Applied Soft Computing 9 (2009) 1102–1110

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

# Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation

M.A. Porta Garcia [a], Oscar Montiel [a], Oscar Castillo [b,*], Roberto Sepúlveda [a], Patricia Melin [b]

[a] Centro de Investigación y Desarrollo de Tecnología Digital del IPN (CITEDI), Av. del Parque 1310, Tijuana, B.C., Mexico
[b] Department of Computing Science, Tijuana Institute of Technology, P.O. Box 4207, Chula Vista, CA, 91909, USA

ARTICLE INFO

*Article history:*
Received 4 September 2007
Received in revised form 24 April 2008
Accepted 26 February 2009
Available online 11 March 2009

*Keywords:*
Ant colony optimization
Autonomous mobile robot navigation
Fuzzy Logic
Path planning
Simple tuning algorithm

ABSTRACT

In the Motion Planning research field, heuristic methods have demonstrated to outperform classical approaches gaining popularity in the last 35 years. Several ideas have been proposed to overcome the complex nature of this NP-Complete problem. Ant Colony Optimization algorithms are heuristic methods that have been successfully used to deal with this kind of problems. This paper presents a novel proposal to solve the problem of path planning for mobile robots based on Simple Ant Colony Optimization Meta-Heuristic (SACO-MH). The new method was named SACOdm, where **d** stands for distance and **m** for memory. In SACOdm, the decision making process is influenced by the existing distance between the source and target nodes; moreover the ants can remember the visited nodes. The new added features give a speed up around 10 in many cases. The selection of the optimal path relies in the criterion of a Fuzzy Inference System, which is adjusted using a Simple Tuning Algorithm. The path planner application has two operating modes, one is for virtual environments, and the second one works with a real mobile robot using wireless communication. Both operating modes are global planners for plain terrain and support static and dynamic obstacle avoidance.



## 1. Introduction

Nowadays, robotics is an essential part in manufacturing processes automatization. Concerning mobile robots, autonomous navigation entails a great challenge. A mobile robot (MR) can be very useful in different situations where humans could be in danger or when they are not able to reach certain targets because of terrain conditions. Then, mobile robotics field is an interesting and challenging subject for science and engineering, and it has many different approaches [12].

The research field of Robot Motion Planning was launched at the middle of the 1960s; however, after the publication of Lozano-Pérez [14] in 1979 the interest in this area grew. After 30 years, the existing research can be widely classified in two main approaches: classical [2] and heuristic. Classical methods dominated this field during the first 20 years; roughly speaking most of them were variations and/or combinations of Potential fields, Roadmaps, Cell Decomposition, and Mathematical programming. Heuristic methods interest were born when scientists confronted many of the drawbacks of classical approaches since the NP-completeness nature of Motion Planning problems. Representative methodologies in the heuristic classification are the Probabilistic Roadmaps, Rapidly Exploring Random Trees, Neural Networks, Genetic Algorithms [1,7,11], Simulated Annealing, Ant Colony Optimization, Particle Swarm Optimizer [3,6], Stigmergy, Wavelets, Tabu Search, and Fuzzy Logic [21]. A general review of the major contributions to the MP field that covers classical and heuristic approaches through a 35-year period is given in [15]; there were surveys of around 1400 papers and cited 82 representative works.

Actual reports [17,22] show that the interest in ant-based algorithm meta heuristics is growing in mobile robotics. ACO-MH is inspired in the foraging behavior of real ants for finding the optimal path from the nest to where the food is. Some ant species, as well as other social insects use an indirect communication method known as stigmergy; this is a concept introduced by the French biologist Pierre-Paul Grassé in 1959. With stigmergy, each ant communicates with another one by modifying their local environment. The ants achieve this task by laying down pheromone along their trails [4]. ACO-MH solves mainly combinatorial optimization problems defined over discrete search spaces. The ant-based algorithms developed as a result of studies of ant colonies are referred as instances of ACO-MH [6].

This work presents a new proposal to solve the problem of path planning for mobile robots; it is based in Ant Colony Optimization

* Corresponding author.
*E-mail addresses:* maporta@gmail.com (M.A. Porta Garcia), o.montiel@ieee.org (O. Montiel), ocastillo@hafsamx.org (O. Castillo), r.sepulveda@ieee.org (R. Sepúlveda), epmelin@hafsamx.org (P. Melin).

1568-4946/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2009.02.014

Meta-Heuristic (ACO-MH) to find the best route according to certain cost functions; the novel method was named SACOmd, where **d** stands for distance, and **m** for memory. One of the main contributions of this work is the inclusion of memory capabilities to the ants in order to avoid stagnation; moreover, the algorithm is able to influence the decision making process based on the exciting distance between the source and target nodes. An other contribution is the use of the fuzzy cost function to evaluate the best routes as well as the use of the method to easily control the performance of the Fuzzy Inference System (FIS).

From the papers surveyed, there was not found any similar proposal. However, there are other works sharing similar ideas in the sense of using ACO-MH as a global planner; for example, in [16] is presented a hybrid model that combines ACO and Artificial Potential Fields (APF) algorithm. In [23] was proposed a new meta-heuristic method of ACO to solve the vehicle routing problem, using a multiple ant colony technique where each colony works separately. In [10] the robot has to visit multiple targets, like the traveling salesman problem but with the presence of obstacles, the robot in this case is modeled as a point robot; that is, the robot occupies an exact cell in the discrete representation of the workspace, using several robots as ants; this robot team architecture has to be in constant communication with each other at all times to share pheromone information.

This paper is organized as follows. Section 2 introduces the reader to the MR field in order to present the general structure of this proposal. In Section 3 the ACO algorithm is explained in the context of mobile robotics. In Section 4 the new proposal of ACO-MH is presented to solve the path planning problem, the specific characteristics of workspace are given, as well as the fuzzy cost function used and the method to tune it; moreover the dynamic characteristic to avoid obstacles is given. The experimental results are given in Section 5. Finally, in Section 6 are the conclusions.

## 2. Navigation architecture proposal

Navigation in mobile robotic ambit is a methodology that allows to guide an MR to accomplish a mission through an environment with obstacles in a good and safe way. The two basic tasks involved in navigation are the environment perception, and path following. The concept of mission, refers to the realization of a set of navigation and operation goals; in this sense, the MR should possess an architecture able to coordinate the on board elements: sensorial system, movement and operation control, in order to achieve correctly the different objectives specified in the mission with efficiency that can be carried out either in indoor or outdoor environments. Generally global planning methods complemented with local methods are used for indoor missions since the environments are known or partially known; for outdoor applications, local planning methods are more suitable, becoming global planning methods a complement because of the scant information of the environment.

The navigation problem of an MR can be divided in four subproblems [20]:

- **World perception.** It senses the world symbolizing it into features.
- **Path planning.** Uses the features to create an ordered sequence of objective points that the robot must attain.
- **Path generation.** The goal is to obtain a path through the sequence of objective points.
- **Path tracking.** It is in charge of controlling that the MR follows a path.

It is common that "path planning" and "path generation" are referred just as "path planning", because some navigation schemes compute the safer instant motion of the vehicle as "path planning", instead of generating a path.

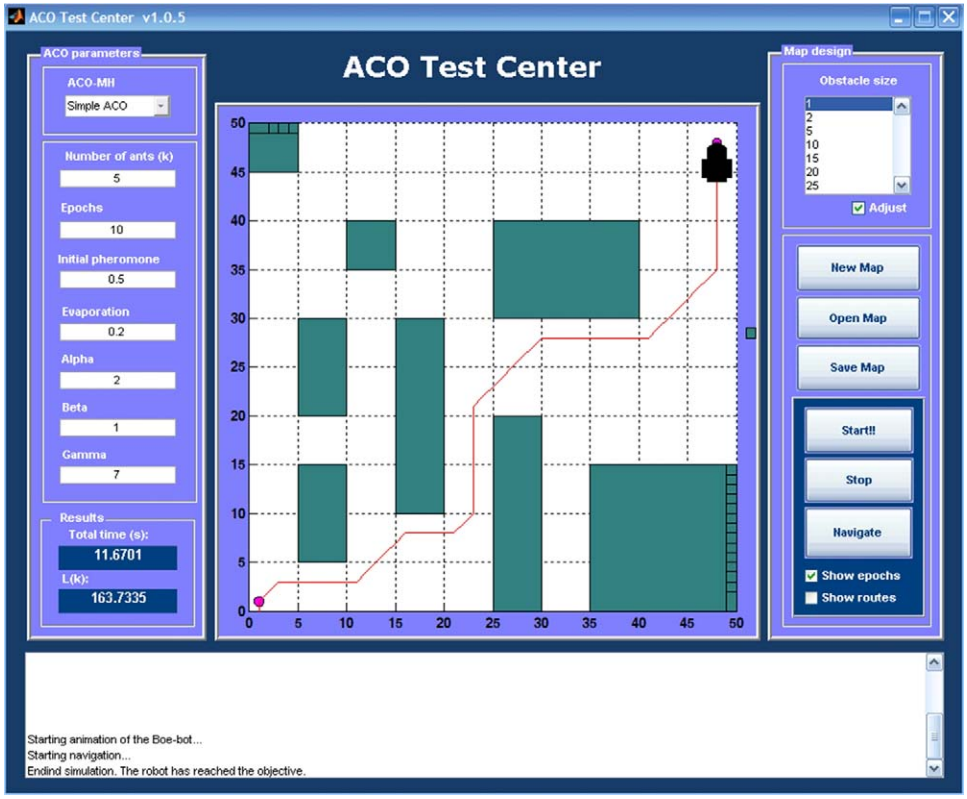


**Fig. 1.** Main screen of the software interface.

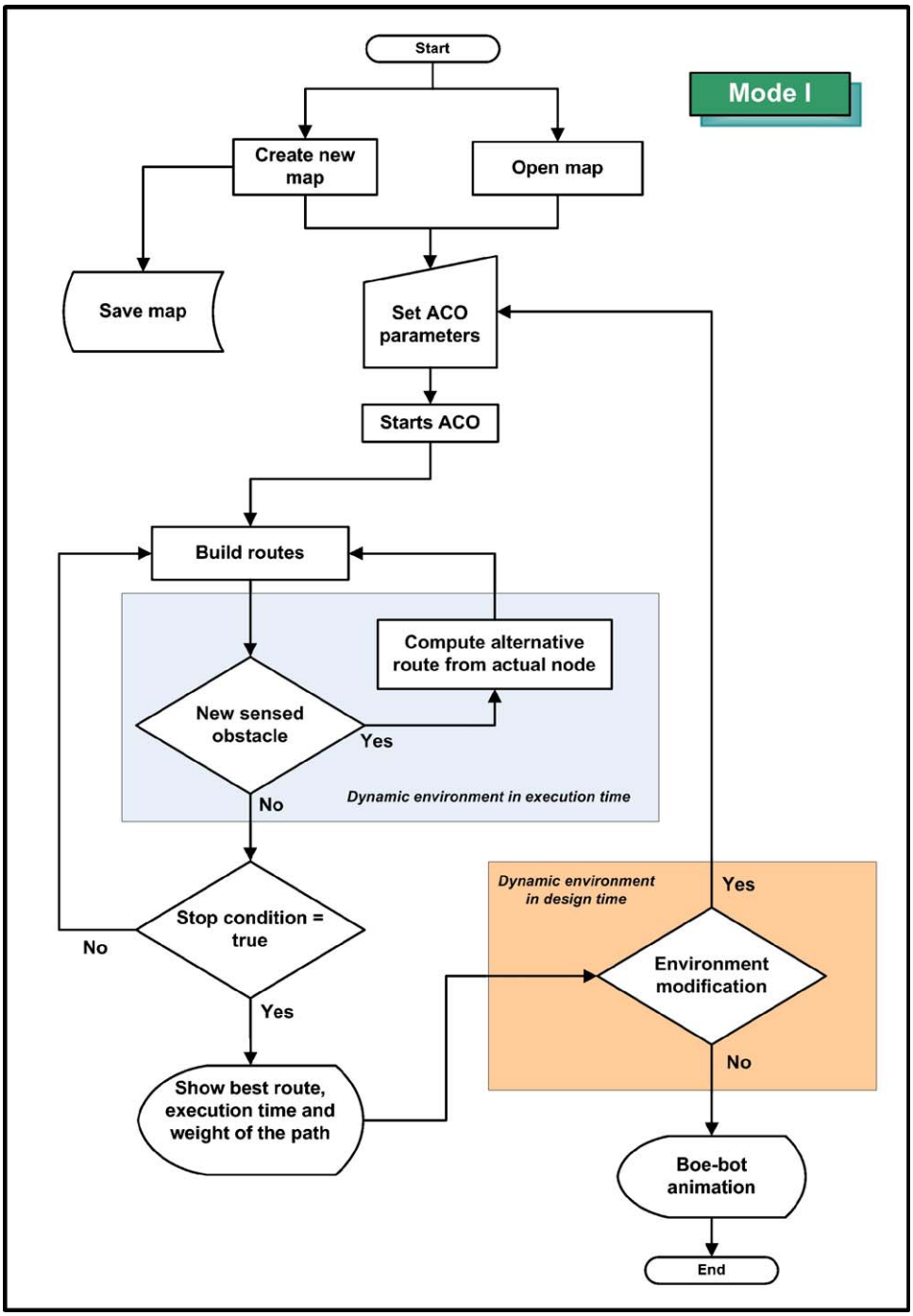


**Fig. 2.** Flow diagram of the framework.

The Navigation solution presented in this work is able to handle the four subproblems aforementioned. Fig. 1 shows the main screen of the ACO Test Center (ACOTC); as it is called this development, it allows to perform path planning with SACO and SACOdm. A graphical interface, as well as the translation of the obtained solution (optimal path) from the virtual to the real world, was implemented to test the proposed methods. The ACOTC has two operational modes: Mode I for virtual environments, allows the user to design maps for indoor conditions over plain terrain, the relational diagram for this mode is given in Fig. 2; Mode II is for on-line navigation of the real MR, in this case the Boe-Bot from Parallax [13,19], its relational diagram is given in Fig. 3.

Mode I is the default operational mode; in this mode, the user can design maps for two dimensions (2D) indoor applications (plain terrain), or a map of real terrain (2D) can be uploaded for virtual testing of the route planning algorithm.

Mode II is selected by pressing the *Navigate* Button, the first task of this mode is to establish communication via Bluetooth between ACOTC and the MR. This mode is the on-line global path planner with dynamic obstacles avoidance. The ACOTC will sent the different objective points (coordinates $(x, y)$) of the optimal path to the MR, in order to achieve coordinated control for tracking the desired path. The MR sensorial on-board equipment will inform to ACOTC whether a new obstacle has appear; if so, the global path

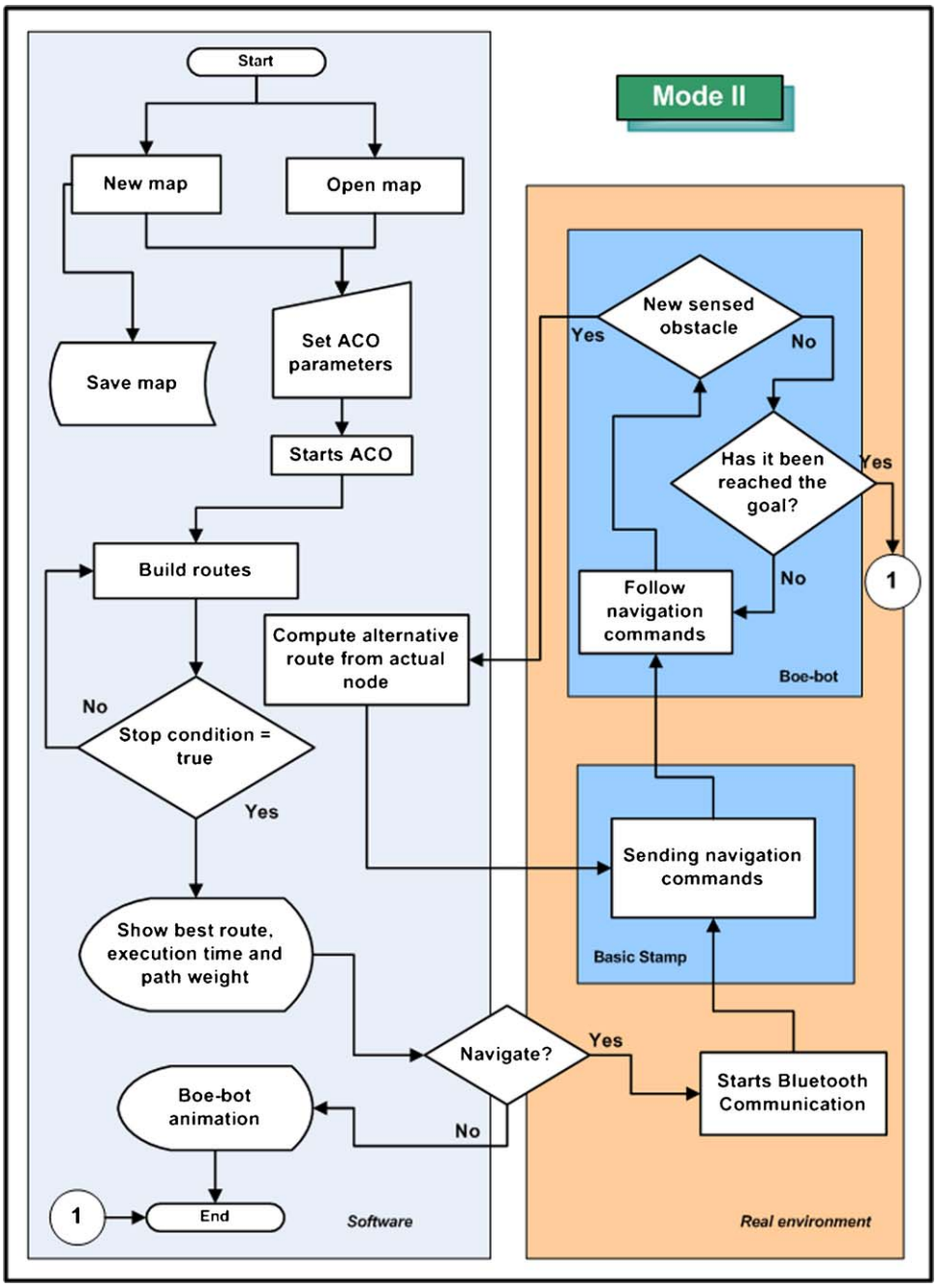


**Fig. 3.** Flow diagram of the framework.

planner will generate a new path starting from the actual MR position.

## 3. Simple ant colony optimization algorithm

The simple ant colony optimization algorithm (SACO) is an ACO-MH implementation that adapts the behavior of real ants to solution of minimum cost path problems on graphs. A number of artificial ants build solutions for the optimization problem by issuing and exchanging information about the quality of these solutions making allusion to the communication system of the real ants [4,5].

Considering an "ant" as a punctual mobile robot MR in 2D, an specification of the robot position in relation to a fixed coordinate system is called a configuration $q$ and it is given by (1), where $p = (x, y)$ is the MR position, and $\theta$ is the orientation,

$$q = (p, \theta) = (x, y, \theta) \tag{1}$$

The set of all the feasible values of $q$ is the Configuration Space $CS$. If the MR is no punctual, it will take up a subspace from CS, hence the MR can be modeled by a circle with radius $\varphi$, and center $p = (x, y)$ in the Cartesian space. The subset of $CS$ that a real robot takes up is defined as $R(q)$,

$$R(q) = \{q_i \in CS / \|q, q_i\| \leq \varphi\} \tag{2}$$

A punctual obstacle in the environment is an object represented by $b_i$, and a set of obstacles is $B$,

$$B = \{b_1, b_2, b_n\} \tag{3}$$

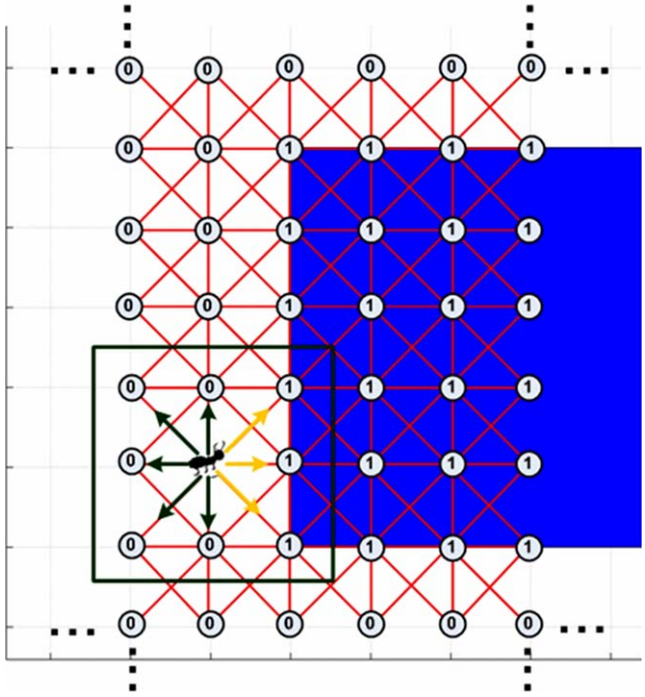

**Fig. 4.** The workspace has been discretized in a matrix of $50 \times 50$ nodes interconnected. The blue box is an obstacle, an ant can have until eight options, only the options with a value of "0" are eligible (green arrows), and the nodes with a value of "1" are not eligible (yellow arrows). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of the article.)

the subset of $CS$ that an obstacle occupies is $b_i(q)$, in such a way that the Configuration Space free of any obstacle is defined as $CS_{\text{free}}$ in (4). Hence, $R(q) \cap B(q) = \varnothing$ for $CS_{\text{free}}$, and $R(q) \cap B(q) \neq \varnothing$ for a $CS$ with forbidden subspace regions because the presence of obstacles.

$$CS_{\text{free}} = \left\{ q \in CS / R(q) \cap \left( \bigcup_{i=1}^{q} b_i(q) \right) = \varnothing \right\} \tag{4}$$

The path planning problem is the search of a succession of punctual configurations $q$ in $CS_{\text{free}}$ called $Q$, such as $q_a$ is the actual configuration, and $q_f$ is the final configuration, this is expressed by (5),

$$Q = \{q_a, \ldots, q_f | q_i \in CS_{\text{free}}\} \tag{5}$$

The paths obtained with (5) are not all feasible, since this expression considers punctual MR; so, $R(q)$ might not fit in many configurations of a real MR. This proposal considers "ants" of size $R(q)$, this consideration will reduce the search space considerably, since the size of cells can be chosen adequately according the MR size. The new expression that considers a configuration of a real robots $R(q)$ instead $q$ can be rewritten as (6),

$$Q(R(q)) = \{R(q_a), \ldots, R(q_f) | R(q_i) \in CS_{\text{free}}\} \tag{6}$$

If $x^k(t)$ denotes a $Q(R(q))$ solution in time $t$, $f(x^k(t))$ expresses the quality of the solution. In general terms, the steps of SACO are as follows:

(1) Each link $(i, j)$ is associated with a pheromone concentration denoted as $\tau_{ij}$.

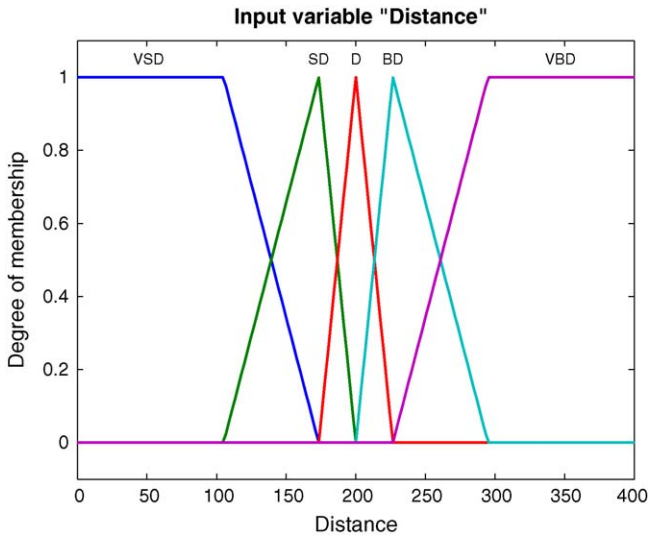


**Fig. 6.** Membership functions of the Distance input once they had been tuned with the STA, the tuning factor is $k = 0.75$.

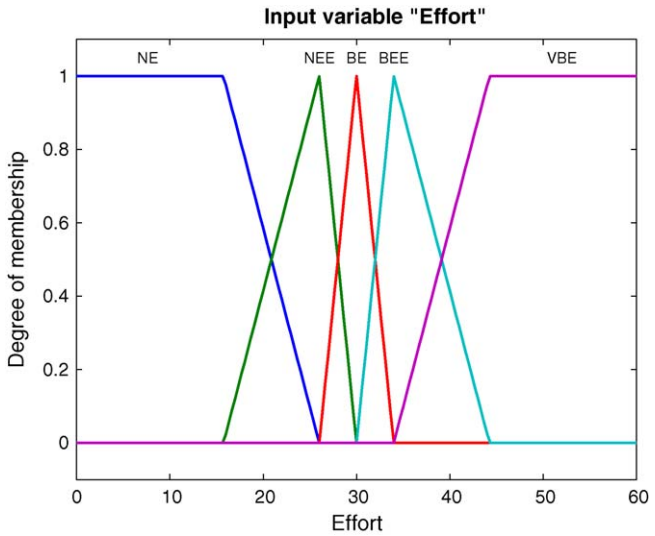


**Fig. 5.** Membership functions of the Effort input once they had been tuned with the STA, the tuning factor is $k = 0.75$.

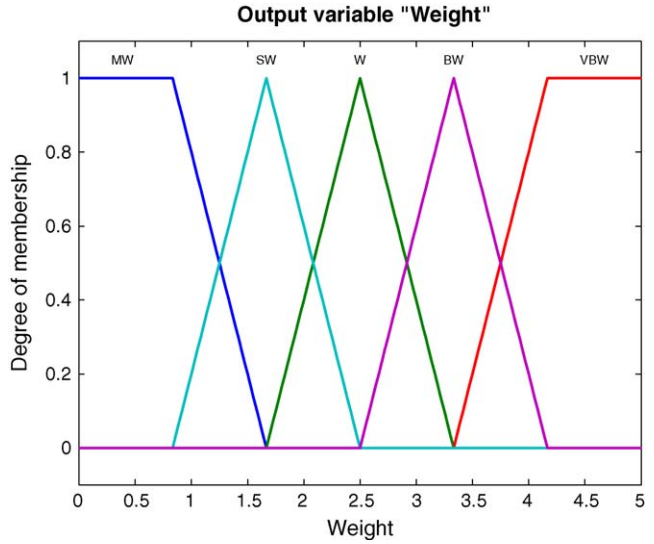


**Fig. 7.** Membership functions of the Weight output. The STA does not require to modify the membership functions parameters of the output.

(2) A number $k = 1, \ldots, n_k$ are placed in the origin node (the nest).
(3) On each iteration or epoch all ants build a path to the destiny node (the food source). For the next node selection it is used the probabilistic formula:

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^\alpha(t)}{\sum_{j \in N_i^k} \tau_{ij}^\alpha(t)} & \text{if } j \in N_i^k \\ 0 & \text{if } j \notin N_i^k \end{cases} \quad (7)$$

In Eq. (7), $N_i^k$ is the set of feasible nodes connected to node $i$ with respect to ant $k$; $\tau_{ij}^\alpha$ is the total pheromone concentration of link $(i, j)$, where $\alpha$ is a positive constant used as gain for the pheromone concentration influence.
(4) 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 node $i$ and ant $k$, $N_i^k = \varnothing$; then predecessor of that node $i$ is included as a former node of the path.
(5) Compute pheromone evaporation using the Eq. (8).

$$\tau_{ij}(t) \leftarrow (1 - \rho)\tau_{ij}(t) \quad (8)$$

In Eq. (8), $\rho$ 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. For $\rho = 1$, the search is completely random. While an ant takes more time for crossing a path, there is more time for the pheromone trail to evaporate. On a short path, which is crossed quickly, the density of the pheromone is higher. Evaporation avoids convergence to local optimums. Without evaporation, the paths generated by the first ants would be excessively attractive for the subsequent ones. In this way, exploration of the search space is not too restricted.
(6) Update pheromone concentration by using Eq. (9).

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \sum_{k=1}^{n_k} \Delta\tau_{ij}^k(t) \quad (9)$$

(7) The algorithm can be ended in three different ways:
a. When a maximum number of epochs have been reached.
b. When it has been found an acceptable solution, with $f(xk(t)) < \epsilon$.
c. When all ants follow the same path.

## 4. SACOdm proposal

Several aspects has been considered to improve the SACO algorithm for MR applications. The original transition formula (7) was modify to accelerate the decision process. The strength of this improvement is better appreciated in free space path optimization. This addition works as follows: $\xi$ is the Euclidian distance between the source and target nodes, and $\beta$ is a value that amplifies the influence of $\xi$, the valid range of $\beta$ is $[0, \infty)$. The new transition formula is (10)

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^\alpha(t)}{\sum_{j \in N_i^k} \xi^\beta \tau_{ij}^\alpha(t)} & \text{if } j \in N_i^k \\ 0 & \text{if } j \notin N_i^k \end{cases} \quad (10)$$

In addition, a memory capability was added to avoid the algorithm stagnation, this capacity is represented by $\gamma$ and basically it is a reference value. A counter keeps track of the already visited nodes by marking them with "1" in the workspace temporally, as if they were obstacles; this is with the intention of avoiding testing again the memorized nodes, once the algorithm reached the $\gamma$ value, the nodes are available for retesting by removing the temporal marks.

For the case of path planning, the algorithm includes a proposal of using a fuzzy cost function based in heuristic knowledge that can be easily adjusted to improve performance using the Simple Tuning Algorithm (STA) [9].

### 4.1. The workspace

The map where the mobile robot navigates is a search space discretized into a matrix representing a graph of $50 \times 50$nodes, where "0" means a feasible node (plain terrain) and "1" are obstacles, see Fig. 4. It is remarkable to say that each artificial ant of the algorithm is a scale representation of the real MR, which means the proposed method considers robot's dimensions; for example, there are going to be situations during the optimization process, where some paths are rejected if the robot does not fit in the space between two obstacles. Under this premise, several computations are saved since some nodes are rejected before the algorithm spends time using them to build paths. The $50 \times 50$ map represents a 4 $m^2$area, in a 1:4 scale (cm).

For this method, it is assumed all nodes are interconnected. In a map with no obstacles, there are 2500 feasible nodes; therefore the matrix of links $E$ would be extremely large. For this reason $E$ is not used, and the pheromone amount value is assigned at each node, which reduces considerably the complexity of the algorithm and then the processing time. This is equivalent to assign the same pheromone concentration to the eight links around every node. If an analogy with reality is made, this can be seen as ants leaving food traces in each node they are visiting, instead of a pheromone trail on the links.

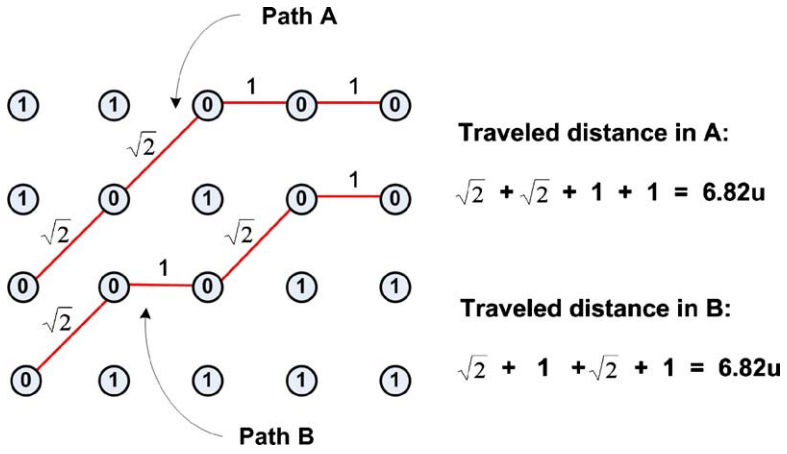


**Fig. 8.** Path A and Path B have the same distance; however path A implies less effort for robot navigation.

Once the ants are placed in the origin node, each ant starts navigating, and the decision process for choosing the next node consist in a $3 \times 3$ window of the whole graph. The ant can choose one of the eight nodes around it using the transition probability formula (10).

### 4.2. The fuzzy cost function

The cost of the path $f(x^k(t))$ to determine the optimal one is evaluated by a Fuzzy Inference System (FIS), which contemplates not only the length of the path but the difficulty for the navigation. The FIS considers two inputs: *Effort* (Fig. 5), and *Distance* (Fig. 6). The first one represents the energy spent by the robot to make turns across the path; for example, the effort become increased if the robot has to make a left turn after a long straight line, because it has to decelerate more; *Distance* is the accumulated Euclidean distance at the moment between the visited nodes. The output, shown in Fig. 7, is a weight assigned to the cost of the path; the more weight is given, the less desirable becomes the path. The output of the FIS is added to the total Euclidean distance of the path, giving the final weight of each one generated by ants. If there are different routes with the same length, the FIS should make a difference of cost giving preference to the straighter paths like those shown in Fig. 8. The FIS variables can be seen in Table 1; and the FIS rule matrix in Table 2.

### 4.3. Tuning of the fuzzy cost function

The FIS can be tuned for a better performance using the Simple Tuning Algorithm (STA) proposed in [8,9], it is applied to facilitate the tuning process of the FIS, since sometimes becomes overwhelming to find the optimal parameters necessaries for a well performance of the fuzzy system. By applying the STA, time and effort are reduced by using a single parameter, the tuning factor $k$. It is based on the properties of the fuzzy surface, allowing the modification of the FIS behavior by means of manipulating the ranges of the membership functions of the input variables, remaining without any modification the output membership functions [8,18]. In this work, the FIS was used as a decision support system to differ the straighter paths from the winding ones. The output surface without applying STA is shown in Fig. 9, and after applying the STA in Fig. 10.

**Table 1**
The FIS has two input variables, Effort and Distance.

| Input variables | | Output variable |
|---|---|---|
| Effort | Distance | Weight |
| NE: Normal Effort | VSD: Very Small Distance | MW: Minimum Weight |
| NEE: Normal Extra Effort | SD: Small Distance | SW: Small Weight |
| BE: Big Effort | D: Distance | W: Weight |
| BEE: Big Extra Effort | BD: Big Distance | BW: Big Weight |
| VBE: Very Big Effort | VBD: Very Big Distance | VBW: Very Big Weight |

The output variable is Weight.

**Table 2**
Fuzzy rule matrix.

| | | Distance | | | | |
|---|---|---|---|---|---|---|
| | | VSD | SD | D | BD | VBD |
| Effort | NE | MW | MW | SW | SW | MW |
| | NEE | MW | SW | W | SW | MW |
| | BE | SW | W | W | W | SW |
| | BEE | BW | BW | W | BW | VBW |
| | VBE | VBW | VBW | BW | VBW | VBW |

There are 25 rules for the two input variables.

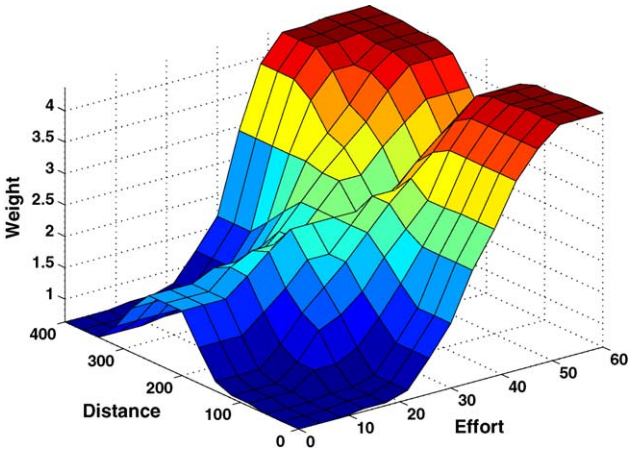


**Fig. 9.** Surface of the FIS output before applying the STA.

The STA method basically consists of four steps:

(1) **Tuning factor selection.** A number $k \in [0, 1]$ is used to define the tuning adjustment level. $k = 0$ is the biggest settling time and $k = 1$ the smallest.
(2) **Normalization of the ranges of the fuzzy controller's variables.** The range of each input fuzzy variable is modified in order to have the lower and upper limits equal to $-1$ and $+1$, respectively.
(3) **Tuning factor processing.** Once the range is normalized, the new vector of operation points will be given by:

$$\mathrm{Vop}_{\mathrm{final}} = (\mathrm{Vop}_{\mathrm{initial}})^{r(k)} \tag{11}$$

where $\mathrm{Vop}_{\mathrm{initial}}$ is a vector with normalized values of the membership in the $x$-axis and $r(k)$ is the polynomial:

$$r(k) = \frac{30k^3 + 37k^2 + 52k + 1}{40} \tag{12}$$

(4) **Renormalization of the ranges of the fuzzy variables.** Convert the normalized range to the previous range of the system. This can be computed multiplying the vector by a constant factor.

### 4.4. Dynamic obstacles avoidance

The algorithm has the capability of sensing changes in the environment, if a new obstacle is placed over the robot's route at

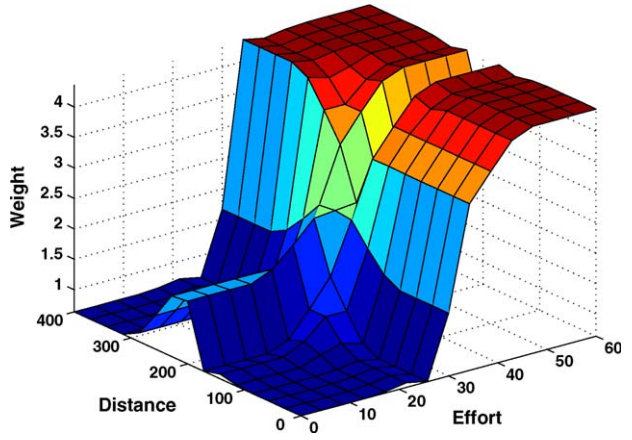


**Fig. 10.** Surface of the FIS output after applying the STA. It was used an adjusting factor $k = 0.75$.

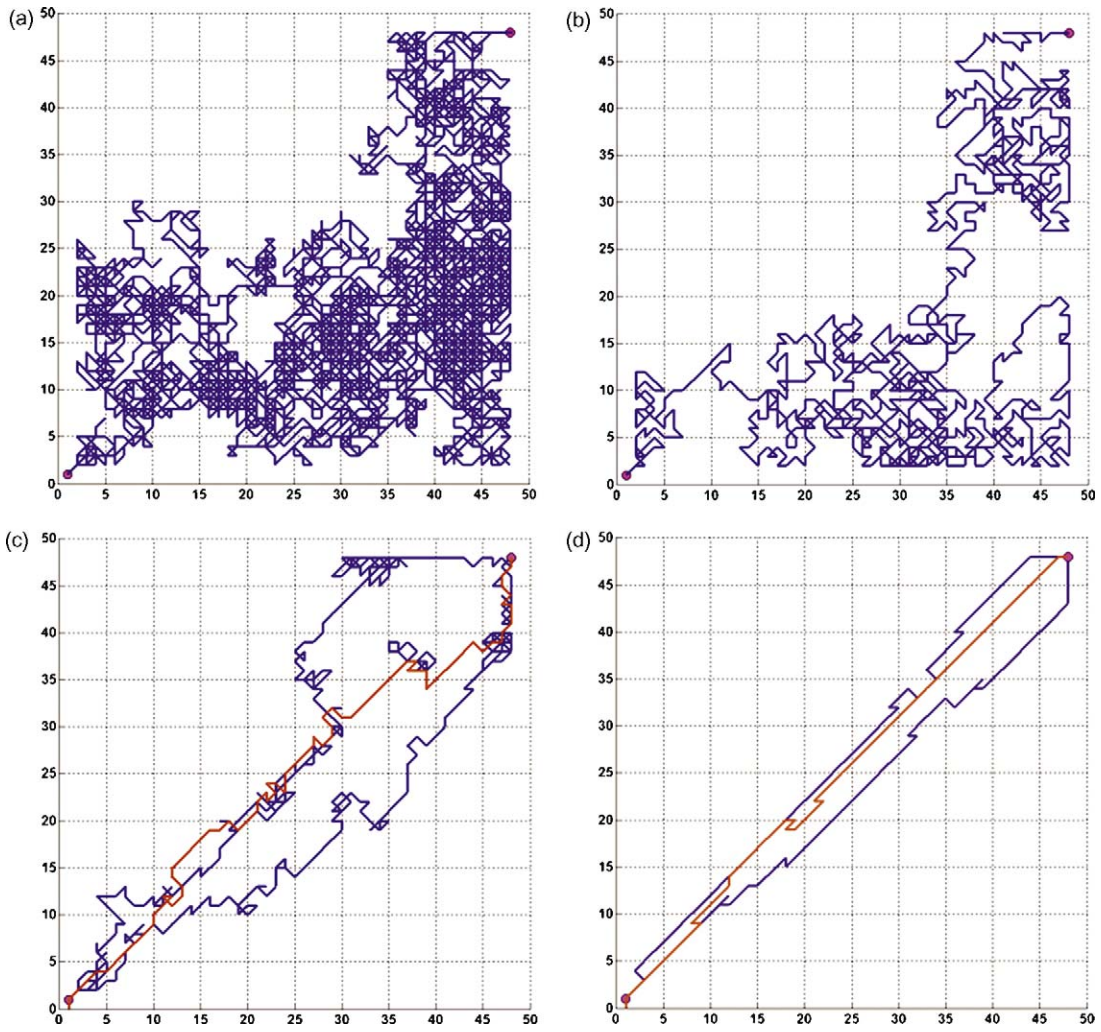


**Fig. 11.** (a) Route generated by the first ant in the first epoch, with $\beta = 0$ and $\gamma = 1$, (b) same situation but with $\beta = 0$ and $\gamma = 100$, (c) The routes of three ants in the first epoch with $\beta = 0.1$ and $\gamma = 1$, (d) same situation but with $\beta = 0.5$ and $\gamma = 1$.

time $t$, it starts a rerouting process in order to avoid the blocking object and get to the destiny node. It has to be considered that after some epochs, the pheromone concentration $\tau_{ij}$ is already increased over the visited nodes; then, when a new obstruction appears, it causes evaporation of the pheromone trail around it. This premise prevents stagnation around the obstacle, and $\tau_{ij}$ of the surrounding area is given by the minimum pheromone value over the search map at $t$.

## 5. Experimental results

Several comparative experiments were conducted to evaluate the new proposed features of SACOdm over SACO. In all the experiments, the starting point is in (2,2), and the target point is in (48,48).

**Experiment 1.** Testing SACO in free space (no obstacles).

The ACOTC was programmed as follows: Ants amount = 3, Epochs = 10, Initial pheromone $(\tau_{ij}) = 0.5$, Evaporation $(\rho) = 0.2$, $\alpha = 2$. Since SACO does not consider $\beta$ and $\gamma$ values, a "0" was given for each variable. We took statistic values of 20 runs. The mean time was 53.99 s with a standard deviation of 19.55. The minimal time to obtain the route was 25.42 s, the maximal time was 91.30 s. The minimal route cost was 97.97 for this route, the maximal cost was 100.41, the mean costs was 98.53 with a standard deviation of 0.73. Eight times of 20 the best route was found. If we let that the algorithm run one or two more epochs, the best route will be found always.

**Experiment 2.** Testing SACOdm in free space.

The benefits of modifying Eq. (10) will be tested. The ACOTC was programmed as follows: Ants amount = 3, Epochs = 10, $\tau_{ij} = 0.5$, $\rho = 0.2$, $\alpha = 2$, $\beta = 1$ and $\gamma = 0$. For 20 runs, the mean time was 4.95 s with a standard deviation of 0.13; the minimal and maximal time were 4.48 and 5.1 s, respectively. The minimal route cost was 97.97 all the times, so the best route was found always. For one "ant" and three epochs, the SACOdm only needed 0.60 s to find the route.

Experiments 1 and 2, considered a scenario with no obstacles, SACOdm was better than SACO finding the optimal route in this kind of scenarios, basically the addition of $\xi^{\beta}$ in Eq. (10) was the reason. Fig. 11 show four tests for the same problem with different parameters.

**Table 3**
In the three tests $\tau_{ij} = 0.5$.

| Exp. 3 | Ants ($k$) | Epochs | *rho* | $\alpha$ | $\beta$ | $\gamma$ | $t(s)$ | $L(k)$ |
|---|---|---|---|---|---|---|---|---|
| 3(a) | 3 | 20 | 0.2 | 2 | 0 | 0 | 88.0 | 127.3 |
| 3(b) | 3 | 15 | 0.2 | 2 | 1 | 7 | 12.14 | 127.3 |
| 3(c) | 3 | 5 | 0.5 | 2 | 1 | 1 | 12.08 | 127.3 |

Test 1 is for SACO, Tests 2 and 3 are for SACOdm.

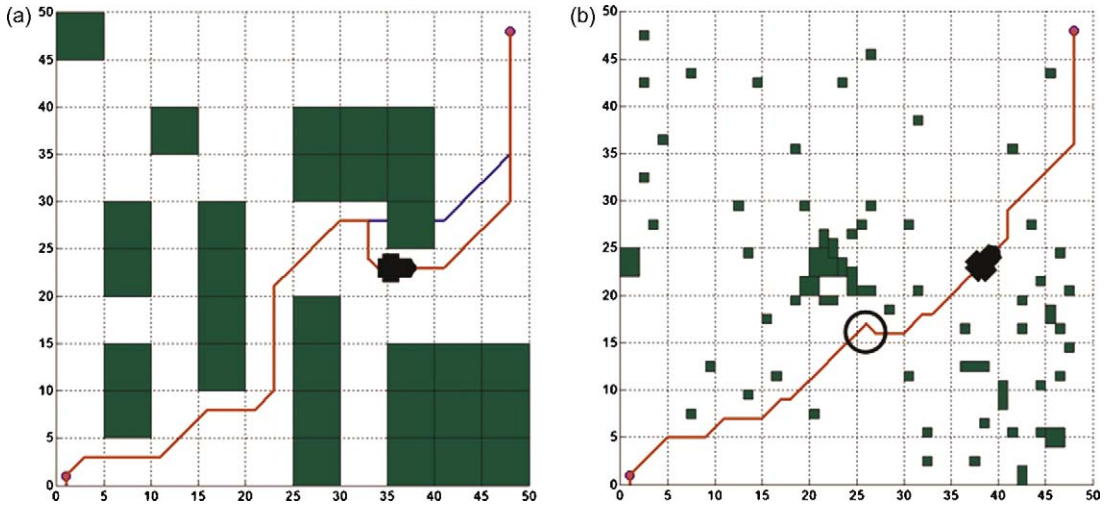


**Fig. 12.** (a) Alternative route given by the algorithm after a new obstacle appears at time $t$ and the Boe-Bot model following the modified path, (b) another search area, with major randomness in the obstacles positioning. The circle indicates a not desirable, but necessary movement in the path because an obstacle was added dynamically.

**Experiment 3.** Comparative table. SACO vs. SACOdm.

In Table 3 the results of several experiments using different parameters are given. The test was made using the map of Fig. 1.

**Experiment 4.** Dynamic obstacle avoidance.

In this example, it is shown how the obstacle is avoided after it appeared once the path was found and the MR was tracking the path. The algorithm searched again for the optimal path from the actual position. See Fig. 12.

## 6. Conclusions

The SACOdm proposed method seems to be a promising path planning system for autonomous mobile robot navigation since the given solutions are not only paths, but the optimal ones. It is remarkable to mention the reduced time of execution of SACOdm against SACO, approximately a 91% (speed up around 10). We tested several maps, there are some maps where both methods needed exhaustive computations that can be reduced changing parameter values. A system that automatically infers these values could be for future work; since SACOdm has some properties that works better in free spaces, and in many maps. However, SACO is more explorative because it is not biased by $\beta$ capability to choose determined nodes, but it is $\beta$ feature that makes faster SACOdm in free spaces. A solution to this problem is to begin the planning task using $\xi^{\beta}$ of SACOdm, then after a time, it is convenient to reduce its influence in SACOdm probabilistic transition equation; by doing this the algorithm will be able to solve maps very fast in free space, specially if there exist big diagonals in the route. The memory capability, $\gamma$, of SACOdm allows to remember some nodes to avoid them in a predetermined time; with this capability no problem was found storing few nodes, less than 100, but it needs further experimentation with different maps. One good feature of SACOdm is that it can be reduced to SACO just programming with "0's" the $\beta$ and $\gamma$ features. The use of a fuzzy cost function that depends on the effort and distance to evaluate the cost of a route worked fine, and we think it is a good idea because we are handling two objectives: the best route, and the effort, as it was a single objective problem.

## References

[1] H. Chen, Z. Xu, Path planning based on a new genetic algorithm, in: International Conference on Neural Networks and Brain, vol. 2, IEEEXplore, (2005), pp. 788–792.
[2] E. Chong, S. Zak, An Introduction to Optimization, Second Edition, Wiley-Interscience Series in Discrete Mathematics and Optimization, United States, 2001.
[3] M. Clerc, Particle Swarm Optimization, ISTE Publishing Company, 2006.
[4] M. Dorigo, M. Birattari, T. Stützle, Ant colony optimization, IEEE Computational Intelligence Magazine (2006) 28–39.
[5] M. Dorigo, T. Stützle, Ant Colony Optimization, MIT Press, England, 2004.
[6] A. Engelbrecht, Fundamentals of Computational Swarm Intelligence, Wiley, Englan, 2005.
[7] M. Gemeinder, M. Gerke, An active search algorithm extending ga based path planning for mobile robot systems, in: Soft Computing and Industry, Springer, Berlin, 2002, pp. 589–596.
[8] E. Gómez-Ramírez, A. Chavez-Plascencia, How to tune fuzzy controllers, in: Proceedings of FUZZ'04, Budapest, Hungary, (2004), pp. 1287–1292.
[9] E. Gómez-Ramírez, Simple Tuning of Fuzzy Controllers, Studies in Fuzziness and Soft Computing, Hybrid Intelligent Systms, Vol. 208, Springer, Berlin, Heidelberg, 2007, pp. 115–133.
[10] K. Gopalakrishnan, S. Ramakrishnan, Optimal Path Planning of Mobile Robot with Multiple Targets Using Ant Colony Optimization, Smart Systems Engineering, New York, 2006, 25–30.
[11] R. Haupt, S. Haupt, Practical Genetic Algorithms, Second Edition, Wiley, NJ, United States, 2004.
[12] J.M. Holland, Designign Autonomous Mobile Robots. Inside the Mind of an Intelligent Machine, Newnes, United States, 2004.
[13] A. Lindsay, Robotics with the Boe-Bot, Parallax Inc., United States, 2004.
[14] T. Lozano-Pérez, M.A. Wesley, An algorithm for planning collision-free paths among polyedral obstacles, Communications of ACM 2 (1979) 959–962.
[15] E. Masehian, D. Sedighizadeh, Classic and heuristic approaches in robot motion planning—a chronological review, in: Proceedings of World Academy of Science, Engineering and Technology, vol. 23, Germany, (2007), pp. 101–106.
[16] H. Mei, Y. Tian, L. Zu, A hybrid ant colony optimization algorithm for path planning of robo in dynamic environment, International Journal or Information Technology 12 (3) (2006) 78–88.
[17] M. Mohamad, W. Dunningan, Ant colony robot motion planning, in: The International Conference on Computer as a Tool, EUROCON, Serbia & Montenegro, (2005), pp. 213–216.
[18] O. Montiel, R. Sepúlveda, P. Melin, O. Castillo, M.A. Porta, I.M. Meza, Performance of a simple tuned fuzzy controller and a PID controller on a DC motor, in: Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence FOCI 2007, United States, (2007), pp. 531–537.
[19] Parallax Inc., Basic Stamp Syntax and Reference Manual, Version 2.2, (2006) ¡ http://www.parallax.com/dl/docs/prod/stamps/web-BSM-v2.2.pdf¿.
[20] D.H. Shin, S. Singh, Path generation for robot vehicles using composite clothoid segments, The Robotics Institute, Internal Report CMU-RI-TR-90-31, Carnegie-Mellon University, 1990.
[21] M. Tarokh, Path planning of rovers using fuzzy logic and genetic algorithm, in: World Automation Conference ISORA-026, United States, (2000), pp. 1–7.
[22] W. Ye, D. Ma, H. Fan, Path planning for space robot based on the self-adaptive ant colony algorithm, in: Proceedings of the 1st International Symposium on Systems and Control in Aerospace and Astronautics ISSCAA, IEEEXplore, (2006), p. 4.
[23] L. Zhishuo, C. Yueting, Sweep based multiple ant colonies algorithm for capacitated vehicle routing problem, in: IEEE International Conference on E-Business Engineering, China, (2005), pp. 387–394.