Optimal Fuzzy Controller: Rule Base Optimized Generation

M. Abdalla*, T. Al-Jarrah**

**The University of Jordan, P.O. Box 13365, Amman 11942, Jordan*
Jordan (Tel: +962-777-966264; e-mail: admin@ mechatronix.us).
***Royal Scientific Society, Amman 11941*

Abstract: A novel Fuzzy Logic controller design methodology is presented. The method utilizes an in-house developed Particle Swarm Optimization (PSO) binary search algorithm to generate the rules for the Fuzzy Logic controller rule-base stage without any human experience intervention. The proposed technique is compared with the well-established Lyapunov based Fuzzy Logic controller design in generating such rules. Finally, the controller's effectiveness and performance are tested, verified and validated using an electrical traction elevator control application. The designed controller's results were compared with the traditional ubiquitous Proportional Integral Derivative controller and classical Fuzzy Logic Controllers.

Keywords: Fuzzy Logic, Membership Functions Optimization, Fuzzy PID Controller, PSO.

1. INTRODUCTION

Currently, all controllers design strategies involve both model-based and model-less approaches. However, when the system's model is not available or very complex to derive then immediately the designers investigate other possibilities such as PID controllers and model-less based designs (*i.e*. Fuzzy Logic, Neural Nets…etc.). Recently, designers and researchers start to use model-less techniques in spite of the model availability due to the designed controllers' simplicity in structure, performance effectiveness, added smartness and robust operation in a wide range of industrial operating conditions. In reality, an optimally tuned Fuzzy Logic controller will at least match a conventional controller, such as a PID controller, on performance. But still, the spread of the PID controllers in the chemical and petroleum industries is noticeable (Abdalla, 2011).

However, tuning the PID controller to find its three optimal parameters is still a challenging task; because the controller's parameters exhibit contradictory effect, nonlinear and time dependent behaviors. Traditionally, the tuning process is performed using a step input and the system's performance is judged based on standard performance measures: steady state error, overshoot, settling and rising time. In literature, several classical on-line and offline auto-tuning techniques were investigated by many researchers, which may be found in a diversity of technical reports and surveys and the majority of the schemes are application dependent (Prashant, 2004; Ang, 2005; Feng, 2000). Artificial intelligence based search techniques are also investigated to optimally tune the PID controllers. Approaches such as Fuzzy, Evolutionary Algorithms and Particle Swarm, which it has demonstrated

effective tuning of the PID controller, seems to be popular as well (Visioli, 1999, 2001; Bandyopadhyay, 2001; Bingul, 2000; Krohling, 1997; Wahsh 2005).

Fuzzy Logic (FL) based controllers is gaining popularity in houseware and industrial applications since the eighties, which owe its effectiveness to the approximate reasoning it generates and its imitation of human decision-making process. The original Mamdani's linguistic FL fixed and simple controller structure had proven robustness and effectiveness for linear and nonlinear applications (Passino, 1998; Galichet, 1995). Unfortunately, the same advantage of the straightforward implementation of the FL controller hinders its effectiveness in controlling dynamical systems when it is improperly tuned. The enormous amount of flexibility available to designers in choosing the controller's Membership Functions (MF) and the rule base made optimal tuning of the MF challenging for certain applications (Woo, 2000). Various researchers investigated optimal tuning of the MF for the Mamdani FL controller type using stochastic and evolutionary approaches. For example, Genetic Algorithms were used to choose and tune the MF of the Mamdani's controller, but they were found to suffer from convergence to global extrema in a reasonable amount of time. Good comparative studies over these techniques may be found in the literature (Vikram, 2014). Newly developed evolutionary Artificial Immune System (AIS) and Particle Swarm Optimization (PSO) optimal tuning of the FL controller MF were also investigated by many researchers (Omkar, 2013; Bingul 2011). Typically, in all the aforementioned techniques the optimization was performed over the MF parameters (*i.e*. widths, centers…etc.) but not the rule base, which is typically selected by the designer.

The Particle Swarm Optimization (PSO) is a relatively new evolutionary algorithm that may be used to find optimal or near optimal solutions in huge search spaces. The PSO algorithms are especially useful for parameter optimization in continuous and multi-dimensional search spaces. The PSO method can generate a high-quality solution within shorter calculation time and it tends to converge very fast when compared to other stochastic methods. The PSO has proven both very effective and quick in diverse set of benchmark optimization problems (Kennedy, 1995; Elwer, 2009; Ahmad

2012; Abdalla, 2009).

In this work, a novel Particle Swarm Optimization (PSO) selection and tuning strategy for the FL Rule-Base (RB) and Membership Function (MF) is proposed. Basically, the PSO is a relatively new evolutionary population-based stochastic optimization technique that may be used to search solution spaces for extrema. The original idea was developed and coined by Eberhart and Kennedy (Kennedy, 1995). The advantage of the PSO method is its speed in converging for optimal solutions compared to other stochastic methods. The PSO effectiveness and performance had been tested using popular benchmark optimization problems (Palupi, 2011; Schoeman, 2010). The use of the PSO to seek optimality technique had been previously proposed by many researchers. In literature, one may find algorithms that describe a swarm based FL controller mobile sensor network approach for collaboratively locating the hazardous contaminants in an unknown large scale area (Cui, 2004). Also, applications that utilize the PSO technique to tune the parameters of the PI-controller for control of electrical motors position and speed are widely used (Allaoua, 2008). Even in robotics, the PSO is used to tune a FL controller for a 2 DOF planar robot (Bingul, 2010).

In this work, the PSO is utilized to select the FL rules as well as it is used to tune the rules for a more FL effective controller design. The proposed method utilizes an in-house developed Particle Swarm Optimization (PSO) modified binary search algorithm to generate the rules for the Fuzzy Logic controller rule-base stage without any human experience intervention.

Furthermore, an elevator system application will be used to demonstrate the effectiveness of a FL rule-base auto-generation using the Particle Swarm Optimization (PSO) algorithm. An overview of the PSO will be presented in subsequent sections; however, the more interested readers are referred to Eberhart and Kennedy original paper, which fully discusses the algorithm that is adopted by the current authors.

2. FUZZY PID CONTROLLER

The Proportional, Integral and Derivative (PID) controllers have been successfully serving in the industry for a long time owing that to their simple structure and robust performance in a wide range of operating conditions. However, lately more demands in performance from such controllers are needed and because the PID controllers are linear in nature this made them not well suited for strongly nonlinear systems. This motivated many researchers to complement the PID controller with Fuzzy Logic (FL) based controller because FL is typically parameterized using rules and membership functions, which made it easy to add nonlinearities, logic, and additional input signals to the controller's control law (Astrom, 2000). The PID controller currently takes many forms: Direct Action (DA), Gain Scheduling (GS) and Hybrid types of FL-PID controllers (Tang, 2001; Kazemian, 2005).

The majority of the FL-PID controllers' applications belong to the Direct Acting (DA) control scheme, where the FL controller part replaces the PD controller part in the conventional PID structure while the I-control action stays the same. However, this modification in the PID structure requires providing two measured inputs for the FL controller. Many researchers had proven that the two PD systems are equivalent with the addition of providing more intelligence due to the FL heuristic nature and ability to handle both linear and nonlinear systems (Carvaja, 2000; Moon, 1995; Georg, 2001).

To handle nonlinear applications with varying properties, characteristics, uncertainties…etc. one may utilize the Gain Scheduling (GS) controller. In the GS controller scheme the FL part provides a dynamical mean to adjust the conventional PID controller's parameters in relation to the current dynamical system behavior. This adaptation turns out to be very useful and provides the required robustness (George, 1999).

In the Hybrid FL-PID controller design, the conventional PID controller effective working space is extended by using a FL based controller. However, both controllers are utilizing the error signal as inputs, where an intelligent switching scheme that makes a decision on the priority of the controllers' contributions may be utilized for further enhancements (Astrom, 2000).

Finally, the common thought-provoking part in all the aforementioned popular FL-PID controllers is the FL controller design itself. In this work, a novel method that utilizes an in-house developed Particle Swarm Optimization (PSO) binary search algorithm is proposed to generate the rules for these ubiquitous FL-PID controllers rule-base stage without any human experience intervention.

2.1. Fuzzy Logic Controller Overview

Since Zadeh had first introduced the fuzzy set theory and Mamdani had applied it to replace operators in control; many newly developed Fuzzy Logic applications are emerging every day. All these overwhelming applications may be categorized into two distinct fields: *control and expert systems*. Both fields apply inference and approximate reasoning using rule bases with the aid of membership functions over fuzzy logic sets [0, 1]; in contrast to classical logic sets {0, 1}. In control applications, the two widely accepted fuzzy linguistic inference tools are the Mamdani and the Takagi-Sugeno methods. This work is considering the Mamdani's based FL controller design (Mamdani, 1981; Sugeno, 1985).

The FL controller provides its crisp output (Defuzzified) based on crisp inputs (Fuzzified internally) by utilizing inference process via a rule base. Hence, the Fuzzy Logic controller design involves designing three stages: Fuzzification, Rule-Base, and Defuzzification stages. Normally the FL controller design is sequential, that is for the Fuzzification stage; the membership functions are first selected and partitioned then the Rule-Base is constructed and finally the Defuzzification stage is implemented.

The difficulty in designing a FL controller emerges from capturing and covering all the aspects of the system's dynamics, which is heavily dependent on the Rule Base (RB) and the selection of the Membership Functions (MF). The

rule base should be constructed in a way that the rules set should span the solution space. On the other hand, the designer must choose the type of fuzzification (singleton or non-singleton), the number of membership functions, the functional forms of the membership functions (piece wise linear, Gaussian, sigmoidal), the parameters of the membership functions (fixed or tuned during a training procedure), the conjunction operator (t-norm, t-conorm), the implication or inference operator (Mamdani, 1981; Sugeno, 1985; Fodor, 2004), the aggregation operator (t-norm, t-conorm) and the type of Defuzzification (centroid, maxima, height). This demonstrates the richness and flexibility of fuzzy controllers but also it reveals the need for some guidelines for their practical design (Bernadette, 2000).

The creation of the FL controller membership functions intrigued many researchers. A closer look into the literature isolates three categories: autogeneration, statistical and psychological methods. Many autogeneration methods utilize Neural Nets, Genetic Algorithms, and Artificial Immune Systems. These techniques are iterative, numerically based and search for optimally adjusted membership functions to satisfy certain objective function (Chonghua, 2015). While the statistical and psychological techniques work on extracting expert information directly from human beings that are necessary to construct the FL controller membership functions.

However, most researchers follow a general frame in determining the FL controller membership functions. For example: it is appropriate to use some piece wise defined linear functions, that lead to a lower sensitivity to an error measure made in their determination: $S(f) = \int_X \left(\frac{df}{dx}\right)^2 dx$. In addition, the minimum completeness level required is $\varepsilon = 0.25$, which guarantees the convergence of control, and a maximum completeness $\varepsilon = 0.5$, consequently it gives a lower overshoot and a lower settling time of the dynamical time response (Chonghua, 2015; Nguyen, 1997).

The FL controller regulates the dynamical system according to a collection set of rules (*i.e.* rule base) in the form of linguistic IF-THEN rules. For example: *IF* e_i *is* B_{i1} *and* de_i/dt *is* B_{i2} *THEN* u_i *is* C_i where: e_i and de_i/dt are the inputs and u_i is the FL controller's rule fired output. Also, the B_{i1} , B_{i1} *and* C_i , $i=\{1,2,..n\}$ are linguistic terms in the Membership Function fuzzy subsets of the universe U of discourse. Traditionally, in literature these set of rules are presented as a table form instead of a cascaded IF-THEN statements.

2.2. FL controller Optimization

Conventional Fuzzy Logic Controllers are designed using Top/Bottom approach (i.e. input/out signals are selected with their membership functions, then the rule base is assigned and finally inference). But in the proposed PSO optimized FL controller design the methodology is different; it is Bottom/Top design approach. Based on a standard full linguistic variables set for the Membership Functions, each input initially should have seven linguistic variables *{NB: Negative Big, NM: Negative Medium, NS: Negative Small, ZE: Zero, PS: Positive Small, PM: Positive Medium and PB: Positive Big}*.

The proposed optimization scheme requires two pass operation. On the first pass the optimization process should first generate the Rule-Base (RB) collection set. While in the second pass, the optimization process will optimize the number of the linguistic variables for each Member Function (MF) within the Fuzzification stage. This might seem counter intuitive and different than the conventional FL controller design. The reason behind this: is the fact that we do not have off hand the optimal linguistic variables; instead a generic full list will be used and it will be updated subsequently by the optimization process.

2.2.1. Rule-Base Selection

Traditionally, the construction of FL controller's rules has been mainly based on expert operator's control experience or actions or by data mining. Unfortunately, acquiring rules from experts is not an easy task; moreover it is very difficult to extract rules from static data bases (data mining). However, selecting a set of the most important and relevant fuzzy rules from a given rule base is an important issue in fuzzy rule base modeling; because it impacts the controller's effectiveness. Thus it is conceivable to eliminate the redundant or less important fuzzy rules from the rule base, which should result in a compact fuzzy model with faster controller's actions. Unfortunately, the decision as to which rules are redundant or less important is also not an easy task to perform.

For example consider a speed/position control of an elevator system application, which is proposed to operate using a FL-PID controller with two inputs and one output. The inputs are the error *e(t)* and the rate of change of the error *de(t)/dt*, while the output is the reference voltage for the discrete Pulse Width Modulation (PWM) drive unit. Assuming, each input to have the full set of the linguistic variables *{NB, NM, NS, ZE, PS, PM and P}*, accordingly, the number of rules generated is 49 (7x7) rules. Consequently, referring to Figure 1, which illustrate a single rule $\{R_i\}$ output, it is evident that each rule could have seven possible outcomes for a single control action (*i.e.* the rule firing output for the controller); namely *{NB, NM, NS, ZE, PS, PM and PB}*. In this work, an optimal outcome for the control action will be determined for each rule of the 49 rules by using the Particle Swarm Optimization (PSO) modified binary search algorithm (discussed in subsequent sections).

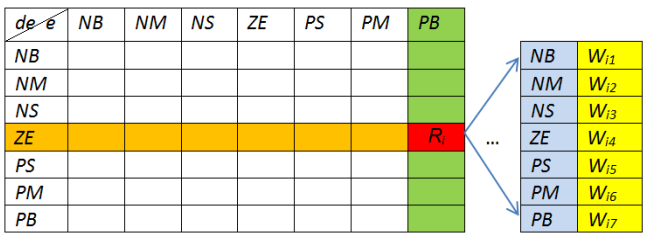


Fig. 1. FL-PID one rule outcome possibilities.

One way to resolve the issue of the huge number of tuning parameters is to implement an optimization process that

utilizes a binary search based approach (*i.e.* either the rule is applied or not). By implementing this methodology the optimization process reduces the number of parameters to 49 parameters instead of 343, which will minimize the computational effort needed in evaluating the system's models and constraints. Each optimization parameter is taken as a decimal number, which when converted into a seven digits binary number it represents a binary weight for the seven possible rule outcomes (ro vector): *[ro(i, 1), ro(i, 2), ro(i, 3), ro(i, 4), ro(i, 5), ro(i, 6), ro(i, 7)]*. This formulation may be described in a matrix form

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ . \\ . \\ P_{49} \end{bmatrix} \rightarrow \begin{bmatrix} ro(1,1), ro(1,2) \ldots\ldots ro(1,7) \\ ro(2,1), ro(2,2) \ldots\ldots ro(2,7) \\ ro(3,1), ro(3,2) \ldots\ldots ro(3,7) \\ . \\ . \\ ro(49,1), ro(49,2) \ldots\ldots ro(49,7) \end{bmatrix} \quad (1)$$

Where: the parameter P_i is the optimized ith rule parameter that has seven possible rule outcomes, which is depicted in Figure 1. For example: [1 0 0 0 0 0 0] represents the ith rule with NB as the optimal rule output for the FL controller, while [0 0 0 0 1 0 0] represents the ith rule with PS as the optimal rule output for the FL controller. This is a binary based selection of a rule's outcomes (*i.e.* only one outcome) that is performed by the first pass of the optimization process.

In contrast to the first stage rules screening, where the controller's rules output were selected based on fixed set of input linguistic variables, the second stage decides on the most significant input linguistic variables that will be kept (*i.e.* the ones that contribute the most for the system's response or the most influential). In order to find those influential linguistic variable rules combination, a weighted fuzzy rule base is proposed. Figure 2 illustrates the proposed weighted sum FL logic controller. The optimization process algorithm must tune these weights and come up with the most sensitive ones for the closed loop dynamical system's response. In the weighted fuzzy rule based system, every rule has a weight (i.e. number between zero and one $\omega_i \in [0,1]$). These rules could be ranked according to their weights; the rules with low weights would be deferred or abandoned, while the high weighted rules would be kept because they are the most influential ones during execution. Through ranking the weights, a simplified rule base that contains the most important ones will be generated.

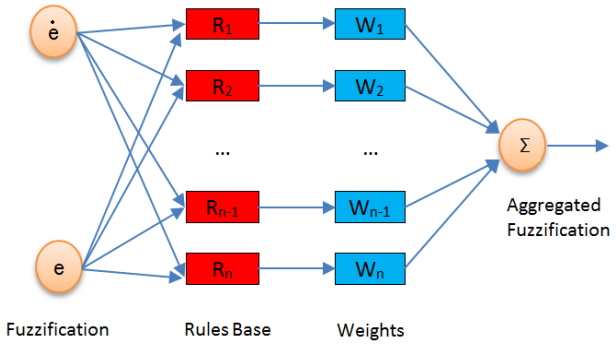


Fig. 2. Weighted Fuzzy Logic controller.

3. PARTICLE SWARM OPTIMIZATION (PSO)

The particle swarm optimization (PSO) has been shown to be an effective optima searching technique for difficult multidimensional none-convex problems (Kennedy, 1995). Recently, the PSO has been successfully applied to solve large scale optimization problems in a diversity of fields. Particle Swarm Optimization was shown that for certain problems it actually outperformed Genetic Algorithms (GA) (Eberhart 1998). It is considered a stochastic evolutionary computational technique with a minimal computational effort when compared with GA.

3.1. PSO Philosophy and Overview

Kennedy and Eberhart have discovered the Particle Swarm Optimizer while they were trying to model the social behavior of artificial life, such as bird flocking, fish schooling, …etc. (Kennedy, 1995). A good metaphor of the Particle Swarm Optimization (PSO) algorithm is to imagine a swarm of bees in a field searching for flowers. The objective of the swarms is to find the location of the most flowers. Basically, each bee will conduct a random search and it will memorize the densest location of flowers it encountered.

Also, by assuming that all the individual bees may share the information about their best findings of locations, then each bee will be guided by its own personal discovery and by the best location reported by the others. Consequently, by altering the direction of their trajectory to fly somewhere between the two locations, the bees will explore the field by overflying locations of greatest concentration and eventually will be drawn to the densest flower location (Robinsons, 2004).

3.2. PSO Algorithm

Based on the aforementioned philosophy in the previous section, the following definitions and notes are needed to realize the PSO algorithm.

- An *n=100* particles will be used in the PSO.
- Every particle is identified with: current position p_i, velocity v_i and p_i_Best in the searched solution space.
- All particles share their information about their best findings of locations.
- The individual particle best position p_i corresponds to its minimum evaluated cost function $J(p_i)$ throughout its search journey.
- The global best position is the best of all the particles bests, p_Global.
- The solution space will be explored by driving the particles to move according to their best findings and also influenced by the best location reported by the other particles.

The mathematical representation of the swarm iterative updates for all particles, *n*, maybe given by the following equation:

$$p_{i\_Best}(t+1) = \begin{cases} p_{i\_Best}(t) & if \quad J\left(p_{i\_Best}(t)\right) \le J\left(p_i(t+1)\right) \\ p_i(t+1) & if \quad J\left(p_{i\_Best}(t)\right) > J\left(p_i(t+1)\right) \end{cases}$$

$$p_{Global}(t)=min\left(J\left(p_{i\_Best}\right), J\left(p_{Global}(t)\right)\right)$$

$$p_i(t)\in \boldsymbol{p}(t),\ \ i=0,\ 1,\ 2,\ \ldots n \tag{2}$$

Figure 3 illustrates how the particles are moving and sweeping the solution space intelligently and it also summarizes the used pseudo-algorithm. Each particle in the swarm tracks its position by means of two vectors; one accelerates the particle in the direction of its own and another towards a global best for the whole swarm (best of the bests). The advancement of the particles in the solution space is controlled by a simple kinematic equation of the form:

$$p_i(t+1)=p_i(t)+\Delta t\ v_i(t) \tag{3}$$

Where the Δt is the advancement in time (increments) and p_i is the current location of the particle (*i.e.* the solution). On the other hand, the particles' velocities are updated during the search with the following formula:

$$v_i(t+1)=wv_i(t)+c_1 rand()\left(p_{i_{Best}}(t)-p_i(t)\right)+ c_2 rand()\left(p_{Global}-p_i(t)\right) \tag{4}$$

Where w is a scalar that generates some form of momentum for the particle taken from the previous iteration. The value that is used in this paper is 0.75, however, this value could be changed dynamically as well (in an adaptive fashion). The constants c_1 and c_2 represent the emphasis toward the particle's best or toward the global swarm's best weighted by random terms. Typical values for these weights are within the interval [0 4], but good reported values in the literature of the scalars are $c_1 = c_2 = 2$. The local best is the best value of the solution attained by the particles and the global best is the best value of the solution attained by the whole swarm. The stochastic behavior of the PSO algorithm may be realized through the given two random sequences: $rand(\ ) \in [0,1]$. Please note that the PSO algorithm is structured in such away it will not get stuck into local optima.

Pseudo Algorithm

Step I: Initialize position, velocity, local best, and global best of all the particle swarms

p_i=rand() ; i=1,2,…, n

v_i=rand() ; i=1,2,…, n

$p_{i\_Best}=p_i$; i=1,2,…, n

$compute\ fitness\ function\ J_i\ for\ each\ p_{i\_Best}$;

$p_{Global} = p_{i\_Best}\ that\ corresponds\ to\ \min(J_i)$;

Step II: Repeat until fitness function of global position is acceptable or m iterations is reached

compute fitness $J_i(p_i)$;

if $J_i(p_i)<J_i\left(p_{i\_Best}\right)$ *then* $p_{i\_Best}=p_i$

$p_{Global}=p_{i\_Best}$ *if* $min\left(J_i\left(p_{i_{Best}}\right)\right)$

update particle velocity

$v_i=wv_i+c_1 rand()\left(p_{i_{Best}}-p_i\right)+\ c_1 rand()\left(p_{Global}-p_i\right)$

update particle position $p_i=p_i+\Delta t v_i$

loop step II

Step III: Optimal solution=the particles with best global fitness values

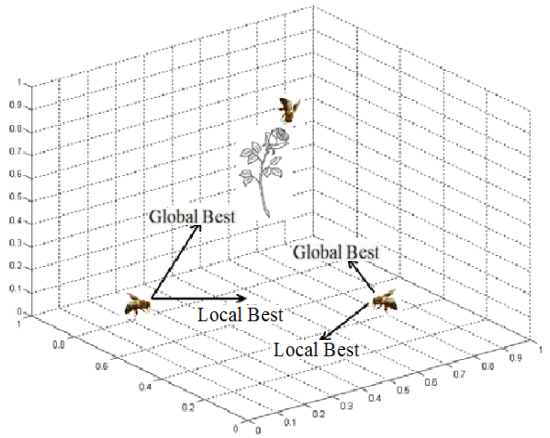


Fig. 3. Particles movement in solution space.

3.3. PSO Implementation for the Rules Auto-generation

Many real life and engineering applications require finding optimal parameters that meet certain objectives (for multi-objectives problems), which are subjective to a diversity of constraints. Typically, in the optimization processes the objective or cost function is a scalar function, which will be denoted by $J(\boldsymbol{p})$, $\boldsymbol{p} \in \mathbb{R}^{\boldsymbol{n}}$.

The FL-PID controller optimization problem that was posed previously maybe formulated as follows:

$$\begin{aligned} &Minimize\ J(\boldsymbol{p}),\ \ \boldsymbol{p}=[p_1,p_2,\ldots p_n]^T \\ &\ p_i \\ &Subject\ to:\quad G_c(\boldsymbol{p}, \boldsymbol{e}),\ \ G_p(\boldsymbol{u}) \end{aligned} \tag{5}$$

Where: $J(\boldsymbol{p})$ is the optimized cost function: in this work it is the error function or deviation between the user's desired trajectory and the system's output (i.e. tracking system) and G_c *and* G_p are the controller's and the system's dynamical equations; respectively.

The optimization problem casted in equation (5) can be solved by a diversity of numerical techniques. However, in this work the Particle Swarm Optimization (PSO) is adopted to search the solution space $\mathcal{M}$ for an extrema. The PSO is considered an evolutionary population-based stochastic optimization technique, which is proven to be effective and easy to implement. The main objective of the optimization process in hand is to estimate the optimal search parameters; p_i, that are needed to reshape the system's dynamic to a desired one (i.e. following user's predefined trajectories).

4. ELECTRICAL ELEVATOR CONTROL APPLICATION

The PSO based FL-PID controller optimization proposed method will be demonstrated over an academic engineering application of an elevator speed profile tracking. The amount of work in literature on elevator systems is overwhelming. However, one may recognize two lines of research: group car control/scheduling and single car position/speed tracking applications. To demonstrate the effectiveness of the proposed approach let's consider an electrical elevator speed tracking, which directly impacts the passengers ride comfort.

For the elevator application, the FL-PID controller objective is to track a user defined velocity transport profile for the elevator's motor. The motivation behind this objective is to create lift quality for the passengers by providing passengers' good ride comfort. Typically, this task involves addressing lateral and vertical vibrations, acceleration and deceleration, and jerk, which requires the controller to regulate the accelerations/deceleration $\ddot{y}(t)$ in m/s^2 and minimize the jerk $\dddot{y}$ in m/s^3. Other performance measures for the elevator systems are also possible; such as noise, door opening speed, safety…etc. For an objective analysis and evaluation of the ride quality, one may consider: ISO Standard 18738 (Boutler, 2000; Abraham, 1984; Li, 2004; Rebai, 2015). Figure 4 depicts the user defined velocity transport profile that is needed to be enforced by the FL-PID controller for ride comfort.

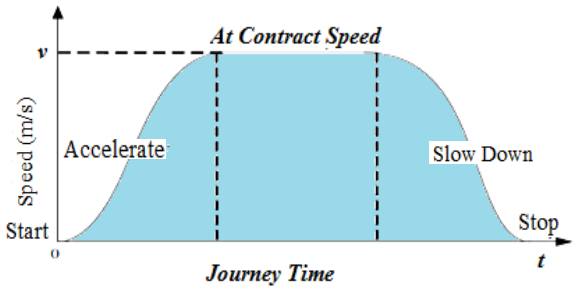


Fig. 4. Elevator car speed profile for an elevator.

Let's consider a 2:1 gearless traction electrical elevator model that is depicted in Figure 5, which will be used for testing the controller's performance as designed. Please note that the complete derivation and analysis of the elevator's dynamical model are omitted due to lack of space, however, interested readers should consult the authors' previous work on the subject (Abdalla, 2011).

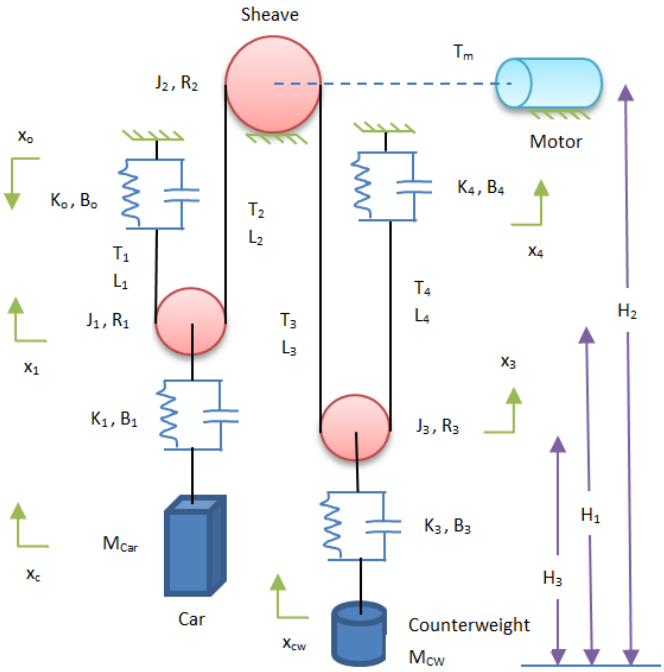


Fig. 5. Gearless Traction Electrical Elevator Model.

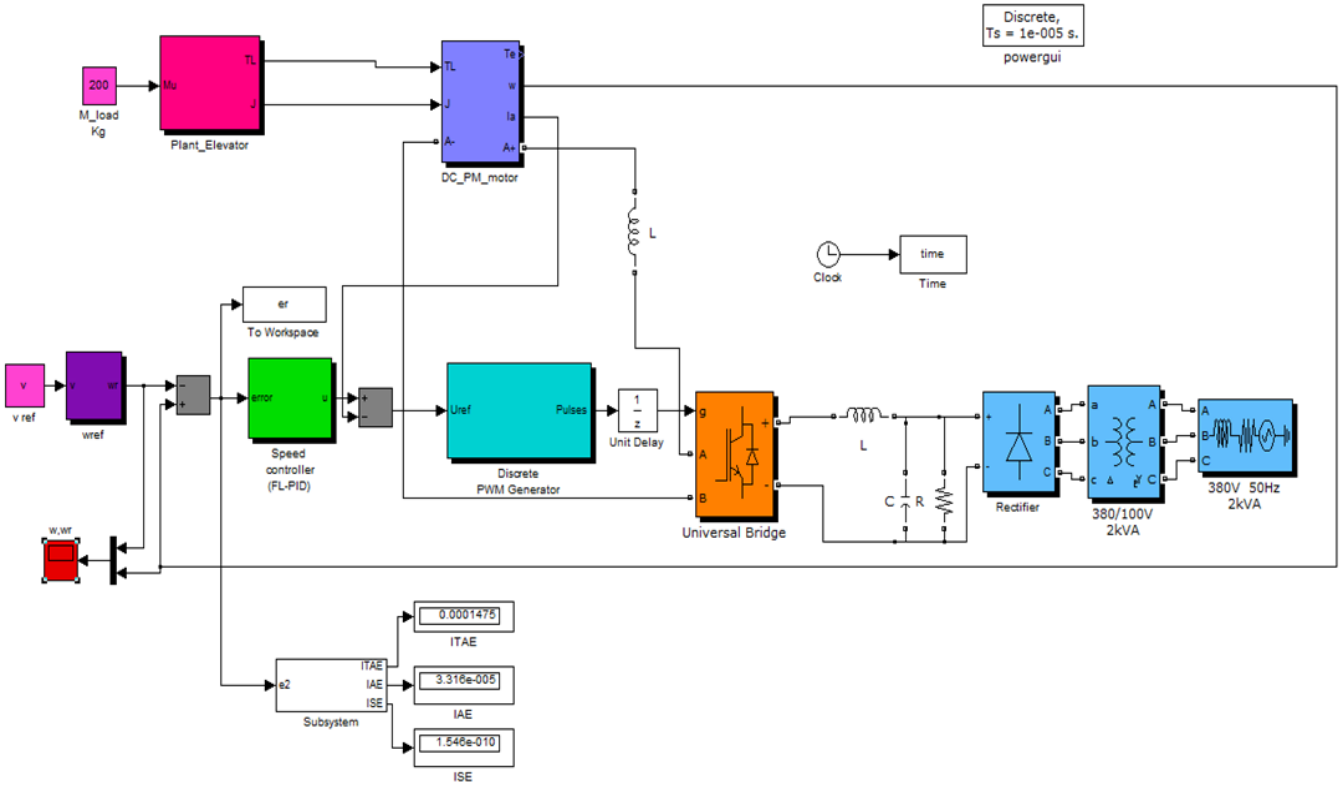


Fig. 6. Elevator simulated model with controller.

Figure 6 depicts the elevator's complete SIMULINK/ MATLAB® implemented model that was used in the simulation and design of the system, which is also complemented with the proposed FL-PID controller. The simulated system model consists of basic modules: a three phase voltage source, universal bridge, Pulse Width Modulation (PWM) and the elevator subsystem that consists of six internal subsystems and the PM-DC motor subsystem (Abdalla, 2011). Table 1 lists all the elevator's simulated parameter values that were used to generate the results.

Table 1. Elevator's Simulated Parameters.

Armature resistance (R_a)	0.49 Ω
Armature inductance (L_a)	4.3 mH
Motor constant (K_e)	0.49
Coulomb friction value (t_f)	0.18
Coefficient of viscous friction (K_f)	4.6e-4
Radius 1 (R_1)	0.2 m
Radius 2 (R_2) Sheave	0.3 m
Radius 3 (R_3)	0.2 m
Moment of inertia (J_1)	0.08 Kg.m^2
Moment of inertia (J_2)	0.15 Kg.m^2
Moment of inertia (J_3)	0.08 Kg.m^2

4.1. FL-PID Controller Design

The PID fuzzy logic based controller structure presented earlier in Figure 4 need to be synthesized. To illustrate the effectiveness of the proposed design approach, two strategies in generating the FL-PID controller will be implemented. The first approach will be the Fuzzy Lyapunov synthesis (analytical techniques) and the second one will be the PSO (stochastic), both will generate the needed rules of the Mamdani-type FL controller for the sake of comparison. Their membership functions, rule base arrays and generated rules control surfaces will be compared.

The major steps in the FL controller design constitutes: creating a knowledge base of the rules (Inference), establishing membership functions for the crisp inputs (Fuzzification) and implementing membership functions for converting to crisp outputs (Defuzzification). In this work, the sensed input signals (measurements) that are fed to the FL controller are the error and its rate of change, $(e(t),\ \dot{e}(t))$.

$$e(t)=v_d(t)-v(t) \tag{6}$$

Where $v_d(t)$ is the desired reference speed of the car (*i.e.* the speed profile) and $v(t)$ is the elevator's actual speed output. For proper velocity profile tracking, the FL-PID controller needs to minimize the deviation or error function $e(t)$.

Fuzzification: The initial design of the proposed FL-PID controller will be based on two inputs: the error $e(t)$ and the rate of change of the error $\dot{e}(t)$, and one output: reference voltage $u(t)$ for the discrete PWM generator. Initially, the FL controller will be implemented with seven linguistic variables that are tentatively selected to span the two inputs and the output ranges; namely: Negative Big (NB), Negative Medium (NM), Negative Small (NS), Zero (ZE), Positive Small (PS), Positive Medium (PM) and Positive Big (PB). However, the initial number of these linguistic variables will be reduced through the optimization process later on.

Defuzzification: A weighted center that utilizes the gravity method will be used to generate the crisp output. Equation 7 illustrates the estimation of the crisp controller output (gravity method).

$$u^*(t)=\frac{\sum_{i=1}^{m} W_i\ \mu_{ik}\ c_i}{\sum_{i=1}^{m} W_i\ \mu_{ik}} \tag{7}$$

Where W_i is the weight of rule, μ_{ik} denotes the degree that instance matches the rule R_i , and $c_i \in \{ C_1, \ldots. C_M\}$ is the centroid of every fuzzy set.

Rule-Base: The rule-base matrix will be derived first by using the Fuzzy Lyapunov synthesis and the second method will be the proposed PSO. Both techniques will generate the needed rules of the Mamdani-type FL controller. The complete design strategies will be presented by the following subsequent sections.

4.2. Rule-Base Generation using Lyapunov Synthesis

The FL-PID controller performance in tracking the user's defined speed profile heavily hinges and affected by the rule-base matrix generation step. A systematic approach in deriving the needed rule-base matrix is the Lyapunov Synthesis approach (Carvajal, 2000).

Define a continuously differentiable function $F(x)_{\ni}\ x(t) = [e(t)\ \ \dot{e}(t)]^T$ as a Lyapunov function with the following standard characteristics:

- $F(x) = 0$, for only $x = 0$,
- $F(x) > 0, x \in \mathbb{R}^2 - \{0\}$,
- $\frac{dF}{dt} < 0, x \in \mathbb{R}^2 - \{0\}$, where $\mathbb{R}^2 - \{0\}$ is the neighborhood of zero; excluding the origin itself.

Now, assuming that the reference speed is v_d and its derivatives $\dot{v}_d$ and $\ddot{v}_d$ are bounded and available to be used by the controller, let's choose $F(x)$ as a Lyapunov function of a quadratic form:

$$F(x)=\frac{1}{2}\left(e^2(t)+\dot{e}^2(t)\right) \tag{8}$$

where the error is given as in equation (6). It is clear that the proposed Lyapunov function $F(x)$ satisfies the first two conditions, while the last condition needs to be fulfilled to establish stability in the sense of Lyapunov. Consequently, the derivative of the Lyapunov function yields the following:

$$\dot{F}(x)=\dot{e}(t)(e(t)+\ddot{e}(t)) \tag{9}$$

By knowing that the second derivative of the error is proportional to the output of the FL controller (u) then the corresponding requirement of Lyapunov stability becomes:

$$\dot{F}(x)=\dot{e}(t)(e(t)+u(t))<0 \tag{10}$$

Consequently, equation (10) must be negative definite at all times, which leads us to draw the following conclusions:

Constraint 1: If $e(t)$ and $\dot{e}(t)$ are both positive then take $u(t) < -e(t)$

Constraint 2: If $e(t)$ and $\dot{e}(t)$ are both negative then take $u(t) > -e(t)$

Constraint 3: If $e(t)$ and $\dot{e}(t)$ have opposite signs then take $u(t) = 0$

Now, using the aforementioned constraints, one can construct the FL-PID controller rule base. Table 2 and Figure 7 depict the Laypunov-based designed rule base control surface and enlist the controller's rules. It is evident that the surface smoothness dictates the effectiveness and smoothness of system's control actions and consequently the controller's performance. In addition, the Lyapunov derived constraints did not provide enough information to reduce the number of the linguistic variables.

Table 2. Lyapunov FL-PID Controller Rules.

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

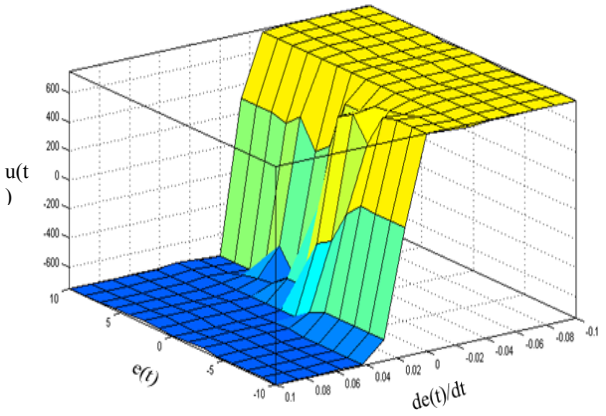


Fig. 7. Lyapunov based FL-PID controller surface.

4.3. Rule-Base Generation using Proposed PSO

The previously introduced Particle Swarm Optimization (PSO) algorithm will be executed to generate an optimal set of FL-PID rule-base matrix. The cost function that was implemented in the PSO optimization process is the norm of the difference in speeds profiles (i.e. the error function or deviation): $J(\boldsymbol{p}) = \| v_d(t) - v(t) \|$.

The optimization process will be implemented on two passes. In the first pass, the PSO will perform a binary search for the controller's actions (i.e. one rule outcome), which was fully illustrated in Figure 5. While the second PSO optimization pass will generate the set of rules' weights that will be implemented in equation (7), which was also presented in Figure 6. The resulted PSO optimization outcomes should result in a controller's output actions that should mimic the system's operator best reaction experiences for the designated inputs and it should provide smooth speed tracking.

In the first pass, the PSO selected the 49 rule base outcomes, $p^* = \{p_1^*, p_2^*, \dots, p_{49}^*\}$, recalling that $p_i \in \mathcal{B}[0,1]$ is a binary byte, refer to equation (1). While in the second pass, the PSO searched the solution space $\mathcal{M}$ for every rule optimal weight, $w^* = \{w_1^*, w_2^*, \dots, w_{49}^*\}$. After applying the second pass, the number of rules reduced from (49) rules to (21) rules by keeping only the significant weights. The corresponding set of significant weights of these rules is given by the following weight vector: *[0.6238, 0.8518, 0.6069, 0.6193, 0.8967, 0.5895, 0.5948, 1.000, 0.5864, 0.2994, 1.000, 0.3128, 0.5912, 1.000, 0.5834, 0.6374, 0.8822, 0.5692, 0.6102, 0.8275, and 0.5934]*. Consequently, this reduction in weights has reduced the linguistic variables set that pertain for the error input from seven linguistic variables to only three linguistic variables. Table 3 and Figure 8, summarizes the PSO results and depicts the FL-PID rule base control surface, which exhibits smoother controller actions with a lower number of rules than the Lyapunov-based design.

Table 3. PSO-based FL PID Rules.

	NS	ZE	PS
NB	PM	PB	PS
NM	PM	PB	PM
NS	PM	PB	PM
ZE	PS	ZE	NS
PS	NM	NB	NM
PM	NM	NB	NM
PB	NS	NB	NM

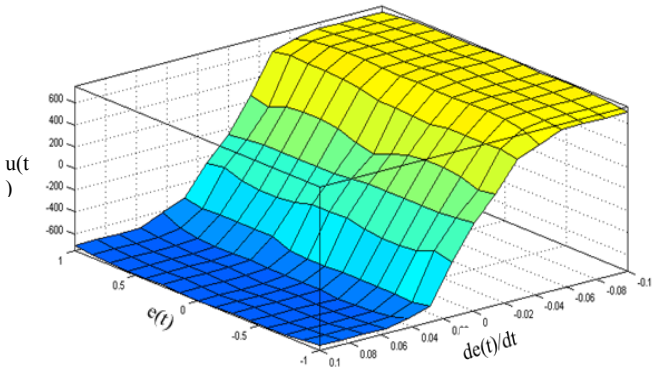
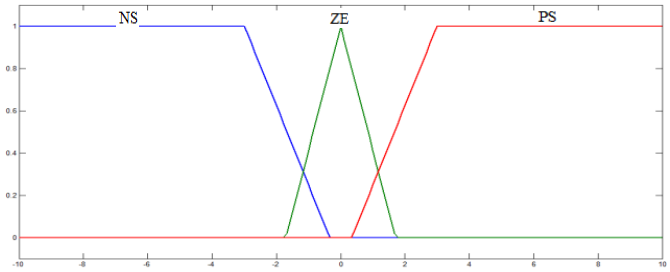


Fig. 8. PSO-based FL PID Control Surface.

Note that since this elevator application is more sensitive to the rate of change of the error (*i.e.* speed profile change); the $\frac{d\dot{e}}{dt}$ linguistic variables are highlighted (fine), while the error in speed is taken as a coarse distribution (*i.e.* no need for large speed deviations). Also, note how the PSO generated weights about the ZE column of the error e(t) were the largest. That is actually expected because fine tuning is performed by the controller in that region (i.e. giving high sensitivity).

The generated PSO based FL-PID controller control surface that is presented in Table 3 and Figure 8 demonstrates smoothness of the controller's control signals (actions) as a function of the controller's inputs, which provides a sense of the controller's trajectories for a two input based systems. In this case, this plot shows the control signals (reference voltage for the discrete PWM generator) as a function of the error and the rate of change of the error. Clearly, all trajectories are directed towards the origin, which gives an indication of the controller's stability. Finally, even with much less number of used rules, the PSO-based tuning generated a control surface that is smoother than the one generated by Lyapunov design, which was depicted earlier in Table 2 and Figure 7.

The corresponding membership functions for the two inputs and the output of the FL-PID controller are shown in Figure 9. The PSO has generated the following membership functions: (i) membership functions for the error, $\mu_A(e)$ (ii) membership functions for the change of error, $\mu_B(\dot{e})$ and (iii) membership functions for the reference voltage for the discrete PWM generator, $\mu_C(u)$.



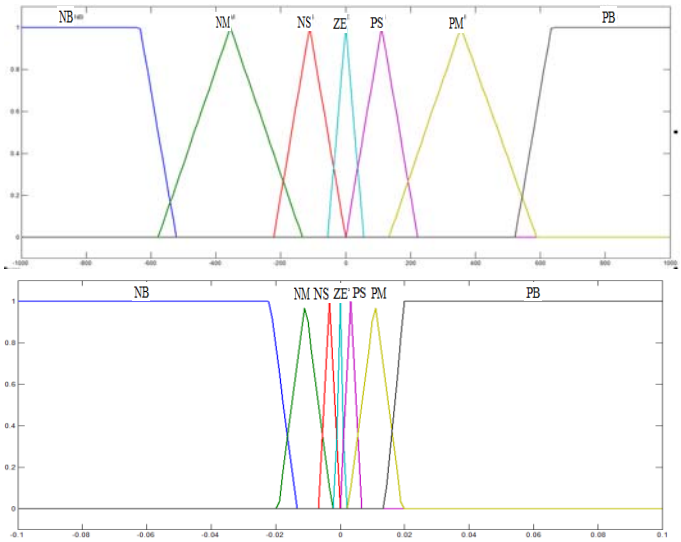


Fig. 9. FL-PID Controller Membership Functions.

To test the effectiveness of the FL-PID controller and to validate its ability to generate proper control laws that will track a designer's velocity profile, Matlab computer simulations have been used for the elevator system. Figure 10 depicts the results for the PSO FL-PID. Each figure illustrates one of the important elevator's physical variables. Typically the acceleration range should be between (-1.5 to 1.5) mps^2 for human comfort, which is obviously met by the controller's output.

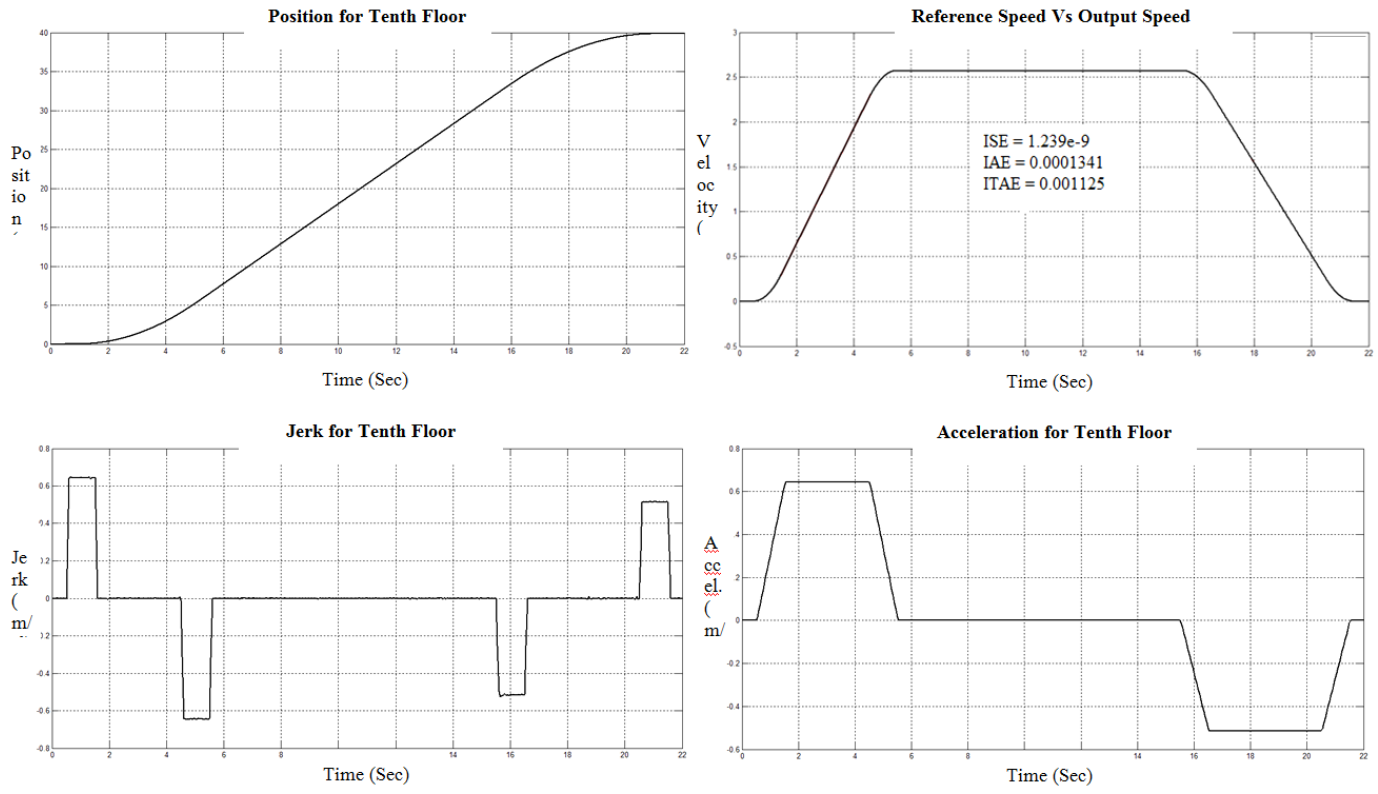


Fig. 10. Response Results using FL-PID Tuned with PSO.

Also, the controller tracking of the speed profile is done nicely with minimal amount of jerk. Figure 10 also depicts the position, velocity, acceleration and jerk trajectories for the tenth floor user calls. The controller's ability to track the designer's speed profile is evident. Also, the acceleration and jerk values are within human tolerated values.

5. CONCLUSIONS

A novel Fuzzy Logic controller design methodology was proposed. The method dealt with the auto generation of a FL-PID rule-base matrix, in contrast, to the classical design methodology that is based on human operator experience and intervention.

Also, the new design methodology offered an optimized number of possible Fuzzy Logic linguistic variables that are needed in the fuzzification and output stages of the controller. This resulted in simplifying the fuzzy rule-base matrix entries that are generated from the first optimal screening.

In the FL-PID rule-base matrix optimization, a modified binary search optimization algorithm, which was based on Particles Swarm Optimization (PSO) strategy was proposed and successfully implemented. The performance of the PSO designed FL-PID controller was demonstrated on controlling and tracking the speed profile of a gearless electrical elevator system. It demonstrated the ability to track both position and speed profile for an electrical elevator system; yet maintaining a minimal amount of acceleration and jerk and creating passengers comfort. The designed controller also exhibited higher performance when compared with a FL-Lyapunov synthesized controller.

REFERENCES

Abdalla M and Al-Jarrah T. (2011). Fuzzy Logic Control of an Electrical Traction Elevator. *JJMIE: Jordan Journal of Mechanical and Industrial Engineering*, 2011(2) 97-106.

Abdalla, M. (2009). Particle Swarm Optimization (PSO) for Structural Damage Detection, *Proceedings of the 3rd Applied Mathematics, Simulation, Modelling, Circuits, Systems and Signals*, Greece, p43-48.

Abraham E. (1984). Performance Criteris: Car Ride Quality. *Elevator World*, I-19-23.

Ahmad A. et. al. (2012). Application Of Particle Swarm Optimization To Optimal Power Systems. *International Journal of Innovative Computing, Information and Control*, 8(3), p1705-1716.

Allaoua B et. al. (2008). The Efficiency of PSO Applied on Fuzzy Logic DC Motor Speed Control. *Serbian Journal of Electrical Engineering*, (5), p247-262.

Ang, K.H. et. al. (2005). PID Control System Analysis, Design, and Technology. *IEEE Transactions on Control Systems Technology*, 13(4), 2005, p559-576.

Åström. K. J. et. al. (2000). The future of PID control. *Control Engineering Practice,* (9), p1163- 1175.

Bandyopadhyay R et al. (2001). Autotuning a PID Controller: A Fuzzy-Genetic Approach. *Journal of Systems Architecture*, (47), p663-673.

Bernadette B., et. al. (2000). On the Choice of Membership Functions in A Mamdani-Type Fuzzy Controller, *Fuzzy Logic Journal*.

Bingul Z. et. al. (2000). A New PID Tuning Technique Using Differential Evolution for Unstable and Integrating Processes with Time Delay. *Neural Information Processing*, Volume 3316 series Lecture Notes in Computer Science, p254-260.

Bingul Z. et. el. (2010). A FL Controller Tuned with PSO for a 2 DOF Robot Trajectory Control. *Expert System Application*, p1017-1031.

Bingul Z. et. al. (2011). A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control. *Expert Systems with Applications*, (38), p1017-1031.

Boutler, B. (2000). *Elevator modeling and DC drive speed controller design*, ApIC S LLC.

Carvajal et. al. (2000). FL-PID Design, Performance, and Stability. *Info Science*, (123), p249-270.

Chonghua W., 2015, *A Study of Membership Functions on Mamdani- Type Fuzzy Inference System for Industrial Decision-Making*, Dessertation, Lehigh Preserve

Cui, X et. al. (2004). A Swarm-based FL Control Mobile Sensor Network for Hazardous Contaminants Localization, *IEEE Int. Conf. on Mobile Ad-hoc and Sensor System*.

Eberhart, R.C. et. al. (1998). Evolving Artificial Neural Networks, *Proceedings of International Conference of Neural Networks and Brain*, Beijing, China.

Elwer A. et. al. (2009). Improved Performance of Permanent Magnet Synchronous Motor by Using PSO Techniques. *Journal of Power Electronics*, (9), p207-214.

Feng, L. (2000). Self-Tuning of PID Controllers by Adaptive Interaction, *Proceedings of the American Control Conference*, Chicago, p3676-3681.

Fodor J. (2004). Left-continuous t-norms in fuzzy logic: An overview. *Acta Polytechnica Hungarica,* 1(2).

Galichet, S. et. al. (1995). Fuzzy controllers: Synthesis and equivalences. *IEEE Trans.Fuzzy Systems*, (3), p140-148.

Georg K I et. al. (2001). Two-Level Tuning of FL-PID Controllers. *IEEE Trans. on System, Man, and Cybernetics*, (31), p263-269.

Georg K I et. al. (1999). Analysis of Direct Action Fuzzy PID Controller Structure. *IEEE Trans. on System, Man, and Cybernetics*, (29), p371-377.

ISO 18738-1 (2012). *Measurement of ride quality -- Part 1*: Lifts (elevators)

Kazemian H B. (2005). Developments of FL-PID controllers. *Expert Systems: The Int. Journal of Knowledge Engineering and Neural Networks*, (22), p254-264.

Kennedy, J. et. al. (1995). Particle Swarm Optimization. *Proceedings of IEEE International Conference on Evolutionary Computation*, p1942-1948.

KK., Li et. al. (2004). A General Survey on Lift Ride Quality at Public Buildings of the Hong Kong Special Adminstrative Region, *Council on Tall Buildings and Urban Habitat: CTBUH 2004 Seoul Conference.*

Krohling, RA. et. al. (1997). Designing PI/PID Controllers for a Motion Control System based on Genetic Algorithms, *Intelligent Control: Proceedings of the IEEE*, p125-130.

Mamdani, E. H., et. al. (1981). *Fuzzy Reasoning and Its Applications*, Academic Press, London

Moon, B.S. (1995). Equivalence between fuzzy logic

controllers and PI controllers for single input systems. *Fuzzy Sets and Systems*, (69), p105-113.

Nguyen H. T. et. al. (1997). *A first course in fuzzy logic*, Boca Raton: CRC Press, Publ info.

Omkar, S. et. al. (2013). An Optimal Fuzzy Logic Controller Tuned with Artificial Immune System, *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications*, p507-518.

Passino, K. et. al. (1998). *Fuzzy Control. Menlo Park*, Addision-WelsleyLongman, California.

Palupi D. et. al. (2011). Particle Swarm Optimization: Technique System and Challenges. *International Journal of Computer Applications*, 14(1), p19-27.

Prashant, M. et. al. (2004). A Method For PID Controller Tuning Using Nonlinear Control Techniques, *Proceedings of the American Control Conference*, Boston, p2925-2930.

Rebai Aissa, et. Al. (2015). Design of an optimized fractional order fuzzy PID controller for a piezoelectric actuator, *CEAI*, 17(3), p41-49.

Robinsons et. al. (2004). Particle Swarm Optimization in Electromagnetics. *IEEE Transactions on Antenas and Propagation*, 52(2), p397-407.

Tang K. S. et. al. (2001). An Optimal FL-PID Controller. *IEEE Transactions on Industrial Electronics*, (48), p757.

Schoeman, I. (2010). *Niching in Particle Swarm Optimization*, Dissertaion, University of Pretoria, Pretoria

Sugeno, M. (1985). *Industrial Application of Fuzzy Control*, North-Holland, New York.

Vikram C. (2014). Comparative Analysis of Tuning a PID Controller using Intelligent Methods. *Acta Polytechnica Hungarica*, 11(8), p235-249.

Visioli A. (1999). FL Based Set-Point Weight Tuning of PID Controllers. *IEEE Trans. on Systems Man and Cybernetics - Systems and Humans*, (29), p587-592.

Visioli A. (2001). Tuning of PID Controllers with FL. *IEEE Control Theory and Application*, (14), p1-8.

Wahsh S et. al. (2005). *Tuning PI-Controller Using PSO for Control of Permanent Magnet Synchronous Motor*, 5th st. Petersburg Simulation Workshop, RUSSIA, p239-244, 2005.

Woo Z W et. al. (2000). FL Controller with Self-tuning Scaling Factors. *Fuzzy Systems*, (115), p321 – 326.