**Mário Sérgio Freitas Ferreira CAVALCANTE[1], Lívia de Maria Calado Machado SOARES[2], Ícaro Bezerra Queiroz de ARAÚJO[2], Fábio Meneghetti Ugulino de ARAÚJO[1]**

Federal University of Rio Grande do Norte - UFRN (1), Federal University of Alagoas - UFAL (2)

doi:10.15199/48.2023.02.06

# Comparative Study of Optimizations for Control Problem Using Fuzzy Type-2

***Abstract.***
*In general, fuzzy logic are able to handle several problems that classic logic is not cabaple, mainly due its capacity to represent the imprecision and uncertanty of human logic and reasoning. But, even classic fuzzy logic or type-1 fuzzy logic are not adquate to fully represent the human knowledge, so type-2 fuzzy logic is more suitable to solve this problem. Controllers based on those logic are known as type-1 and type-2 fuzzy controllers, these controllers are hard to tune due its large number of parameters. In literature, there are a lot of strategy to solve this problem for both controllers based on meta-heuristics. To investigate and validate the controllers obtained it was used a Servo motor from Quanser, a control problem which requires precision and velocity in error correction. We tested several controlers and optimization techniques based on classic PI controllers and particle swarm optimization, genetic algorithms and ant colony optimization based on three diferents avaliation index IEA, ITEA e Goodhard index. By analyzing the results obtained, the type-2 fuzzy controller showed significant gain for the control of this plant, when optimized with the PSO method. From the results, it can also be inferred that the ant algorithm was not suitable for this problem, with the proposed evaluation function.*

***Streszczenie.***
*W przemyśle kilka strategii i algorytmów sterowania jest już używanych i opisanych w literaturze. Wśród istniejących technik, regulatory rozmyte wyróżniają się zdolnością do radzenia sobie z poważnymi nieliniowościami występującymi w rzeczywistych instalacjach oraz zdolnością do reprezentowania wiedzy eksperckiej, która jest nieprecyzyjna i matematycznie niedokładna. W tej pracy zbadano dwa typy istniejących regulatorów rozmytych opartych na modelu Sugeno, są to rozmyte typu 1, tutaj sklasyfikowane jako konwencjonalne rozmyte i rozmyte typu 2. Analizując otrzymane wyniki, regulator rozmyty typu 2 wykazał znaczny zysk w sterowaniu tą instalacją, gdy został zoptymalizowany metodą PSO. Z wyników można również wywnioskować, że algorytm mrówkowy nie był odpowiedni dla tego problemu z zaproponowaną funkcją ewaluacyjną. (Badanie porównawcze optymalizacji problemu sterowania przy użyciu algorytmu rozmytego)*
***(Badanie porównawcze optymalizacji problemu sterowania przy użyciu algorytmu rozmytego)***

**Keywords:** CAD, optimal design, deterministic algorithms, evolutionary algorithms
**Słowa kluczowe:** CAD, algorytm ewolucyjny, logika rozmyta

## Introduction

The main aspects of information are imprecision and uncertainty. Among the existing theories to deal with these characteristics, the most prominent are set theory and probability theory. But both are not sufficient to deal with the wealth of information provided by humans (1).

In the quest to treat this imprecision present in knowledge (2) proposed a new type of logic, known as Fuzzy Logic (*Fuzzy Logic*). Fuzzy Logic introduced the concept of fuzzy sets capable of handling the vague aspect of information (1). Instead of limiting the element of a set to fully contained or not contained, this new logic allows working with the uncertainty of information in a universe of discourse, which allows the element of analysis to be "partially" contained in a certain area of information.

To use this technique in control systems, two fronts of studies stood out. Sugeno (1972) proposed a new concept of fuzzy measures, which uses linear functions in the output of inference, and with this, a new fuzzy model emerged, known as Fuzzy Takagi-Sugeno(-Kang) (TSK) or Fuzzy-Sugeno. Parallel to this, (3) proposed a new model for this logic to calculate the inference of this logic, which uses membership functions both in the input logic as in the output, and this model became known as Fuzzy-Mamdani. This technique was tested by the authors themselves to control a steam engine.

Applications using fuzzy logic to control nonlinear systems can be seen in petrochemical (4), energy (5; 6; 7; 8), aeronautical (12) industries. However, the use of this technique is not restricted only to control (9), several areas such as: signal processing, communication, agriculture (10), expert systems (11), medical and psychological also use this technique (13).

One of the industries that most proves the efficiency and quality of this control technique is the automotive area. Practical problems related to the control of suspension (14), the control of electronic vehicles (15) and the control of braking systems with *Antilock Braking System* (ABS) (16).

The work of Mirzaei *et al.* (2015) used a conventional fuzzy logic-based controller, based on the Sugeno model, to control a vehicle ABS system. Due to the difficulty in tuning each parameter of the controller, the authors used a genetic algorithm to find the best values to perform the plant control. The experiment developed used different terrains to simulate the behavior and tune the controller. The results obtained by the authors showed that, when compared with proportional-integrative (PI) controllers for the tests performed, the fuzzy controller obtained better results, by generating less oscillations when the ABS is triggered, fast convergence to the *setpoint* and more satisfactory performance for the different conditions of the specified trajectory (16).

Fuzzy controllers were used to increase the charging speed of electric vehicle stations. The controller used three inputs: error, integral of the error and the state of charge in the battery. The results were obtained under different operating conditions and prove that the proposed controller can be an alternative to the currently used controllers (17) .

Industrial applications of fuzzy controllers emphasize two important characteristics related to this control strategy (PRECUP; HELLENDOORN, 2011):

- When the plant presents large nonlinearities, obtaining the mathematical model of the plant and tuning a controller becomes a complex process. The fuzzy controller is a more viable alternative, when compared to the classic control (conventional) for not basing its tuning on linearized models;

- Compared to classical proportional-integrative-derivative (PID) control, fuzzy control can be strongly based and focused on the human operator's experience, and the fuzzy controller can more accurately model the experience (linguistically), when compared to classical controllers.

In (18) the pitch angle of an aero generator was con-

trolled. In this work, the authors used a fuzzy-Mamdani controller with two inputs and one output, in which the two inputs correspond to the error and the derivative of the error and the output is the control signal that is applied directly to the actuator of the plant, which is a servo motor that acts on the aero generator, which modifies its angle.

There is also, in some works, the use of this technique in conjunction with other control techniques, in order to facilitate the tuning (19) who used a methodology of adaptive fuzzy controller to control an electric servo motor.

In this study, the designed fuzzy controller generates a reference tracking with zero error. The advantage of using fuzzy controller strategies joined with other techniques is that it allows the tuning of the controller rules to be done automatically and thus makes the system acquire a satisfactory response.

**Type-2 Interval Fuzzy Logic**

Type-2 fuzzy logic is well characterized by the uncertainty spot present in a pertinence function associated with a primary pertinence function. This "spot" of uncertainty represents the uncertainty associated with the different values that the function can obtain in the same universe of discourse.

Thinking about a type-1 pertinence function, by including a spot of uncertainty, on the position and even the shape, in all its domain, one obtains a type-2 pertinence function, as can be seen in Fig. 1. In this case, as shown in the figure, an input value x has different pertinence values in each membership function. Performing this operation for

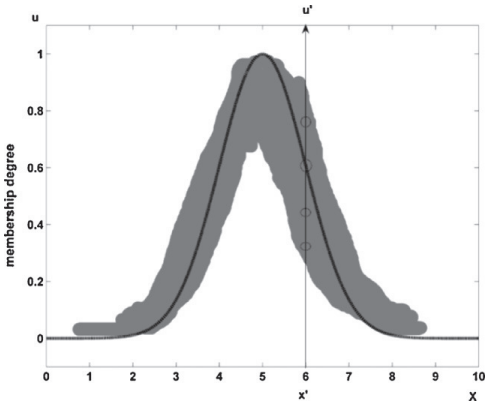


Fig. 1. Type-2 pertinence function similar to a type-1 pertinence function with uncertainty patch (20).

all possible values of x belonging to the universe of discourse ($x \in X$), we obtain a three-dimensional membership function, which according to (21) characterizes a type-2 fuzzy membership set. According to (20), a fuzzy set A is characterized by the membership function [(20),(22),(23)]: $A = \{((x,u), \mu_A(x,u)) | \forall x \in X, \forall u \in J_X \subseteq [0,1]\}$

In which, $0 \leq \mu_A(x,u) \leq 1$, $J_X \subseteq [0,1]$ represents the primary pertinence degree of x in in set A and $\mu_A(x,u)$ is a pertinence set known as secondary set (20).

The uncertainty represented by the region is called "*footprint of uncertainty*" (FOU). When $\mu_A(x,u) = 1$, $\forall u \in J_X \subseteq [0,1]$, one has the so-called interval type-2 fuzzy, which can be seen in Fig. 2. Perceived uniform distribution of the uncertainty patch represents the interval fuzzy set that can be described in terms of an upper($\mu_A(x)$) and a lower($\underline{\mu_A(x)}$) membership function.

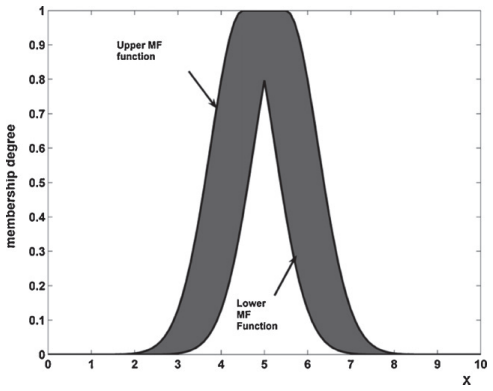


Fig. 2. Type-2 membership function similar to a type-1 membership function with uncertainty patch (20).

A type-2 fuzzy logic system is also characterized by the use of IF-SO rules, so that the antecedents or the consequents are type-2 membership functions and, as in type-1 fuzzy, there are the processes of fuzzification, rule base, inference machine, and defuzzification, as described in the subsections below.

**Fuzzification** The fuzzification process consists of mapping real inputs (*crisps*) into type-2 fuzzy sets, that is, into numerical values from the fuzzy universe.

**Rules** The rule structures in both the type-1 fuzzy and the type-2 fuzzy remain the same, so that it is represented in rules as follows for a MISO (multiple-input, single-output) system:

(1)

$$R^1 : \text{IF} x_1 \text{ is } F_i^l \text{ and } \cdots \text{ and } \cdots x_p \text{ is } F_p^l, \text{ THEN } y \text{ is } G^l$$

In which, $R^i$ represents the rule number, $p$ the amount of inputs of $x_p \in X_p$, $y$ the output of the system, and $F_i^l$ and $G^l$ represent the type-2 fuzzy membership functions being, respectively, predecessors and consequents.

**Inference** The inference process combines rules and makes a mapping from type-2 fuzzy input sets into type-2 fuzzy outputs, in type-2 fuzzy logic systems, for this it is necessary to define the union ($\sqcup$) and intersection ($\sqcap$) operations of the relations referring to the type-2 sets. Given a type-2 rule: IF $F_i^l \times \dot{\cdot}\cdot \times F_p^l = A^l$, 1, it can be rewritten as follows (20):

$$(2) \quad R^l : F_1^l \times \dot{\cdot}\cdot \times F_p^l \rightarrow G^l = A^l \rightarrow G^l \quad l = 1, ..., M$$

So that $R^l$ is described through the membership functions $\mu_{R^l}(x,y) = \mu_{R^l}(x_1, ..., x_p, y)$, in which:

$$(3) \qquad \mu_{R^l}(x,y) = \mu_{A^l \rightarrow G^l}(x,y)$$

that can be rewritten as:

$$(4) \qquad \mu_{R^l}(x,y) = [\prod_{i=1}^{p} \mu_{F_i^l}(x_i) \prod \mu G^l(y)]$$

In fuzzy logic-based systems, using interval fuzzy sets and intersection operated via t-norm product, then the result of the input and the fired rules $\prod_{i=1}^{p} \mu_{F_i}(x'_o \equiv F^l(x')$, is an interval fuzzy set of type-1, which follows Eq. 5.

$$(5) \qquad F^l(x') = [\underline{f}^l(x'), f_l(\bar{x'})] \equiv [\underline{f}^l, \bar{f}^l]$$

Wherein:

$$f^l(x') = \underline{\mu}_{F_P^l}(x'_1) * ... * \underline{\mu}_{F_P^l}(x'_p) \tag{6}$$

and

$$f^l(x') = \bar{\mu}_{F_P^l}(x'_1) * ... * \bar{\mu}_{F_P^l}(x'_p) \tag{7}$$

Where $*$ represents the t-norm operation.

**Type reductor** The type reduction in type-2 fuzzy consists of generating a type-1 output fuzzy set, which is subsequently converted into a numerical value after the defuzzification step. Using the centroid method to obtain the reduction, this method can be expressed according to Eq. 9

$$Y(x) = [y_l, y_r] = \int_{y_l^1 \in y_r^1} \cdots \int_{y_M^1 \in y_r^M} \int_{f^1 \in [\underline{f}^1, \bar{f}^1]} \cdots \times \int_{f^1 \in [\underline{f}^M, \bar{f}^M} \frac{1}{\sum_{i=1}^M f^i y^i / \sum_{i=1}^M f^i} \tag{8}$$

This interval set is determined by two endpoints, $y_l$ and $y_r$, which corresponds to the centroid of interval type-2 fuzzy sets, one can obtain:

$$y_l = \frac{\sum_{i=1}^M f_l^i y_l^i}{\sum_{i=1}^M f_l^i} \tag{9}$$

$$y_r = \frac{\sum_{i=1}^M f_r^i y_r^i}{\sum_{i=1}^M f_r^i} \tag{10}$$

The values $y_l$ and $y_r$ define an interval output of fuzzy logic type-2.

**Defuzzification** From the type reduction given by Eq.9 and Eq. 10, one can use the average between $y_l$ and $y_r$, as shown in the equation:

$$y(x) = \frac{y_l + y_r}{2} \tag{11}$$

**Bio-inspired Optimization Methods**

This section will explain some basic concepts of the optimization methods used for tuning the fuzzy controllers used in this work.

**Observations about fuzzy logic optimization** Type-1 and type-2 fuzzy controller tuning problems can be solved with any of the methods that will be mentioned in the following subsections. The big problem is the form of representation of fuzzy logic in each of these methods. In genetic algorithms, for example, the logic must be represented in the form of chromosomes, in particle swarm in the form of particles that move in a universe of speech and, in optimization by ant colony, in graphs representing paths by which ants can move (20).

**Genetic Algorithm Optimization** Genetic algorithms are global optimization algorithms. As the name implies, they are based on genetic and evolutionary mechanisms, proposed by Holland in 1975 (24) (25). These algorithms exploit prior information to find optimal values in a search space. For this purpose, new test points are generated at each iteration.

At each iteration of the algorithm, known as generation, the concepts of natural selection and reproduction are applied to a population of individuals in the universe of discourse. Through selection, a probability is generated that determines which individuals will reproduce. This probability of reproducing, determined by the fitness index. Thus, candidates with a higher fitness index are assigned higher probability values and, consequently, higher chances of reproducing and thus passing their fitness "histories" on to the next generations.

Because genetic algorithms do not use the gradient as information, this type of strategy can be effective regardless of the nature of the objective function. It combines the use of randomly generated numbers and information from previous generations to evaluate and improve a population of potential candidates rather than a single point at a time (26).

Genetic algorithms have quite significant characteristics when compared to other search methods (27):

- They are based on a set of possible solutions;
- They do not involve problem modeling (modeling is restricted to solutions);
- The algorithm does not result in a single solution, but rather a population of solutions ranked qualitatively by natural selection;
- It is a probabilistic method, not a deterministic one. In other words, the same population will hardly present the same results for the same problem.

Although different authors use a significantly different methodology for implementing genetic algorithms, the basic structure of these algorithms remains the same, as can be seen in Figure 3.

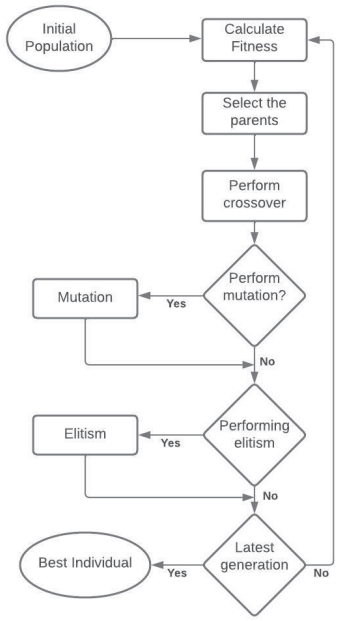


Fig. 3. Development flowchart for optimization with Genetic Algorithm (28)

In general, genetic algorithms consist of four main steps, being the selection step, crossover, mutation, and finally, the stopping criteria.

Selection methods can be used both for choosing which individuals to progeny, and for choosing the best adapted ones to pass to the next generation (29). Selection is based on a method of evaluating the fitness of individuals, that is, natural selection in which the individuals that are best adapted to the environment are chosen.

With selection, a fixed quantity of individuals are selected that will reproduce in order to generate individuals for the next generation. In this process, individuals with low suitability will have a high probability of disappearing from the population, while the most suitable will tend to ensure their survival.

Crossover is a process involving one or two individuals and emulates the exchange between pairs of chromosomes. In its simplest form, it is a random process that occurs with a user-defined probability.

In the mutation step, initially the mutation rate in the population is defined. Based on this rate, the probability that an individual will or will not mutate is given.

In most publications, mutation is completely random, both with respect to the rate of occurrence in each generation, and with respect to the individual probability. However, parameterization of this genetic operator allows greater control over the diversity and evolution of the population, which may be desirable when adapting a genetic algorithm to different problems, or when aiming to establish optimal search parameters (27).

The definition of the stopping criterion of an optimization algorithm is very important, since it would be ideal if the algorithm would terminate as soon as the maximum point is found. In some cases, the stopping criterion can be defined by the number of iterations of the algorithm or by population stagnation. As in any optimization algorithm, there is the possibility that a population may converge to a solution that is not necessarily a global maximum (27).

**Evolutionary Algorithms - Ant Colony**

Ant Colony Optimization (ACO) is a heuristic that was formulated in the 1990s by Marco Dorigo. The idea was inspired by the behavior of real ants, related to their abilities to find the shortest path between the nest and the food source. The search is carried out by exploiting the pheromone trails, a chemical substance deposited by the ants during their journey.

Due to this cooperative and effective foraging behavior, the ants will build better alternatives on the way to find food. In nature, real ants move randomly in search of food, i.e., they perform exploratory searches for possible solutions. When they find food they return to the nest and deposit pheromone. The greater amount of pheromone means that more ants have found this path, reinforcing the trail with their own pheromone, which in turn increases the probability that this is the best or shortest path (30).

Based on this behavior, Dorigo (1992) proposed a mathematical method that simulates the pheromone trail in an artificial system. The "artificial ants" that generate the pheromone are called agents.

The main features of this algorithm are (31):

- The positive feedback as a function of the pheromone trails, that is, the higher the pheromone level the better the quality of the solution;
- The virtual pheromone, whose quality is increased in good solutions and decreased in other solutions that are not effectively good, avoiding stagnation;
- The cooperative behavior of the agents once exploration is collective;
- Pheromone reinforcement in trails that have reached better performances.

The figure 4 presents a flowchart of the algorithm proposed in this work.

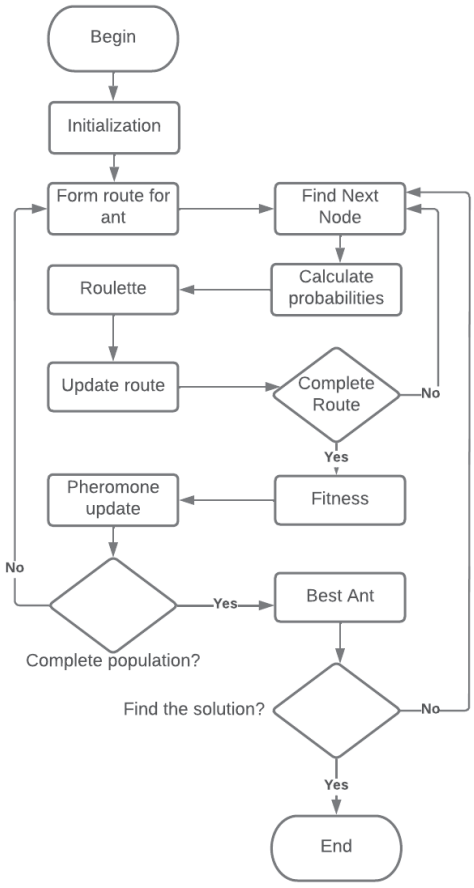


Fig. 4. Flowchart of the generic ACO algorithm (32)

The ants start from the node representing the nest, the search for the next node for the ant takes into account the pheromone level and the heuristic value present on each edge.

In general, for the first ant population, the pheromone levels are equal, i.e., the ant's first choices are based only on the heuristic factor.

The transition probability from node $i$ to node $j$ is defined as the probability that an ant will choose the $ij$ path (33). This can be calculated from the equation 12.

(12)

$$p(t)_{i,j} = \begin{cases} \frac{(\tau(t)_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}{\sum_{l} (\tau(t)_{il})^{\alpha} \cdot (\eta_{il})^{\beta}}, \text{if the } j \text{ path is allowed} \\ 0, \text{ if contrary} \end{cases}$$

Where:

$\eta_{ij}$ : It represents the heuristic component of each edge;

$\tau(t)_{ij}$ : Corresponds to the pheromone deposited at time $t$ on edge $ij$;

$l$ : Total population of individuals;

$\alpha$ and $\beta$ :They balance the contributions of the pheromone and the heuristic factor, respectively.

After the probability is selected, a node is selected by roulette drawing, i.e., even nodes with low probability can still be selected for the ant's route.

When the ant's route is complete, the route cost is calculated based on a previously established function, or (performance), of the controller found, this cost serves as a basis for calculating the pheromone that is deposited in each route. With the value of the route cost, the pheromone level on each edge is updated following the equation 13.

$$\tau_{ij}(t) = \tau_{ij}(t) + \Delta\tau ij(t) \tag{13}$$

Thus, the amount of pheromone deposited by the ant depends on the quality of the route (Q) and the cost of each route (C(n)), as shown by the equation 14 (32).

$$\Delta\tau_{ij}(t) = \frac{Q}{C(n)} \tag{14}$$

In order to avoid excessive accumulation of pheromone in a particular route, which would result in a stagnation of the ants for only one route, we opt to use a rate of dissipation of the amount of pheromone, in which, at each generation of ants, a percentage of the pheromone deposited in all is dissipated.

**Particle Swarm Algorithm**

Particle Swarm Optimization (PSO) is a technique initially inspired by bird displacement, and is another form of evolutionary computation that is stochastic in nature, like the genetic algorithm (GA). However, unlike GA, PSO does not have evolutionary operators such as crossover and mutation. In PSO, particles (which correspond to potential solutions) move through the problem space, following the current best particles.

In other words, each particle has a position and velocity of displacement in the universe of discourse, the position defines performance, or how well the particle fits the solution. Each particle also keeps stored the position of the best solution it has found, this value is called *personal best* or *pbest*. When all particles are evaluated, it is possible to select among them which presented the best solution to the problem presented, the best value is stored, called *global best* or *gbest*.

These values (gbest and pbest) determine the way each particle moves, changing the direction and speed with which they move through the solution space. The basic equations for updating the velocity and position of each particle are given in the equations 15 and 16, respectively (34).

$$v_i(k+1) = v_i(k) + \gamma_{1i}(p_i - x_i(k)) + \gamma_{2i}(G - x_i(k)) \tag{15}$$

$$x_i(k+1) = x_i(k) + v_i(k+1) \tag{16}$$

Being:
$i$ - Index of the particle;
$k$ - Index of the particle;;
$v_i$ - Velocity of the ith particle;
$x_i$ - Position of the ith particle;
$p_i$ - Best position found by the ith part (*pbest*);
$G$ - Best position found by all particles;
$\gamma_{1,2}$ Random values between [0,1] applied on the ith particle.

The concept of particle swarm optimization consists, in each time step, of changing the velocity of each particle towards its *pbest* and the global solution, *gbest*.

Figure 5 shows a flowchart representing the basic algorithm for this optimization.

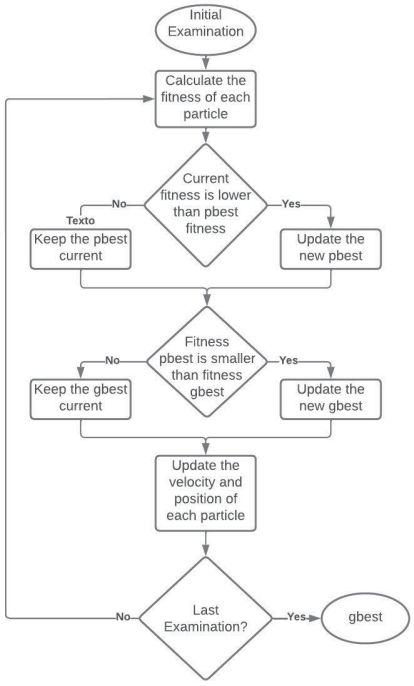


Fig. 5. Flowchart of PSO algorithm (35)

In recent years, the PSO algorithm has been used successfully in several research and application areas. It has been shown that this algorithm achieves better results faster and cheaper compared to genetic algorithm (34).

There is a wide range of variations of the PSO algorithm, the most widely used includes inertia in the velocity equation which represents the "difficulty" the particle encounters in changing its position. In most works, the inertia of the particle is given as a constant 1.4 or a factor decreasing in time as the algorithm runs from 0.9 to 0.4. Two other parameters $r_1$ and $r_2$ are also added to the velocity equation and represent the "reliability" of the particle in its own solution and in the global solution, respectively (35). Thus, the velocity of the particle is then calculated according to the equation 17.

$$v_i(k+1) = \omega_i v_i(k) + r_{1i}[\gamma_{1i}(p_i - x_i(k))] + r_{2i}[\gamma_{2i}(G - x_i(k))] \tag{17}$$

**Controllers Obtained**

The next subsections will describe the controllers that were obtained through the optimizations, in this section is the description of the type-1 and type-2 fuzzy controllers, as well as their membership functions and, finally, the rule matrix of each controller.

**GA-based controller** For the Genetic Algorithm a maximum number of 100 generations was used as a stopping criterion, once it was observed that this number was enough to obtain answers that provide satisfactory performance of the control system in a population with 20 individuals. The parameters used in the GA for the three controllers are shown

in Table 1.

Table 1. Genetic algorithm optimization parameters

| | |
|---|---|
| Population | 20 Individuals |
| Binary representation | 32 Bits |
| Crossover Rate | 80 % |
| Selection | Roulette |
| Mutation Rate | 1 % |
| Elitism | 20 % |
| Stop Criteria | Population Stagnation |
| | 100 Generations |

For the first input, the error, the optimized membership functions using genetic algorithm are shown in Figure 6.

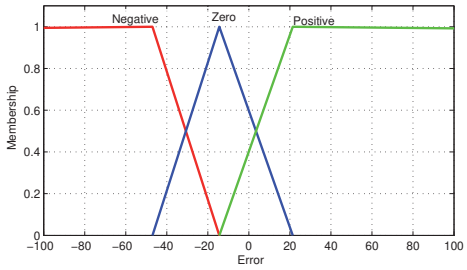


Fig. 6. Input membership functions: Error optimized with GA

For the second input, (error derivative), the membership functions used are shown in Figure 7. Note that the membership function for the zero derivative function tends towards the positive side, while for the negative derivative it is not restricted only to the negative axis and partially occupies the positive axis. In the case of the positive derivative, the membership values are restricted to positive values.

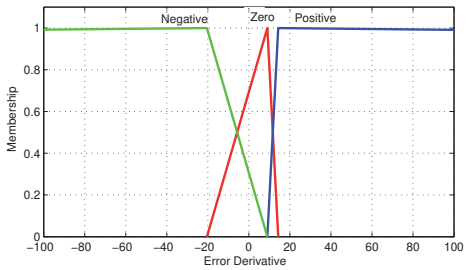


Fig. 7. Input membership functions: Error Derivative optimized with GA

The output Sugeno functions are shown in Table 2. One notices deviating values of the multiplier of the first input to that of the second input, one also notices that the values of the constants of the output Sugeno functions were close to zero.

Table 2. Output Sugeno's functions for the Conventional Fuzzy

| Name | Error multiplier | Error derivative multiplier | Constant |
|---|---|---|---|
| S1 | 48,57 | 2,38 | 0,0 |
| S2 | 35,04 | 10,76 | 0,01 |
| S3 | 5,12 | 8,22 | 0,02 |

The rules of the optimized fuzzy controller are represented in Table 3. In this figure, the rules are presented in the form of a *Fuzzy Associative Matrix* (FAM). The two inputs are related by the connective AND inferred by the minimum operator (MIN).

Table 3. Conventional Fuzzy Rules for the Genetic Algorithm

| $\frac{de(t)}{dt}/e(t)$ | Negative | Zero | Positive |
|---|---|---|---|
| Negative | S2 | S1 | S3 |
| Zero | S2 | S1 | S1 |
| Positive | S3 | S1 | S3 |

Figure 8 shows the membership functions optimized by the Genetic Algorithm. Note the non-symmetry of the membership functions around the zero value on the *X* axis. Clearly, the positive error function also contains part of the negative error axis.

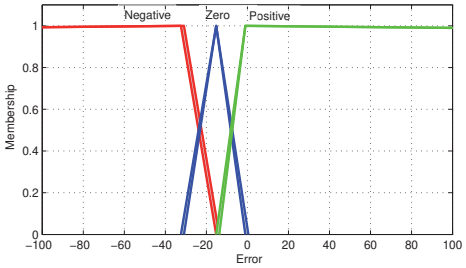


Fig. 8. Pertinence functions for Error optimized with GA

In Figure 9 are the membership functions for the second input of the type-2 fuzzy. According to the figure, the membership functions are not symmetric, that is, the membership function that represents the positive error derivative contains a part of the negative axis. In this case, the zero membership function is not in the center of the axes, but is positioned around -20.

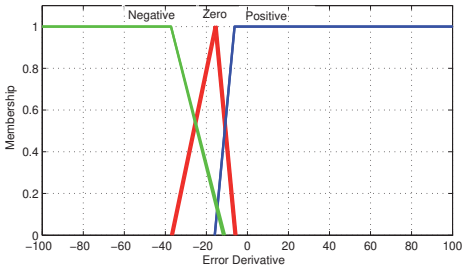


Fig. 9. Membership functions for the Error Derivative optimized with GA

The output Sugeno functions can be seen in Table 4. The first value of the Sugeno function multiplies the first input, the second value multiplies the second input, and the constant adds a *offset* in the output calculation.

Table 4. Sugeno Output Functions for Fuzzy Type-2

| Name | Error multiplier | Error derivative multiplier | Constant |
|---|---|---|---|
| S1 | 7,37 | 3,13 | 0,01 |
| S2 | 1,62 | 4,65 | 0,1 |
| S3 | 8,71 | 2,18 | 0,02 |

The rules for the optimized fuzzy controller are presented in Table 5. The rules are presented in the form of FAM. The two inputs are related by the connective AND inferred by the minimum operator (MIN).

Table 5. Rules for the Type-2 fuzzy controller

| $\frac{De(t)}{dt}/e(t)$ | Negative | Zero | Positive |
|---|---|---|---|
| Negative | S2 | S1 | S3 |
| Zero | S2 | S1 | S1 |
| Positive | S2 | S3 | S3 |

**PI Controller**

The parameters for the ant colony based optimization (ACO) are presented in Table 12.

The optimized PI controller found close values for $K_p$ and $K_i$. The values of $K_p$ and $K_i$ can be found in table 6.

Table 6. Optimized PI Values with GA

| $K_p$ | $K_i$ |
|---|---|
| 2,35 | 3,77 |

For the first input, Error, the optimized membership functions using PSO are shown in Figure 10.

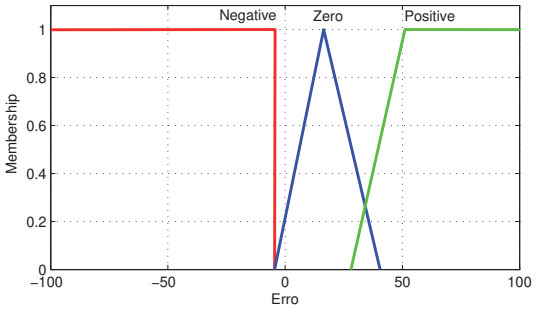


Fig. 10. Input membership functions: Error optimized with PSO

For the second input, Error Derivative, the optimized membership functions are shown in Figure 11.

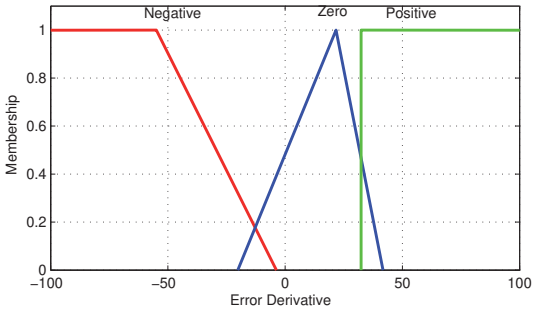


Fig. 11. Membership functions for input: Error Derivative optimized with PSO

The Output Sugeno functions can be found in Table 7.

Table 7. Sugeno output functions for conventional fuzzy with PSO

| Name | Error multiplier | Error derivative multiplier | Constant |
|---|---|---|---|
| S1 | 14,40 | 4,19 | 0,08 |
| S2 | 81,68 | 51,21 | 0,01 |
| S3 | 8,53 | 0,73 | 0,02 |

The rules of the optimized fuzzy controller are presented in Table 8.

Table 8. Rules for conventional fuzzy controller with PSO

| Error/Derivative | Negative | Zero | Positive |
|---|---|---|---|
| Negative | S2 | S1 | S3 |
| Zero | S2 | S1 | S1 |
| Positive | S3 | S1 | S3 |

For the first input, Error, the optimized membership functions using PSO are shown in Figure 12.

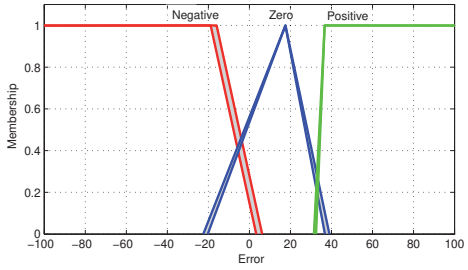


Fig. 12. Input membership functions: Error optimized with PSO

For the second input, Error Derivative, the optimized membership functions are shown in Figure 13.

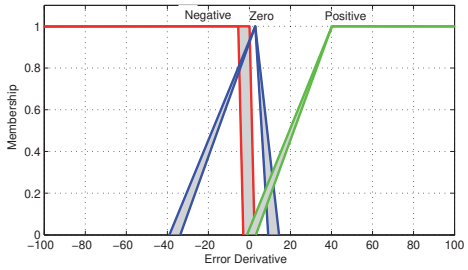


Fig. 13. Membership functions for input: Error Derivative optimized with PSO

Table 9 contains the Sugeno output functions for the *fuzzy* type-2 controller. Note-2 that the values for the constants are close to zero. This is because the *offset* sometimes includes a permanent regime error.

Table 9. Sugeno output functions for conventional fuzzy with PSO

| Name | Error multiplier | Error derivative multiplier |
|---|---|---|
| S1 | 14,40 | 4,19 |
| S2 | 81,68 | 51,21 |
| S3 | 8,53 | 0,73 |

The rules for the optimized fuzzy controller are presented in Table 10. The rules are presented in the form of a *fuzzy associative matrix*. The two inputs are related by the AND connective inferred by the minimum (MIN) operator.

Table 10. Rule for Type-2 Fuzzy Controller

| Error derivative | Negative | Zero | Positive |
|---|---|---|---|
| Negative | S2 | S1 | S3 |
| Zero | S2 | S1 | S1 |
| Positive | S3 | S1 | S3 |

The optimized PID controller found close values for $K_p$ and $K_i$. The $K_p$ and $K_i$ values are shown in Table 11.

Table 11. PSO optimized PI values

| $K_p$ | $K_i$ |
|---|---|
| 1,47 | 2,29 |

**Ant Colony** To perform the optimization of fuzzy controllers, it is necessary to represent the parameters in the form of a graph, in which each ant will travel a path and update the pheromone by the route in which it travels, at the end of the cycle, when arriving at the food, which for this case represents the last tuning value of the fuzzy controller, the pheromone deposited by all map. Figure 14 represents the graph generated by the parameters of the fuzzy controller.

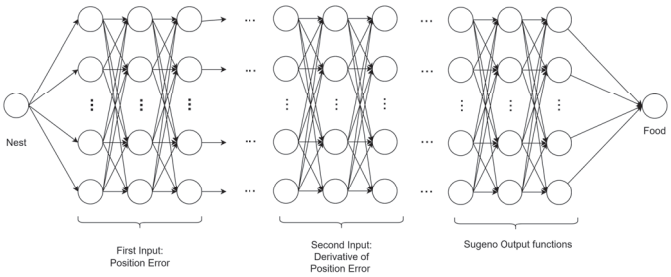


Fig. 14. Representation of the Map with Fuzzy Parameters

In this work, each route completed by the ant represents a controller. Thus, each parameter of the controllers is associated with a node chosen by the ant in its route. All links must be tested in order to respect the format of the chosen membership function. On Tab. 12 is the parameters for ACO optimization.

Table 12. Parameters for Ant Colony Optimization

| Population | 20 Ants |
|---|---|
| $\alpha$ - Contribution of the pheromone | 0,7 |
| $\beta$ - Heuristic Contribution | 0,3 |
| Ridge Selection | Roulette |
| Stop Criteria | Population Stagnation |
| | 50 Generations |

For the first input we obtained the membership functions shown in Figure 15.

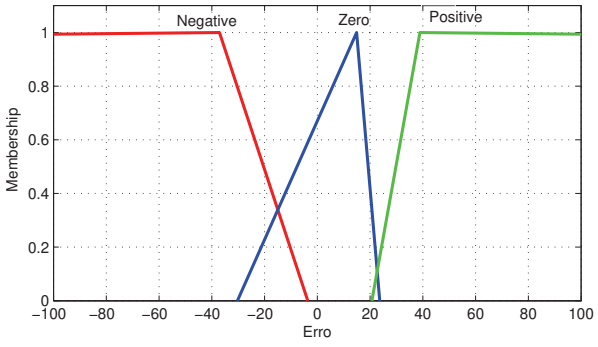


Fig. 15. Pertinence functions for input Error optimized with ACO

For the input derived from the error, the configuration of membership functions shown in Figure 16.

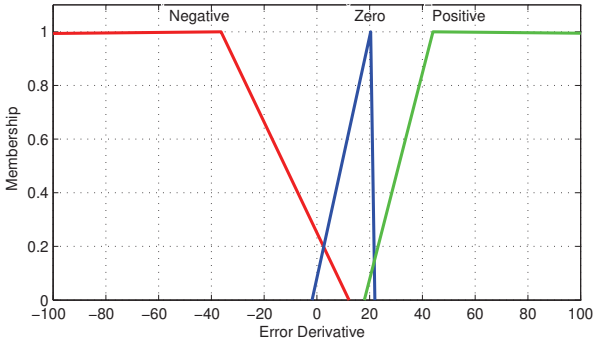


Fig. 16. Input membership functions: Error Derivative optimized with ACO

The output Sugeno functions obtained are shown in Table 13, for which we note very close values for the multipliers of the first and second inputs and values close to zero for the output constant.

Table 13. Sugden output functions for conventional fuzzy for ACO

| Name | Error multiplier | Error derivative multiplier | Constant |
|---|---|---|---|
| S1 | 4,10 | 5,10 | 0,01 |
| S2 | 7,20 | 5,00 | 0,1 |
| S3 | 3,30 | 4,00 | 0,02 |

The rules of the optimized fuzzy controller are represented in Table 14. The rules are presented in the form of a FAM. The two inputs are related by the connective AND inferred by the minimum operator (MIN).

Table 14. Rules for conventional fuzzy controller with GCA

| Error/derivative | Negative | Zero | Positive |
|---|---|---|---|
| Negative | S2 | S1 | S3 |
| Zero | S2 | S1 | S1 |
| Positive | S3 | S1 | S3 |

Figure 17 shows the graphical representation of the membership functions for the first input.

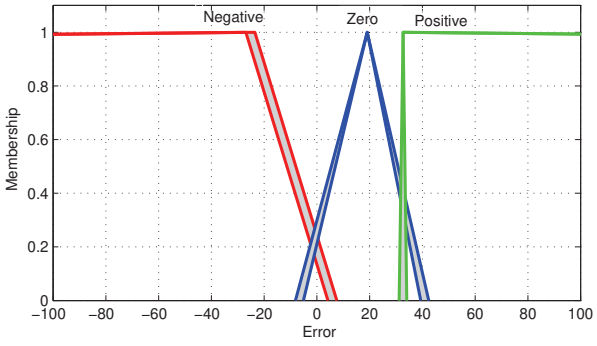


Fig. 17. Type-2 Input membership functions: Error optimized with ACO

Figure 18 represents the membership functions for second input. It can be seen that the membership functions have small patches of uncertainty around the pertinence functions.

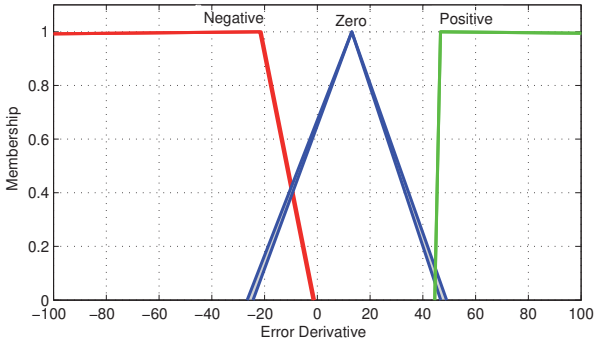


Fig. 18. Input membership functions: Error Derivative optimized with ACO

Table 15 shows the output Sugeno functions for the optimized controller. Note that, like other controllers, the value of the Sugeno function constant approaches zero, as well as the values that multiply the input are larger than those that multiply the second input (for functions S2 and S3).

Table 15. Sugeno output functions for fuzzy Type-2 for ACO

| Name | Error multiplier | Error derivative multiplier | Constant |
|---|---|---|---|
| S1 | 1,4 | 3,9 | 0,2 |
| S2 | 10,9 | 3,7 | 0,1 |
| S3 | 18,9 | 0,6 | 0,4 |

The rules for the type-2 fuzzy controller are shown in Table 16.

Table 16. Rules for the Fuzzy Type-2 controller for ACO

| Error/Derivative | Negative | Zero | Positive |
|---|---|---|---|
| Negative | S2 | S1 | S3 |
| Zero | S2 | S1 | S1 |
| Positive | S2 | S3 | S3 |

The optimized PI controller found close values for $K_p$ and $K_i$. The values of $K_p$ and $K_i$ are shown in Table 17.

Table 17. ACO-optimized PI values

| $K_p$ | $K_i$ |
|---|---|
| 5,7 | 4,1 |

**Results and discussion**

**Genetic Algorithms** With the genetic algorithm obtained the controllers shown in the following subsections. The processing time for the conventional fuzzy controller was 157.31 seconds, while for the fuzzy type-2 was 157.84 seconds, indicating that for this case, the processing between the two logics is similar, ie, the addition of the FOU did not present great computational expense. For the PI controller, the processing time was 23.81 seconds.

Figure 19 shows the response of the system. The system with this controller presents small *overshoot* and a low stabilization time. The system response exhibits an oscillation relative to the noise present in the tests performed.

Figure 20 presents the control signal of the system. The control signal presents large peaks when there is a change of *setpoint*, presenting only a few oscillations due to the presence of noise in the tests performed.

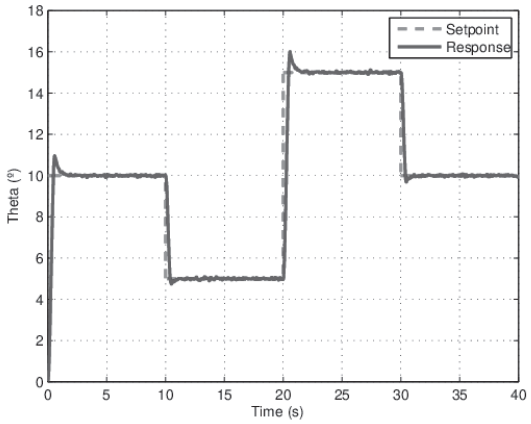


Fig. 19. Control signal for conventional fuzzy controller with GA

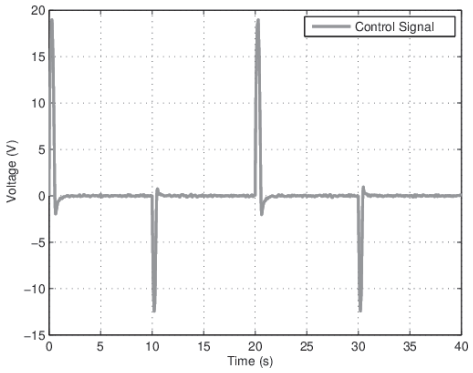


Fig. 20. Response for conventional fuzzy controller with GA

Figure 21 presents the system response with the fuzzy type-2 controller optimized by the Genetic Algorithm. In the figure, it can be seen that the system rise time is low, which leads to a more aggressive controller and thus a higher *overshoot*, however, the system response presents a low stabilization time using the 2 % criterion.

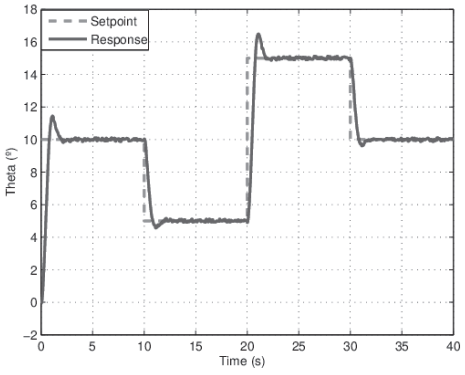


Fig. 21. Response for FT2 controller with GA

The control signal, presented in Figure 22, shows voltage peaks when there is a change in the *setpoint*, but these peaks do not reach the controller saturation. It is also observed small oscillations due to the presence of noise in the plant.

Figure 23 shows the response of the system with the optimized PI controller. Note that the system has a large *overshoot* for this controller.

The control signal is shown in Figure 24. As in the pre-

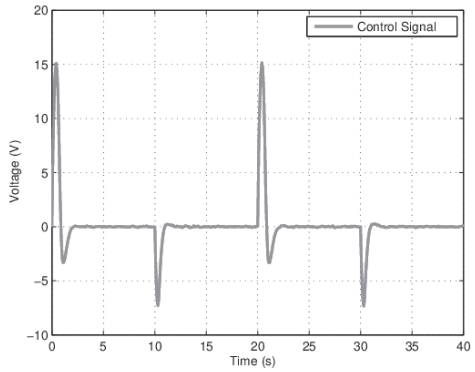


Fig. 22. Control signal to FT2 controller with GA

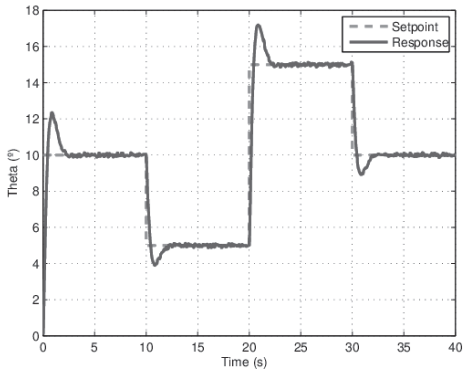


Fig. 23. Response for PI controller with GA

vious case, the control signal presents voltage peaks when there is a change of *setpoint*. The oscillations present in the control signal are relative to the noise present in the system. The control signal does not reach the saturations of the system.

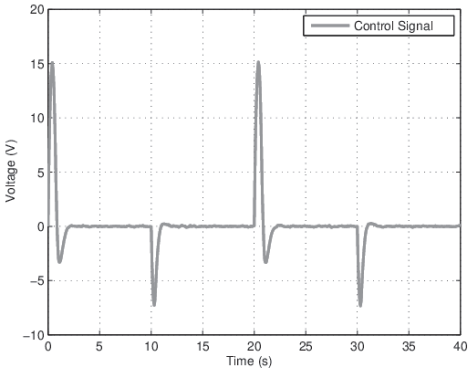


Fig. 24. Control signal for PI controller with GA

From the values presented in Table 18, it can be seen that the PI controller presented worse results than the other controllers. Due to its slower behavior than the fuzzy controllers, higher *overshoot* and longer stabilization time for permanent regime.

The conventional fuzzy controller achieved lower rates of IEA and ITEA than the type-2 fuzzy controller. This is due to the more aggressive behavior of the fuzzy controller type-2 controller. Thus, the system response for the conventional controller presents less rise time and *overshoot* than the other controllers.

However, all the aggressiveness of the conventional

Table 18. Index for controller evaluation.

| Controller/Index | IEA | ITEA | IG |
|---|---|---|---|
| Conventional Fuzzy | $3,82 \times 10^{-1}$ | $1,82 \times 10^{-1}$ | $2,99 \times 10^{0}$ |
| Fuzzy T2 | $4,72 \times 10^{-1}$ | $2,75 \times 10^{-1}$ | $1,93 \times 10^{0}$ |
| PI | $5,26 \times 10^{-1}$ | $4,24 \times 10^{-1}$ | $7,80 \times 10^{0}$ |

fuzzy controller causes the index of Goodhart is higher than the fuzzy controller type-2, since this index weights more negatively the oscillation and peaks of the controllers.

In changes of *setpoint*, the fuzzy controller type-2 has smaller voltage peaks than the conventional fuzzy controller, but the conventional controller has smaller *overshoots* than the type-2.

**Particle Swarm** The particle swarm optimization generated the controllers shown in the following subsections. The execution time of the optimization algorithm for the conventional fuzzy controller was 157.31 seconds, for fuzzy type-2 157.84, which indicates that for this case, the processing between the two similar logics, i.e., the addition of the FOU did not present great computational expense. For the PI controller the processing time for the optimization was 45.89 seconds.

Figure 25 represents the response for the optimized conventional fuzzy controller with PSO controller. The system response exhibits small *overshoots* for all applied steps and a small steady-state error correction time. The oscillation present in the system response is relative to noise.

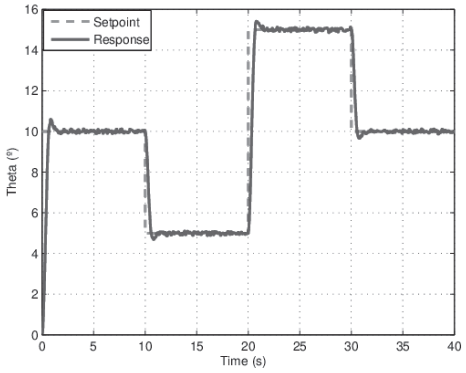


Fig. 25. Response for conventional fuzzy controller with PSO

The control signal present in Figure 26 presents a behavior with high peaks when changing *setpoints*. The stabilization of the control signal occurs in a short time.

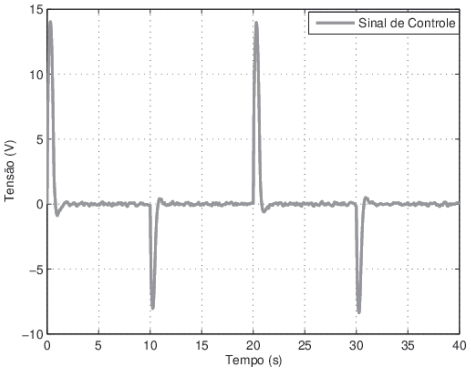


Fig. 26. Control signal for conventional fuzzy controller with PSO

Figure 27 represents the response for the optimized

type-2 fuzzy controller with PSO controller. It can be seen that the system response exhibits small *overshoots* for all steps applied to the reference and a small steady-state error correction time. The oscillation present in the system response is related to noise.

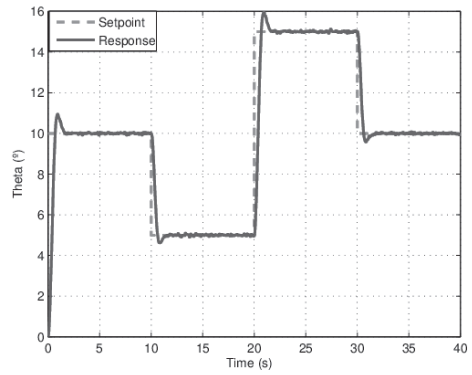


Fig. 27. Response for fuzzy controller type-2 with PSO

The control signal present in Figure 28 presents a behavior with high peaks when changing *setpoints*, but the stabilization of the control signal happens in a short time.

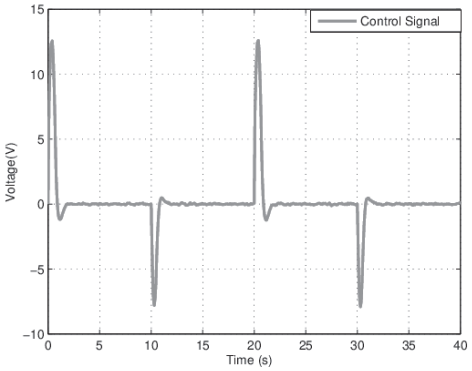


Fig. 28. Control signal for fuzzy controller type-2 with PSO

The PI controller generates the response shown in Figure 29 for this system. The system response exhibits *overshoot* and a high permanent regime error correction time, i.e., even with the presence of the overshoot, the system remains slow.

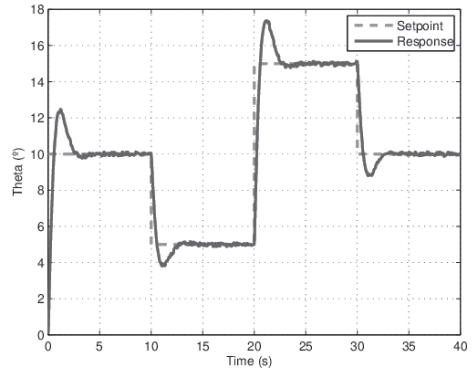


Fig. 29. Response for the optimized PI controller with PSO

The control signal, presented in Figure 30, shows voltage peaks when there is a change in the *setpoint*, but these peaks do not reach the controller saturation. It is also observed small oscillations due to the presence of noise in the plant.

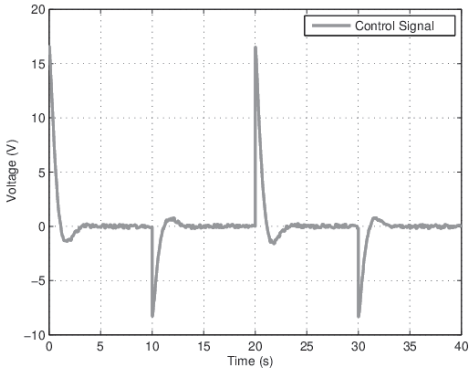


Fig. 30. Control signal for the optimized PI controller with PSO

Based on the behavior presented by the system the performance indexes of the controllers obtained through the PSO algorithm can be calculated. The indexes can be found in Table 19.

Table 19. Key figures for controller evaluation.

| Controller / Index | IEA | ITEA |
|---|---|---|
| Conventional Fuzzy | $4,50 \times 10^{-1}$ | $3,04 \times 10^{-1}$ |
| Fuzzy T2 | $4,34 \times 10^{-1}$ | $2,20 \times 10^{-1}$ |
| PI | $6,06 \times 10^{-1}$ | $5,60 \times 10^{-1}$ |

The conventional fuzzy controllers and fuzzy type-2 optimized by the PSO algorithm obtained better results compared to the PI controller.

Among the fuzzy controllers, by presenting lower stabilization and rise time, the type-2 fuzzy controller obtained better values of performance index.

In changes of *setpoint*, the PI controller presents a large *overshoot*, reaching 20% of the value of the applied pulse. The control signal also reaches large peaks when there is a change of reference.

Table 20. *Goodhart* indices for the controllers

| Controller/Index | IG |
|---|---|
| Conventional Fuzzy | $2,36 \times 10^{0}$ |
| Fuzzy T2 | $1,80 \times 10^{0}$ |
| PI | $5,15 \times 10^{0}$ |

For the index of Goodhart, seen from Table 20, it is observed that the type-2 fuzzy controller has a better result than the conventional fuzzy controller, because of the higher weighting of the control signal. The conventional fuzzy controller has larger peaks than the fuzzy controller type-2. The PI controller has the highest Goodhart index among the controllers, due to the presence of large voltage peaks when there is change of *setpoint*.

The optimization of the type-2 fuzzy controller provided a gain of about 32.79% in controller quality when examined by the evaluation function, corresponding to a significant gain for the plant.

The addition of the FOU presents a significant gain in the performance of the optimized controller.

**Ant Colony**

With the Ant Colony algorithm obtained the controllers shown in the following subsections. The processing time for the conventional fuzzy controller was 160.63 seconds, for the fuzzy type-2 was 217.63 seconds and 17.92 seconds for the PI controller. The addition of the FOU causes an increase in the complexity of the algorithm, due to the increase in the number of possible routes for the ants to travel.

Figure 31 represents the behavior of the plant with the conventional fuzzy controller optimized with ACO. In the figure, it is noted that the system response presented small *overshoots* for steps 10 and 15 in the reference, but in the down steps, the system response became slow and without *overshoot*.

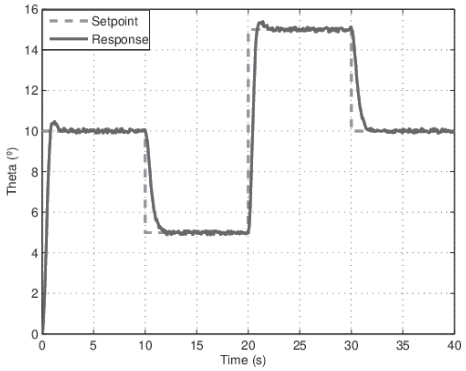


Fig. 31. Response for conventional fuzzy controller with ACO

The figure 32 presents the control signal generated by the controller. It can be seen that the control signal presents large voltage peaks when there is a change of *setpoints* and small oscillations due to the presence of noise.

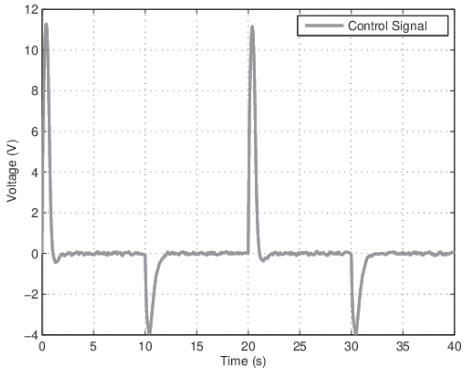


Fig. 32. Control signal for Conventional Fuzzy controller with ACO

Figure 33 presents the system behavior for the type-2 fuzzy controller optimized with ACO. Note that for this controller, the system response presented small rise times, but large *overshoots*, for the steps of 10 and 15, and small *overshoots* for the descent steps. Note that the signal became oscillatory due to the presence of noise, but the controller was able to reject this noise.

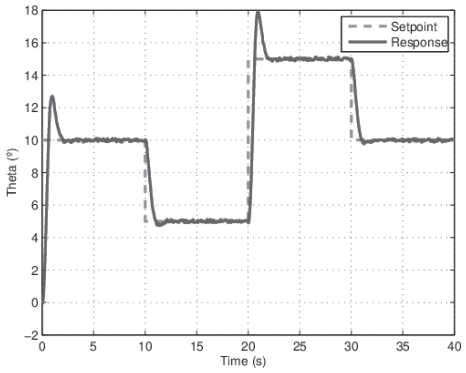


Fig. 33. Response for FT2 controller with ACO

The control signal that generates the system response is shown in Figure 34. It can be seen that for rising steps, the control signal presents voltage peaks that do not reach controller saturation. The control signal also presents the oscillations related to noise.

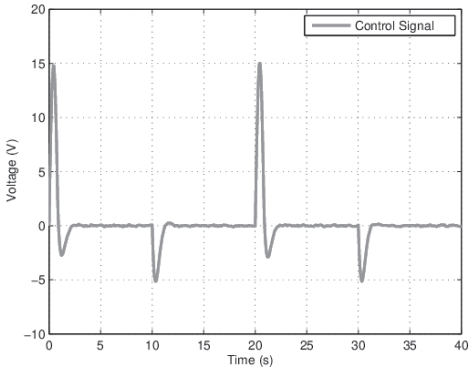


Fig. 34. Control signal to FT2 controller with ACO

The Figure 35 presents the system response for the implemented PI controller. Note a short rise time, but large oscillations for the stabilization of the steady state error, this causes a high stabilization time for all steps in the reference.

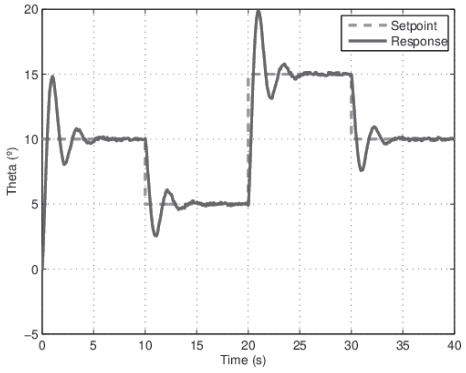


Fig. 35. Response for the optimized PI controller with ACO

Figure 36 shows the curve of the system control signal. When there are changes of *setpoint*, the control signal presents voltage peaks and some oscillation until it manages to zero the regime error. The influence of this oscillation, made the system slower.

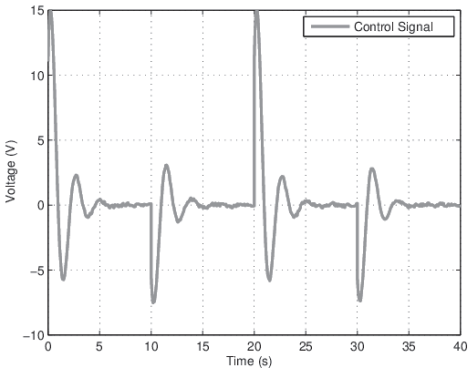


Fig. 36. Control signal for the optimized PI controller with ACO

**Analysis of Results** In possession of the results, it was possible to analyze the behavior of each system response and control signal according to the evaluation indexes. In Table 21 are the indices studied in this work.

Table 21. Evaluation indexes of controllers optimized with Ant Colony.

| Controller / Index | IEA | ITEA | IG |
|---|---|---|---|
| Conventional Fuzzy | $5,89 \times 10^{-1}$ | $3,77 \times 10^{-1}$ | $1,74 \times 10^{0}$ |
| Fuzzy Tipo-2 | $5,70 \times 10^{-1}$ | $2,43 \times 10^{-1}$ | $1,42 \times 10^{0}$ |
| PI | $8,58 \times 10^{-1}$ | $9,61 \times 10^{-1}$ | $5,36 \times 10^{0}$ |

The fuzzy controller type-2 presented lower index values for the presented problem, especially, regarding the ITEA, due to the presence of FOU and noise. The FOU has as its main objective to better handle the uncertainties present in the system (in this case represented by noise). Thus, the system behavior has lower steady state error than the other controllers.

The PI controller showed higher indexes than the conventional fuzzy and type-2 fuzzy controllers, due to its oscillatory behavior when there is a change of *setpoint* and its delay to stabilize the system, even with a high rise time. The control signal of this controller is also quite oscillatory.

In changes of *setpoint*, the type-2 fuzzy controller has smaller voltage peaks in the control signal than conventional fuzzy. Regarding the response, the changes of *setpoint* cause greater *overshoot* for the fuzzy controller type-2 than for the conventional controller.

For this relationship of *overshoot* and reference tracking speed, for fuzzy controllers, the values of the IEA index are very close .

In changes of *setpoint*, the fuzzy controller type-2 presents smaller voltage peaks than the conventional fuzzy controller, but the conventional controller presents smaller *overshoots* than the type-2. Therefore, the values of AEI, which weights the error equally by time, are very close for these two controllers with the type-2 fuzzy controller behaving better for the system.

In the Goodhart index, it is observed that the type-2 fuzzy controller has a better result than the conventional fuzzy controller, this occurs due to higher weighting of the control signal. The conventional fuzzy controller showed higher voltage peaks than the fuzzy controller type-2.

**Conclusion**

This work showed the potential of fuzzy controllers, both conventional and fuzzy type-2 in controlling a nonlinear system more efficiently than the classic PI controller. It is important to note that for all tests performed, with different optimizations, the PI controller showed worse results compared to fuzzy controllers.

The presence of the uncertainty spot (FOU) presented a gain of the type-2 fuzzy controller over the other controllers. Given that the goal of the FOU is to assist in the treatment of inaccuracies present in the system, in this study represented by the inaccuracy of the sensor and the noise, even a small FOU, proved efficient in improving the quality of the system response.

The type-2 fuzzy controller showed better results than the conventional fuzzy controller. Despite being computationally more costly, the tests of practical implementations showed that there is little difference between the optimization of conventional and type-2 fuzzy controllers. With good programming practices, it is possible to decrease the processing time of both controllers.

By analyzing the performance indices, the results presented showed that for the particle swarm and ant colony optimization, the type-2 fuzzy controller showed better performance than the other controllers, with lower values of ITEA, IEA and IG.

All the responses obtained for the system studied were fast, with the smallest possible regime errors, even in the presence of noise, thus obtaining small evaluation index values.

Among all optimization methods, the PSO presented the best results for the type-2 fuzzy controller, which in turn obtained better *fitness* than the other controllers with the other optimization methods. For the case of conventional and PI fuzzy controllers, the Genetic Algorithm proved to be more effective for the control problem studied.

The limitation and need to represent fuzzy logic as a graph, made the optimization by Ant Colony to find the best result for complex controllers, as conventional fuzzy and fuzzy type-2, was not as efficient as the optimization methods for the proposed evaluation function.

***Authors**: Msc. Mário Sérgio F. F. Cavalcante, Department of Computer Engineering and Automation, Federal University of Rio Grande do Norte - Technology Center - Natal, Rio Grande do Norte, Brazil, email: mariocavalcante@dca.ufrn.br, Livia M. C. M. Soares, Dr. Ícaro B. Q. Araújo, Federal Univesity of Alagoas - Maceió, Alagoas, Brazil email: lmcms@ic.ufal.br email: icaro@ic.ufal.br Dr. Fábio M. U. Araújo, Federal University of Rio Grande do Norte - Technology Center - Natal, Rio Grande do Norte, Brazil, email: meneghet@dca.ufrn.br*

REFERENCES

[1] S. SANDRI e C. Correa,*Lógica nebulosa*, Instituto Tecnológico da Aeronáutica–ITA, V Escola de Redes Neurais, pp. C073-c090, São José dos Campos, 1999.

[2] A. Zadeh,*Fuzzy sets*, Information and control, Elsevier, v. 8, n. 3, p. 338–353, 1965.

[3] E. Mamdani, *Application of fuzzy algorithms for control of simple dynamic plant*, IET. Proceedings of the Institution of Electrical Engineers. [S.l.], 1974. v. 121, n. 12, p. 1585–1588.

[4] R. Liao,C. Chan,J. Hromek,G. Huang e L. He,*Fuzzy logic control for a petroleum separation process*, Engineering Applications of Artificial Intelligence, Elsevier, v. 21, n. 6, p. 835–845, 2008.

[5] E. BONABEAU;M. DORIGO;G. Theraulaz,*Swarm intelligence: from natural to artificial systems* [S.l.]: Oxford university press, 1999.

[6] ABDALLAH, Abdelkader *An adaptive RST fuzzy logic and an adaptive PI fuzzy logic controllers of a DFIG in Wind Energy Conversion.* PRZEGLĄD ELEKTROTECHNICZNY, 2021. DOI: 10.15199/48.2021.11.02.

[7] JUSOH, Mohd. Fuzzy logic-based control strategy for hourly power dispatch of grid-connected photovoltaic with hybrid energy storage. PRZEGLĄD ELEKTROTECHNICZNY, 2022. DOI: 10.15199/48.2022.01.02.
[8] KHELIFA, Siham. Power quality enhancement in wind turbine based DFIG using fuzzy control and neural-RIP for harmonic mitigation. PRZEGLĄD ELEKTROTECHNICZNY, 2022. DOI: 10.15199/48.2022.05.26.
[9] ZAGHRAT, Fatiha. Robust fuzzy sliding mode control implementation for DC motor. PRZEGLĄD ELEKTROTECHNICZNY, 2022. DOI: 10.15199/46.2022.05.18.
[10] ZAINUDIN, M. N. Shah. A Framework for Chili Fruits Maturity Estimation using Deep Convolutional Neural Network. PRZEGLĄD ELEKTROTECHNICZNY, 2021. DOI: 10.15199/48.2021.12.13.
[11] ALKHAYYAT, Mahmood. Adaptive Neuro-Fuzzy Controller Based STATCOM for Reactive Power Compensator in Distribution Grid. PRZEGLĄD ELEKTROTECHNICZNY, 2022. DOI: 10.15199/48.2022.04.23.
[12] BERENJI, H. R.; SARAF, S.; CHANG, P.-W.; SWANSON, S. R. *Pitch control of the space shuttle training aircraft*. IEEE Transactions on Control Systems Technology, IEEE, v. 9, n. 3, p. 542–551, 2001
[13] O. Erdinc, B. Vural, M. Uzonoglu,*A wavelet-fuzzy logic based energy management strategy for a fuel cell/battery/ultra-capacitor hybrid vehicular power system*, Journal of Power sources, Elsevier, v. 194, n. 1, p. 369–380, 2009.
[14] CAPONETTO, R.; DIAMANTE, O.; FARGIONE, G.; RISITANO, A.; TRINGALI, D. A soft computing approach to fuzzy sky-hook control of semiactive suspension. *Control Systems Technology, IEEE Transactions on*, IEEE, v. 11, n. 6, p. 786–798, 2003.
[15] SCHOUTEN, N. J.; SALMAN, M. A.; KHEIR, N. A. Fuzzy logic control for parallel hybrid vehicles. *Control Systems Technology, IEEE Transactions on*, IEEE, v. 10, n. 3, p. 460–468, 2002.
[16] A. Mirzaei, M. Moallen, B. Mirzaeian e B. Fahimi, *Design of an optimal fuzzy controller for antilock braking systems*, In: IEEE. Vehicle Power and Propulsion, 2005 IEEE Conference. [S.l.], 2005. p. 823–828.
[17] GARCIA-TRIVINO, P.; FERNANDEZ-RAMIREZ, L. M.; TORREGLOSA, J. P.; JURADO, F. Fuzzy logic control for an electric vehicles fast charging station. In: IEEE. 2016 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM). [S.l.], 2016. p. 1099–1103.3
[18] CIVELEK, Z.; ÇAM, E.; LÜY, M.; GÖREL, G. A new fuzzy controller for adjusting of pitch angle of wind turbine. The Online Journal of Science and Technology-July, v. 6, n. 3, 2016.
[19] J. Wai, M. LEE,*Intelligent optimal control of single-link flexible robot arm*. Industrial Electronics, IEEE Transactions on, IEEE, v. 51, n. 1, p. 201–220, 2004.
[20] O. Castillo e P. Melin, *A review on the design and optimization of interval type-2 fuzzy controllers*. Applied Soft Computing, v. 12, n. 4, p. 1267-1278, 2012.
[21] O. Castillo, P. Melin, J. Kacprzyk e W. Pedrycz,*Type-2 fuzzy logic: theory and applications*, Granular Computing,GRC 2007. IEEE International Conference on. IEEE, 2007.
[22] J. M. Mendel, *Uncertain rule-based fuzzy logic systems: introduction and new directions*, Upper Saddle River: Prentice Hall PTR, 2001.
[23] J. M. Mendel, R. I. John e F. Liu, *Interval type-2 fuzzy logic systems made simple*. IEEE transactions on fuzzy systems, v. 14, n. 6, p. 808-821, 2006.
[24] J. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, U Michigan Press,1975.
[25] A. Freitas, *A survey of evolutionary algorithms for data mining and knowledge discovery*, Advances in evolutionary computing. Springer, 2003. p. 819–845
[26] T. Marler e J. Arora, *Survey of multi-objective optimization methods forengineering*, Structural and multidisciplinary optimization, Springer, v. 26, n. 6, p.369–395, 2004.
[27] F. Bueno, *Métodos heurísticos-teoria e implementações* .IFSC. Araranguá, 2009.
[28] A. Pires, A. Figueiredo, A. G. Severino e F. Araújo, *Otimização de controle fuzzy usando algoritmo genético*, Simpósio Brasileiro de Automação Inteligente - (SBAI), 2013.
[29] F. Santos, *Implementação eficiente de busca em plataforma paralela*. Congresso da Sociedade Brasileira de Computação, Florianópolis, 2002.
[30] R. Koide, *Algoritmo de colônia de formigas aplicado à otimização de materiais compostos laminados*, Dissertação (Mestrado em Engenharia) – Programa de Pós-graduação em Engenharia Mecânica e de Materiais, Universidade Tecnológica Federal do Paraná, Curitiba, 2010.
[31] E. Bonabeau, M. Dorigo, G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. [S.l.]: Oxford university press, 1999
[32] M. Farias, P. Oliveira, L. Barros, L. Siquera Júnior, *Projeto de sistema de controle multivariável baseado em otimização por colônia de formigas*.V Simpósio Brasileiro de Sistemas Elétricos, 2014.
[33] M. Dorigo,*Optimization, learning and natural algorithms*.Ph. D. Thesis, Politecnicodi Milano, Italy, 1992
[34] B. Birge.*Psot-a particle swarm optimization toolbox for use with matlab*.SIS, v. 3, p.973–990, 2003
[35] A. Severino, L. Linhares, F. Araújo, *Optimal design of digital-low pass finite impulse response filter using particle swarm optimization and bat algorithm*. IEEE: Informatics in Control, Automation and Robotics (ICINCO), 2015-12th International Conference, 2015. v. 1, p. 207–214