Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Parameter Tuning of Membership Functions of a Type-1 and Type-2 Fuzzy Logic Controller for an Autonomous Wheeled Mobile Robot Using Ant Colony Optimization

R. Martínez-Marroquín, O. Castillo, J. Soria
Division of Graduate Studies and Research
Tijuana Institute of Technology
Tijuana, Baja California, México
e-mails: mc.ricardo.mtz@gmail.com, ocastillo@hafsamx.org, jsoria@cox.net

***Abstract*— In this paper we describe the application of a Simple ACO (S-ACO) as an optimization method for the membership functions' parameters of a fuzzy logic controller (FLC) in order to find the optimal intelligent controller for an Autonomous Wheeled Mobile Robot. Simulation results show that ACO outperforms a GA in the optimization of FLCs for an autonomous mobile robot.**

KEYWORDS: fuzzy logic, Swarm Intelligence, Mobile robots, Optimization, Intelligent control.

I. INTRODUCTION

Nowadays, fuzzy logic is one of the most used methods of computational intelligence and with the best future; this is possible thanks to the efficiency and simplicity of fuzzy systems since they use linguistic terms similar to those that human beings use.

The complexity in developing fuzzy systems can be found at the time of deciding which are the best parameters for the membership functions, the number of rules or even the best granularity that could give us the best solution for the problem that we want to solve.

A solution for the above mentioned problem is the application of Bio-inspired algorithms for the optimization of fuzzy systems. Bio-inspired algorithms for optimization can be a useful tool because of their capabilities of solving nonlinear problems, well-constrained or even NP-hard problems. Among the most used methods for optimization we can find: Genetic Algorithms, Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), etc.

This paper describes the application of ACO as an optimization method of the membership functions parameters of the type-1 and type-2 FLC, in order to find the best intelligent controller for an autonomous wheeled mobile robot.

This paper is organized as follows: Section 2 shows the concept of ACO and a description of S-ACO, which is the technique that was applied for optimization. Section 3 presents the problem statement and the dynamic and kinematic model of the unicycle mobile robot. Section 4 shows the proposed fuzzy logic controller and in Section 5 the development of the optimization method is described. In Section 6 the simulation results are shown. Finally, Section 7 shows the Conclusions.

II. S-ACO ALGORITHM

ACO is a probabilistic technique that can be used for solving problems that can be reduced to finding good paths along graphs. This method is inspired on the behavior presented by ants in finding paths from the nest or colony to the food source.

The S-ACO is an algorithmic implementation that adapts the behavior of real ants to solutions of minimum cost path problems on graphs [11]. A number of artificial ants build solutions for a certain optimization problem and exchange information about the quality of these solutions making allusion to the communication system of real ants [5].

Let us define the graph $G = (V, E)$, where V is the set of nodes and E is the matrix of the links between nodes. G has $n_G = |V|$ nodes. Let us define L^K as the number of hops in the path built by the ant k from the origin node to the destiny node. Therefore, it is necessary to find:

$$Q = \{q_a, ..., q_f | q_1 \in C\} \quad (1)$$

where Q is the set of nodes representing a continuous path with no obstacles; $q_a, ..., q_f$ are former nodes of the path and C is the set of possible configurations of the free space. If $x^k(t)$ denotes a Q solution in time t, $f(x^k(t))$ expresses the quality of the solution. The S-ACO algorithm is based on Equations (2), (3) and (4):

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^k}{\sum_{j \in N_{ij}^k} \tau_{ij}^\alpha(t)} & if \quad j \in N_i^k \\ 0 & if \quad j \notin N_i^k \end{cases} \quad (2)$$

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) \quad (3)$$

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \sum_{k=1}^{n_k} \Delta\tau_{ij}^k(t) \quad (4)$$

Equation (2) represents the probability for an ant *k* located on a node *i* selects the next node denoted by *j*, where, N_i^k is the set of feasible nodes (in a neighborhood) connected to node *i* with respect to ant *k*, τ_{ij} is the total pheromone concentration of link *ij*, and α is a positive constant used as a gain for the pheromone influence.

Equation (3) represents the evaporation pheromone update, where $\rho \in [0,1]$ is the evaporation rate value of the pheromone trail. The evaporation is added to the algorithm in order to force the exploration of the ants, and avoid premature convergence to sub-optimal solutions [11]. For $\rho = 1$ the search becomes completely random [11].

Equation (4), represents the concentration pheromone update, where $\Delta\tau_{ij}^{k}$ is the amount of pheromone that an ant *k* deposits in a link *ij* in a time *t*.

The general steps of S-ACO are the following:

1. Set a pheromone concentration τ_{ij} to each link (*i,j*).
2. Place a number k=1, 2,…, n_k in the nest.
3. Iteratively build a path to the food source (destiny node), using Equation (2) for every ant.
   - Remove cycles and compute each route weight $f(x^k(t))$. A cycle could be generated when there are no feasible candidates nodes, that is, for any *i* and any *k*, $N_i^k = \varnothing$; then the predecessor of that node is included as a former node of the path.
4. Apply evaporation using Equation (3).
5. Update of the pheromone concentration using equation (4)
6. Finally, finish the algorithm in any of the three different ways:
   -When a maximum number of epochs has been reached.
   -When it has been found an acceptable solution, with $f(x_k(t)) < \varepsilon$.
   -When all ants follow the same path.

III. Problem Statement

The model of the robot considered in this paper is a unicycle mobile robot (see Figure 1), that consists of two driving wheels mounted of the same axis and a front free wheel.

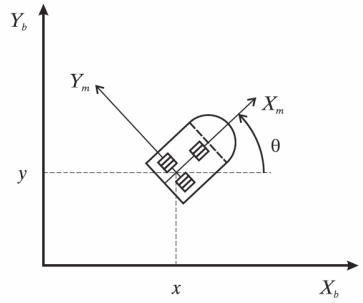


Fig. 1. Wheeled Mobile Robot [10]

A unicycle mobile robot is an autonomous, wheeled vehicle capable of performing missions in fixed or uncertain environments. The robot body is symmetrical around the perpendicular axis and the center of mass is at the geometrical center of the body. It has two driving wheels that are fixed to the axis that passes through *C* and one passive wheel prevents the robot from tipping over as it moves on a plane. In what follows, it´s assumed that motion of the passive wheel can be ignored in the dynamics of the mobile robot presented by the following set of equations [8]:

$$M(q)\dot{\vartheta} + C(q,\dot{q})\vartheta + D\vartheta = \tau + F_{ext}(t) \tag{5}$$

$$\dot{q} = \underbrace{\begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix}}_{J(q)} \underbrace{\begin{bmatrix} v \\ w \end{bmatrix}}_{\vartheta} \tag{6}$$

where $q = (x, y, \theta)^T$ is the vector of the configuration coordinates; $\vartheta = (v, w)^T$ is the vector of linear and angular velocities; $\tau = (\tau_1, \tau_2)$ is the vector of torques applied to the wheels of the robot where τ_1 and τ_2 denote the torques of the right and left wheel respectively (Figure 1); $F_{ext} \in \mathbb{R}^2$ uniformly bounded disturbance vector; $M(q) \in \mathbb{R}^{2\times 2}$ is the positive-definite inertia matrix; $C(q,\dot{q})\vartheta$ is the vector of centripetal and Coriolis forces; and $D \in \mathbb{R}^{2\times 2}$ is a diagonal positive-definite damping matrix. Equation (6) represents the kinematics of the system, where (x, y) is the position of the mobile robot in the X-Y (world) reference frame, θ is the angle between heading direction and the *x*-axis *v* and w are the angular and angular velocities, respectively.

Furthermore, the system (5)-(6) has the following non-holonomic constraint:

$$\dot{y}\cos\theta - \dot{x}\sin\theta = 0 \tag{7}$$

which corresponds to a no-slip wheel condition preventing the robot from moving sideways[9]. The system (6) fails to meet Brockett's necessary condition for feedback stabilization [2], which implies that non-continuous static state-feedback controller exists that stabilizes the close-loop system around the equilibrium point.

The control objective is to design a fuzzy logic controller of τ that ensures:

$$\lim_{t\to\infty} \| q_d(t) - q(t) \| = 0 \tag{8}$$

for any continuously, differentiable, bounded desired trajectory $q_d \in \mathbb{R}^3$ while attenuating external disturbances.

A more detailed description can be found on reference [10].

IV. Fuzzy Logic Control Design

In order to satisfy the control objective it is necessary to design fuzzy logic controllers for the real velocities of the mobile robot. To do that, Takagi-Sugeno fuzzy logic controllers were designed, using linguistic variables in the input and mathematical functions in the output. The errors of

linear and angular velocities (e_v, e_w respectively), were taken as inputs variables, while the right (τ_1) and left (τ_2) torques were taken as outputs. The membership functions used on the input are trapezoidal for the negative (*N*) and positive (*P*), and a triangular one was used for the zero (*C*) linguistic terms. The interval used for the fuzzy controllers is [-50 50] [10]. Figure 2 shows the input and output variables.

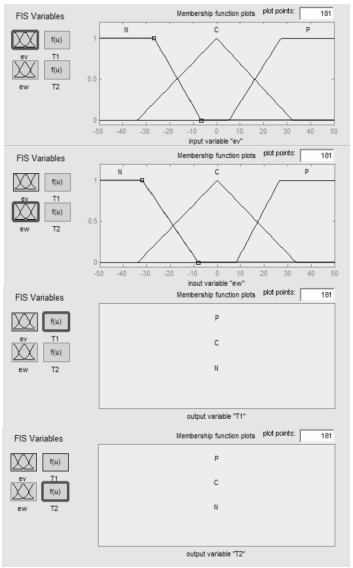


Fig. 2. Input /output variables: (a) Linear velocity error. (b) Angular velocity error. (c) Right output (τ_1). (d) Left output (τ_2).

The rule set of the FLC contains 9 rules, which governs the input-output relationship of the FLC and this adopts the Takagi-Sugeno style inference engine [10] it is used with a single point in the outputs, which means that the outputs are constant values, obtained using a weighted average defuzzification procedure. In Table 1 we present the rule set whose format is established as follows:

Rule i : if e_v is G_1 and e_w is G_2 then F is G_3 and N is G_4

where $G_1..G_4$ are the fuzzy set associated to each variable i=1,2,…,9.

TABLE 1. FUZZY RULE SET

e_v / e_w	**N**	**C**	**P**
N	N / N	N / C	N / P
C	C / N	C / C	C / P
P	P / N	P / C	P / P

To find the best type-1 FLC, we used a S-ACO to find the parameters of the membership functions. Table 2 shows the parameters of the membership functions, the minimal and maximum values in the search range for the S-ACO algorithm to find the best type-1 fuzzy logic controller.

TABLE 2. PARAMETERS OF THE MEMBERSHIP FUNCTIONS

MF TYPE	**Point**	**Minimal Value**	**Maximal Value**
Trapezoidal	a	-50	-50
	b	-50	-50
	c	-15	-5.1
	d	-1.5	-0.5
Triangular	a	-5	-1.8
	b	0	0
	c	1.8	5
Trapezoidal	a	0.5	1.5
	b	5.1	15
	c	50	50
	d	50	50
Constant (N)	a	-50	-50
Constant (C)	a	0	0
Constant (P)	a	50	50

It is important to remark that values shown in Table 2 are applied to both inputs and both outputs of the fuzzy logic controller.

V. ACO ARCHITECTURE FOR THE FLC OPTIMIZATION

A S-ACO algorithm was applied for the optimization of the membership functions for the type-1 fuzzy logic controller. For developing the architecture of the algorithm it was necessary to follow the next steps:

1. Marking the limits of the problem in order to eliminate unnecessary complexity.
2. Representing the architecture of the FLC as a graph that artificial ants could traverse.
3. Achieving an adequate handling of the pheromone but permitting the algorithm to evolve by itself.

A. Limiting the Problem and Graph Representation

One of problems found on the development of the S-ACO algorithm was to make a good representation of the FLC. First we reduced the number of elements that the method needed to find by deleting the elements whose minimal value and maximal values are the same (see Table 2) and therefore if they were included they will not change any way. Table 3 shows the parameters of the membership functions included in the search.

TABLE 3. PARAMETERS OF THE MEMBERSHIP FUNCTIONS INCLUDED IN S-ACO SEARCH

Mf Type	**Point**	**Minimal Value**	**Maximal Value**
Trapezoidal	c	-15	-5.1
	d	-1.5	-0.5
Triangular	a	-5	-1.8
	c	1.8	5
Trapezoidal	a	0.5	1.5
	b	5.1	15

The next step was to represent those parameters shown in Table 3; to do that it was necessary to discretize the parameters in a range of possible values in order to represent every possible value as a node in the search graph. The level of discretization between minimal and maximal value was of 0.1 (by example: -1.5, -1.4, -1.3,…, -0.5).

Table 4 shows the number of possible values that each parameter can take.

TABLE 4. NUMBER OF POSSIBLE VALUES OF THE PARAMETERS OF MEMBERSHIP FUNCTIONS OF THE TYPE-1 FLC

Mf Type	**Point**	**Combinations**
Trapezoidal	c	100

	d	15
Triangular	a	33
	c	33
Trapezoidal	a	15
	b	100

Figure 3 shows the search graph for the proposed S-ACO algorithm, the graph can be viewed as a tree where the root is the nest and the last node is the food source.

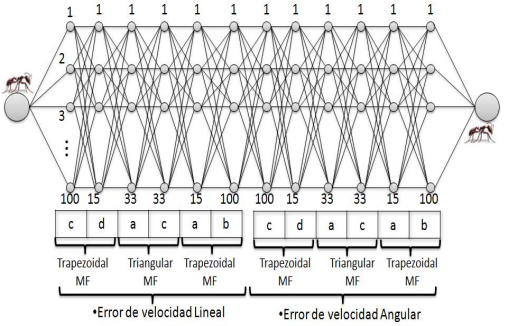


Fig. 3. S-ACO Architecture

B. References Updating Pheromone trail

An important issue is that the update of pheromone trail has to be applied in the best way possible. In this sense we need to handle the evaporation (Equation 3), and increase or deposit of pheromone (Equation 4), where the key parameter in evaporation is denoted by ρ that represents the rate of evaporation and in deposit of pheromone is denoted by $\Delta\tau$ that represents the amount of pheromone that an ant k deposits in a link ij in a time t. For ρ we assign a random value and Equation 9 shows the way of how the increase of pheromone is calculated.

$$\Delta\tau = \frac{(e_{\max} - e_k)}{e_{\max}} \tag{9}$$

where $e_{\max} = 10$ is the maximum error of control permitted and e_k is error of control generated by a complete path of an ant k. We decided to allocate $e_{\max} = 10$ in order to stand $\Delta\tau \in [0,1]$. We also defined $\alpha = 0.2$ (see equation 2) for all experiments presented in this paper because of the results obtained after the tests. And finally we use a random value for the evaporation coefficient ρ = [0,1] (see equation3) in each iteration in order to simulate the uncertain environment conditions which affects the real pheromone trails deposited by real ants.

VI. Simulation Results

In this section we present the results of the proposed controller to stabilize the unicycle mobile robot, defined by Equation (5) and Equation (6), where the matrix values

$$M(q) = \begin{bmatrix} 0.3749 & -0.0202 \\ -0.0202 & 0.3739 \end{bmatrix},$$

$$C(q,\dot{q}) = \begin{bmatrix} 0 & 0.1350\dot{\theta} \\ -0.150\dot{\theta} & 0 \end{bmatrix},$$

and $D = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$

were taken from [6]. The evaluation was made through computer simulation performed in MATLAB® and SIMULINK®.

The desired trajectory is the following one:

$$\vartheta_d(t) = \begin{cases} v_d(t) = 0.2(1-\exp(-t)) \\ w_d(t) = 0.4\sin(0.5t) \end{cases} \tag{10}$$

and was chosen in terms of its corresponding desired linear v_d and angular w_d velocities, subject to the initial conditions

$$q(0) = (0.1,0.1,0,)^T \text{ and } \vartheta(0) = 0 \in \mathbb{R}^2$$

The gains γ_i, i=1, 2, 3 of the kinematic model (see [10]) are $\gamma_1 = 5$, $\gamma_2 = 24$ and $\gamma_3 = 3$ were taken from [10].

A. S-ACO Algorithm Results for the Optimization of the Type-1 FLC

Table 5 shows the results of the type-1 FLC, obtained varying the values of maximum iterations and number of artificial ants, where the highlighted row shows the best result obtained with the method. Figure 4 shows the optimization behavior of the method.

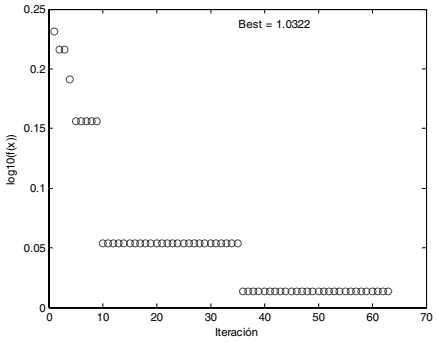


Fig. 4. Optimization Behavior for the S-ACO on type-1 FLC Optimization.

Figure 5 shows the membership functions of the FLC obtained by S-ACO algorithm.

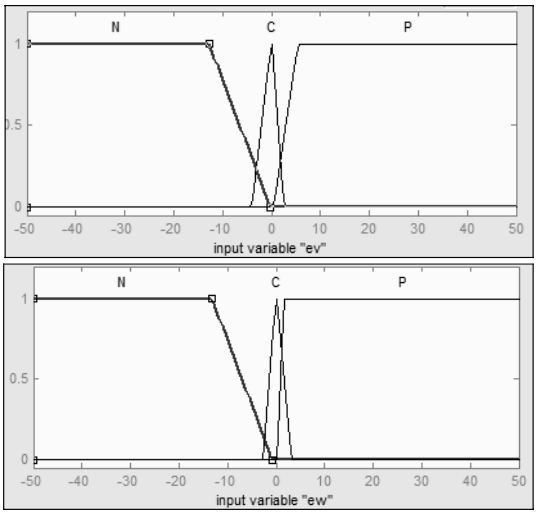


Fig. 5. Membership Functions: (a) Linear velocity error, (b) angular velocity error optimized by S-ACO algorithm.

The block diagram used for the FLC that obtained the best results can be seeing in [10]. Figure 6 shows the desired trajectory and obtained trajectory.

TABLE 5. S-ACO RESULTS FOR TYPE-1 FLC OPTIMIZATION.

Experiment	Iterations	Ants	Average Error	Time
01	25	80	0.1263	00:14:55
02	25	80	0.1432	00:15:00
03	25	80	0.1069	00:14:52
04	25	80	0.1119	00:15:41
05	50	80	0.1100	00:28:43
06	50	80	0.0982	00:29:22
07	100	18	0.1078	00:13:59
08	100	18	0.1095	00:13:16
09	100	18	0.1197	00:12:09
10	15	20	0.1343	00:02:19
12	15	20	0.1308	00:02:17
13	15	20	0.1353	00:02:18
14	50	20	0.1222	00:07:30
15	50	20	0.1233	00:07:29
16	50	20	0.1098	00:07:30
17	50	20	0.1215	00:08:09
18	50	20	0.1142	00:07:14
19	50	20	0.1164	00:07:39
20	50	20	0.1217	00:07:22
21	100	200	0.1087	02:15:46
22	50	80	0.1086	00:29:56
23	60	90	0.1338	00:41:56
24	60	90	0.1091	00:39:48
25	60	90	0.1044	00:40:00
26	17	28	0.1211	00:04:44
27	17	28	0.1211	00:05:51
28	40	50	0.1211	00:17:51
29	40	50	0.1211	00:18:14

B. S-ACO Algorithm Results for the Optimization of the Type-2 FLC

Table 6 shows the results of the type-2 FLC, obtained varying the values of maximum iterations and number of artificial ants, where the highlighted row shows the best result obtained with the method. Figure 7 shows the optimization behavior of the method.

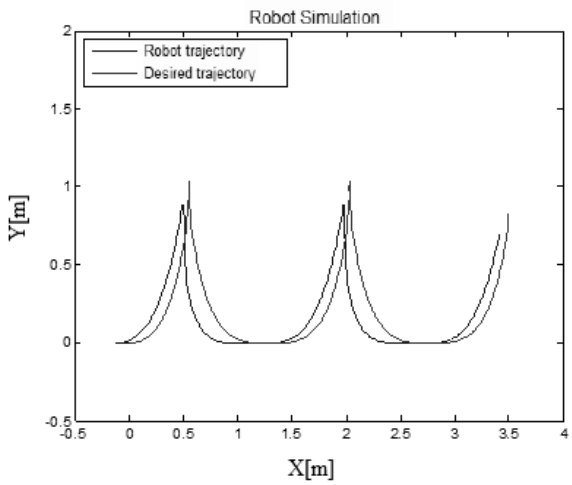


Fig. 6. Obtained trajectory with type-1 FLC optimization

TABLE 6. S-ACO RESULTS FOR TYPE-2 FLC OPTIMIZATION.

Experiment	Iterations	Ants	Average Error	Time
01	10	10	0.0866	00:30:43
02	10	10	0.0756	00:57:06
03	10	10	0.0730	00:32:07
04	10	10	0.0690	00:31:15
05	18	13	0.0659	01:14:49
06	18	13	0.0666	01:15:53
07	100	14	0.0663	05:27:42
08	45	14	0.0679	02:25:38
09	45	14	0.0675	02:26:55
10	45	14	0.0717	02:18:58
11	45	14	0.0691	02:16:57
12	25	15	0.0745	01:26:11
13	25	15	0.0710	01:28:04
14	25	15	0.0765	01:30:24
15	25	15	0.0666	01:30:25
16	27	17	0.0724	01:49:05
17	27	17	0.0657	01:30:53
18	27	17	0.0748	01:40:17
19	20	20	0.0741	02:11:05
20	20	20	0.0666	04:11:49
21	20	20	0.0733	02:08:06
22	20	20	0.0656	02:59:22
23	12	23	0.0663	01:32:07
24	12	23	0.0663	01:28:43
25	12	23	0.0781	01:29:23

Figure 8 shows the membership functions of the FLC obtained by S-ACO algorithm. Figure 9 shows the desired trajectory and obtained trajectory.

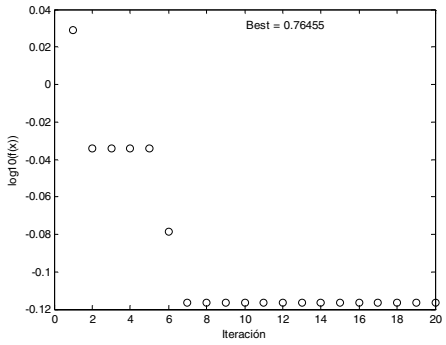
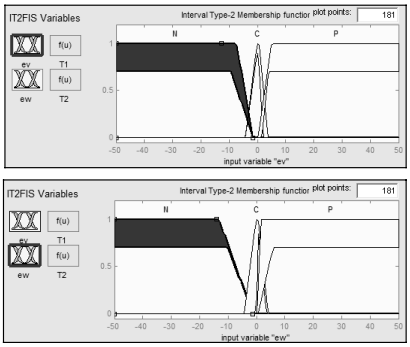


Fig. 7. Optimization Behavior for the S-ACO on type-2 FLC Optimization.



(a)

(b)

Fig. 8. Membership Functions: (a) Linear velocity error, (b) angular velocity error optimized by S-ACO algorithm.

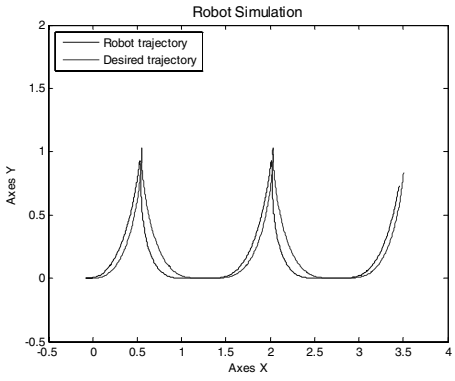


Fig. 6. Obtained trajectory with type-2 FLC optimization

C. Comparison of S-ACO and GA

Table 7 shows the comparison between the best result obtained by the S-ACO algorithm and the best result obtained the GA algorithm in type-1 FLC optimization and type-2FLC optimization (See Table 5 and Table 6 in [13] for more details about the GA results). The average error is significantly lower, although computer time is larger for type-1 FLC optimization and for type-2 FLC optimization it is clear that S-ACO outperformed GA.

TABLE 7. COMPARISON OF S-ACO ALGORITIHM AND GA ALGORITIHM FOR TYPE-1 FLC OPTIMIZATION.

S-ACO Type-1 FLC Optimization	Iterations	Ants	Average Error	Time
	50	80	0.0982	00:29:22
GA Type-1 FLC Optimization	Generations	Ind.	Average Error	Time
	26	30	0.4082	00:06:40
S-ACO Type-2 FLC Optimization	Iterations	Ants	Average Error	Time
	20	20	0.0656	02:59:22
GA Type-2 FLC Optimization	Generations	Ind.	Average Error	Time
	42	35	0.3989	07:11:52

VII. CONCLUSION

A trajectory tracking controller has been designed based on the dynamics and kinematics of the autonomous mobile robot through the application of ACO for the optimization of membership functions of the type-1 and type-2 fuzzy logic controller with good results obtained after simulations. As future work, a PSO algorithm will also be applied to the optimization of type-1 and type-2 FLCs for the autonomous robot. After this, a statistical comparison among ACO, PSO and GA in the optimization of the type-2 FLCs will be made.

REFERENCES

[1] A.M. Bloch, Nonholonomic mechanics and control, Springer Verlag, New York, 2003.

[2] R.W. Brockett, Asymptotic stability and feedback stabilization. In: R.S. Millman and H.J. Susman (Eds.) , Diferential Geometric Control Theory, Birkhauser Boston, 1983, pp. 181 -181.

[3] O. Castillo, P. Melin, Soft Computing for Control of Non-Linear Dynamical Systems, Springer-Verlag, Helderberg, Germany, 2001.

[4] M. Dorigo., M. Birattari, and T. Stützle., "Ant Colony Optimization", IEEE Computational Intelligence Magazine, November 2006, pp. 28-39.

[5] M. Dorigo, T. Stützle, Ant Colony Optimization, Bradford, Cambridge, Massachusetts, 2004.

[6] K. Duc Do, Zhong-Ping and J. Pan, "A global output-feedback controller for simultaneous tracking and stabilizations of unicycle-type mobile robots", IEEE Trans. Automat. Contr., col. 20, issue 3, 2004, pp. 589-594.

[7] A.P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, Wiley, J England, 2005.

[8] T-C Lee, K-T. Song, C-H. Lee and C-C. Teng, "Tracking control of unicycle-modeled mobile robot using a saturation feedback controller.", IEEE trans. Contr. Syst. Technol., vol. 9, issue 2, 2001, pp. 305-318.

[9] D. Liberzon, Switching in Systems and control, Bikhauser, 2003

[10] R. Martínez, O. Castillo and L. Aguilar, "Intelligent control for a perturbed autonomous wheeled mobile robot using type-2 fuzzy logic and genetic algorithms.", JAMRIS, Vol. 2, n°1, p1-11, 2008.

[11] Porta-García M., Montiel O. and Sepulveda R., "An ACO path planner using a FIS for a Path Selection Adjusted with a Simple Tuning Algorithm.", JAMRIS, Vol. 2, n°1, p1-11, 2008.

[12] R. Sepúlveda, O. Castillo, P. Melin, and O. Montiel, An Efficient Computational Method to Implement Type-2 Fuzzy Logic In control Applications, Analysis and Design of Intelligent Systems Using Soft Computing Techniques, Advances in Soft Computing, vol. 41, June 2007, pp. 45-52.

[13] R. Martínez, O. Castillo, L. Aguilar, "Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms", Information Sciences 179, 2158–2174, 2009.