Chemical Reaction Algorithm for Type-2 Fuzzy Control Optimization in Mobile Robots

Submitted: 25th August 2018; accepted: 20th September 2018

David de la O, Oscar Castillo, José Soria

DOI: 10.14313/JAMRIS_3-2018/14

Abstract:
In this work the optimization process of the tracking and reactive controllers for a mobile robot are presented. The Chemical Reaction Algorithm (CRA) is used to find the optimal parameter values of the membership functions and rules for the reactive and tracking controllers. In this case, we are using five membership functions in each variable of the fuzzy controllers. The main goal of the reactive controller is aimed at providing the robot with the ability to avoid obstacles in its environment. The tests are performed on a benchmark maze problem, in which the goal is not necessarily to leave the maze, but rather that the robot avoids obstacles, in this case the walls, and penalizing for unwanted trajectories, such as cycles. The tracking controller’s goal is for the robot to keep into a certain path, this in order that the robot can learn to react to unknown environments. The optimization algorithm that was used is based on an abstraction of chemical reactions. To perform the simulation we use the “SimRobot” toolbox, the results of the tests are presented in a detailed fashion, and at the end we are presenting a comparison of results among the CRA, PSO and GA methods.

Keywords: Chemical Reaction Algorithm, control, fuzzy logic, robotics

1. Introduction
Lofti Zadeh (1965) proposed fuzzy logic and rules-based procedures as a means to model and capture the human knowledge and deal with uncertainty in the real word. These methods have been applied to ill defined industrial processes, since these methods are usually based on experienced people who usually obtain good results, regardless of whether they receive imprecise information [10–14, 23]. The methods have also been applied to Control of a Mobile Robot using Fuzzy Bee Colony Optimization Algorithm [2, 8], Particle Swarm [3, 16–18, 21–24, 26, 42–46], Genetic Algorithms [11, 15, 29, 47], Differential evolution [30] and Ant Colony Optimization [25, 30, 37, 41]. The origin of these impreciseness can be related to a variation of time concerning the application of a control signal and the warning of its effect [2], and nonlinearities in the dynamics of the system or sensor degradation [21]. The processes in which the fuzzy rule-based approximation have been applied include the automated process of the Operation of a Public Transport System [38], water tank control [1], [20] and sewage treatment plants [47].

We use the word fuzzy because fuzzy systems have to be precisely defined, a fuzzy controller operates as a non-linear controller that is defined with precision. Essentially what we want to emphasize is that although the phenomenon described by this theory may be fuzzy, the theory itself is accurate.

The CRA optimization algorithm was originally developed by Astudillo et al. [6], which is based on a metaheuristic of a population that does not change in size, in addition to applying a generalization of chemical reactions as exploration and exploitation of mechanisms. The algorithm uses chemical reactions by changing at least one of the substances (element or compound), changing their composition and sets of properties. We take as a basis the tests performed by Melendez et al. [32–36] and de la O [13], in which a fuzzy system is designed for the navigation of an Autonomous Mobile Robot, it uses 2 controllers, a reactive controller and a tracking controller, and then optimizes the parameters and rules of the controllers using GA [32–36].

This work is organized as follows: Section 2 refers to the concepts of fuzzy logic systems, Section 3 describes the Chemical Optimization Paradigm used in the present paper. Sections 4 and 5 define the fundamental methodology of this work and the benchmark functions used. Section 6 shows the results of the simulations, comparisons and Section 7 presents the conclusion.

2. The CRA Paradigm
The algorithm of chemical reactions was developed by Astudillo et al. in 2011 [6], this algorithm is a new paradigm which is inspired by natural behavior of the chemical reactions, makes the population come together to find an optimal result in the search space supported by several intensifier/diversifier mechanisms.

One might think that chemical theory and it is descriptions are difficult and that have no relation with the optimization theory, but only the general scheme is considered as the basis of the chemical reaction optimization algorithm.

Astudillo et al. [4–7], defined the elementary terminology for characterizing and classifying artificial chemicals. Because the laws of reaction and rep-
2.2.1. Combination Reactions

In this type of reactions, two of the substances that can be elements or compounds are combined to form the product. Reactions of this type are classified as combining synthesis, and are generally represented as follows:

\[ B + X \rightarrow BX \] (1)

2.2.2. Decomposition Reactions

In a decomposition reaction, a single substance decomposes or breaks, producing two or more distinct substances. The starting material must be a compound and the products can be elements or compounds. The general form of this equation is the following:

\[ BX \rightarrow B + X \] (2)

2.2.3. Substitution Reactions

In a simple substitution reaction an element reacts with a compound and takes the place of one of the elements of the compound, producing a different element and also a different compound. The general formula for this reaction is:

\[ X + AB \rightarrow AX + B \] (3)

2.2.4. Double-substitution Reactions

In a double substitution reaction, two compounds exchange pairs with each other to produce distinct compounds. The general form of these equations is [4]:

\[ AB + CD \rightarrow AC + BD \] (4)

The flowchart for this optimization method can be found in Figure 1, and the following list of steps is presented:

- We start by generating an initial set of elements/compounds.
- We evaluate the original set of elements, to obtain a fitness value.
- Based on the above evaluation, we select some of the elements/compounds to “induce” a reaction.
- Taking into consideration the result of the reaction, evaluations of the new element/compounds are obtained and selected elements are those of greater fitness.
- Repeat the steps until the algorithm meets the terminating criteria (the desired result in the maximum number of iterations is reached) [6].

![Fig. 1. Flowchart of the CRA](image-url)
This algorithm consists of a metaheuristic based on a static population, and applies an abstraction of chemical reactions as intensifying mechanisms and diversification. It also uses an elitist reinsertion strategy which allows for the perpetuity of the best elements and, therefore, the average fitness of the whole set of elements increases with each iteration.

The reactions of synthesis and decomposition are used for exploration in the search space of the solutions: These procedures demonstrate to be effective and promptly lead to the results of a desired optimal value.

The single and double substitution reactions allow the algorithm to search for obtaining optimal values around a previously found solution.

We start the algorithm by randomly generating a set of elements/compounds under the uniform distribution space of possible solutions, and this is represented as follows:

$$X = \{x_1, x_2, ..., x_n\}, \quad (5)$$

Where $$x_n$$ is used to represent the element/compound.

The total number and the representation of the original elements depend on the complexity of the problem that is solved.

In order to find the best possible controllers we use a metaheuristic strategy, which has proven to produce good results, and this is achieved by applying the CRA (see Fig. 2). In this case the algorithm will search the solution space of the problem to be solved. Combining the values of the best controllers and generating new controllers. The goal is to optimize the parameters of the membership functions and fuzzy rules.

### 3. Description of the Tool for Simulation

The SIMROBOT toolbox software [30] enables performing robot simulations and is used for testing the fuzzy controllers. The mobile robot has two controlled and sensed wheels, in addition to an uncontrolled and un-sensed wheel. Figure 3 illustrates this type of robot. This robot has two degrees of freedom: y-translation and x-translation or z-rotation.

---

**Fig. 2. General flowchart of the chemical reaction algorithm optimizing the fuzzy controllers**

**Fig. 3. Kinematic coordinate system [27]**

The kinematic equations of the mobile robot are as follows:

- Equation 6 shows the sensed forward velocity solution:

$$\begin{cases}
V_{b_x} = \frac{R}{2l_a} \left( -l_b -l_a -l_b \right) \\
V_{b_y} = \frac{R}{2l_b} \left( -l_b -l_a -l_b \right) \\
\alpha_{a_x} = \frac{1}{R(l_b^2 + 1)} \left( -l_b l_b - l_b \right) \\
\alpha_{a_y} = \frac{1}{R(l_b^2 + 1)} \left( -l_b l_b - l_b \right)
\end{cases} \quad (6)$$

- Equation 7 shows the Actuated Inverse Velocity Solution:

$$\begin{cases}
\alpha_{a_x} = \frac{1}{R(l_b^2 + 1)} \left( -l_b l_b - l_b \right) \\
\alpha_{a_y} = \frac{1}{R(l_b^2 + 1)} \left( -l_b l_b - l_b \right)
\end{cases} \quad (7)$$

where (in the metric system):

- $$V_{b_x}, V_{b_y}$$ are the translational velocities of the robot’s body [m/s],
- $$\alpha_{a_x}, \alpha_{a_y}$$ is the robot’s z-rotational velocity [rad/s],
- $$V_{b_x}, V_{b_y}$$ are the wheels’ rotational velocities [rad/s],
- $$R$$ the actuated wheel radius [m],
- $$l_a, l_b$$ are the distances between the wheels and the robot’s axes [m].

### 4. Simulations and Tests

Two different control tests were performed to experiment with the performance of the algorithm in control problems, and each test is described as follows.

We used the Chemical Reaction Algorithm (CRA) to optimize the parameters and rules of the reactive and tracking fuzzy controllers, one test was to optimize only the parameters of the fuzzy controller, leaving the controller’s fuzzy rules fixed. A second test was to optimize parameters and fuzzy controller rules, and for this test we executed two CRAs simultaneously, one that optimizes the parameters and another one that optimizes the rules, alternating in each iteration.

The CRA performs the task of initializing the parameters of each fuzzy controller, selecting the elements and the chemical reactions that will be applied, evaluating the results and, through the simulation, performing an elitist reinsertion and putting them back into the population.
4.1. Reactive Controller

The reactive control has the purpose to achieve the same capability that a person has when driving, that is, to react to unanticipated circumstances, road traffic congestions, traffic of the signs, etc., but in a more elementary level. We use a maze to test possible solutions, in which the objective is not to guide the robot through the maze to the exit, but rather obstacle avoidance. The objective is to optimize the robot controller to find the maze output, use the maze to optimize the reactive control due to the characteristic of the situation of the simulation, i.e. it is a confined space in which the robot can not move easily and each wall is considered as an obstacle for the robot to avoid them while moving. We use a Mamdani type FIS, which consists of 3 inputs, that are the distances obtained by the robot sensors, as mentioned in Section 2, and 2 outputs that control the speed of the servomotor in the robot, and all this information is encoded in each element.

4.1.1. Reactive Controller with Type-1 Fuzzy Logic

We encode each membership function of the fuzzy reactive controller, represented by an element, into 25 positions of a vector of real values, which represents the values of each parameter of the triangular membership function, which has five membership functions in each of its variables (see Fig. 4).

![Fig. 4 Structure of the element to fuzzy parameters](image)

We encode the values of the rules of the fuzzy reactive controller, represented by an element, into 250 positions of a vector of integer values, which represent the values of the rules set of the fuzzy controller (see Fig. 5).

![Fig. 5. Structure of the element to fuzzy rules](image)

Fig. 6. Fuzzy reactive control inputs

The controller is a Mamdani fuzzy system and it has 3 inputs (the sensors of the robot) and two outputs which control the speed of each servomotor of the robot. This is illustrated in Figure 6.

4.1.2. Reactive Controller with Type-2 Fuzzy Logic

We encode each membership function of the fuzzy reactive controller, represented by an element, into 50 positions of a vector of real values, which represent the values of each parameter of the triangular membership function, which has five membership functions in each of its variables (see Fig. 7).

![Fig. 7. Element Encoding to fuzzy parameters](image)

We encode the values of the rules of the fuzzy reactive controller, represented by an element, into 250 positions of a vector of integer values, which represent the values of the rules set of the fuzzy controller (see Fig. 8).

![Fig. 8. Structure of the element to fuzzy rules](image)

The controller is a Mamdani fuzzy system and it has 3 inputs (the sensors of the robot) and two outputs which control the speed of each servomotor of the robot, respectively. This is illustrated in Figure 9.

![Fig. 9. Fuzzy reactive control inputs](image)
4.2. Tracking Controller

The goal of the tracking controller is to keep the robot on the right path, in a given reference. The robot will be able to move about the reference and stay on the road, being able to move from point A to B, without obstacles present in the path.

4.2.1. Tracking Controller with Type-1 Fuzzy Logic

We encode each membership function of the fuzzy tracking controller, represented by an element, into 20 positions of a vector of real values, which represents the values of each parameter of the triangular membership function, which has five membership functions in each of its variables (see Fig. 10).

![Fig. 10. Structure of the element to fuzzy parameters](image)

We encode the values of the rules of the fuzzy reactive controller, represented by an element, into 50 positions of a vector of integer values, which represent the values of rules set of the fuzzy controller (see Fig. 11).

![Fig. 11. Structure of the element to fuzzy rules](image)

The controller will take into account the errors $(\Delta e_p, \Delta \theta)$ in its minimum values, Figure 12, the minimum values to which we refer are the relative error of the orientation of the left front and the relative error of the position. We used a Mamdani Fuzzy system and its 2 inputs are $(\Delta e_p, \Delta \theta)$ and two outputs which control the speed of each servomotor of the robot and this is illustrated in Figure 12.

![Fig. 12. Fuzzy controller inputs $e_p, \theta$](image)

To calculate the performance of the controller we use the equation of the mean square error between the reference and the path of the robot.

4.2.2. Tracking Controller with Type-2 Fuzzy Logic

We encode each membership function of the fuzzy tracking controller, represented by an element, into 40 positions of a vector of real values, which represent the values of each parameter of the triangular membership function, which has five membership functions in each of its variables (see Fig. 13).

![Fig. 13. Structure of the element to fuzzy parameters](image)

We encode the values of the rules of the fuzzy reactive controller, represented by an element, into 50 positions of a vector of integer values, which represent the values of rules set of the fuzzy controller (see Fig. 14).

![Fig. 14. Structure of the element to fuzzy rules](image)

The controller will take into account the errors $(\Delta e_p, \Delta \theta)$ in its minimum values, Figure 7, the minimum values to which we refer are the relative error of the orientation of the left front and the relative error of the position. We used a Mamdani Fuzzy system and its 2 inputs are $(\Delta e_p, \Delta \theta)$ and two outputs which control the speed of each servomotor of the robot and this is illustrated in Figure 15.

![Figure 15. Fuzzy controller inputs $e_p, \theta$](image)
To calculate the performance of the controller we use the equation of the mean square error between the reference and the path of the robot.

### 4.3. Objective Function for Both Controllers

The CRA starts by creating elements to be evaluated by the Simrobot toolbox which will assign a crisp value that will represent the performance of the controller taking into account the criteria that we want to achieve. To achieve this, we must provide the CRA with a good evaluation criterion which is capable of penalizing undesirable behaviors and rewarding with higher fitness values those elements that yield the performance we desire in the controller. If we do not provide a correct evaluation method, we can guide the population of elements to suboptimal solutions or even not to a solution at all [5], [17]–[21], [31]. The algorithm has fixed parameters for the chemical reactions, for our tests and based on the proposed by Astudillo et al., we use for each reaction a value of 0.2, which corresponds to take 20% of the amount of elements and react in each of the 4 reactions.

#### 4.3.1. Reactive Controller Objective Function

In order to measure the performance of the controller, we will use the following criteria:
- Distance traveled,
- Time used to travel the distance,
- Battery life.

In order to measure these criteria we will use a Fitness FIS, which will provide the desired fitness value, adding very basic fuzzy rules that will give greater fitness to the controller that provided the longer trajectories in smaller times and a longer battery life. This seems to be a good strategy that will guide the algorithm to evolve and provide optimal control, but we have noticed that this strategy is not able to do just that on its own: it is also necessary to have a robot trajectory supervisor to make sure that there is a forward movement path and free of loops. For this purpose, it uses a neural network (NN) that is capable of detecting trajectories with cycles that do not have the desired forward displacement behavior, assigning a low activation value and higher activation values to those that are cycle free. The NN consists of two inputs and one output, and two hidden layers, see Figure 16.

To perform the evaluation of the reactive controller we will use the method of integrating both the FIS and the NN where the final fitness value for each element will be calculated with Equation 7. Taking into account the NN response, the activation value is set to 0.35, this means that any activation less than 0.35 will be penalized in the ability given by the FIS.

Equation 8 expresses how to calculate the fitness value of each individual

$$f(i) = \begin{cases} f_{v,nv,nv} < 0.35 & \text{if } f_{v,nv} \geq 0.35 \\ f_{v,nv} \geq 0.35 & \text{otherwise} \end{cases}$$

where:
- $f_i$ – fitness value of the $i$-th individual,
- $f_{v}$ – crisp value out of the fitness FIS,
- $n_{v}$ – looping trajectory activation value.

#### 4.3.2. Tracking Controller Objective Function

To measure the performance of the tracking controller we use the root-mean-square error (RMSE) between the given reference and the path achieved by the robot. We will perform the test three times for each element and take the average of the three tests. The initial position of the robot with respect to the reference is random, but it ensures that a vertical position of the robots is above the reference and in another test it is below it (Fig. 17) [32–36].

![Figure 16. Fitness Function for the Reactive Controller](image1)

![Figure 17. Fitness Function for the Tracking Controller](image2)
5. Simulation Results

We present the results of the tests performed for each of the controllers: reactive and tracking. In order to perform these tests we have used as mentioned before the SimRobot Software and the Matlab language. To determine the suitability of each controller we use the simulation software. In the tests of the reactive controller, the robot must be able to react in a closed environment, avoiding hitting the obstacles (walls). In the tracking controller test the robot must be able to stay above the given reference.

The results will be presented in two subsections:
- Reactive Controller,
- Tracking Controller.

5.1. Reactive Controller

In this section, we show the results of the tests with the reactive controller, and to determine the suitability of each controller we use the simulation tool. In the tests of the reactive controller the robot must be able to react in a closed environment, avoiding hitting the obstacles (walls).

5.1.1. Reactive Controller with Type-1 Fuzzy Logic

We can find the configuration of the CRA and the results of the simulation tests in Table 1, where we can find the fitness value obtained in each of the experiments. We can also find statistical values which are the mean, variance, best and the worst obtained values.

<table>
<thead>
<tr>
<th>Table 1. Summary of type 1 reactive controls results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>Fitness</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>Average</td>
</tr>
<tr>
<td>Best</td>
</tr>
<tr>
<td>Poor</td>
</tr>
<tr>
<td>Std Dev</td>
</tr>
</tbody>
</table>

5.1.2. Reactive Controller with Type-2 Fuzzy Logic

We can find the configuration of the CRA and the results of the simulation tests in Table 2, where we can find the fitness value obtained in each of the experiments. We can also find statistical values which are the mean, variance, best and the worst obtained values.

<table>
<thead>
<tr>
<th>Table 2. Summary of type 2 reactive controls results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>Fitness</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>Average</td>
</tr>
<tr>
<td>Best</td>
</tr>
<tr>
<td>Poor</td>
</tr>
<tr>
<td>Std Dev</td>
</tr>
</tbody>
</table>

We can find the configuration of the CRA and the results of the simulation tests in Table 3, and we can also find the fitness value obtained in each of the experiments. We can also notice statistical values which are the mean, variance, best and the worst obtained values.

<table>
<thead>
<tr>
<th>Table 3. Summary of Tracking Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>Fitness</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>Average</td>
</tr>
<tr>
<td>Best</td>
</tr>
<tr>
<td>Poor</td>
</tr>
<tr>
<td>Std Dev</td>
</tr>
</tbody>
</table>

We can state that the results are good, because the average of the results is 0.2880, the best result is 0.2260, and when comparing with other algorithms (i.e. PSO) the result of the CRA is better.

5.3. Comparison of Results

We compare against Melendez et al. [4], and Tables 4 and 5 summarize the results presented in [4], where we have added the results obtained using the CRA.

5.3.1. Reactive Controller

In this section we do a comparison of the CRA, Genetic Algorithm(GA) and Particle Swarm Optimization.

We can find the parameters used in the CRA, GA and PSO in Table 3, we note that the parameters that the CRA uses less iterations than the GA and PSO, and also a smaller population size.
Table 4. Parameters of the CRA, GA and PSO

<table>
<thead>
<tr>
<th></th>
<th>CRA</th>
<th>GA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration</td>
<td>Elements</td>
<td>Rate-DoksSub</td>
<td>Iteration</td>
</tr>
<tr>
<td>30</td>
<td>10 RateDoks</td>
<td>0.2 Rate-Sub</td>
<td>8000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.02</td>
</tr>
<tr>
<td>Particle</td>
<td>Iteration</td>
<td>CL, C2</td>
<td>Node Weight</td>
</tr>
<tr>
<td>20</td>
<td>500</td>
<td>1.4952</td>
<td>LD</td>
</tr>
</tbody>
</table>

We can find the results of CRA, GA and PSO in Table 5, where we have the best, the worst, standard deviation and the average of each method.

Table 5. Comparison of Results of CRA and GA

<table>
<thead>
<tr>
<th></th>
<th>CRA</th>
<th>GA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank</td>
<td>Fitness</td>
<td>Active Rules</td>
<td>Fitness</td>
</tr>
<tr>
<td>1</td>
<td>0.3198</td>
<td>119</td>
<td>0.3607</td>
</tr>
<tr>
<td>2</td>
<td>0.2470</td>
<td>17</td>
<td>0.3304</td>
</tr>
<tr>
<td>3</td>
<td>0.3017</td>
<td>75</td>
<td>0.3800</td>
</tr>
<tr>
<td>4</td>
<td>0.2618</td>
<td>34</td>
<td>0.3900</td>
</tr>
<tr>
<td>5</td>
<td>0.3253</td>
<td>25</td>
<td>0.3999</td>
</tr>
<tr>
<td>6</td>
<td>0.2589</td>
<td>31</td>
<td>0.3299</td>
</tr>
<tr>
<td>7</td>
<td>0.3089</td>
<td>41</td>
<td>0.3226</td>
</tr>
<tr>
<td>8</td>
<td>0.2463</td>
<td>34</td>
<td>0.3179</td>
</tr>
<tr>
<td>9</td>
<td>0.4416</td>
<td>38</td>
<td>0.3143</td>
</tr>
<tr>
<td>10</td>
<td>0.2534</td>
<td>30</td>
<td>0.3126</td>
</tr>
<tr>
<td>Best</td>
<td>0.4416</td>
<td>0.3607</td>
<td>0.4110</td>
</tr>
<tr>
<td>Worst</td>
<td>0.2463</td>
<td>0.3267</td>
<td>0.3551</td>
</tr>
<tr>
<td>Average</td>
<td>0.2965</td>
<td>0.3278</td>
<td>0.3758</td>
</tr>
<tr>
<td>Std Dev</td>
<td>0.0597</td>
<td>0.0135</td>
<td>0.0201</td>
</tr>
</tbody>
</table>

To compare the algorithms we performed an ANOVA test for the three samples with a significance level of 0.05. Using the following parameters for the ANOVA test:

H0: All means are equal
H1: At least one mean is different
α = 0.05

To perform the analysis, the variances are assumed to be the same. We can conclude that the results of the ANOVA test is to reject the null hypothesis. Because of this, the Tukey test is also performed. Tukey’s comparisons indicated with 95% confidence that PSO was statistically better than GA and CRA, which were assumed to be statistically the same.

5.3.2. Tracking Controller

We can find the parameters used for CRA, GA and PSO in Table 6, we can note that the parameters of CRA uses less iterations than the GA and PSO, and also a smaller population size.

Table 6. Comparison of Results of CRA and PSO

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>t</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRA</td>
<td>0.2880</td>
<td>0.0386</td>
<td>-2.03</td>
<td>0.073</td>
</tr>
<tr>
<td>PSO</td>
<td>0.3129</td>
<td>0.00334708</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. Conclusion

In the present work, we used the CRA to optimize the parameters and rules of the fuzzy controllers, both for reactive and tracking behaviors. The reactive controller aims at giving the robot the ability to avoid obstacles in its environment. The tests were performed in a maze, in which the goal is not to leave the maze, but that the robot avoids obstacles, in this case the walls, and penalizing the unwanted trajectories as cycles. The tracking controller’s goal is for the robot to be able to stay on a certain path, this test was performed 3 times for each element, this in order that the robot can react to unknown environments. After performing the tests, analyzing and comparing the results, we can notice that the algorithm is not statistically better GA, and its performance is similar to the PSO, because it is a newly created algorithm. We propose the following tasks that could be performed to improve the performance of the algorithm: to use a fuzzy controller that controls the parameters of the chemical reactions, since these values are fixed during the execution of the test, with this we will be able to do at the beginning an exploration and at the end focus on the exploitation. It is important to mention that the advantage of the algorithm is that we use it for minimizing time, less population (elements) and less iterations.

AUTHORS

David de la O, Oscar Castillo*, José Soria – Tijuana Institute of Technology, 22379, Tijuana, Mexico. Emails: ddsh@live.com, ocastillo@tectijuana.mx, jsoria57@gmail.com

*Corresponding author.

REFERENCES


DIO: 10.1007/978-3-642-16773-7_41.


DIO: 10.1007/978-3-319-05170-3_19.


DIO: scholar.waset.org/1307-6892/391.


DIO: 10.1007/978-3-642-33021-6-8.


DIO: 10.1016/S0965-9978(99)00054-X.


