four different noise levels. For each case study, a comparative study of the results of the average of two T1 FLSs and the corresponding IT2 FLS, obtained through computer simulations in MATLAB environment validated that IT2 FLS performance is equivalent to the average of two T1 FLSs; that proves the effectiveness of the realization approach.

The design of IT2 fuzzy logic controller chip presented in this paper is based on the architecture shown in Fig. 3. This architecture uses two T1 FLSs to emulate an IT2 FLS and uses the first approach for obtaining two T1 FLSs. Here, the first T1 FLS is constructed using UMFs and the second one with the LMFs so as to emulate the FOUs of all IT2 FSs in an IT2 FLS. The fuzzification, fuzzy inference and defuzzification are done as traditionally for two T1 FLSs and the outputs are then averaged to yield the final output of the IT2 FLS. The advantage of using this realization methodology is that it avoids the complications and intensive computations required for type reduction.

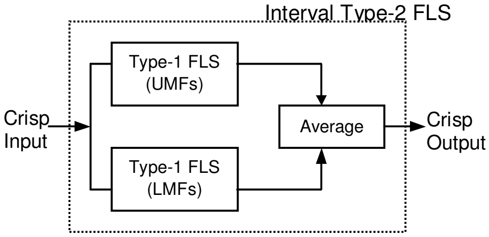

FIGURE 3: Realization Methodology for IT2 FLS with T1 FLSs.

3.2 Analog Functional Blocks of IT2 FLS

In this section, we describe the complete structure of the designed IT2 fuzzy processor in detail. A zero order TSK fuzzy model is used for implementing each T1 FLS i.e. the rule consequents are constant values called singletons and each rule has the format described in (4).

IF (x is A) AND (y is B)
THEN z = c (4)

In the above, *x* and *y* are input variables, *A* and *B* are linguistic variables of *x* and *y,* defined by FSs. Furthermore, *z* is an output variable and *c* is some constant. The output is computed from a weighted average represented by (5), in which each consequent value z_i is weighted by the activation degree w_i of its corresponding rule, α_i being the weight associated to i^{th} rule.

$$Output = \frac{\sum_i \alpha_i w_i z_i}{\sum_i \alpha_i w_i} \tag{5}$$

The complete schematic arrangement for the hardware implementation of the IT2 FLS is shown in Fig. 4. It has the following functional blocks:

a) Fuzzifier block fuzzifies the inputs and it contains membership function generators (MFGs) that generate MFs of different shapes viz. Z, trapezoidal, triangle and S.
b) MIN circuit is used in the inference engine for computing the activation degree of each rule.
c) Scalar circuits are used to weight the singleton consequents.
d) Multiplier-Divider circuits are used for calculating the defuzzified output of each T1 FLS.
e) Averager circuit calculates the defuzzified output of the IT2 processor by computing the average of the two defuzzified values obtained from both T1 FLSs.

In the present work, an IT2 fuzzy chip for two input variables, partitioned into three FSs, and one output having five singletons is designed. Therefore, each T1 FLS viz. the T1 FLS (UMFs) and the T1 FLS (LMFs), have 2 inputs (3 MFs for each input) and 1 output (5 singletons). We use MIN method for the inference engine of T1 FLSs. For the defuzzification of each T1 FLS, weighted average method is used. Detailed description of the circuits used for each functional block of the IT2 fuzzy chip is given below.

3.2.1 Fuzzifier Circuit [31]

Fuzzifier, which converts a crisp input to a fuzzy set, is the first stage in a fuzzy controller. We have used transconductance mode CMOS based circuits for implementing the fuzzifier block and its schematic is shown in Fig. 5. It consists of two differential amplifiers with one PMOS current mirror load. V_{ref1} and V_{ref2} are the control voltages that are fed to one input of each differential pair. And V_{IN} is applied to the second inputs of both the differential pairs. I_{out} can be written as in (6).

$$I_{out} = I_{D2} + I_{D4} \tag{6}$$

Since all transistors in this circuit operate in saturation region i.e. $V_{GS}>V_T$ and $V_{DS}>V_{GS}-V_T$ for each MOS transistor, therefore their drain currents can be defined by (7) and (8).

$$I_{D_{1,2}} = K_1(V_{GS_1} - V_T)^2 \tag{7}$$

$$I_{D_{3,4}} = K_2(V_{GS_2} - V_T)^2 \tag{8}$$

$$\text{where } K_1 = \frac{K(W/L)_1}{2},\ K_2 = \frac{K(W/L)_2}{2},\ K \text{ is the transconductance parameter}$$

$$\mathrm{V_{GS_1} = V_{IN} - V_{ref1}} \quad \text{and} \quad \mathrm{V_{GS_2} = V_{IN} - V_{ref2}}$$

$(W/L)_1$ =size of M1 & M2, $(W/L)_2$ =size of M3 & M4

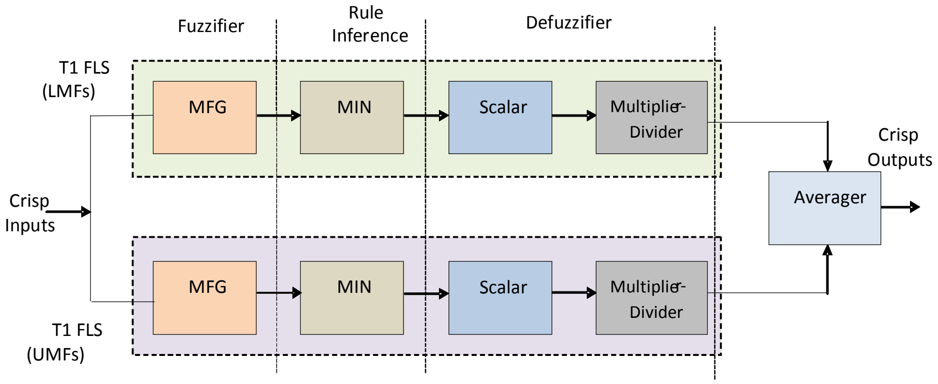

FIGURE 4: Functional Blocks of IT2 FLS.

For MOS transistors operating in saturation region, the drain currents can be approximated in a quadratic form [14]. So (7) and (8) are written in quadratic from and are given in (9) and (10) respectively.

$$I_{D_{1,2}} = \frac{I_S}{2} \pm \frac{\alpha_1}{2}\sqrt{2\beta_1 - \beta_1^{\,2}\alpha_1^{\,2}} \tag{9}$$

$$I_{D_{3,4}} = \frac{I_S}{2} \pm \frac{\alpha_2}{2}\sqrt{2\beta_2 - \beta_2^{\,2}\alpha_2^{\,2}} \tag{10}$$

+ sign for I_{D1} and I_{D3}
– sign for I_{D2} and I_{D4}

Where α_1, α_2, β_1 and β_2 are defined as in (11) and (12)

$$\alpha_1 = \frac{V_{IN} - V_{ref_1}}{V_T} \quad , \qquad \alpha_2 = \frac{V_{IN} - V_{ref_2}}{V_T} \tag{11}$$

$$\beta_1 = \frac{K\,V_T}{2\,I_S}\left(\frac{W}{L}\right)_1 \quad , \qquad \beta_2 = \frac{K\,V_T}{2\,I_S}\left(\frac{W}{L}\right)_2 \tag{12}$$

Using the values of I_{D2} and I_{D4} as obtained from above equations and putting them in (6), the output current of the circuit can be written as (13)

$$I_{out} = I_S - \frac{\alpha_1}{2}\sqrt{2\beta_1 - \beta_1^{\,2}\alpha_1^{\,2}} - \frac{\alpha_2}{2}\sqrt{2\beta_2 - \beta_2^{\,2}\alpha_2^{\,2}} \tag{13}$$

Thus α and β are the two control parameters of this circuit, which tune the position and slope of the MF respectively. The values of these parameters should be so chosen as to obtain the desired shape of the MF. As suggested by (11), the value of α can be varied by varying the value of V_{ref} for each differential pair. Similarly (12) suggests that β can be changed by changing the (*W/L)* of the differential pairs.

The results from Cadence Spectre simulation run for trapezoidal, S and Z shapes implemented by the fuzzifier circuit are shown in Figs. 6 (a) to (d). For trapezoidal and triangular shapes, the characteristics of I_{out} are shifted up because two currents I_{D2} and I_{D4} are added up. Suitable current mirrors are used to scale output currents of all MFGs in the same range. Figs. 6 (b) and 6 (c) show how the programmability of Z and S shaped MFs can be affected by varying the difference in V_{ref1} and V_{ref2}. Fig. 6 (d) shows the slope tuning of a trapezoidal MF. By varying $(W/L)_1$, the left hand slope of this curve changes and by varying $(W/L)_2$, the right hand slope of the curve changes. Thus by varying both the *(W/L)* ratios together, the width of the curve can be changed. Similarly, the slopes of Z and S MFs can be changed. When symmetrical MFs are desired, the $(W/L)_1$ must match $(W/L)_2$. All MFs are symmetrical in the current design.

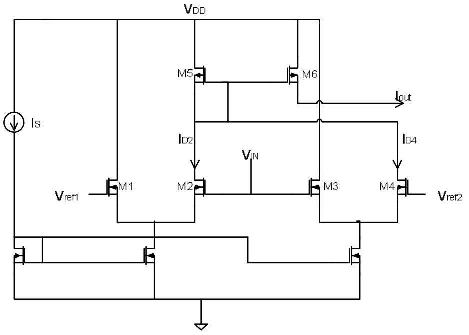

FIGURE 5: Membership Function Generator (MFG) circuit.

3.2.2. MIN-MAX Circuits

The most popular fuzzy logic operators used to compute the inference of a rule are logical “AND” and logical “OR”. MIN and MAX modules can be used to implement the AND and OR operations respectively. We have used current mode MIN circuits to implement the rule base. One MIN is required for calculating the inference of each rule. The circuit schematics of a two-input MIN is shown in Fig. 7 (a) [16]. It consists of MAX circuit block as shown in Fig. 7 (b) with extra current sources to complement the directions of currents [32]. Transistors M_1 and M_3 are source follower transistors. M_2 and M_4 are current sensor transistors that can sink high current. The value of V_{Bias}, which is applied to M_1, M_3 and M_5 transistor gates, is calculated from (14).

$$V_{Bias} = V_{GS_{1,3}} + \Delta V_{2,4} \qquad (14)$$

Where, V_{GS}= Transistor gate-source voltage, and Δ=overdrive voltage

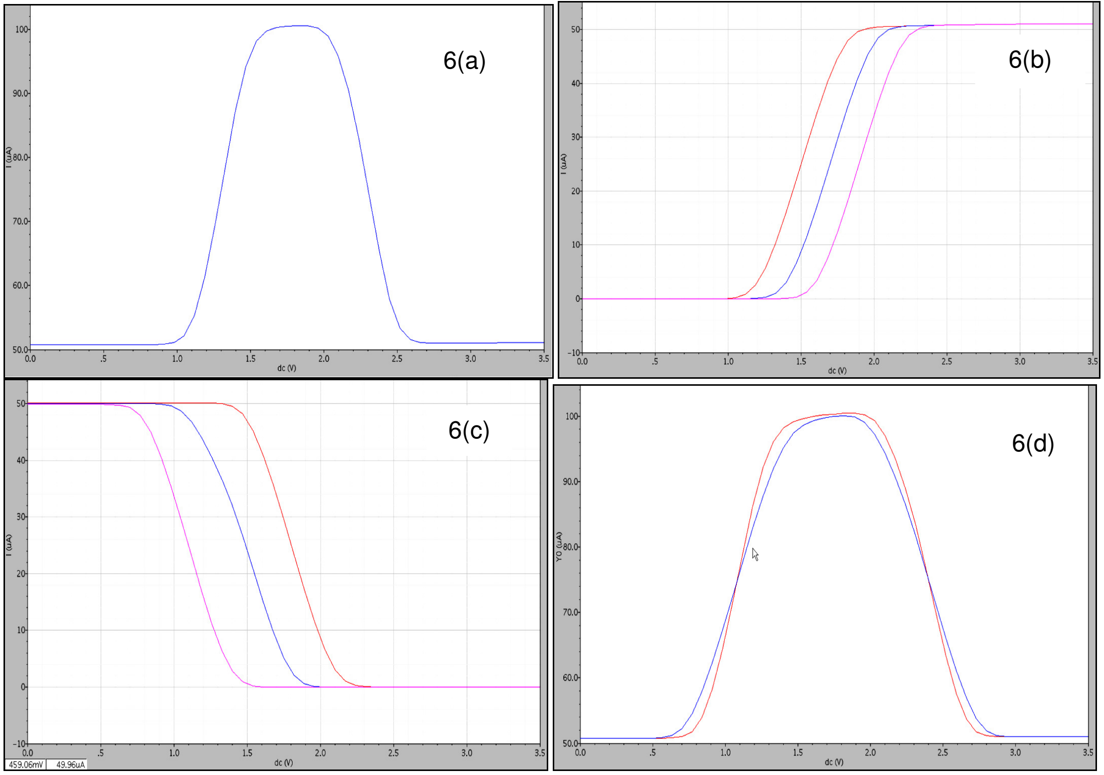

FIGURE 6: (a). Trapezoidal curve obtained through simulation of MFG circuit V_{ref1}=1.5V, V_{ref2}=2V, *W/L*=5 6(b). S-shaped curve obtained through simulation of MFG circuit V_{ref1}=0V, V_{ref2}=1V, 1.3V,

1.5V, *W/L*=3 6(c). Z-shaped curve obtained through simulation of MFG circuit V_{ref2}=0V, V_{ref1}=500mV, 1V, 1.5V, *W/L*=3 6(d). Slope tuning of trapezoidal MF V_{ref1}=1.5V, V_{ref2}=2V, *W/L*=5,3.

If $I_1>I_2$ in the MAX circuit, M_1 and M_2 transistors will be in the saturation region, M_3 and M_4 will be in triode and cutoff regions respectively because of current mirror circuits. Thus, M_1 current I_1 would mirror in to the output. The MIN circuit operation is very similar to the MAX circuit, with the difference that the currents I_1 and I_2 are being stolen from the transistors M_1 and M_3. Therefore, in the MIN circuit, the branch from which we steal lesser current would mirror its current into the output. These circuits can work with low power supply; the minimum power supply for these circuits is calculated from (15).

$$V_{DD\min} = V_{GS} + 2V_{DS(Sat)} \tag{15}$$

Since there are a large number of MIN circuits used in a fuzzy controller, power consumption of chip will be decreased significantly with these MIN circuits working on low voltage. Size of each MIN depends upon the number of inputs only and we can increase the number of inputs of these circuits only by adding two transistors for each input such as M_1 and M_2. The design presented here targets 2 inputs, and therefore, two inputs MIN circuits are required. Inputs to each MIN circuit are the outputs of two MFGs from the fuzzifier block, which correspond to the antecedent part of the rule in consideration and the output of each MIN is the firing rule strength w_i of that rule.

Simulation results of a two input MIN circuit are given in Figs. 8 (a) and (b). Fig. 8 (a) is the DC output characteristic for different values of I_2 and Fig. 8 (b) is the transient response for two different shapes of I_1 and I_2. Fig. 9 (a) and (b) are the DC output characteristics and transient response respectively for a two input MAX circuit.

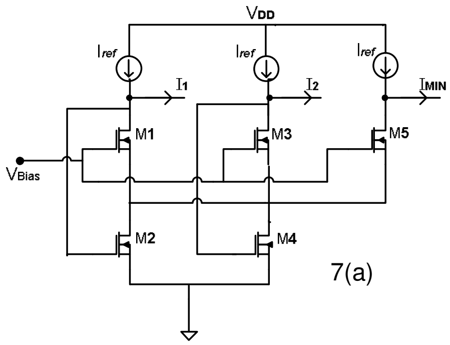

FIGURE 7 (a): MIN circuit.

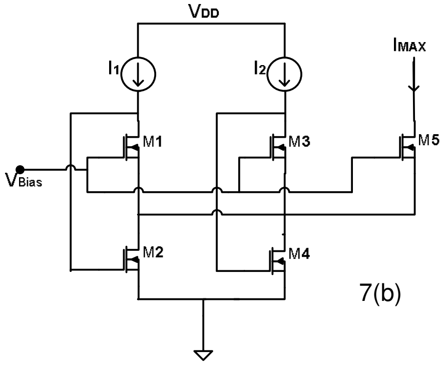

FIGURE 7 (b): MAX circuit.

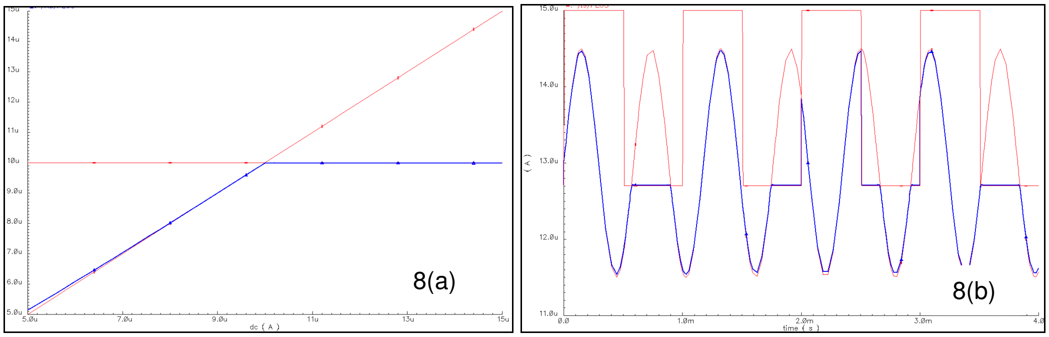

FIGURE 8: (a). DC response of MIN circuit with two inputs
(b). Transient response of MIN circuit with two inputs

3.2.3. Scalar Circuit

Scalar circuit provides many current sources of scaled value of the input current. Scalar circuit is based on current mirror as shown in Fig. 10. I_{in} is the input current and I_{o1}, I_{o2},…, I_{oi} are the output currents of 1st, 2nd, and i^{th} stage mirrors, respectively. Since, transistor M_1 is in saturation region, I_{in} can be written as (16).

$$I_{in} = 0.5K\ (W/L)_{in}(V_{GS} - V_T)^2 \qquad (16)$$

Current through the i^{th} current mirror can be written as (17).

$$I_{oi} = 0.5K\ (W/L)_i(V_{GS} - V_T)^2 \qquad (17)$$

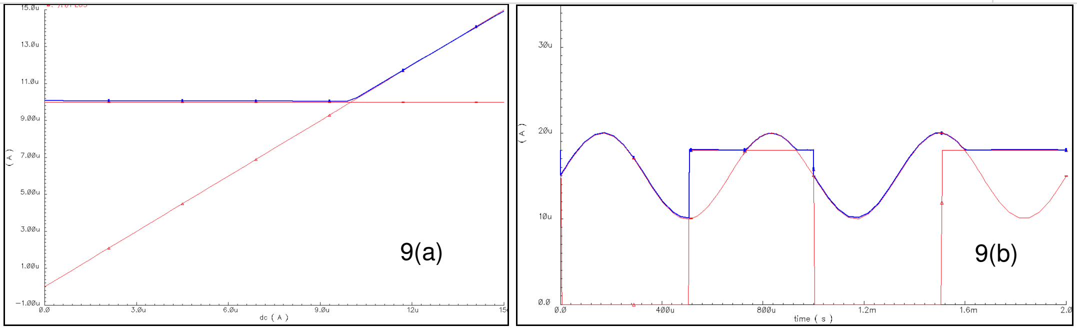

FIGURE 9: (a). DC response of MAX circuit with two inputs
(b). Transient response of MAX circuit with two inputs.

From (16) and (17), I_{oi} can be simplified to (18).

$$I_{oi} = \alpha_i I_{in} \qquad (18)$$

Where, α_i is the scaling factor of the ith stage current mirror, given by (19).

$$\alpha_i = \frac{(W/L)_i}{(W/L)_{in}} \qquad (19)$$

Response of the scalar circuit is shown in Fig. 11 for different values of α (0.5, 1, and 2).

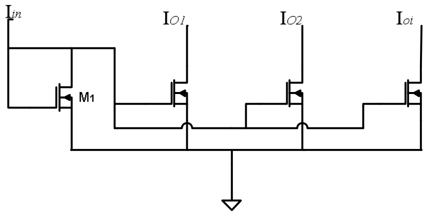

FIGURE 10: Scalar circuit

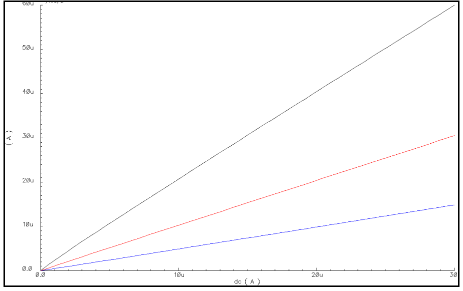

FIGURE 11: Response of Scalar circuit (for α =0.5, 1, and 2).

3.2.4. Multiplier-Divider Circuit

Multiplier-divider circuit shown in Fig. 12 is used in the defuzzifier section [15]. It works on the principle of translinear circuits where all the transistors are operating in saturation region. The output of the circuit can be expressed as (20)

$$I_{out} = \frac{I_{in} I_{b1}}{I_{b2}} \qquad (20)$$

Block diagram of the defuzzification scheme followed here is shown in Fig. 13. It consists of scalar circuits in the first stage. The scalar takes the rule strength w_i calculated from the MIN circuit as the input current, and generates the weighted rule strength $\alpha_i w_i$. Outputs of all scalars are wired to produce the sum of these weighted rule strengths. The resultant current output I of the current mirror is given by (21).

$$I = \sum_i \alpha_i w_i \tag{21}$$

Inputs to the multiplier-divider circuits are the corresponding values of the weighted rule strengths $\alpha_i\, w_i$, the corresponding consequent z_i, and the sum I. Each multiplier-divider circuit multiplies $\alpha_i\, w_i$ with corresponding z_i, and divides by I. The outputs of all multiplier-divider circuits are wired to give the global defuzzified output, as given in (5).

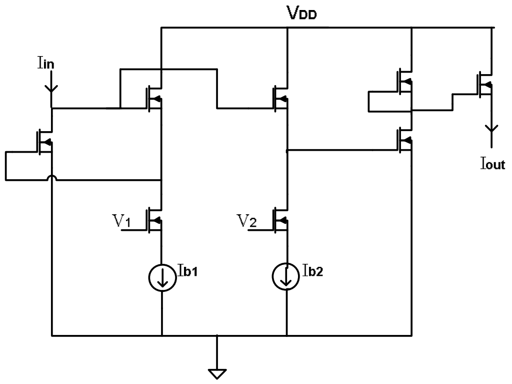

FIGURE 12: Multiplier-Divider circuit.

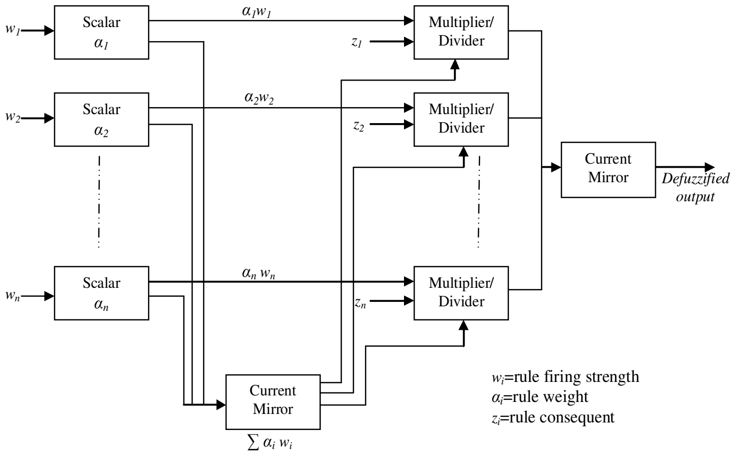

FIGURE 13: Block diagram of Defuzzifier.

The performance of Multiplier-Divider circuit is tested as a multiplier by fixing the values of I_{in} and I_{b2}, and sweeping the values of I_{b1}. Fig. 14 (a) shows the simulation run for three different values of I_{in} viz. 10µA, 12µA, 15µA, I_{b2} fixed at 10µA, and I_{b1} swept across from 5µA to 40µA. The same circuit is tested as a divider by fixing the values of I_{in} and I_{b1}, and sweeping the values of $I_{b2.}$ Fig 14 (b) shows the simulation run for three different values of I_{in} viz. 5µA, 10µA, 15µA, I_{b1} fixed at 10µA, and I_{b2} swept across from 5µA to 40µA.

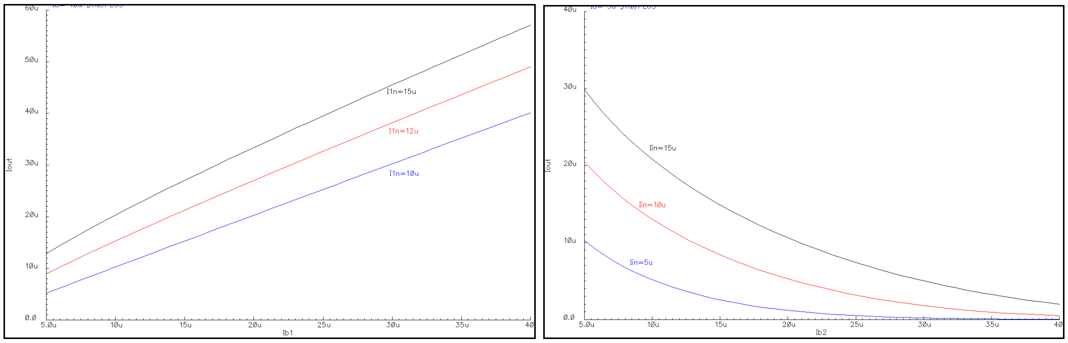

FIGURE 14: (a). Multiplier-Divider circuit acting as multiplier (I_{in} =10µA, 12µA, 15µA, I_{b2} = 10µA).
(b). Multiplier-Divider circuit acting as divider (I_{in} =5µA, 10µA, 15µA, I_{b1} = 10µA).

3.2.5 Averager Circuit

The averager circuit computes the average of the defuzzified outputs of two T1 FLSs, which is the final defuzzified output of the IT2 FLS. The averager circuit works on the principle of current mirror. Defuzzified outputs of both the T1 FLSs are wired so that the sum of both becomes the drain current of M_1 as shown in Fig. 15 and as represented by (22). Sizes of M_1 and M_2 are related by (23).

$$I_{D1} = I_1 + I_2 \tag{22}$$

$$(W/L)_2 = \frac{1}{2}(W/L)_1 \tag{23}$$

$$\text{Therefore,}\quad I_{D2} = \frac{I_1 + I_2}{2} \tag{24}$$

Current output from M_2 is the average of the two input currents I_1 and I_2, where I_1 is the output current from T1 FLS (UMFs) and I_2 is the output current from T1 FLS (LMFs). Thus, this circuit gives the average of the two T1 FLSs. Fig. 16 shows the simulation result of Averager circuit.

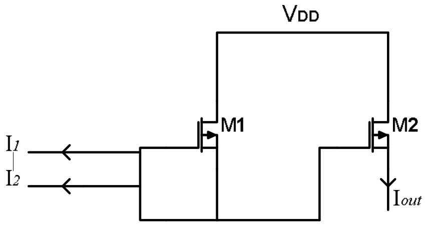

FIGURE 15: Averager circuit

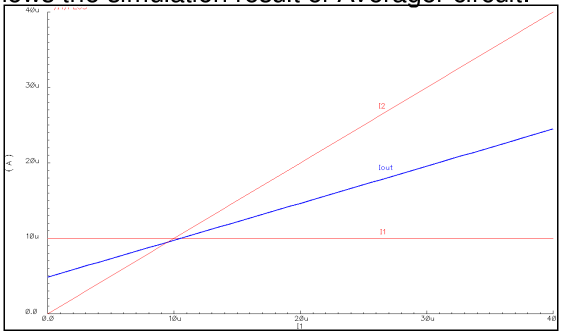

FIGURE 16: Response of Averager circuit.

4. ANALOG IT2 FUZZY LOGIC CONTROLLER CHIP

In this section, fuzzy functional blocks which have been described in the previous section, are combined into an IT2 fuzzy chip and the arrangement is shown in Fig. 17. Current mirrors are used wherever required to change the current directions. Both T1 FLSs differ only in the designs of their fuzzifiers, specifically, the sizes of the differential pair MOS transistors of the MFGs. For generating two different slopes corresponding to the UMFs and LMFs of the FOUs, *W/L*=4 and *W/L=3* respectively are selected. Designs of all other modules viz. MIN, scalar, defuzzifier are same in both T1 FLSs of the IT2 fuzzy chip.

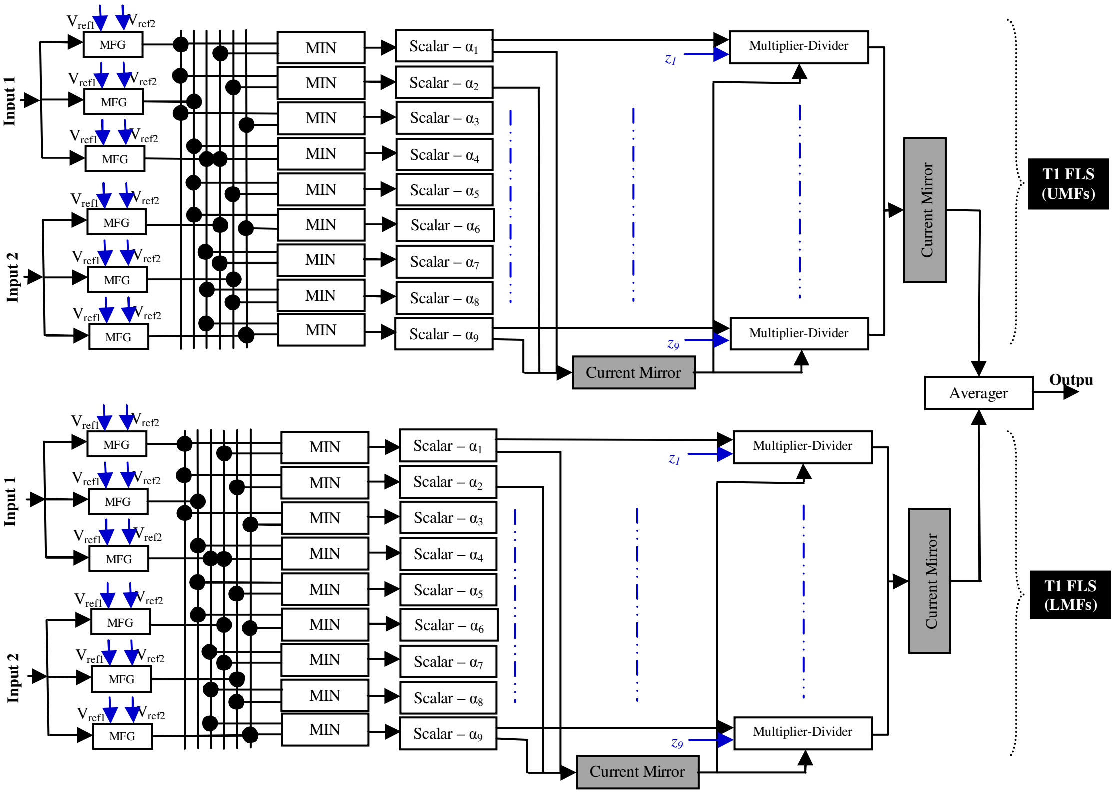

FIGURE 17: Arrangement of Fuzzy functional blocks for IT2 Fuzzy Chip realization

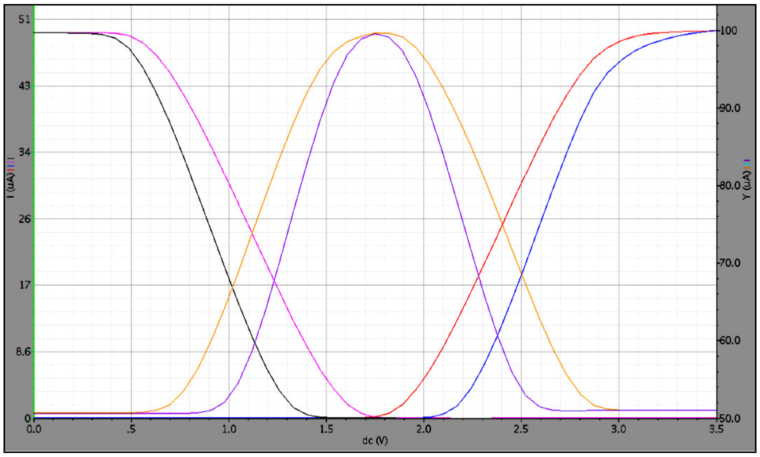

FIGURE 18: UMFs and LMFs of 3 FOUs for one variable obtained through simulation of fuzzifier circuit *W/L*=4 for UMFs, 3 for LMFs

Voltage Sources	0.9V
	1.0V
	1.2V
	2.1V
	2.2V
	2.4V
	2.5V

Current Sources	10μA
	20μA
	30μA
	40μA
	50μA

TABLE 1: On-chip Voltage and Current sources.

LMFs	Vref1	Vref2
1	1	5
2	2.2	1.0
3	0	2.4

UMFs	Vref1	Vref2
1	0.9	5
2	2.1	1.2
3	0	2.5

TABLE 2: Reference Voltage (V) Settings

Pins Details	**Number of Pins**
V_{DD}	1
GND	1
Inputs	2
Output	1
Consequents	5
On-chip Current Sources	5
On-chip Reference Voltage Sources	7
Vref1 and Vref2 for all the MFGs for T1 FLS (UMFs) and T1 FLS (LMFs) 2*[2*3+2*3]	24
Total	**46**

TABLE 3: External pins of IT2 Fuzzy Logic Controller Chip

4.1. Pulse Response of IT2 Fuzzy Logic Controller Chip

In order to determine the speed of the chip, a square pulse is applied to one input, while the other input is set to 0V. The input MFs for this test are shown in Fig. 18. Rule base for both T1 FLSs is taken arbitrarily and is listed in Table 4 in indexed form. The numbers in the input and output columns refer to the index number of membership functions.

Results of this test obtained through Cadence Spectre Simulation are shown in Fig. 19. The response of this chip to pulse input shows a maximum delay of 50ns. This corresponds to a speed of 20 MFLIPS (mega fuzzy logic inferences per second) including the defuzzification process. Since, rule by rule architecture has been followed in this realization; the fuzzy inferences are performed in parallel. Hence, the inference speed is independent of the number of rules and number of MFs. This speed is in a good range for most applications. The chip occupies an area of 0.32 mm^2.

Rule Number	**Input #1**	**Input #2**	**Output**
1	1	1	1
2	1	2	2
3	1	3	3
4	2	1	2
5	2	2	3
6	2	3	4
7	3	1	3
8	3	2	4
9	3	3	5

TABLE 4: Fuzzy Rule base in Indexed form

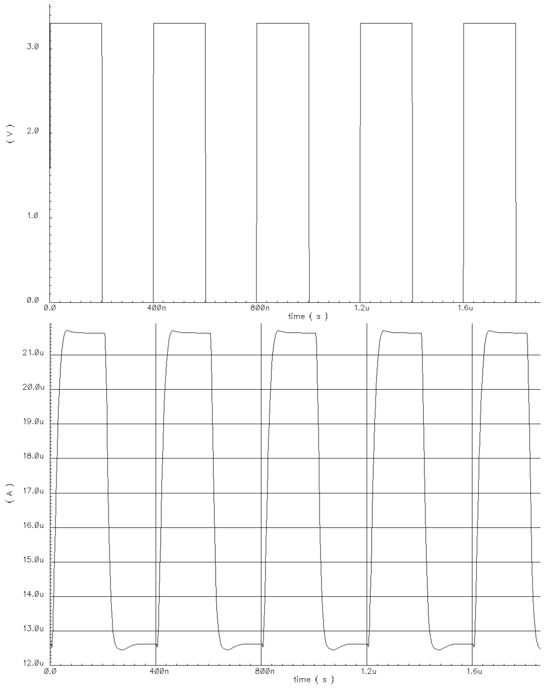

FIGURE 19: Pulse response of the IT2 Fuzzy Logic Controller Chip

The comparison of the proposed design with the existing designs on different target technologies is presented in Table 5. The proposed design has achieved a considerable high speed along with a significant reduction in power and area. Although the achieved speed is less than the FPGA based design [18], however, a severe limitation of FPGA based implementation is that it requires external memory that grows with resolution, number of inputs, and number of MFs.

References	**[17]**	**[18]**	**[21]**	**Proposed**
Target Technology	Microcontroller	FPGA	0.35 μm (Digital CMOS)	0.18 μm (Analog CMOS)
Design Specifications	2 inputs with 2 sets per input, 4 rules, 4 consequents	2 inputs, 1 output, 9 rules	2 inputs, 1 output, 64 rules	2 inputs with 2 fuzzy sets per input, 1 output, 9 rules
Power	*Not Specified*	*Not Specified*	*Not Specified*	20 mW
Area	-	-	5957 μm x 5954 μm (35.46 mm^2)	0.32 mm^2
Speed (FLIPS)	29.17 (Inference time: 34.28 ms)	30 x 10^6	3.125 x 10^6	20 x 10^6
Additional Memory Requirements	RAM: 1024 bytes Flash: 4096 bytes	Highly Memory Intensive	-	No Additional Memory Required

TABLE 5: Comparison of the Proposed Design with previous work

5. CONCLUSIONS

We have presented here the design of an analog CMOS IT2 fuzzy logic controller chip in 0.18μm technology. The design is based on the realization methodology of averaging of two T1 FLSs. The basic fuzzy functional blocks viz. fuzzifier, inference engine, defuzzifier and averager, all are analog circuits. General features of analog fuzzy circuits are high speed, low power and small size. Furthermore, due to parallelism in the architectures of the fuzzifier and the inference engine, the speed of the chip is independent of the number of inputs and the number of rules. However, the power consumption will increase with the number of inputs and the number of MFs used to fuzzify each input.

The shapes and positions of the MFs are tunable through IC control pins. The rule base is also programmable through control pins provided on IC. Further some references voltage sources and reference consequent current sources are designed on chip. The chip has a speed of 20 MFLIPS and power consumption of 20mW and it occupies an area of 0.32mm^2. The chip features are listed in Table 6.

Description	**Features**
Technology	0.18µm
No. of Inputs	2 (3 MFs each)
No. of Outputs	1 (5 singletons)
No. of pins	46
Supply Voltage	3.3V
Power Consumption	20 mW
Inference speed	20MFLIPS
Area	0.32mm^2

TABLE 6: IT2 Fuzzy Logic Controller Chip Features

6. ACKNOWLEDGMENT

We would like to thank Government of India for providing facilities to our institute under SMDP-II project. This work has been carried out using lab facilities created under SMDP-II at Electronics & Communication Engineering Department provided by Ministry of Communication and Information Technology, New Delhi, Government of India.

7. REFERENCES

[1] J.M. Mendel and R.I. John, “Type-2 Fuzzy Sets Made Simple”, IEEE Transaction on Fuzzy Systems, vol. 10, no. 2, 2002, pp. 117–127.

[2] Q. Liang and J. M. Mendel, “Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters”, IEEE Trans. Fuzzy Syst., vol. 8, no. 5, 2000, pp. 551–563.

[3] Q. Liang and J. M. Mendel, “MPEG VBR video traffic modeling and classification using fuzzy technique”, IEEE Trans. Fuzzy Syst., vol. 9, no. 1, 2001, pp. 183–193.

[4] H. B. Mitchell, “Pattern recognition using type-II fuzzy sets,” Inf. Sci., vol. 170, no. 2–4, 2005, pp. 409–418.

[5] J. Zeng and Z. Q. Liu, “Type-2 fuzzy hidden Markov models and their application to speech recognition”, IEEE Trans. Fuzzy Syst., vol. 14, no. 3, 2006, pp. 454–467.

[6] P. Melin and O. Castillo, “A new method for adaptive control of nonlinear plants using type-2 fuzzy logic and neural networks”, J. Gen. Syst., vol. 33, no. 2/3, 2004, pp. 289–304.

[7] H. Hagras, “A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots”, IEEE Trans. Fuzzy Syst., vol. 12, no. 4, 2004, pp. 524–539.

[8] Baturone, Barriga, Carlos Jimenez-Fernandez and Diego, R. Lopez, Microelectronic Design of Fuzzy-Logic Based Systems, CRC Press, 2000.

[9] H. Eichfeld, T. Künemund, and M. Menke, “A 12b general-purpose fuzzy logic controller chip”, IEEE Trans. Fuzzy Syst., vol. 4, no. 4, 1996, pp. 460–475.

[10] M.J. Patyra, J.L. Grantner and K. Koster, “Digital fuzzy logic controller: Design and implementation”, IEEE Trans. Fuzzy Syst., vol. 4, no. 4, 1996, pp. 439–459.

[11] J.M. Jou, P.Y. Chen and S.F. Yang, “An adaptive fuzzy logic controller: Its VLSI architecture and applications”, IEEE Trans. Very Large Scale Integr. Syst., vol. 8, no. 1, 2000 pp. 52–60.

[12] V. Salapura, “A fuzzy RISC processor”, IEEE Trans. Fuzzy Syst., vol. 8, no. 6, 2000, pp. 781–790.

[13] D. Kim, “An implementation of fuzzy logic controller on the reconfigurable FPGA system”, IEEE Trans. Ind. Electron., vol. 47, no. 3, 2000, pp. 703–715.

[14] Y. Ota and B. Wilamowski, “CMOS implementation of a Voltage-mode fuzzy Min-Max Controller”, Journal of Circuits, Systems, and Computers, Vol. 6, No. 2, 1996, pp. 171-184.

[15] M.F. Azeem, K.P. Govila, “Design of Analog CMOS Based Fuzzy Inference System”, IEEE International Conference on Fuzzy Systems, Vancouver, 2006, pp. 1715-1720.

[16] V. F. Dinavari, A. Khoei, K. Hadidi, M. Soleimani, H. Mojarad, “Design of a Current-Mode Analog CMOS Fuzzy Logic Controller”, IEEE Eurocon, St. Petersburg, 2009, pp. 211-217.

[17] J. Bulla, G. Sierra and M. Melgarejo “Implementing a Simple Microcontroller-Based Interval Type-2 Fuzzy Processor”, Proceedings of 51st Midwest Symposium on Circuits and Systems (MWSCAS), Knoxville (TN), 2008, pp. 69-72.

[18] M. A. Melgarejo and C. A. Pena-Reyes, “Hardware architecture and FPGA implementation of a type-2 fuzzy system”, Proc. ACM GLSVLSI, Boston, MA, 2004, pp. 458–461.

[19] M. A. Melgarejo, R. A. Garcia and C. A. Pena-Reyes, “Pro-two: A hardware based platform for real time type-2 fuzzy inference”, Proc. IEEE Int. Conf. Fuzzy Syst., vol. 2, 2004, pp. 977–982.

[20] M. Melgarejo and C. A. Pena-Reyes, “Implementing interval type-2 fuzzy processors”, *IEEE Comput. Intell. Mag.*, vol. 2, no. 1, 2007, pp. 63–71.

[21] S. H. Huang and Y. R. Chen, “VLSI implementation of type-2 fuzzy inference processor”, Proc. IEEE Int. Symp. Circuits Syst., vol. 4, 2005, pp. 3307–3310.

[22] H. Hagras, “A Type-2 Fuzzy Logic Controller for Autonomous Mobile Robots”, in Proceeding of IEEE FUZZ Conference, Budapest, Hungary, July 2004, pp. 965-970.

[23] Q. Liang, N. N. Karnik, and J. M. Mendel, “Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems”, *IEEE Trans. Syst., Man, Cybern., C, Appl. Rev.*, vol. 30, no. 3, 2000, pp. 329–339.

[24] Q. Liang and J. M. Mendel, “Overcoming time-varying co-channel interference using type-2 fuzzy adaptive filter”, *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, No. 12, 2000, pp. 1419–1428.

[25] Mamta Khosla, R K Sarin, Moin Uddin, Arun Khosla and Satvir Singh, *Realizing Interval Type-2 Fuzzy Systems with Type-1 Fuzzy Systems.* Book Chapter for Book titled Cross-

Disciplinary Applications of Artificial Intelligence and Pattern Recognition: Advancing Technologies, IGI Global, USA. (Accepted for Publication).

[26] Roberto Sepúlveda, Oscar Castillo, Patricia Melin, Oscar Montiel, “An Efficient Computational Method to Implement Type-2 Fuzzy Logic in Control Applications”, Analysis and Design of Intelligent Systems using Soft Computing Techniques, Springer, 2007, pp. 45-52.

[27] Roberto Sepúlveda, Oscar Montiel, Gabriel Lizárraga, Oscar Castillo, “Modeling and Simulation of the Defuzzification Stage of a Type-2 Fuzzy Controller Using the Xilinx System Generator and Simulink”, Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control, Springer, 2009, pp. 309-325.

[28] Jerry M. Mendel, “Type-2 fuzzy sets and Systems: an Overview”, IEEE Computational Intelligence Magazine, 2007, pp. 20-29.

[29] Qilian Liang and Jerry M. Mendel, “Interval Type-2 Fuzzy Logic Systems: Theory and Design”, IEEE Transactions on Fuzzy Systems, Vol. 8, No. 5, 2000, pp.535-550.

[30] J. M. Mendel, R. I. John, and F. Liu, “Interval Type-2 Fuzzy Logic Systems Made Simple”, *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 6, 2006, pp. 808-821.

[31] Mamta Khosla, R K Sarin, Moin Uddin and Ajay Sharma, “Analog Realization of Fuzzifier for IT2 Fuzzy Processor,” 3rd International Conference on Electronics Computer Technology-ICECT 2011, Kanyakumari, pp 239-245.

[32] Y. Ota and B. Wilamowski, “Current-mode CMOS Implementation of a Fuzzy Min-Max Network”, Proc. World Congr. Neural Networks, vol. II, 1995, pp. 480 - 483.

Design PID-Like Fuzzy Controller With Minimum Rule Base and Mathematical Proposed On-line Tunable Gain: Applied to Robot Manipulator

Farzin Piltan SSP.ROBOTIC@yahoo.com
Department of Electrical and Electronic
Engineering, Faculty of Engineering,
Universiti Putra Malaysia 43400
Serdang, Selangor, Malaysia

N. Sulaiman nasri@eng.upm.edu.my
Department of Electrical and Electronic
Engineering, Faculty of Engineering,
Universiti Putra Malaysia 43400
Serdang, Selangor, Malaysia

Arash Zargari *SSP.ROBOTIC@yahoo.com*
Industrial Electrical and Electronic
Engineering SanatkadeheSabze
Pasargad. CO (S.S.P. Co), NO:16
,PO.Code 71347-66773, Fourth floor
Dena Apr , Seven Tir Ave , Shiraz , Iran

Mohammad Keshavarz *SSP.ROBOTIC@yahoo.com*
Industrial Electrical and Electronic
Engineering SanatkadeheSabze
Pasargad. CO (S.S.P. Co), NO:16
,PO.Code 71347-66773, Fourth floor
Dena Apr , Seven Tir Ave , Shiraz , Iran

Ali Badri *SSP.ROBOTIC@yahoo.com*
Industrial Electrical and Electronic
Engineering SanatkadeheSabze
Pasargad. CO (S.S.P. Co), NO:16
,PO.Code 71347-66773, Fourth floor
Dena Apr , Seven Tir Ave , Shiraz , Iran

Abstract

In this study, an on-line tunable gain model free PID-like fuzzy controller (GTFLC) is designed for three degrees of freedom (3DOF) robot manipulator to rich the best performance. Fuzzy logic controller is studied because of its model free and high performance. Today, robot manipulators are used in unknown and unstructured environment and caused to provide sophisticated systems, therefore strong mathematical tools are used in new control methodologies to design adaptive nonlinear robust controller with acceptable performance (e.g., minimum error, good trajectory, disturbance rejection). The strategies of control robot manipulator are classified into two main groups: classical and non-classical methods, however non linear classical theories have been applied successfully in many applications, but they also have some limitation. One of the most important nonlinear non classical robust controller that can used in uncertainty nonlinear systems, are fuzzy logic controller. This paper is focuses on applied mathematical tunable gain method in robust non classical method to reduce the fuzzy logic controller limitations. Therefore on-line tunable PID like fuzzy logic controller will be presented in this paper.

Keywords: Tunable Gain, Robot Manipulator, Fuzzy Logic Controller, Classical Control, Non-Classical Control, on-line Tunable Gain.

1. INTRODUCTION

Controller design is the main part in robotic manipulator as well as the major objectives stability and robustness. Consequently to improve the system's performance lots of researchers are about control systems [2].

Some of robot manipulators which work in industrial processes are controlled by linear PID controllers, but design linear controller for robotic manipulators is extremely difficult because they are nonlinear, uncertainty, multi input multi output (MIMO) and time variant [1, 3]. To eliminate the above problems control researchers applied PID methods in nonlinear robust controller (e.g., fuzzy logic controller).

After the invention of fuzzy logic theory in 1965, this theory was used in wide range applications that fuzzy logic controller (FLC) is one of the most important applications in fuzzy logic theory because the controller has been used for nonlinear and uncertain (e.g., robot manipulator) systems controlling. However pure FLC works in many areas but calculation and tune the PID coefficient most of time is challenge [4-7, 15-24].

On-line tuning control is used in systems with various dynamic parameters and need to be training on line. Combined on-line tuneable gain method for artificial controllers can solve the uncertainty challenge in uncertain nonlinear systems [8, 15-24].

This paper is organized as follows: In section 2, main subject of modelling three degrees of freedom robot manipulator formulation are presented. Detail of fuzzy logic controllers with on-line tuneable gain is presented in section 3. In section 4, the simulation result is presented and finally in section 5, the conclusion is presented.

2. ROBOT MANIPULATOR DYNAMIC FORMULATION

The equation of an *n-DOF* robot manipulator governed by the following equation [1, 3, 15-24]:

$$M(q)\ddot{q} + N(q, \dot{q}) = \tau \quad (1)$$

Where τ is actuation torque, M (q) is a symmetric and positive define inertia matrix, $N(q, \dot{q})$ is the vector of nonlinearity term. This robot manipulator dynamic equation can also be written in a following form:

$$\tau = M(q)\ddot{q} + B(q)[\dot{q}\,\dot{q}] + C(q)[\dot{q}]^2 + G(q) \quad (2)$$

Where B(q) is the matrix of coriolios torques, C(q) is the matrix of centrifugal torques, and G(q) is the vector of gravity force. The dynamic terms in equation (2) are only manipulator position. This is a decoupled system with simple second order linear differential dynamics. In other words, the component $\ddot{q}$ influences, with a double integrator relationship, only the joint variableq_i, independently of the motion of the other joints. Therefore, the angular acceleration is found as to be [3, 15-24]:

$$\ddot{q} = M^{-1}(q).\{\tau - N(q, \dot{q})\} \quad (3)$$

This technique is very attractive from a control point of view.

3. PID LIKE FUZZY INFERENCE SYSTEM WITH ON-LINE TUNABLE GAIN

In recent years, artificial intelligence theory has been used in robotic systems. Neural network, fuzzy logic, and neuro-fuzzy are combined with tuneable methods and used in nonlinear, time variant, and uncertainty plant (e.g., robot manipulator). After the invention of fuzzy logic theory in 1965 by Zadeh [4], this theory was used in wide range area. Fuzzy logic controller (FLC) is one of the most important applications of fuzzy logic theory. This controller can be used to control of nonlinear, uncertain, and noisy systems. This method is free of some model-based techniques that used in classical controllers. It should be mentioned that fuzzy logic application is not only limited to the modelling of nonlinear systems [5-9] but also this method can help engineers to design easier controller.

The main reasons to use fuzzy logic technology are able to give approximate recommended solution for unclear and complicated systems to easy understanding and flexible. Fuzzy logic provides a method which is able to model a controller for nonlinear plant with a set of IF-THEN rules, or it can identify the control actions and describe them by using fuzzy rules. Besides using fuzzy logic in the main controller of a control loop, it can be used to design adaptive control, tuning parameters, working in a parallel with the classical and non classical control method [5, 15-18].

3.1 Fuzzy Inference System

However the application area for fuzzy control is really wide, the basic form for all command types of controllers consists of;

- Input fuzzification (binary-to-fuzzy[B/F]conversion)
- Fuzzy rule base (knowledge base)
- Inference engine
- Output defuzzification (fuzzy-to-binary[F/B]conversion) [5, 15-18].

The basic structure of a fuzzy controller is shown in Figure 1.

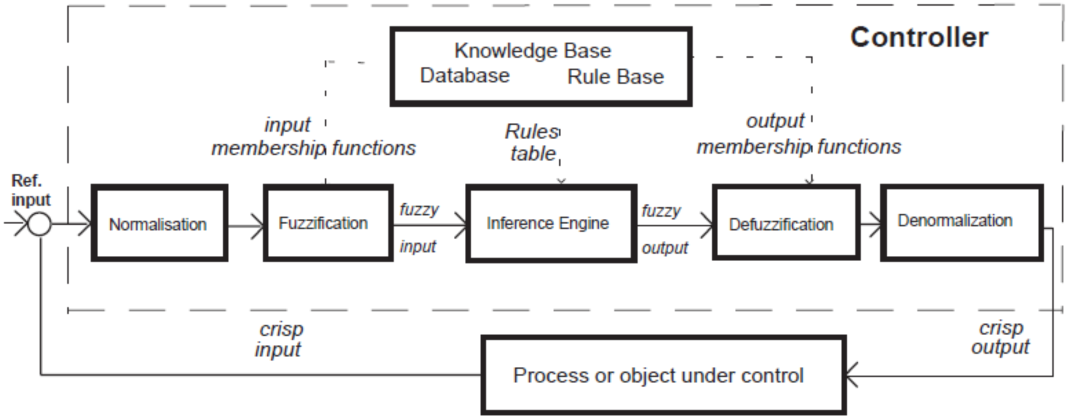

FIGURE 1: Block diagram of a fuzzy controller with details.

3.2. PID Fuzzy Logic Controller

A PID fuzzy controller is a controller which takes error, integral of error and derivative of error as inputs. Fuzzy controller with three inputs is difficult to implementation, because it needs large number of rules, in this state the number of rules increases with an increase the number of inputs or fuzzy membership functions [10-12]. In the PID FLC, if each input has 7 linguistic variables, then $7 \times 7 \times 7 = 343$ rules will be needed. The proposed PID FLC is constructed as a parallel structure of a PD FLC and PI FLC (Figure 2), and the output of the PID FLC is formed by adding the output of two fuzzy control blocks. This work will reduce the number of rules needed to $7 \times 7 + 7 \times 7 = 98$ rules only.

As a summary the design of PID like fuzzy logic controller based on Mamdani's fuzzy inference method has four steps , namely, fuzzification, fuzzy rule base and rule evaluation, aggregation of the rule output (fuzzy inference system), and deffuzzification [15-18].

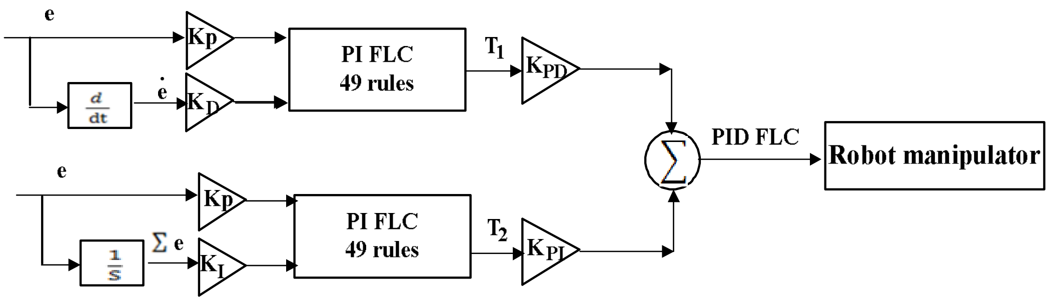

FIGURE 2: Proposed PID controller with minimum rule base

Fuzzification: the first step in fuzzification is determine inputs and outputs which, it has two inputs ($e,\dot{e}$) or $(e,\sum e)$ and one output ($\tau_{1fuzzy} or\ \tau_{2fuzzy}$). The inputs are error (e) which measures the difference between desired and actual output position, the change of error ($\dot{e}$) which measures the difference between desired and actual velocity and the summation of error $(\sum e)$ which measured the difference between desired and actual summation of error. The second step is chosen an appropriate membership function for inputs and output which, for simplicity in implementation and also to have an acceptable performance the researcher is selected the triangular membership function that it is shown in Figure 3. The third step is chosen the correct labels for each fuzzy set which, in this research namely as linguistic variable. The linguistic variables for error (e) are; Negative Big (NB), Negative Medium (NM), Negative Small (NS), Zero (Z), Positive Small (PS), Positive Medium (PM), Positive Big (PB), and it is quantized in to thirteen levels represented by: -1, -0.83, -0.66, -0.5, -0.33, -0.16, 0, 0.16, 0.33, 0.5, 0.66, 0.83, 1 the linguistic variables for change of error ($\dot{e}$) are; Fast Left (FL), Medium Left (ML), Slow Left (SL),Zero (Z), Slow Right (SR), Medium Right (MR), Fast Right (FR), and it is quantized in to thirteen levels represented by: -6, -5, -0.4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, the linguistic variables for summation of error ($\sum e$) are; Negative Big (NB), Negative Medium (NM), Negative Small (NS), Zero (Z), Positive Small (PS), Positive Medium (PM), Positive Big (PB), and it is quantized in to thirteen levels represented by: -1, -0.83, -0.66, -0.5, -0.33, -0.16, 0, 0.16, 0.33, 0.5, 0.66, 0.83, 1 and the linguistic variables to find the output are; Large Left (LL), Medium Left (ML), Small Left (SL), Zero (Z), Small Right (SR), Medium Right (MR), Large Right (LR) and it is quantized in to thirteen levels represented by: -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6.

Fuzzy rule base and rule evaluation: the first step in rule base and evaluation is provide a least structured method to derive the fuzzy rule base which, expert experience and control engineering knowledge is used because this method is the least structure of the other one and the researcher derivation the fuzzy rule base from the knowledge of system operate and/or the classical controller. Design the rule base of fuzzy inference system can play important role to design the best performance of fuzzy sliding mode controller, that to calculate the fuzzy rule base the researcher is used to heuristic method which, it is based on the behavior of the control of robot manipulator suppose that two fuzzy rules in this controller are;

F.R^1: *IF e is NB and $\dot{e}$ is FL, THEN τ is LL.* **(4)**
F.R^2: *IF e is PS and $\dot{e}$ is FL THEN τ is ML*

The complete rule base for this controller is shown in Table 1. Rule evaluation focuses on operation in the antecedent of the fuzzy rules in fuzzy sliding mode controller. This part is used AND/OR fuzzy operation in antecedent part which AND operation is used.

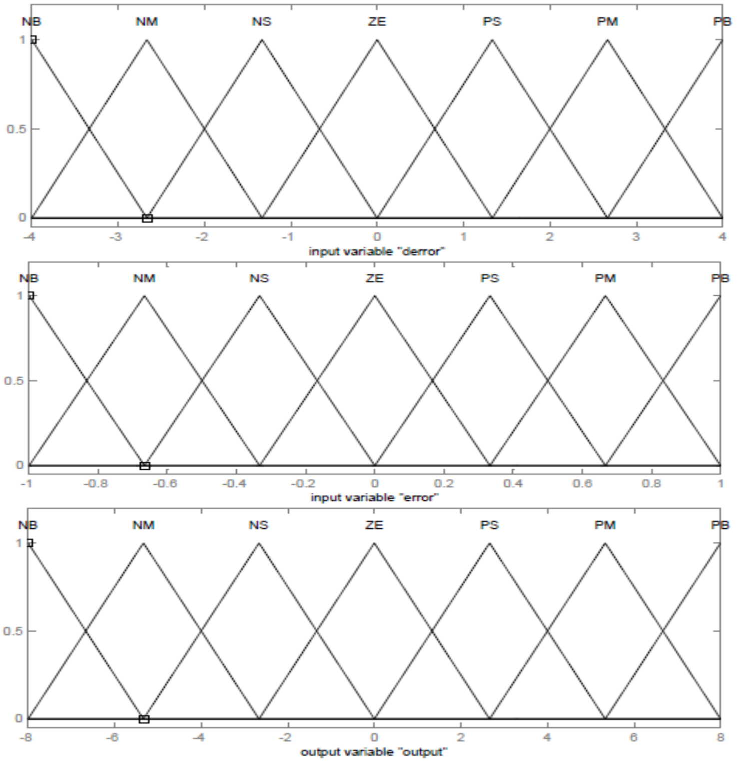

FIGURE 3: Membership function: a) error b) change of error c) output

Aggregation of the rule output (Fuzzy inference): Max-Min aggregation is used to this work which the calculation is defined as follows;

$$\mu_U(x_k, y_k, U) = \mu_{\cup_{i=1}^{r} FR^i}(x_k, y_k, U) = max\left\{min_{i=1}^{r}\left[\mu_{R_{pq}}(x_k, y_k), \mu_{p_m}(U)\right]\right\} \quad (5)$$

Deffuzzification: The last step to design fuzzy inference in our fuzzy sliding mode controller is defuzzification. This part is used to transform fuzzy set to crisp set, therefore the input for defuzzification is the aggregate output and the output of it is a crisp number. In this design the Center of gravity method (COG) is used and calculated by the following equation;

$$COG(x_k, y_k) = \frac{\sum_i U_i \sum_{j=1}^{r} \mu_u(x_k, y_k, U_i)}{\sum_i \sum_{j=1}^{r} \mu_u(x_k, y_k, U_i)} \quad (6)$$

$\dot{e}$ e	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NM	NS	NS	ZE
NM	NB	NM	NM	NM	NS	ZE	PS
NS	NB	NM	NS	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PS	PM	PB
PM	NS	ZE	PS	PM	PM	PM	PB
PB	PS	PS	PM	PB	PB	NB	ZE

TABLE 1: Modified fuzzy rule base table

This table has 49 cells, and used to describe the dynamics behavior of fuzzy controller.
Table 2 is shown the lookup table in fuzzy logic controller which is computed by COG deffuzzification method. These output values were obtained from trial and error after some manual adjustment to reach the best performance in fuzzy logic controller.

$\dot{e}$ → e ↓	**Membership Function**												
	-1	**-0.83**	**-0.66**	**-0.5**	**-0.33**	**-0.16**	**0**	**0.16**	**0.33**	**0.5**	**0.66**	**0.83**	**1**
-1	-5.6	-5.4	-5	-4.8	-4.8	-4.7	-4.7	-4.6	-4.5	-4.4	-4.3	-4.3	-4.2
-0.83	-4.7	-4.5	-4.4	-4.3	-4.2	-4.1	-4	-3.9	-3.8	-3.8	-3.7	-3.6	-3.5
-0.66	-3.7	-3.6	-3.5	-3.2	-3	-3	-3	-2.9	-2.9	-2.8	-2.8	-2.7	-2.7
-0.5	-2.0	-2.0	-2.0	-1.9	-1.9	-1.8	-1.8	-1.7	-1.7	-1.6	-1.5	-1.4	-1.3
-0.33	0.0	-0.8	-1.0	-1.2	-1.7	-2.3	-2.2	-2.2	-2.0	-2.0	-1.0	-1.0	0.0
-0.16	1.0	1	0.5	-0.5	-1.0	-1.2	-1.5	-1.7	-1.0	-1.0	-1.0	-0.5	-0.5
0	1.3	1.2	1.0	0.8	0.6	0.0	-0.2	-0.4	-0.6	-0.8	-1.0	-1.0	-1.0
0.16	2.0	2.0	1.9	1.8	1.8	1.8	1.8	1.8	1.5	0.0	-0.3	-1.0	-0.8
0.33	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	1.2	0.8	0.0	0.0
0.5	2.0	2.1	2.3	2.5	2.5	2.6	2.6	2.7	2.7	2.8	2.9	2.9	3.0
0.66	2.7	2.7	2.8	2.9	2.9	3.0	3.1	3.1	3.2	3.2	3.6	3.7	3.9
0.83	3.6	3.6	3.7	3.8	3.9	4.0	4.1	4.2	4.2	4.3	4.3	4.6	4.7
1	4.4	4.4	4.5	4.6	4.7	4.8	5.0	5.0	5.1	5.2	5.3	5.6	5.6

TABLE 2: COG lookup table in fuzzy sliding mode controller

Table 2 has 169 cells to shows the PD fuzzy part behavior. For instance if $e = 0$ and $\dot{e} = 0$ then the output= -0.2.

3.3 On-line Tunable Gain to Adjust Fuzzy Logic Controller

All conventional fuzzy logic controller have common difficulty, they need to find several parameters. Tuning PID like FLC method can tune automatically the scale parameters using new method. To keep the structure of the controller as simple as possible and to avoid heavy computation, a mathematical supervisor tuner is selected [13-14]. In this method the tuneable controller tunes the input scaling factors using gain updating factors. In this method the first gain updating factor, k_1, is updated by a new coefficient factor, k_n, Where k_n is a function of system error.

$$K_n = e^2 - \frac{(r_v - r_{vmin})^2}{1 + |e|} + r_{vmin} \qquad (7)$$

$$r_v = \frac{\dot{e}(t) - e^{\cdot}(t-1)}{\dot{e}(0)}$$
$$if\; e^{\cdot}(0) = \begin{cases} \dot{e}(t) & if\; e^{\cdot}(t) \geq \dot{e}(t-1) \\ \dot{e}(t-1) & if\; e^{\cdot}(t) < \dot{e}(t-1) \end{cases} \tag{8}$$

Figure 3 is shown the PID like fuzzy logic controller with proposed tunable gain.

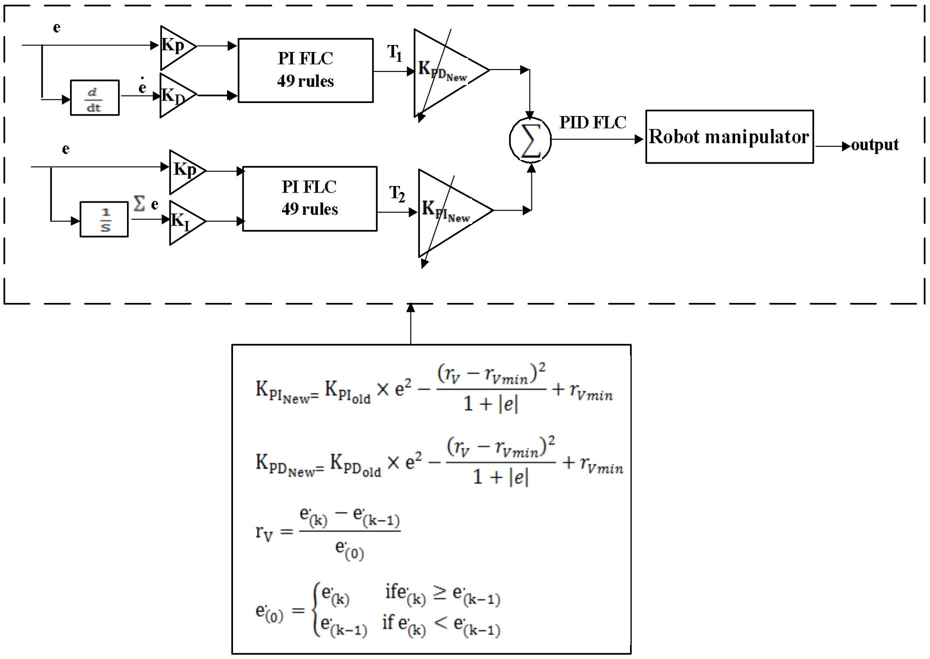
e
Kp
$\frac{d}{dt}$
ė
KD
PI FLC
49 rules
T1
$K_{PD_{New}}$
PID FLC
Robot manipulator
output
e
Kp
$\frac{1}{s}$
Σ e
KI
PI FLC
49 rules
T2
$K_{PI_{New}}$
$K_{PI_{New}} = K_{PI_{old}} \times e^2 - \frac{(r_V - r_{Vmin})^2}{1+|e|} + r_{Vmin}$
$K_{PD_{New}} = K_{PD_{old}} \times e^2 - \frac{(r_V - r_{Vmin})^2}{1+|e|} + r_{Vmin}$
$r_V = \frac{e^{\cdot}_{(k)} - e^{\cdot}_{(k-1)}}{e^{\cdot}_{(0)}}$
$e^{\cdot}_{(0)} = \begin{cases} e^{\cdot}_{(k)} & if e^{\cdot}_{(k)} \geq e^{\cdot}_{(k-1)} \\ e^{\cdot}_{(k-1)} & if\ e^{\cdot}_{(k)} < e^{\cdot}_{(k-1)} \end{cases}$

FIGURE 3: PID like fuzzy logic controller with proposed tunable gain

4 SIMULATION RESULTS

Pure fuzzy logic controller (FLC) and gain tuning PID like fuzzy controller (GTFLC) are implemented in Matlab/Simulink environment. Tracking performance, disturbance rejection and error are compared.

4.1 Tracking Performances

From the simulation for first, second and third trajectory without any disturbance, it was seen that FLC and GTFLC have the same performance. This is primarily because this system is worked on certain environment. The GTFLC gives significant trajectory good following when compared to FLC. Figure 4 shows tracking performance without any disturbance for FLC and GTFLC.

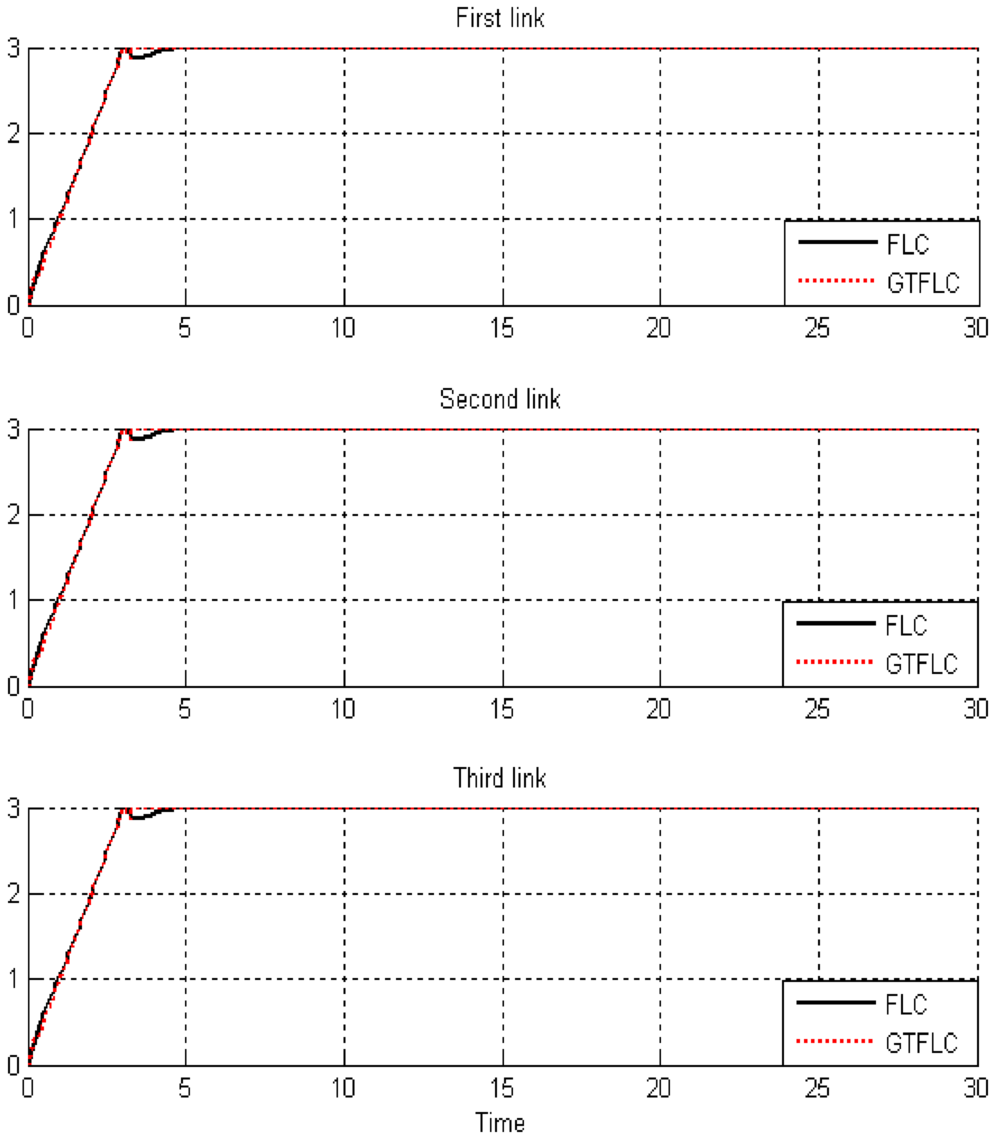

FIGURE 4: FLC and GTFLC for First, second and third link trajectory

By comparing trajectory response without disturbances in FLC and GTFLC it is found that the GTFLC overshoot **(0%)** is lower than FLC's **(1%)**, although both of them have about the same rise time.

4.2 Disturbance Rejection

Figure 5 has shown the power disturbance elimination in FLC and GTFLC. The main target in these controllers is disturbance rejection as well as the other responses. A band limited white noise with predefined of 40% the power of input signal is applied to the FLC and GTFLC. It found fairly fluctuations in FLC trajectory responses.

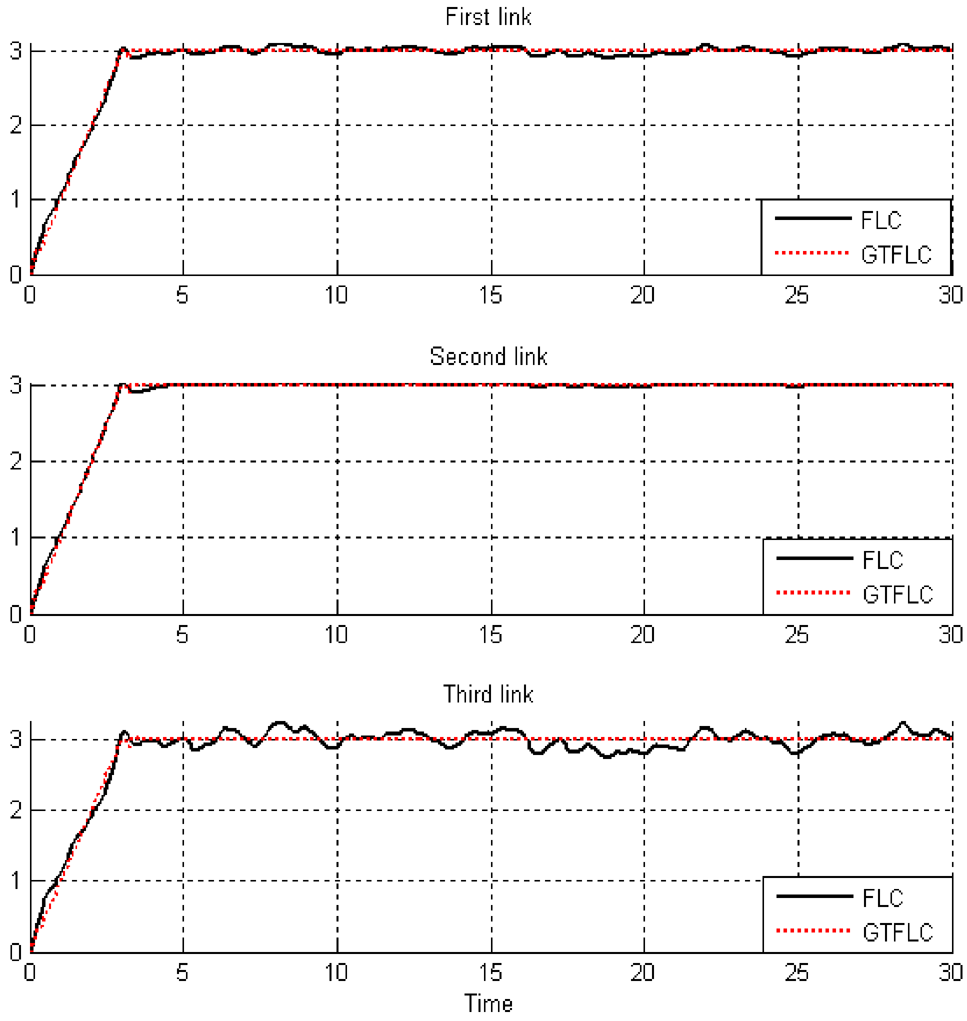

FIGURE 5: FLC and GTFLC for First, second and third link trajectory with external disturbances.

Among above graph relating to trajectory following with external disturbance, FLC has fairly fluctuations. By comparing some control parameters such as overshoot and rise time it found that the GTFLC's overshoot **(0%)** is lower than FLC's **(4%)**, although both of them have about the same rise time**.**

4.3 Calculate Errors

Figure 6 has shown the error disturbance in FLC and GTFLC. The controllers with no external disturbances have the same error response. By comparing the steady state error and RMS error it found that the GTFLC's errors **(Steady State error = -0.0007 and RMS error=0.0008)** are fairly less than FLC's **(Steady State error ≅ $\mathbf{0.0012}$ and RMS error=$\mathbf{0.0018}$)**, When disturbance is applied to the FLC error is about 13% growth.

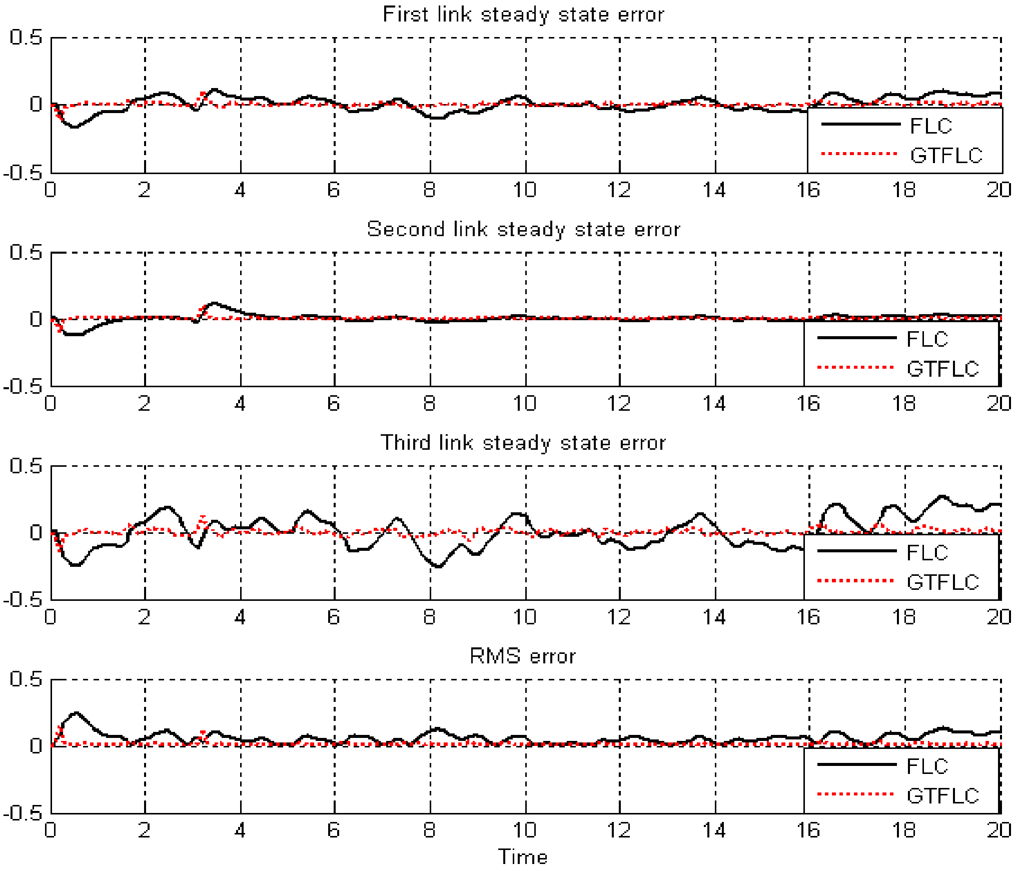

FIGURE 6: FLC and GTFLC for First, second and third link steady state and RMS error with external disturbances.

5 CONCLUSIONS

This paper presents a new methodology for designing s online tunable PID like fuzzy logic controller with minimum rule bases and high performance for 3 DOF robotic manipulator. From the simulation, it found that proposed PID fuzzy gain tuning has 98 rule base for main controller but in normal PID like fuzzy controller by the other researcher has about 343 rules for main controller. In GTFLC, the mathematical tunable controller can changed K_{PD} & K_{PI} to achieve the best performance. In this method the proposed mathematical supervisory controller is changed the gain updating factor of main FLC to get the best performance.

REFERENCES

[1] T. R. Kurfess, *Robotics and automation handbook*: CRC, 2005.

[2] K. Ogata, *Modern control engineering*: Prentice Hall, 2009.

[3] B. Siciliano and O. Khatib, *Springer handbook of robotics*: Springer-Verlag New York Inc, 2008.

[4] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy Sets and Systems,* vol. 90, pp. 111-127, 1997.

[5] L. Reznik, *Fuzzy controllers*: Butterworth-Heinemann, 1997.

[6] J. Zhou and P. Coiffet, "Fuzzy control of robots," 2002, pp. 1357-1364.

[7] S. Banerjee and P. Y. Woo, "Fuzzy logic control of robot manipulator," 2002, pp. 87-88.

[8] K. Kumbla*, et al.*, "Soft computing for autonomous robotic systems," *Computers and Electrical Engineering,* vol. 26, pp. 5-32, 2000.

[9] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. I," *IEEE Transactions on systems, man and cybernetics,* vol. 20, pp. 404-418, 1990.

[10] R. J. Wai*, et al.*, "Implementation of artificial intelligent control in single-link flexible robot arm," 2003, pp. 1270-1275.

[11] R. J. Wai and M. C. Lee, "Intelligent optimal control of single-link flexible robot arm," *Industrial Electronics, IEEE Transactions on,* vol. 51, pp. 201-220, 2004.

[12] M. B. Menhaj and M. Rouhani, "A novel neuro-based model reference adaptive control for a two link robot arm," 2002, pp. 47-52.

[13] S. Mohan and S. Bhanot, "Comparative study of some adaptive fuzzy algorithms for manipulator control," *International Journal of Computational Intelligence,* vol. 3, pp. 303–311, 2006.

[14] B. K. Yoo and W. C. Ham, "Adaptive control of robot manipulator using fuzzy compensator," *Fuzzy Systems, IEEE Transactions on,* vol. 8, pp. 186-199, 2002.

[15] Piltan, F., et al., "Design sliding mode controller for robot manipulator with artificial tunable gain," Canaidian Journal of pure and applied science, 5 (2): 1573-1579, 2011.

[16] Piltan, F., et al., "Design Artificial Nonlinear Robust Controller Based on CTLC and FSMC with Tunable Gain," International Journal of Robotic and Automation, 2 (3): 205-220, 2011.

[17] Piltan, F., et al., "Design Mathematical Tunable Gain PID-Like Sliding Mode Fuzzy Controller with Minimum Rule Base," International Journal of Robotic and Automation, 2 (3): 146-156, 2011.

[18] Piltan, F., et al., "Design of FPGA based sliding mode controller for robot manipulator," International Journal of Robotic and Automation, 2 (3): 183-204, 2011.

[19] P

iltan, F., et al., "Novel sliding mode controller for robot manipulator using FPGA," Journal of Advanced science and Engineering Research, 1 (1): 1-22, 2011.

[20] Piltan, F., et al., "Design a New sliding mode Adaptive Hybrid Fuzzy controller," Journal of Advanced science and Engineering Research, 1 (1): 115-123, 2011.

[21] Piltan, F., et al., "Design Artificial Robust Control of Second Order System Based on Adaptive Fuzzy Gain Scheduling," International Journal of Robotic and Automation, 2 (4): 2011.

[22] Piltan, F., et al., "Design of Pc-Based Sliding Mode Controller and Normalized Sliding Surface Slope Using PSO Method for Robot Manipulator," International Journal of Robotic and Automation, 2 (4): 205-220, 2011.

[23] Piltan, F., et al., "Design of Model Free Adaptive Fuzzy Computed Torque Controller: Applied to Nonlinear Second order System," International Journal of Robotic and Automation, 2 (4): 232-244, 2011.

[24] Piltan, F., et al., "An Adaptive Sliding Surface Slope Adjustment in PD Sliding Mode Fuzzy Control for Robot Manipulator," International Journal of Control and Automation, 4 (3): 65-76, 2011.

INSTRUCTIONS TO CONTRIBUTORS

The main aim of International Journal of Artificial Intelligence and Expert Systems (IJAE) is to provide a platform to AI & Expert Systems (ES) scientists and professionals to share their research and report new advances in the field of AI and ES. IJAE is a refereed journal producing well-written original research articles and studies, high quality papers as well as state-of-the-art surveys related to AI and ES. By establishing an effective channel of communication between theoretical researchers and practitioners, IJAE provides necessary support to practitioners in the design and development of intelligent and expert systems, and the difficulties faced by the practitioners in using the theoretical results provide feedback to the theoreticians to revalidate their models. IJAE thus meets the demand of both theoretical and applied researchers in artificial intelligence, soft computing and expert systems.

IJAE is a broad journal covering all branches of Artificial Intelligence and Expert Systems and its application in the topics including but not limited to technology & computing, fuzzy logic, expert systems, neural networks, reasoning and evolution, automatic control, mechatronics, robotics, web intelligence applications, heuristic and AI planning strategies and tools, computational theories of learning, intelligent system architectures.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJAE.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Starting with volume 2, 2011, IJAE appears in more focused issues. Besides normal publications, IJAE intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

We are open to contributions, proposals for any topic as well as for editors and reviewers. We understand that it is through the effort of volunteers that CSC Journals continues to grow and flourish.

LIST OF TOPICS
The realm of International Journal of Artificial Intelligence and Expert Systems(IJAE) extends, but not limited, to the following:

- AI for Web Intelligence Applications
- AI Parallel Processing Tools
- AI Tools for Computer Vision and Speech Understand
- Application in VLSI Algorithms and Mobile Communication
- Case-based reasoning
- Derivative-free Optimization Algorithms
- Evolutionary and Swarm Algorithms
- Expert Systems Components
- Fuzzy Sets and logic
- Hybridization of Intelligent Models/algorithms
- Inference
- Intelligent Planning
- AI in Bioinformatics
- AI Tools for CAD and VLSI Analysis/Design/Testing
- AI Tools for Multimedia
- Automated Reasoning
- Data and Web Mining
- Emotional Intelligence
- Expert System Development Stages
- Expert-System Development Lifecycle
- Heuristic and AI Planning Strategies and Tools
- Image Understanding
- Integrated/Hybrid AI Approaches
- Intelligent Search

- Intelligent System Architectures
- Knowledge-Based Systems
- Logic Programming
- Multi-agent Systems
- Neural Networks for AI
- Parallel and Distributed Realization of Intelligence
- Reasoning and Evolution of Knowledge Bases
- Rule-Based Systems
- Uncertainty
- Knowledge Acquisition
- Knowledge-Based/Expert Systems
- Machine learning
- Neural Computing
- Object-Oriented Programming for AI
- Problem solving Methods
- Rough Sets
- Self-Healing and Autonomous Systems
- Visual/linguistic Perception

CALL FOR PAPERS

Volume: 3 - **Issue:** 1 - February 2012

i. Paper Submission: November 30, 2011 **ii. Author Notification:** January 01, 2012

iii. Issue Publication: January / February 2012

CONTACT INFORMATION

Computer Science Journals Sdn BhD
B-5-8 Plaza Mont Kiara, Mont Kiara
50480, Kuala Lumpur, MALAYSIA

Phone: 006 03 6207 1607
006 03 2782 6991

Fax: 006 03 6207 1697

Email: cscpress@cscjournals.org

CSC PUBLISHERS © 2011
COMPUTER SCIENCE JOURNALS SDN BHD
M-3-19, PLAZA DAMAS
SRI HARTAMAS
50480, KUALA LUMPUR
MALAYSIA

PHONE: 006 03 6207 1607
006 03 2782 6991

FAX: 006 03 6207 1697
EMAIL: cscpress@cscjournals.org