Low-cost fuzzy-based obstacle avoidance method for autonomous agents

Luis C. Gonzalez-Sua, Ivan Gonzalez, Leonardo Garrido, and Rogelio Soto

Tecnológico de Monterrey
Monterrey, NL, Mexico 64849
{lc.gonzalezsua,ni.gonzalez.phd.mty,
leonardo.garrido,rsoto}@itesm.mx
http://www.itesm.mx

Abstract. In this article, there lies an implementation of a novel fuzzy-based methodology that will reduce the amount of operations per iteration, reducing the computational cost of traditional fuzzy systems. This was achieved by implementing adaptive control theories like gain scheduling. Also, the method was implemented with a self-tuning technique using Genetic Algorithms (GA) that increases the method's efficiency, improving the following metrics: amount of collisions and time spent to finish the task. Two experiments were implemented to test the previously mentioned metrics; the first experiment will test the agent with several intersecting dynamic obstacles, in a long path. The second experiment will deal with two aggressive agents that will attempt to collide with the test agent. For comparison purposes, the Potential Fields' Method (PFM) was implemented and tested under the same metrics and experiments. The obtained results show for experiment 1 an improvement on the average collisions by 25.03% less compared to the PFM statistics and only and increase of 1.43% of the average time spent. And for experiment 2, there was an improvement of 93.46% of less average collisions compared to PFM and a reduction of 10.27% of average time spent. The contribution of this work is to implement a faster and less processing expensive method than traditional fuzzy ones.

Keywords: Fuzzy logic, Self-tuning, Autonomous agents, Obstacle avoidance, Potential fields

1 Introduction

Obstacle avoidance and navigation are closely together, however it is not the same thing, because navigation requires to create a path with several waypoints from a point A to a point B, but in the middle of the way they will meet some other points according to the terrain and some obstacles [4]. However, this article will focus on the obstacle avoidance part due to its importance to a safe navigation [7]. There has been plenty of research for obstacle avoidance [14], nonetheless, fuzzy has demonstrated to be quite a good choice [16], it has demonstrated to be a worthy option [11], and has demonstrated that in can be a

feasible option for obstacle avoidance [8]. Notwithstanding, there is the need for comparison, and potential fields has proven to be a worthy adversary [10]. The use of fuzzy for obstacle avoidance purposes gives a more human approach [9], but it has shown some drawbacks [2], highlighting the high number of required calculations and the necessity for an optimal tuning [15]. Withal the difficulties, fuzzy still can be use as a practical obstacle avoidance method [5]. In this article, the authors proposed two practical solutions to the previously mentioned drawbacks of fuzzy. To resolve the high number of calculations requirements, they establish a novel methodology based on the gain-scheduling theory and obtain a method that can maintain the fuzzy original structure, and also, can process several inputs without having an exponential increase in the number of calculation per iteration. The other drawback, as mentioned before, was the tuning issue. Implementing standard simple tuning algorithms [6] was not a viable solution due to the high level of uncertainties of the model, however, heuristic methods offers and alternative. Previous research has demonstrated that using genetic algorithms (GA) could be a feasible way to find a viable solution [13], [1].

Some recent work has demonstrated that by implementing GA with another optimization method like particle swarm optimization (PSO), can obtain better results [12], and that some other biology inspired methods like ant colony optimization (ACO) can also improve the performance [3]. Although, those methods requires a higher order magnitude (like 10 or 100 times bigger than implemented) and communication among the searching agents, but they excel in parallel simulations. Therefore, the singular case treated in this article was not able to take advantage of this features and GA was the best available option.

The following sections explains the proposed method and the implementation of GA to obtain an applicable option with an acceptable performance.

2 Methods

This section contains all the implemented methods and a brief explanation, commencing with the potential fields method, the fuzzy staged method and the optimisation by genetic algorithms.

2.1 Potential Fields Method (PFM)

The PFM has proven to be a feasible method for obstacle avoidance purposes, due to the effectiveness and relatively easy implementation, this method is by far one of the most used methods [17]. The easy implementation of the PFM is due to the simplicity of the algorithm, it calculates a resulting vector based on the distance and direction to goal and obstacles, the magnitude of the vector is given by a particular formula, for this article, an inverse equation was implemented as the equation (1) shows.

$$f(x) = \frac{a}{dist - b} + c \tag{1}$$

Where a, b and c are constants that modify the behaviour of the graph and *dist* is the distance between the agent and the goal/obstacle.The implemented PFM uses the following values for the goal: $a = -20$, $b = -0.9$ and $c = -0.5$, and for the obstacles $a = 12.5$, $b = -0.8$ and $c = -1.5$. The figure 1 shows the graphs for the behaviour towards the goal and the obstacles.

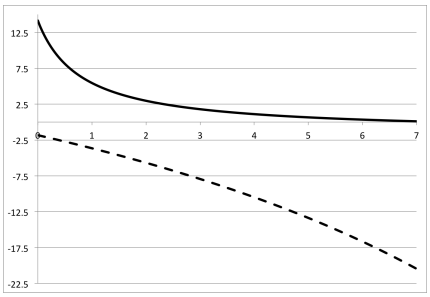


Fig. 1. Graphs for potential fields equations. The solid line shows the behaviour for obstacles and dashed line shows the behaviour for goal.

It can be noticed, that the obstacles will generate a higher repel force the closer it gets, and the goal generates a higher attraction the far it gets. this was considered to make the agent to go straight to the goal the far it gets and react if gets close to any obstacle. This configuration was selected to make the agent faster and less reactive, because if the agent's reaction to obstacles is too sensitive, the agent will spend most of the time changing directions and not moving. Also, if the reaction is too loose, the collisions will be more common.

2.2 Proposed Fuzzy Staged Method (FSM)

The Fuzzy Staged Method, proposes a new alternative to traditional fuzzy systems when the computational costs are important. This method split the current set of rules in several stages according to the relations between the inputs. In this article, the authors implemented this method to an obstacle avoidance algorithm, that requires six input variables to control three outputs, that will manipulate an agent through a field avoiding any type of obstacles without getting too far from the destination. If a traditional fuzzy method were implemented, the total amount of evaluated rules per iteration will be as high as 810 ($2*3*3*3*5*3$) rules per iteration, because a fuzzy system to have a proper function needs that for every combination of inputs a rule will be evaluated. Therefore, instead of evaluating so many rules, the FSM will reduce the total number of rules per iteration to only 30 ($2*3+3*3+3*5$) without breaking the condition that for each combination of inputs there must be a rule.

To implement FSM, first, it is necessary to analyse the relationships among the variables to group them in stages. In this case, the relation between the

variables was palpable among pairs, for the first stage, the stamina and the effort both are variables directly related to the agent's motion model that determines how fast it moves. Then, for the second stage, the distance to the target and the relative speed are linked by a a derivate, because the relative speed is the change of the distance through time. And the same happens int the third stage with the direction to obstacle and angular speed of the obstacle, because the angular speed is the change of the direction through time. The stages where implemented as shown on figure 2.

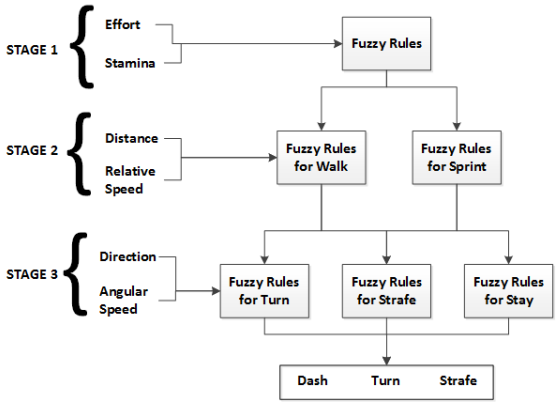


Fig. 2. Block Diagram for Fuzzy Staged Method

The rules implemented in each stage will be explained in the following subsections, is worth to notice that all the outputs go at the bottom of the block diagram, however, the power output can be separated from the rest to simplify the implementation (less rules to be written and most of them will be repetitive).

Stage 1 As mentioned before, the stamina and the effort defines the rules for the first stage, it also controls directly the power output, because the effect of the other stages is minimum and negligible, thus it can be isolated from the rest of the rules, therefore instead of writing six sets of rules for the third stage, there was only needed to write three sets of rules. However, the total amount of written rules does not affect the number of evaluated rules per iteration, it only simplifies the implementation process.

The table 1 shows the implemented rules for this stage. Notice that for every combination of inputs, there is a rule to be evaluated, getting only six rules at this stage.

Stage 2 The stage 2 was composed by distance and relative speed inputs, that as mentioned before, they are strongly related. The total amount of rules to be

Table 1. Rules for Stage 1

	Effort	
Stamina	**Tired**	**Fresh**
Critical	Walk	Walk
Normal	Sprint	Walk
High	Sprint	Sprint

Table 2. Rules for Stage 2

	Distance		
Rel. Speed	**V. Close**	**Close**	**Far**
Away	Stay	Stay	Stay
Slow	Dodge	Turn	Stay
Fast	Dodge	Turn	Stay

evaluated for this stage is nine. Compared to the first stage, the outputs could not be separated from the rest, because the turn angle and the strafe angle are mutually exclusive, because if the agent strafes and turns in the same cycle, will result in a very inefficient move.

The output from this stage does not controls directly any output, but they selects. as stage 1, the set of rules that will apply to the next stage. For the current one, the rules are shown on table 2.

Stage 3 This is the final stage, from now on, the output from here will affect directly the direction where to the agent will displace. The variables are direction and angular speed. These two variables were left last, because the direction by itself do not give an accurate localisation of the obstacle, it means that it will only describe where is the obstacle but not far, and it is necessary to know how far the obstacle is, because if the obstacle is too far away, there is no need for course corrections. However, if the agent is too close, it is necessary to determine if the obstacle will have a chance get in the agent's path or if only will be close but not in the middle of the way, that is why three set of rules was defined from the previous stage.

In this last stage, the three set of rules decide if whether do nothing about the obstacle (Refer to table 3 Stay), make a slight correction (Refer to table 4 Turn) or make a strong course correction (Refer to table 5 Strafe). Despite that the strong correction will be mostly efficient to avoid obstacles, it has a trade off, because the higher the strafing angle, the less effective will be the dash.

2.3 Optimization by Genetic Algorithm

To improve the FSM efficiency, a genetic algorithm was implemented to find a combination of membership functions. The enforced GA searches for the best

Table 3. Rules for Stage 3 - Stay

	Angular Speed		
Direction	**To Left**	**Static**	**To Right**
Far Left	Straight/Face	Straight/Face	Straight/Face
Left	Straight/Face	Straight/Face	Straight/Face
Center	Straight/Face	Straight/Face	Straight/Face
Right	Straight/Face	Straight/Face	Straight/Face
Far Right	Straight/Face	Straight/Face	Straight/Face

Table 4. Rules for Stage 3 - Turn

	Angular Speed		
Direction	**To Left**	**Static**	**To Right**
Far Left	Straight/Face	Straight/Face	Straight/Face
Left	Straight/Face	Right/Face	Hard Right/Face
Center	Right/Face	Right or Left*/Face	Left/Face
Right	Hard Left/Face	Left/Face	Straight/Face
Far Right	Straight/Face	Straight/Face	Straight/Face

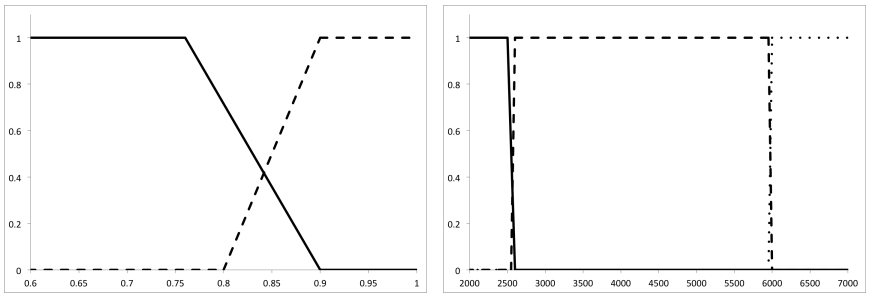
possible combination by obtaining the maximum possible value of the equation (2).

$$eval = 2000 + (800 - t) - 100 * coll \tag{2}$$

Where *eval* stands for the evaluation value, t is the time spent for the current individual and *coll* is the times that the agent collides with an obstacle. The value of 2000 was selected to set an acceptable evaluation value of the individual (i.e. individuals with 2000 evaluation was considered as decent). The value of 800 was taken as the minimum time to be beaten, so if the time that the agent spent to complete the test was higher, it will take it as a punishment. The 100 coefficient that multiplies the collisions, was set to define a higher punishment for each collision, this was designed to make the GA to find a viable combination to reduce the collisions to a minimum.

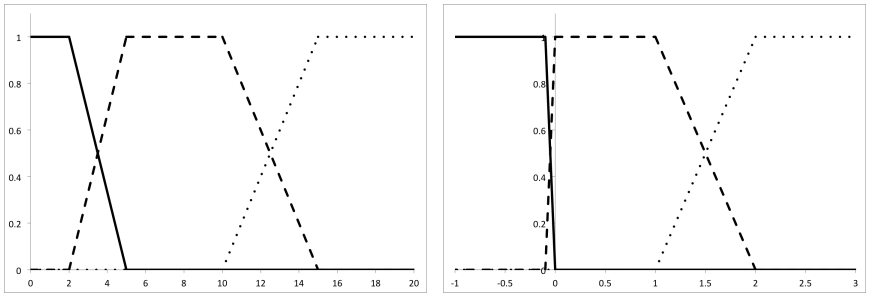
The GA was conceived with a 0.8 cross probability and 0.2 mutating probability. Also, the individual chromosome has 92 segments of 32 bits each, which defines the different values of the membership functions, therefore, more than one segment was required to define a single membership function. The initial population was of 30 individuals, and left searching for 56 generations. The best individual was taken and enforced, the values were fine tuned to get a more centred approach (remove the unbalance in the MFs), resulting in the values shown in the figures 3, 4, 5, 6 and 7.

The resulting individual was tested at the same conditions and experiments of the PFM. The following section will explain in detail the experiments and the results from both methods.



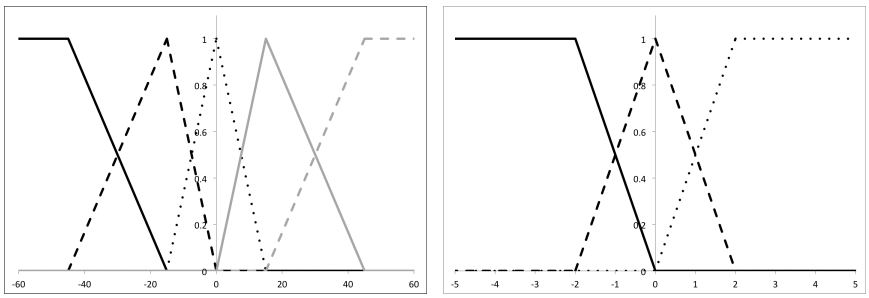
(a) Effort's MFs (b) Stamina's MFs

Fig. 3. Membership functions for inputs of Stage 1. Figure 3(a) is the Effort Membership Function Values. The solid line is the Tired MF and the dashed line is the Fresh MF. Range (-1, 1). Figure 3(b) is the Stamina Membership Function Values. The solid line is the Critical MF, the dashed line is the Normal MF and the doted line is High MF. Range (0, 8000).



(a) Distance's MFs (b) Relative Speed's MFs

Fig. 4. Membership functions for inputs of Stage 2. Figure 4(a) is the Distance Membership Function Values. The solid line is the Very Close MF, the dashed line is the Close MF and the doted line is Far MF. Range (0, 150). Figure 4(b) is the Relative Speed Membership Function Values. The solid line is the Away MF, the dashed line is the Slow MF and the doted line is Fast MF. Range (-4, 4).



(a) Direction's MFs (b) Angular Speed's MFs

Fig. 5. Membership functions for inputs of Stage 3. Figure 5(a) is the Direction Membership Function Values. The solid black line is the Far Left MF, the dashed black line is the Left MF, the doted black line is the Center MF, the solid grey line is the Right MF and the dashed grey line is Far Right MF. Range (-90, 90). Figure 5(b) is the Angular Speed Membership Function Values. The solid line is the To Left MF, the dashed line is the Static MF and the doted line is To Right MF. Range (-180, 180).

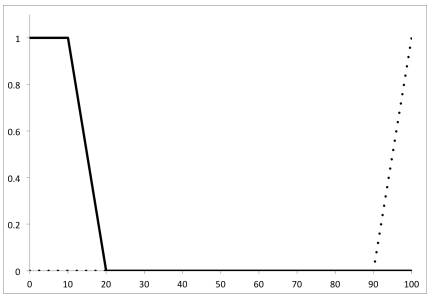
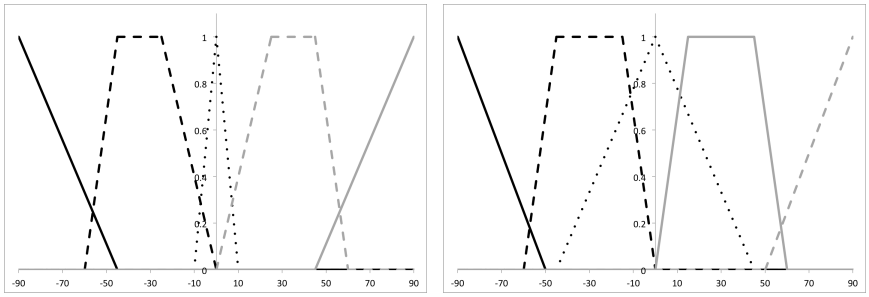


Fig. 6. Membership functions values for output Power. The solid line is the Walk MF and the dashed line is Sprint MF.

Table 5. Rules for Stage 3 - Strafe

	Angular Speed		
Direction	**To Left**	**Static**	**To Right**
Far Left	Straight/Face	Straight/Face	Straight/Face
Left	Straight/Face	Straight/Dodge Right	Straight/Avert Right
Center	Straight/Dodge Right	Straight/Dodge Left or Right*	Straight/Dodge Left
Right	Straight/Avert Left	Straight/Dodge Left	Straight/Face
Far Right	Straight/Face	Straight/Face	Straight/Face



(a) Turn's MFs (b) Strafe's MFs

Fig. 7. Membership functions for outputs Turn and Strafe. Figure 7(a) is the Turn Membership Function Values. The solid black line is the Hard Left MF, the dashed black line is the Left MF, the doted black line is the Straight MF, the solid grey line is the Right MF and the dashed grey line is Hard Right MF. Range (-90, 90). Figure 7(b) is the Strafe Membership Function Values. The solid black line is the Dodge Left MF, the dashed black line is the Avert Left MF, the doted black line is the Face MF, the solid grey line is the Avert Right MF and the dashed grey line is Dodge Right MF. Range (-90, 90).

3 Experiments and Results

In order to test the proposed method two types of experiments were implemented in the Robocup 2D simulator server. The first experiment was designed to analyse the behaviour of the agent in long runs, with some static and dynamic obstacles. The second experiment was conceived in a more aggressive approach to watch the ability of the agent to react facing incoming obstacles. The following subsections will explain in detail how the experiments were created and implemented.

3.1 Experiment 1

The experiment number 1 is an hexagonal path with six obstacles and six waypoints, the first three obstacles that the agent encounters are three dynamic ones that maintains a constant speed with small differences among them in order to collide with the agent if the agent tries a straight line path at maximum

speed. The other three agents are static in a straight line pattern, hence if the agent tries to go straight from one waypoint to the next one, it will collide with each obstacle. The figure 8 shows the current global coordinates on the server from the left side. This experiment was designed to test the agent's ability to maintain maximum speed through a better stamina management, because the server simulates fatigue by giving the agent a value of a maximum stamina of 8000, and if the level drops below 2500, the effort value decreases and the agent cannot reach maximum velocity, meaning that the agent will use more stamina but obtains less impulse, the equation (3) shows the current relation between consumed power and the effective dash power.

$$edp = e * dpr * p \tag{3}$$

Where edp is the effective dash power which is the final value that will impulse the agent. The e stands for the agent's current effort value, dpr is dash power rate, which is a constant of the agent, and p that is the value of the energy to be spent to move the agent. It is clear that the relationship between the effective dash power and the power is lineal, however, because the value of dash power rate is constant, the effort will be the only variable that affects the effective dash power, thus if this value is less than 1, the effective dash power will be less than supplied power, translating in an inefficient use of stamina and less speed.

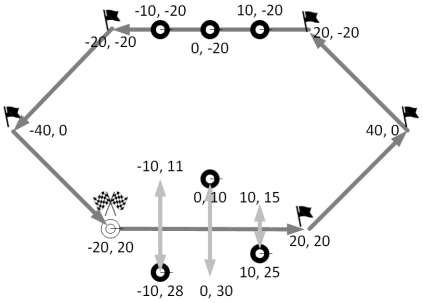


Fig. 8. Layout for Experiment 1. Dark grey lines show the agent's path and light grey shows the obstacle's path. Flag conventions: checkered flags - start/end, solid flags - waypoints

A total of 7000 experiments were made, for each of the methods (FSM and PFM), the results are shown in the table 6. It worth to notice that the PFM's statistics shows a very good performance regarding to the collisions average, however, the FSM demonstrates that it was able to even improve the amount of collision compared to its counterpart reducing it by a 25.03%. However, the time spent does not show an improvement, the FSM have an average 1.43% higher that the PFM in the time spent metrics, maybe, due to the evaluation function that gives the collisions more punishment than the reward given to time savings.

Table 6. Results for experiment 1

Simulations	7000			
	Time Spent Avg.	Std. Dev.	Collisions Avg.	Std. Dev.
FSM	850.25	38.49	0.71	1.15
PFM	838.20	24.23	0.95	1.72

3.2 Experiment 2

The second experiment was designed to test the method's ability to avoid incoming obstacles. It consists of two aggressive agents located along a straight path from the test agent's start position to the goal zone. The aggressive agents will focus on the test agent and will attempt to ram it at maximum speed, until the test agent reaches the goal zone or the aggressive agents loss sight of the test agent. Figure 9 shows the initial coordinates of the agents, the initial positions and the goal position.

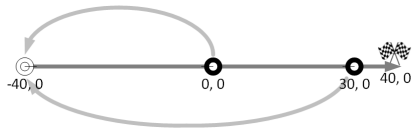


Fig. 9. Layout for Experiment 2. Dark grey arrows shows the agent's path and light grey indicates that the obstacles will go to the current agent's position. The checkered flags signals the goal

To evaluate both methods in this test, a total of 12000 experiments were made for each of the methods. Table 7 shows the obtained statistics for both methods. It is noticeable that the FSM excels the PFM in this tests, a reduction of 93.46% in the collisions average stablishes that the FSM is better to avoid dynamic obstacles such as the aggressive agents, specially because they do not move in a predictable path like the obstacles from experiment 1, because they change their path according to the current location of the test agent. Also, due to the fact that the test agents has less collisions, the average time spent is also improved by 10.27%, indicating that the FSM can be faster than the PFM.

Table 7. Results for experiment 2

Simulations	12000			
	Time Spent Avg.	Std. Dev.	Collisions Avg.	Std. Dev.
FSM	102.78	6.71	0.18	0.57
PFM	114.55	12.97	2.71	2.40

4 Conclusions and Future Work

The obtained results shows that the FSM can successfully avoid obstacles even better than the PFM, however, the tuning of the inputs and outputs is crucial to the method's efficiency. GA offers a viable option for tuning purposes, although, the evaluation function's focus is critical to decide whether the agent is aimed to spent less time to complete the test or if the agent focus on reducing the collisions to a minimum. In this work, as mentioned before, the target was to reduce the amount of collisions, rather than the time spent, but the GA provides a solution that reduces the collisions' average and maintains an average value close to the time spent of the PFM. Is also worth to be mentioned, that the more generations that are allowed to run the GA, the better solution it will obtain, but again it strongly relies on the evaluation function. As the main contribution of this work, the total amount of operations per iteration, was significantly reduced. For this particular case, a traditional fuzzy method will need 810 rules to work properly. This method only uses 30 rules per iteration, a reduction of almost 96% of the rule evaluations per iteration.

As future work, the authors propose, to try the GA with different evaluation functions in order to obtain a more general solution that will not only have a good performance on a particular test, but will have a decent behaviour on almost every scenario, and will reduce the complexity of the individual to converge faster into an optimal solution. Also, as a part of a wider program, the authors will implement the method on mobile robots, provided by the Tecnológico de Monterrey. The implementation of those tests, will have some limitations (i.e. limited space, limited amount of bots, battery limitations etc.) and may will take longer than the experiments with the simulated agents.

Acknowledgments. This works was possible thanks to the support of the CONACYT and the Tec de Monterrey through the Autonomous agents and e-Robots research chairs. Also the authors will like to extend a special thanks to Carlos Hernandez, Cesar Osimani, David Chapa and Jazzmin Novelo for their collaboration during this work. Also a special thanks to Claudia Valenzuela for her support and patience. Last but not least, the authors appreciate the never-ending work of Nova, Athos, Wolf and the rest of the SC lab crew.

References

1. Karim Benbouabdallah and Zhu Qi-dan. Genetic fuzzy logic control technique for a mobile robot tracking a moving target. *International Journal of Computer Science Issues*, 10(1):607–613, 2013.
2. Oscar Castillo and Patricia Melin. A review on the design and optimization of interval type-2 fuzzy controllers. *Applied Soft Computing*, 12(4):1267 – 1278, 2012.
3. Oscar Castillo and Patricia Melin. A review on interval type-2 fuzzy logic applications in intelligent control. *Information Sciences*, 279:615–631, 2014.

4. Jesús S Cepeda, Luiz Chaimowicz, Rogelio Soto, José L Gordillo, Edén A Alanís-Reyes, and Luis C Carrillo-Arce. A behavior-based strategy for single and multi-robot autonomous exploration. *Sensors*, 12(9):12772–12797, 2012.
5. Chian-Song Chiu, Kuang-Yow Lian, and P. Liu. Fuzzy gain scheduling for parallel parking a car-like robot. *Control Systems Technology, IEEE Transactions on*, 13(6):1084 – 1092, nov. 2005.
6. JC Cortes-Rios, E Gomez-Ramirez, HA Ortiz-De-La-Vega, O Castillo, and P Melin. Optimal design of interval type 2 fuzzy controllers based on a simple tuning algorithm. *Applied Soft Computing*, 23:270–285, 2014.
7. D. Fox, W. Burgard, S. Thrun, and A.B. Cremers. A hybrid collision avoidance method for mobile robots. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1238–1243 vol.2, 1998.
8. Luis C Gonzalez-Sua, Olivia Barron, Rogelio Soto, Leonardo Garrido, Ivan Gonzalez, José Luis Gordillo, and Alejandro Garza. Design and implementation of a fuzzy-based gain scheduling obstacle avoidance algorithm. In *Artificial Intelligence (MICAI), 2013 12th Mexican International Conference on*, pages 45–49. IEEE, 2013.
9. Chih-Lyang Hwang and Li-Jui Chang. Internet-based smart-space navigation of a car-like wheeled robot using fuzzy-neural adaptive control. *Fuzzy Systems, IEEE Transactions on*, 16(5):1271–1284, 2008.
10. Jinseok Lee, Yunyoung Nam, Sangjin Hong, and Weduke Cho. New potential functions with random force algorithms using potential field method. *Journal of Intelligent & Robotic Systems*, 66(3):303–319, 2012.
11. Tsong-Li Lee and Chia-Ju Wu. Fuzzy motion planning of mobile robots in unknown environments. *Journal of Intelligent and Robotic Systems*, 37(2):177–191, 2003.
12. Ricardo Martínez-Soto, Oscar Castillo, and Luis T Aguilar. Type-1 and type-2 fuzzy logic controller design using a hybrid pso-ga optimization method. *Information Sciences*, 2014.
13. Ricardo Martínez-Soto, Oscar Castillo, and Juan R Castro. Genetic algorithm optimization for type-2 non-singleton fuzzy logic controllers. In *Recent Advances on Hybrid Approaches for Designing Intelligent Systems*, pages 3–18. Springer, 2014.
14. Ellips Masehian and Davoud Sedighizadeh. Classic and heuristic approaches in robot motion planning–a chronological review. In *Proc. World Academy of Science, Engineering and Technology*. Citeseer, 2007.
15. Abraham Meléndez, Oscar Castillo, Fevrier Valdez, Jose Soria, and Mario Garcia. Optimal design of the fuzzy navigation system for a mobile robot using evolutionary algorithms. *Int J Adv Robotic Sy*, 10(139), 2013.
16. Saroj Kumar Pradhan, Dayal Ramakrushna Parhi, and Anup Kumar Panda. Fuzzy logic techniques for navigation of several mobile robots. *Applied soft computing*, 9(1):290–304, 2009.
17. Saroj Kumar Pradhan, Dayal Ramakrushna Parhi, Anup Kumar Panda, and Rabindra Kumar Behera. Potential field method to navigate several mobile robots. *Applied Intelligence*, 25(3):321–333, 2006.