FUZZY CONTROL SYSTEM NAVIGATION USING PRIORITY AREAS

M. A. OLIVARES*, P. CAMPOY, C. MARTINEZ, J. F. CORREA and I. MONDRAGÓN

DISAM, Universidad Politécnica de Madrid,
Madrid, 28006, SPAIN
Computer Vision Group
**E-mail: mig_olivares@hotmail.com*
www.disam.upm.es/vision

This paper presents an improvement for the software implementation (MOFS) of a user adaptive fuzzy control system for autonomous navigation of mobile robots in unknown environments. This improvement consists of a priority areas definition where the environment is measured by a PLS laser sensor, in order to get a reduction in the number of fuzzy rules and also in the computational cost, and hence obtaining improvements in the trajectory. This system has been tested in a pioneer mobile robot and on a robotic wheelchair, odometry sensors are used to localize the robots and the goal positions. The system is able to drive the robots to their goal position avoiding static and dynamic obstacles, without using any pre-built map. This approach improves the way to measure the danger of the obstacles, the way to follow the walls of corridors and the detection of doors. These improvements reduce the zigzag effect of the previous system by making the trajectories significantly straighter and hence reducing the time to reach the goal position.

Keywords: Fuzzy logic, autonomous navigation, adaptive control, fuzzy control, mobile robots, collision avoidance, reactive navigation, priority areas.

1. Introduction

One of the most investigated problems in the robotic community is the autonomous navigation of mobile robots. There are different ways to face this problem either by reactive, delivered or hybrid methods. The main goal of this research is to reduce the computational and memory cost required in autonomous navigation of mobile robots. This approach reduces the computational cost by avoiding the use of environment maps, with the additional advantage that it can be used regardless of the place. To approach the computational cost fuzzy logic techniques for the interaction between the robot and it's surroundings.

In the following paper we remark on the usefulness of the two improvements deeply explained in ([1]). Which are, the more important and comprehensive sector for the definition of each fuzzy-variable and a user adaptive learning algorithm based of the synapses-weight idea of the brain operation. For this purpose a versatile software was created called MOFS (**M**iguel **O**livares **F**uzzy **S**oftware) which uses fuzzy logic techniques that can learn from experience regardless of the intelligent system architecture. The improvement made in this work is to measure the environment by creating different areas with different priority values.

The next section of this paper presents the autonomous fuzzy navigation control system in detail. Section III defines the developed user adaptive learning algorithm. Section IV is dedicated to the explanation of the environment division using priority areas. Section V describes the fuzzy-software implementation. Section VI shows the experimental results. Section VII consists of the conclusion and future works.

2. Autonomous Fuzzy Navigation Control System

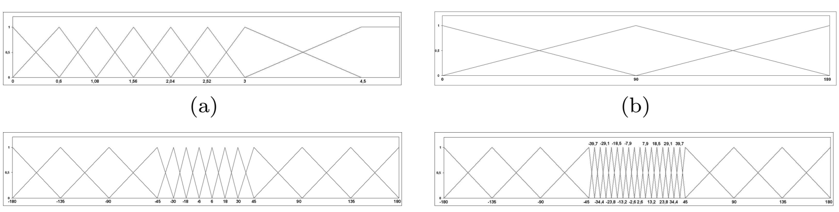
For this work we use the MOFS to developed an autonomous fuzzy navigation system. The structure of the fuzzy control is divided into three inputs variables for interaction with the environment and one output. The data we use to check each situation is defined by the distance (in meters) between the robot and the obstacle with the biggest priority, the angle direction inside the selected priority area and the angle to the goal from the robots position (both in degrees). The angles and distance are recovered by using a 180 range laser sensor and an odometry system.

The figure 1 shows the inputs and the output of the fuzzy control system. In the figure 1 (a), the distance to obstacles is shown from 0 meters (very near) to 4.5 meters (far away), values with are equal to or greater that 4.5 have the same interpretation. The second input variable is shown in figure 1 (b), splits the environment in three areas (Left, Center and Right) and provides where is the obstacle. The detection of the angle to the goal position (fig. 1 (c)) is based on the data provided by a 2D odometry system, which return the angle between that position and the orientation of the mobile robot. Taking into account that this is a representation of 360 degrees area, the left extreme and the right extreme represent the back side of the mobile robot. The front side of the robot has a bigger number of sets in order to define a better behavior when it is very near to the goal position. Finally the control system evaluates all the values of the inputs using a base of rules and giving how many degrees must the mobile robot turn in order to avoid the obstacles and face the goal position. The range of this variable is defined in the same way than the angle to the goal position, where the left extreme and the right extreme represent a big turn of 180 degrees (fig. 1 (d)).

In previous work different velocities were declared as constants depending on the danger of the situation which was based on the distance to the near obstacle, but in this work the priority areas allow side obstacles to be considered less dangerous than obstacles in front of the mobile robot. The system scan time has been increased to 6 times per seconds, double that of the previous work ([1]).

The fuzzification of the inputs and outputs are defined by using a triangular membership function. This function has been used in many robotics applications for navigation purposes (see,[2],[3],[4],[5],[56]).

The system defines a more important and comprehensive sector in a variable, with the aim of increasing the number of fuzzy-sets only in the sector where the variable has been invoked. With this improvement we avoid an increase in the number of fuzzy-sets in all space of a variable when a fault occurs in one sector, as shown in Fig. 1. This is very important because there exists situations where the number of fuzzy-rules are increased, making the system slower. This could be possible by the statistical study of the uses of the fuzzy-subsets of each variables. To compare the same system response without this more important sector definition, the rule-based size was increased by 40%, this yielded to a 78% improvement in the computational time, 1.26s (improved) vs 2.25s (not improved). With regard to the time spent to check each situation the system spent 17% more, $0,059$s



(a) (b) (c) (d)

Fig. 1. Fuzzy sets and membership functions for the mobile robot: (a) distance to the near obstacle $(0, 4'5)$ meters(b) angle, inside the selected priority area, between robot and the priority obstacle, $(0, 180)$ (c) angle between the robot and goal position, $(-180, 180)$ (d) motor command (output), the direction to take, $(-180, 180)$

(improved) vs $0,06903s$ (not improved).

For the inference process (in the defuzzification) we used an adaptation of the minimum and product classic method (see,[2],[78]), with the consideration of the weights assigned at each fuzzy-rule. This idea will be explained in the next section. For the defuzzification part itself, we used the Height Method (see,[5],[49]), but with the adaptation of weighting of the fuzzy-rules (Eq. 1a, 1b), where w^l is the rule weight and W is the maximum weight possible.

$$y = \frac{\sum_{l=1}^{M} \overline{y}^l \, min\big(\mu_{B'}(\overline{y}^l)\big) \, \frac{w^l}{W}}{\sum_{l=1}^{M} \, min\big(\mu_{B'}(\overline{y}^l)\big) \, \frac{w^l}{W}} \tag{1a}$$

$$y = \frac{\sum_{l=1}^{M} \overline{y}^l \, \prod \big(\mu_{B'}(\overline{y}^l)\big) \, \frac{w^l}{W}}{\sum_{l=1}^{M} \prod \big(\mu_{B'}(\overline{y}^l)\big) \, \frac{w^l}{W}} \tag{1b}$$

3. User adaptive learning algorithm

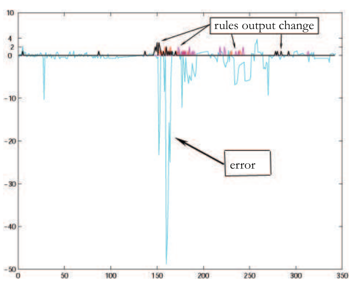
This algorithm is based on the idea of the synapse-weight, where a weight variable for each rule is defined, that represents the contribution of each rule in the system output. The default value of the weight is 0.3 and the maximum value is 3. The decision of the user will make the weight of the rules increase or decrease depending on the situation and in accordance with the system output. For each situation 8 rules are selected, given the fact that we use a simple overlap in the variables. In the training period the system compares each rule output and with the user decision, which had been fuzzificated by using the same proccess. When the rule output and the user decision are different the system will decrease the weight of the rule, and in the case of a negative rule weight the system will change the output of the rule by using the set of user decisions that has the highest value. In the case where the rule output is like one of the two sets of the user decision the system will increase the weight by a quarter of the set membership value. So with this method the system will adapt its behavior to the users' behavior. In the training phase the user just needs to make some tours and the system will learn from this, evaluating all the situations that the user makes in the tours. If the tours have many different situations the system will learn early.

In the execution phase, when the user isn't making a decision the system continues learning, based on the rules that have higher weights, which represents the users behavior. The way in which the system compares the output of the selected rules for the situation with the output of the system, is the same way as in the training phase.

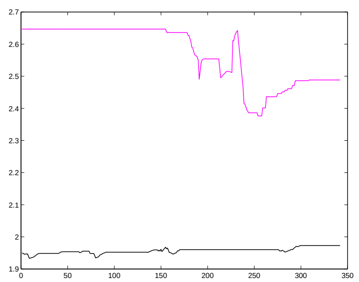
The learning is illustrated in Fig. 2 (a), where the error between the system output and the user decision is represented by the label 'error' (cyan color). The other lines referenced with the label 'rules output change' represent the rules output change at specific proximities to the obstacle. When the learning algorithm makes the output of the rules change the system reduces the error. The more characteristic weight changes are represented in Fig. 2(b).

4. environment division using priority areas

The division of the environment into different areas with specific priority values is the new improvement introduced in the latest version of MOFS. We try to make a radical reduction in the number of rules and in the same way reduce the size of the base of fuzzy rules, moreover we try to get better trajectories without increasing the complexity of the software. Better trajectories are measured by the a more direct path to the goal, hence reducing the zigzag effect of the previous work ([1]). The environment which is described by the angle to the obstacles in front of the mobile robot where divided into three areas, the first one from 0 to 60 degrees, the second one from 60 to 120 degrees, and the last one from 120 to 180 degrees. It is important to note that the PLS laser gives the environment information 180 degrees in front of the robot, from the left to the right. Furthermore each segment is divided into three concentric areas, given a total of 9 areas and mixing the two fuzzy variables, angle and distance between the obstacle and the robot. The assignment of the priorities are shown in the figure 3. There are some areas that has equal priority, so in that cases we consider



(a)



(b)

Fig. 2. (a) Outputs modifications. (b) Weight modifications.

the priority obstacle the near one.

By the fact of giving more importance to obstacles in the middle segment of the priority area the robot behavior through a corridor is greatly improved by making a much straighter path, also doors detection in indoor environments is more efficient.

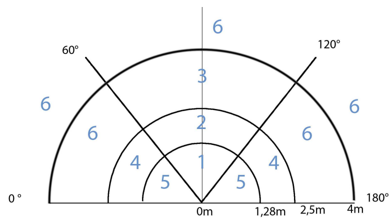


Fig. 3. Priority Areas.

5. Software Implementation

The software was independently designed for the c++ platform. We defined one class for each part of the fuzzy-logic environment in order to facilitate the future updates and to be easy to work with. There are different classes for variables, rules, membership functions and defuzzification modes (Fig. 4). Depending on the system we want to create we can define the number of inputs and outputs that we prefer or make some different system in serial mode, where the output of one system can be the input of another. In the same way we can define the different characteristics of the variables, the type of the fuzzification inference type or the defuzzification mode. At the beginning the rule-base is implemented randomly, making this process so fast, and depending on the size of the rule-base the user adaptive learning algorithm in the training mode modified this in a few or little more tours or uses. For this initial training it is possible to introduce a pattern list. In this paper we make an initial rule base and then we use it for adaptation of two users. With this initial rule base, where the weight of all the rules are reset at the default weight, we can make a basic adaptation in 4 or 5 tours, taking into account its complexity, with the robotic-wheelchair.

The updates of the software can be implemented in each of the ways of the fuzzy-logic parts, like introducing different membership functions, fuzzy inference types or introducing another kind of defuzzification mode.

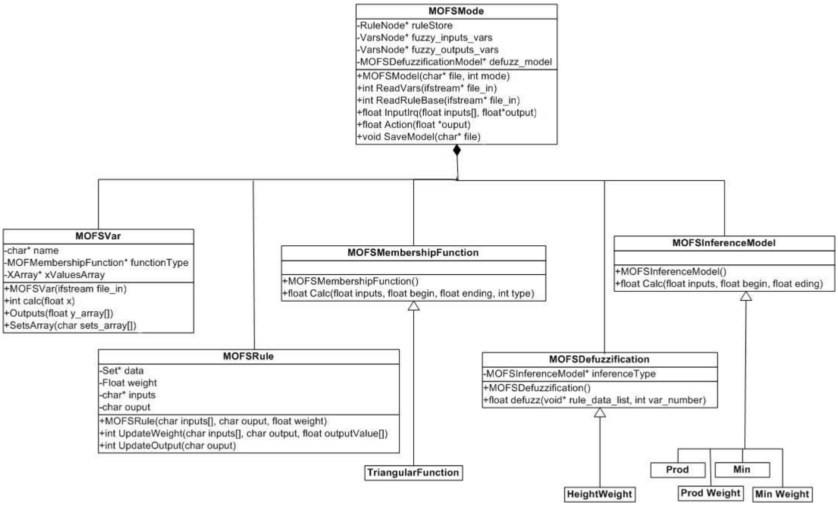
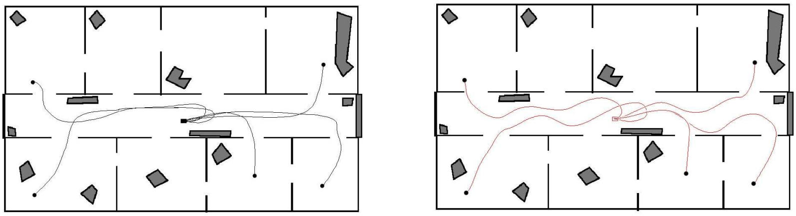


Fig. 4. Software definition.

6. Experiments Results

The experimental results presented in this section are a comparison with the previous experiments realized in ([1]).

Using this new improvement a 75% reduction in the size of the base of rules is gain, leaving a group of just 252 rules, from the 1680 rules of the previous work. Subsequently a large reduction in computational cost is obtained, with the previous system spending 0.059 seconds for each iteration and the new approach just 0.028 seconds, the 52% less. We must note that the time spent in order to reach the same goal point is reduced, this is because the velocity control is no longer based only on the distance to the near obstacle, but takes the obstacle priority area into account. This gives much greater importance to obstacles directly in front of the robot rather then the obstacles on each side. For example, in a corridor the walls don't reduce the robots velocity in the situation where there are more priority obstacles directly in front of the robot. Consequently, the zigzag effect present in the previous work obtained when the robot travels to the goal position, is greatly reduced. This can be seen by comparing figures 5 (a) and 5 (b), with the much straighter trajectories of the latest research clearly visible. Also the doors detection while traveling to the goal position is more accurate.



(a) (b)

Fig. 5. (a) Paths with priority areas. (b) Paths without priority areas.

7. Conclusions and Future works

In this paper, a fuzzy-based, priority areas navigator system for mobile robots has been proposed, which implements obstacle avoidance and solves navigation problems in unknown environments. We remark the previous improvements of user adaptive navigation system and add the priority area definition of the environment. The main goals of this approach are to reduce the computational and memory cost of the system evaluation of each situation and to improve the trajectory by implementing more efficient evaluation of the environment without using a map database. When the computational costs of the previous work is compared to a map based navigation system a 90% (0.059 seconds for each iteration)was obtained. With this latest approach we get a further reduction of 52%, just 0.028 seconds per iteration.

The system gives a good improvement in the time spent to reach the same goal from the same initial point with a reduction of 12%. The straight trajectory gives more robustness and softer movements when avoiding obstacles. For future works, we have planned to greatly develop the software by

	time spent in each iteration	Rule base size
previous work	0.059 s	1680
last work	0.028 s	252
percentage reduction	52%	75%

implementing different methods of fuzzy-inference, defuzzification or membership functions. We plan to include different bases of rules that represent different behaviors to the autonomous navigation control system, for example following some robots or signs, escape from deadend roads, etc. This can be achieved by storing different rule-bases and including more capabilities for recognition of these situations or simply just changing the rules when the system has too many continuous errors. We are considering to include more sensors and/or real-time map creation that stores repetitive paths, using SLAM techniques.

Currently we are trying to implement computer vision techniques to improve the recognition of obstacles in the environment, and embed the fuzzy-system navigation control into different kinds of UAS's.

Additionally we intend to use the software in a different type of robotic system that do not have navigation tasks as the main objective, such as medical robots, car shock absorber systems or other security car systems.

Acknowledgment

The authors would like to thanks Deane Chadwick, Dr. Luis Mejias for support and help in the development of this work.

References

1. M. A. Olivares and J. A. F. Madrigal, Fuzzy logic user adaptive navigation control system for mobile robots in unknown environments, in *Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on*, 2007.
2. G. Castellanoa and A. Fanelli, A self-organizing neural fuzzy inference network, in *in Proc. of IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, (Como, Italy, 2000).
3. N. K. Kasabov, Learning fuzzy rules through neural networks *Artificial Neural Networks and Expert Systems, 1993. Proceedings., First New Zealand International Two-Stream Conference on* 1993.
4. S.-J. Huang and C.-F. Hu, Predictive fuzzy controller for robotic motion control, in *Industry Applications Conference, 1995. Thirtieth IAS Annual Meeting, IAS '95., Conference Record of the 1995 IEEE*, 1995.
5. P. G. Zavlangas, S. G. Tzafestas and K. Althoefer, Fuzzy obstacle avoidance and navigation for omnidirectional mobile robots *ESIT 2000, Aachen, Germany* 2000.
6. C.-J. Wu, Fuzzy robot navigation in unknown enviroments *National Yunlin Institute of Technology* 1995.
7. R. Araujo, U. Nunes and A. de Almeida, Fuzzy-based reinforcement learning of a robot force control skill, in *in Proc. of IEEE International Symposium on Industrial Electronics (ISIE'96)*, (Warsaw, Poland, 1996).
8. A. Pesonen and A. Wolski, Quatified and temporal fuzzy reasoning for active monitoring in rapidbase *Symposium on Tool Environments and Development Methods for Intelligent Systems, TOOLMET* 2000.
9. S. Thongchai, Behavior-based learning fuzzy rules for mobile robots *American Control Conference, 2002. Proceedings of the 2002* 2000.