A Fuzzy Logic Based Robot Navigation System

John Yen and **Nathan Pfluger**
Center for Fuzzy Logic and Intelligent Systems
Department of Computer Science
Texas A&M University
College Station, TX 77843

Abstract

One of the key issues in mobile robot navigation is coping with uncertainty in a dynamic environment. One possible solutions lies in behaviors, but there is a problem with standard behavior arbitration techniques. To combat this problem, David Payton introduced a command arbitration net that minimizes the loss of information between behaviors. We have extended his approach using fuzzy logic. Some benefits of using fuzzy logic include simplicity, extensibility and understandability, especially in the area of sensor fusion.

Introduction

Mobile robot path planning and path execution is a significant problem in the process of building autonomous robots. The ideal place for such a robot would be in a space station, or other isolated area where it is hard or impossible for human life to exist.

There are many limitations to current approaches. Methods based on potential fields and stimulus response paradigms have problems finding paths, even when they exist. A standard graph decomposition method always gives a path, but requires complete knowledge of the environment, and gives a path that is not easily followed. Also, graph decomposition methods plan in a static environment, while most useful environments in the real world are dynamic.

Therefore, important issues in mobile robot planning and execution in the real world include:

1. Dealing with a dynamic environment.
2. Dealing with the problems of incomplete and/or inaccurate knowledge.
3. Coping with the problems of limited sensor information.

The main focus of our work has been on developing a fuzzy controller that will take a path based on known information, which may or may not be complete and/or accurate, and adapt it to a given environment, using limited sensing apparatus. The robot's only source of information is gathered from these sensors, and yet it has the ability to avoid dynamically placed obstacles near and along the path. This gives the robot the ability to adjust to plans made with incomplete or inaccurate maps. These benefits are the result of using a method similar to Payton's command arbitration network.

By using fuzzy logic, our project has been able to improve some areas of Payton's approach. These improvements include smoother control commands and a simpler arbitration network, due to the nature of fuzzy logic and fuzzy rule based systems. Our work has been tested on a simulation test bed that uses a robot with five to ten range finding sensor's in fixed postions. The system has shown the ability to follow both nonsmooth paths and gradient field information while dynamically avoiding unplanned obstacles.

Previous Work

In this section we will discuss previous approaches to mobile robot path planning and path execution. We will then then give a background on the workings of fuzzy logic.

Mobile Robot Path Planning

The basic problem in mobile robot path planning is getting the robot from point A to point B without colliding into any walls or obstacles, in, hopefully, close to the shortest amount of time. Two basic methods have been proposed to accomplish this task.

The first method works by using a method called behaviors. The method is best compared to the biological paradigm of stimulus response. The robot is given a start and a goal. The goal attracts the robot, while walls and obstacles repel the robot. Other behaviors, such as open space finding, wall hugging, or door This method of path planning, unfortunately, is sometimes unable to find paths, even though they exist. Another problem is that the robot sometimes becomes undirected, wandering about the map searching for doors and passages in an area where they do not exist.

The other method is based on graph decomposition. The space is partitioned into convex free areas, which

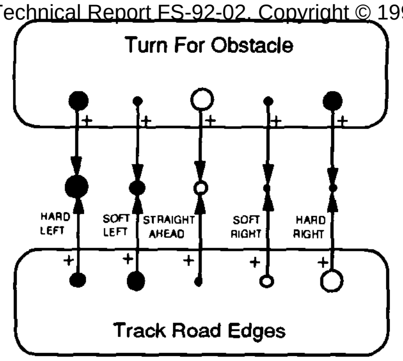

Figure 1: Connectionist network for fusing turn rate commands.

can be represented by a graph where each node is a free area and the presence of an edge between areas represents the fact that the areas touch. A graph search is then used to find a path from the start node to the goal node in the graph. Some major problems with graph-decomposition are that it gives a non-smooth path for the robot to follow and that all obstacles in the environment must be known and modeled for the robot to avoid them.

More recently, a new approach for controlling the robot was proposed by Payton [Payton *et al.* 1990]. This approach combined the two basic methods by giving the robot a non-smooth path and forcing it to avoid obstacles in the path. The approach was based on combining the output of the path following routine with data collected from the sensors, see figure 1. The robot would decide that it needed to turn by sensing an object in the forward sensor, and then would combine this knowledge with which way the path went to determine a final turn.

However, Payton's approach seems to be limited by the granularity of the network, allowing only a finite number of control commands. Also, the interactions between sensors during sensor fusion is handled by adding additional layers and links, making for a complex system. Finally, Payton's approach seems to have some problems with extensibility, for example adding a new sensor to the robot requires all modules, whether or not they have a direct link to the sensor, to be reconfigured.

Fuzzy Logic and Fuzzy Control

Fuzzy logic was first proposed in the early 1960's by Lotfi Zadeh[Zadeh 1965, Zadeh 1978]. Fuzzy logic is based on the idea that humans do not think in terms of crisp numbers, but rather in terms of concepts. The degree of membership of an object in a concept may be partial, with an object being partially related with many concepts. By characterizing the idea of partial membership in concepts, fuzzy logic is better able to convert natural language control strategies which are used by humans to a form usable by machines.

Experience has shown that a controller based on fuzzy logic yields superior results than conventional control algorithms, and sometimes even better results than human operators[Lea 1988, Lee 1990a].

Some major advantages of fuzzy logic are that it allows a human expert to express his knowledge in a natural way, and that fewer rules in general are needed to express concepts, therefore saving time and space when a search for which rules for a given situation must be executed. The final advantage of fuzzy logic is that since there are no crisp transitions between states, the system tends to be much more resistant to input deviations.

A Proposed Architecture

We now give an overview of the total system. Although most of the current research has been concentrated on the path execution subsystem, we wish to give an overview of a system where this subsystem will be useful. After defining our complete system, we will give the details involved in designing our path execution subsystem.

System Overview

Our architecture is based on the hierarchical approach to planning. This methodology allows us to divide the problem into levels of abstraction, inviting modular development. The current design of our architecture contains three principle layers.

1. **Task Planner (TP):** This planner works at the level where it must be decided which tasks or portion of tasks must be accomplished next. It must be able to coordinate multiple tasks that may or may not by independent. This planner decides the ordering of tasks, and determines the start point and goal point for each task. This planner works with the path generator to work out strategies for combining tasks.
2. **Path Generator (PG):** This planner uses a start point and a goal point given by the Task Planner and uses current maps to decide which is the best path to follow, based on the length and safety of a path. It supplies a rough path, more like a general plan than an actual path, to the Controller for execution.
3. **Fuzzy Controller (FC):** This subsystem, which is the actual controller for the robot, is given a path from a starting point to the goal point, and directs the robot along it. This controller works closely with the sensors to detect and avoid walls and obstacles, adapting the given path to the current situation. If the robot is forced too far from the path, it has the option to request the higher level planners for a new path to follow.

Fuzzy Controller

Most of the early work has been done on the Fuzzy Controller. We believe that this controller is a natu-

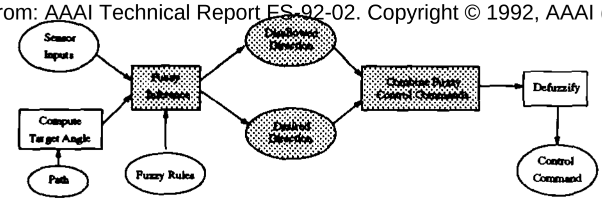

Figure 2: Data Flow in the FC

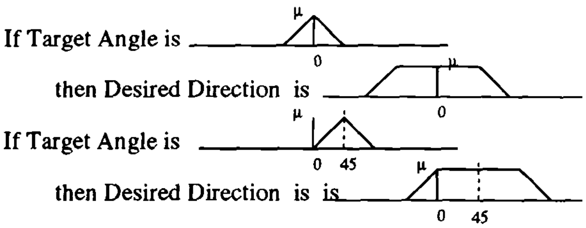

Figure 3: Determining Desired Direction

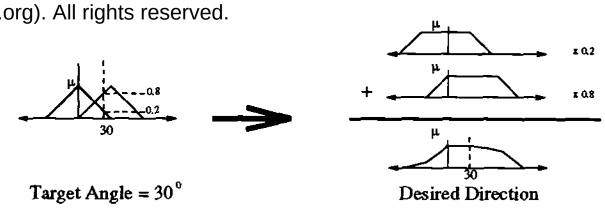

Figure 4: Example Desired Direction Calculation

Sensor Angle	Distance	μ_{near}
-90	20	0.9
-45	10	1.0
0	100	0.2
45	200	0.0
90	60	0.4

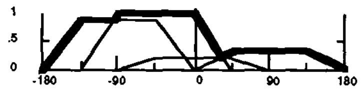

Figure 5: Example Sensor Fusion

ral extension of the controller proposed by Payton et al.[Payton *et al.* 1990]. The largest difference is that where Payton uses discrete sets to represent the turn needed and the sensor readings, we use fuzzy sets and relations.

The controller is able to take a path, generated either by a simple graph decomposition method or by a gradient field type method, and follow it to the goal. The controller is able to cut corners on the path, while avoiding walls and obstacles, thereby adapting the path to the environment.

Overview The algorithm used by the Fuzzy Controller works in four steps.

1. Determine the target angle, which is the angle between the current direction and desired direction. The desired direction is computed by choosing a target point on the path for the robot to aim at, irregardless of obstacles and walls. Fuzzify the target angle to make it into the more general desired direction.
2. Integrate the sensors using fuzzy sets to determine disallowed turning angles due to blocked sensors.
3. Combine the desired direction and the disallowed direction to find a set that represents directions that are both desired and not disallowed.
4. Defuzzify the resultant combination to determine the control command for the change in direction.

The flow of data is shown in figure 2. All information passed between modules is in fuzzy format. The process of defuzzification, shown last, converts a fuzzy set, representing the concept of both desired and not disallowed, to a crisp number for use by the system.

Determining Desired Direction Fuzzification is performed in the first step by matching the crisp number corresponding to the target angle to a fuzzy set representing the different types of turns. By computing the outcome of applying a set of fuzzy rules, for which a subset is given in figure 3, the concept of wanting to turn by x degrees is broadened to the concept of turning in the general direction of x.

This broadening of the target angle allows the system to adapt the path to the environment by not constraining the robot to go only in direction of the target angle, but in any angle in the general direction of the target angle, with a preference in going nearly the same angle as desired. An example calculation of a turn of thirty degrees is given in figure 4.

Determining Disallowed Direction The fuzzification process of the sensors is slightly different. The distances returned by the sensors are compared to a fuzzy set representing the concept of 'near'. The degree of membership in near determines how disallowed that direction is. The use of a fuzzy set to determine nearness helps to overcome uncertainty in sensor readings by avoiding the sharp cutoffs that are present in standard rule based sysetms.

Once a sensor determines the nearness of the nearest obstacle, it must determine how disallowed the directions around the sensor are. Since most obstacles in the real world have some size, this is a good way to generalize a blocked sensor, since, in general, a blocked sensor in a direction implies that all angles in that direction would be impassable. This generalizing of a single sensor to cover an area around the sensor allows the robot to use less sensors.

Once all of the sensors have determined how disallowed the angles around them are, they are combined using the fuzzy logic operator 'or', see figure 5 for an example of combining five sensors. Notice that the area each sensor has control over overlaps some with other sensors. This both allows for a limited number of sensors to reason about all possible angles, and for each sensor to influence its neighbors, but not totally. In other words, if the sensor to the right is blocked,

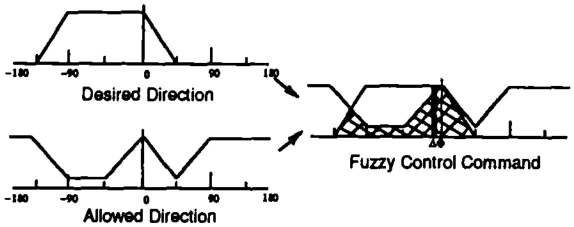

Figure 6: Combining the Desired and Allowed Directions

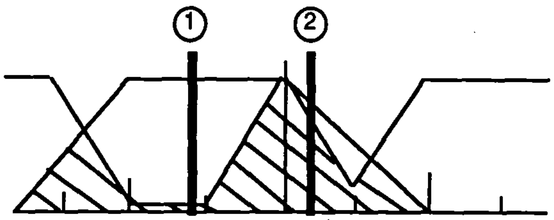

Figure 7: Defuzzification Strategies

then the right part of the current area is also blocked, giving an interaction between the sensors.

Combining disallowed direction directly with the desired direction isn't possible, since one is a positive concept and the other is a negative concept. Therefore, we instead use the complement of disallowed direction, which can be thought of as the allowed direction of travel. This set can now be combined directly with desired direction to obtain a fuzzy control command.

Combining Desired Direction and Allowed Direction The process of combining the desired direction and the allowed direction is simple. In fuzzy logic terms the process is to take the fuzzy set that results as the minimum at each corresponding point between the desired direction and allowed direction, i.e. the fuzzy *and* operator. This process is shown in figure 6. To obtain the final control command, the fuzzy set that results from the combination must be defuzzified.

Defuzzifying the Result Defuzzification is process of converting a fuzzy set into a crisp number that represents some property of the fuzzy set. There are three major defuzzification strategies, the results of two of which are shown in figure 7. The first and simplest strategy, which isn't shown, is to take the highest point along the set. In standard fuzzy logic terminology this is call the Max defuzzification strategy. It has been shown[Lee 1990a, Lee 1990b] that other defuzzification strategies, in general, give better results due to the fact that only the strongest rule will affect the final decision, and fuzzy logic gets much of its strength from the

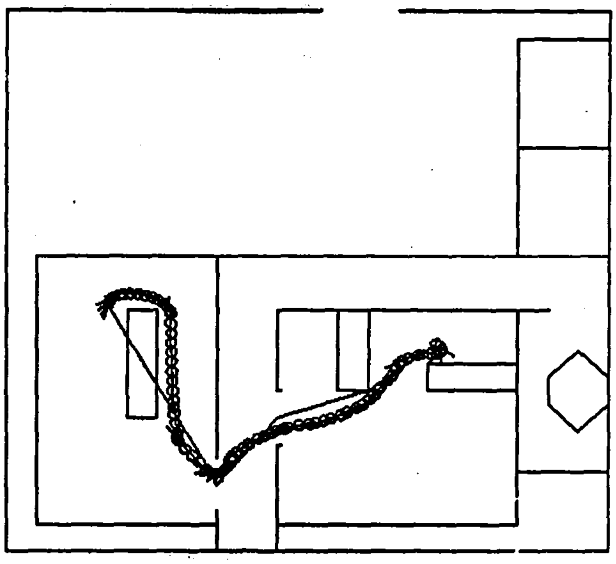

Figure 8: Simulations of Fuzzy Navigation Controller

blending of rules.

Therefore, the next strategy is to take a weighted sum of the set, otherwise known as the centroid strategy. This is shown as marker 1 in figure 7. Although this strategy works well for normal fuzzy sets, it can lead to problems as shown. Since this set represents the combination of allowed and desired directions, we need our final command to be in an area that is strong in both concepts, and marker 1 is not strong in both.

Finally, we developed a strategy where we analyze the fuzzy set and break it into regions. We then take the weighted average of the largest region, shown as marker 2 in figure 7. This technique satisfies both the conditions of using all (or most) of the set and giving a command that is strong in both the desired and allowed directions. For a more detailed description of this technique, which has been named centroid of largest area, see Pfluger et al. [Pfluger *et al.* 1992].

Benefits

One of the major advantages of our control system is the robot's ability to cut corners when allowed and avoid unplanned obstacles. The target angle is determined from a point that is well along the path, and this allows the robot to look ahead. In a situation where a corner is being approached, the robot knows both which way a corner needs to be cut, and when the turn is allowed by sensors. In addition, if an obstacle is directly in the path, the robot can sometimes avoid the obstacle while still traveling in the general direction of the path, see figure 8

Although most of these benefits can be accomplished by Payton's system, addition benefits of our system compared to his include simplicity, smoothness of control command and exstensibility. Finally, due to the nature of fuzzy logic, our control system is easily understandable, and therefore easier to create and maintain.

Path Generation

If the robot wasn't given a path, but only a goal, then this controller would simulate a behavior like system, i.e. going toward the goal and avoiding sensed obstacles. The strength of this overall approach lies in its ability to use path information.

The robot has the ability to use a planned path between the robot's current position and the goal. Due to the nature of the lookahead feature, the robot does not have to stay directly on the path in order for it to achieve its goal, see figure 8. The robot is able to cut corners, follow a path along an obstacle, and even follow a path that goes through the middle of obstacle, a situation that could occur if there are unknown obstacles in the environment. This allows the path generation module to not have to deal with a dynamic environment, or in other words the path generator doesn't need to plan a new path every time a new obstacle is discovered near on in the current path planned.

This gives three types of path information useful to a mobile robot. At the global level is the direction of the goal, which is used in potential field methods. At the local level are gradient fields which act as a constant direction finder[Payton *et al.* 1990]. At a level inbetween these two types of path information are non smooth paths which act like a subgoal system.

Conclusions

We have used fuzzy logic to extend Payton's behavioral architecture for mobile robot control. By using fuzzy rules to explicitly capture heuristics implicit in Payton's behaviors, we obtain a mobile robot controller that is extensible and understandable, yet can effectively cope with unforseen obstacles and other problems of imperfect paths in a dynamic environment. Improving the adaptability of the mobile robot controller also has an important impact on the task of path generation. Since the generator is no longer required to produce perfect smooth paths, the complexity of its algorithm can be reduced significantly.

Future work of our research includes investigating the role of fuzzy logic in dealing with uncertainty in path generation and task scheduling for mobile robots, working on a fuzzy mapping element to facilitate information gathering and handling, and working on control systems to interconnect and arbitrate between the Fuzzy Controller, the Path Planner and the Task Planner units.

References

[Albus, 1990] J. S. Albus. "A Theory of Intelligent Systems". In *Fifth IEEE Int. Symp. Intelligent Control*, Philadelphia PA, Sept 1990.

[Anderson and Donath, 1988] T.L. Anderson and M. Donath. "Synthesis of Reflexive Behavior for a Mobile Robot Based Upon a Stimulus-Response Paradigm". In *Mobile Robots III: Proceedings of the SPIE*, volume 1007, Cambridge MA, Nov 1988.

[Brooks, 1990] R.A. Brooks. "A Robust Layered Control System for a Mobile Robot". In G. Shafer and J. Pearl, editors, *"Uncertain Reasoning"*, pages 204–213. Morgan Kaufmann, 1990.

[Clancey, 1984] William J. Clancey. "Classification Problem Solving". In *Proceedings of the AAAI*, pages 49–55, 1984.

[Langari and Tomizuka, 1990] G. Langari and M. Tomizuka. "Stability of Fuzzy Linguistic Control Systems". In *Proceedings of the IEEE Conference on Decision and Control*, Dec 1990.

[Lea, 1988] R. Lea. "Automated Space Vehicle Control and Rendezvous Proximity Operations". *Telematics and Informatics*, 5(3):179–185, 1988.

[Lee, 1990a] C.C. Lee. "Fuzzy Logic in Control Systems: Fuzzy Logic Controller - Part I". *IEEE Trans. Systems, Man, and Cybernetics*, 20:404–418, Mar 1990.

[Lee, 1990b] C.C. Lee. "Fuzzy Logic in Control Systems: Fuzzy Logic Controller - Part II". *IEEE Trans. Systems, Man, and Cybernetics*, 20:419–435, Mar 1990.

[Payton *et al.*, 1990] D.W. Payton, J.K. Rosenblatt, and D.M. Keirsey. "Plan Guided Reaction". *IEEE Trans. Systems, Man, and Cybernetics*, 20:1370–1382, Nov 1990.

[Payton, 1986] D. W. Payton. "An Architecture for Reflexive Autonomous Vehicle Control". In *IEEE Conference on Robotics and Automation*, pages 1838–1845, 1986.

[Pfluger *et al.*, 1992] N. Pfluger, J. Yen, and R. Langari. "A Defuzzification Strategy for a Fuzzy Logic Controller Employing Prohibitive Information in Command Formulation". In *Proceedings of the Fuzz-IEEE Conference*, San Diego, CA, March 1992.

[Yen and Pfluger, 1991a] J. Yen and N. Pfluger. "Designing an Adaptive Path Execution System". In *Proceedings of the IEEE/SMC Conference*, pages 1459–1464, Charlottesville, VA, October 1991.

[Yen and Pfluger, 1991b] J. Yen and N. Pfluger. "Path Planning and Execution Using Fuzzy Logic". In *Proceedings of the AIAA Conference on Guidance, Navigation, and Control*, pages 1691–1698, New Orleans, LA, August 1991.

[Zadeh, 1965] L. A. Zadeh. "Fuzzy Sets". *Inform. Contr.*, 8(3):338–353, June 1965.

[Zadeh, 1978] L. A. Zadeh. "Fuzzy Sets as a Basis for a Theory of Possibility". *Fuzzy Sets and Systems*, 1:3–28, 1978.