International Conference on Fuzzy Logic and Applications
(Fuzzy'97), May 18-21, 1997, Zichron Yaakov, Israel.

REAL-WORLD ROBOT NAVIGATION USING FUZZY REACTION AND DELIBERATION

Gal A. Kaminka
University of Southern California
Information Sciences Institute and Department of Computer Science
4676 Admiralty Way, Marina del Rey, CA 90292 USA
galk@isi.edu

Abstract

We report on the development of a fuzzy system providing a low-level obstacle-avoidance navigation for a robot in an office environment. The system employs reactive and deliberative rule-based behaviors in a unified manner, allowing them to compete and compromise in controlling the robot. The system is constructed hierarchically, and successfully employs less than 100 rules to control a 28-variable space. We show that the system acts distinctively different than crisp behavior arbitration approaches in that it allows for all behaviors to influence the control of the robot in parallel, each to degree to which it is appropriate. This approach is a step towards a unified view of reaction and deliberation as responses to an external and internal environments, respectively.

1. INTRODUCTION

Autonomous mobile robots have served for many years as valuable test-beds for AI research. Numerous studies have been performed towards developing robots that can carry out their tasks in complex dynamic environments while avoiding physical obstacles (e.g., [1], [2], [3], [5], [6]). We report on the design and implementation of a low-level navigation system which avoids obstacles while attempting to reach goal locations. The system is part of the YODA/F project, developing a robot that can navigate and carry out simple tasks (e.g., mail distribution, floor tours) in an office environment. As part of the ongoing work on this project, an earlier version of the robot (YODA)[1] has successfully participated (winning second-place) in the AAAI-96 National Robot Competition [10], [11].

---

[1] The older version uses the same architecture, but a non-fuzzy navigation controller.

YODA/F is a three-layer software architecture in the spirit of Atlantis [6] running on top of a DMR/MRV-3 robot about 120cm tall and weighing more than 50kg. The robot has 24 ultra-sonic long-range sonar sensors spread about the robot body and internal position and orientation detectors (x,y,ϕ). It accepts primitive commands controlling velocity, heading, etc. The software consists of a planner, responsible for high-level scheduling and planning tasks; an executive layer, responsible for making sure the plan is carried out, and the controller described in this report.

The controller is responsible for getting a sequence of goal locations from the executive layer, and driving the robot from one location to the next while avoiding any obstacles, inanimate or otherwise. If it cannot drive the robot all the way to the goal, it reports the actual distance successfully traveled to the executive layer and allows it to replan. We use the term *goal-directed obstacle avoidance* to note this local-minimum search, where the robot must try to get closer and closer to the goal if it is possible, as opposed to a smarter mechanism employing a global optimization search, where the robot may choose to get farther from the goal in order to bypass a known obstacle. The global behavior is carried out by the higher layers of the architecture, and is outside the scope of this report.

The design and implementation of the controller were influenced heavily by several important constraints. Robustness of the obstacle-avoidance system is critical because of the safety requirements which follow from the robot's mass and size. However, this constraint is hard to maintain: the sonar sensors are not reliable[2], the position and orientation sensors lose accuracy as the robot moves around, the distances are given only in approximate terms (±30cm), etc. In short, we are seeking reliable performance from a system whose sensors and information are inherently unreliable and inaccurate.

2. FUZZY BEHAVIORS

Based on the idea of reactive behavior-based architectures ([1], [2], [3]) we have modeled the design of the controller on a competition between two behaviors for control of the robot. An *obstacle-avoidance* behavior selfishly attempts to keep the robot at a distance from any physical object which it can sense through the sonars. It will turn the robot away from obstacles, and reduce the speed or even stop the robot if they are too close. This behavior is *reactive*, in the sense that it reacts to the environment in attempting to carry out its task, without considering internal models of the

[2] Objects as close as 10 cm were sometimes sensed to be as much as 2 meters away.

world or goals. If the robot is not close to any object, this behavior will not change its heading or position. If the robot is inside a hallway, this behavior will attempt to keep the robot away from the walls, but will have no "preference" on which way is otherwise best.

A second behavior is a *goal-seeking* behavior. It attempts to drive the robot closer and closer to the location provided by the executive layer of the architecture. It doesn't take into account any obstacles that are in the way. It is *deliberative* in the sense that the behavior mostly ignores the robot's surroundings, responding instead to internal goals based on models of the world (as existing in the higher levels of the architecture).

While crisp examples of behavior-based competition choose between all the behaviors competing for influence ([1], [2], [3], [10], [11]) the fuzzy approach allows for gradual compromises to be made, allowing both behaviors to influence the output to the degree that they are appropriate. Thus, if the robot is very close to a wall, the influence of the obstacle-avoidance behavior is stronger, and causes the robot to turn away and slow down. When the way is clear, the influence of the goal-seeking behavior is stronger, and leads the robot towards the goal. In most cases, a compromise results where the robot both avoids obstacles to the degree that they are close, and heads towards the goal, to the degree that it is far.

3. COUNTERING RULE EXPLOSION

Given the inherent approximate nature of the information processed in the system, a fuzzy rule-based approach seems natural for implementing the behaviors. However, the system must deal with a large variable space: 27 input variables (24 sonars, orientation, goal distance and heading), and 2 outputs (velocity, and the change to the current heading). Since fuzzy systems are vulnerable to the problem of an exponential number ([7], [8]) of rules (in the size of the variable space), we needed ways to reduce the number of rules while still maintaining acceptable performance. We have managed to develop a working controller which implements the two behaviors in under 100 rules by utilizing two strategies: we built the controller as a hierarchical system, and we preferred writing rules with large coverage of the input space.

3.1 Hierarchical Rule-Based System

It is important to note that sonars positions are absolute - as the robot turns, its wheels turn in place without rotating the body, and the sonars do not shift their position; i.e., if a sonar was pointing North, it will point North always, regardless of whether the robot has turned. The implication of this is that the sonar sensor pointing in the forward direction may actually be a different sensor from reading to reading, depending on the current orientation of the robot. In other words, any rule referring to the relative-direction sonar reading (e.g., the forward sonar), such as:

IF <u>ForwardSonar=*Blocked*</u> **THEN** <u>Speed=*Stop*</u>

is actually to be implemented in the controller by a rule for each of the 24 sonar/orientation combinations:

IF <u>Sonar-1=*Blocked*</u> **AND** <u>Orient=*Towards-1*</u> **THEN** <u>Speed=*Stop*</u>

.....

IF <u>Sonar-24=*Blocked*</u> **AND** <u>Orient=*Towards-24*</u> **THEN** <u>Speed=*Stop*</u>

This expansion will occur whenever we refer to the any sonar reading by its relative position. Assuming we refer to each sonar in ω rules, we would require $24*\omega*24 = 576*\omega$ rules. Not only does this explosion hurt severely the expected performance of the controller, but it also makes the rules tedious to specify and harder to maintain. It is easier for the rule-writer to conceptualize the behavior of the robot as in the single-rule version above.

We therefore structured the controller as two rule-bases (Fig. 1) which work in sequence, creating a two-level hierarchical system [9]. The first rule-base simply computes the relative readings - Forward, Back-Left, Right, etc. - and passes the results for processing in the second rule-base, which can now be much smaller and easier to work with. Notice that this eliminates the need for the variable holding the current orientation of the robot. The number of rules now needed is 24*24 for the first rule-base, and $24*\omega$ for the second, for a total of $576+24*\omega$. To find out the minimal ω that will make sure the trade off is beneficial in the number of rules we solve the inequality $576+24*\omega < 576*\omega$ and get $\omega>24/23$. Intuitively, the more we use the same variable combination, the more we save from computing it in the first stage.

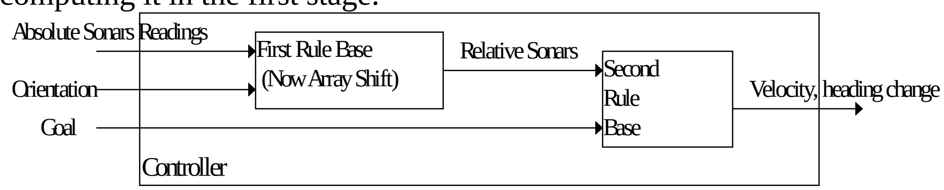

Figure 1.

This restructuring reduces substantially the number of rules required to process, and enhances the readability and maintainability of the rules. An additional advantage is the added modularity of the controller. Any of the two serial components may be replaced by an equivalent algorithm if it proves to be better. Such was the case with the first rule-base, where the relative-directions can be more efficiently computed by sensor-array shifts.

3.2 Simple Rules - Broad Coverage

The second rule-base, although simpler, is still handling 28 variables, and is thus still very much at risk of rule-explosion. Another strategy effective in reducing the number of rules was to employ a *selfish rule strategy*. We preferred writing rules which do not attempt to make decisions based on a global criteria, but instead tested only a single sonar sensor on the IF side, with appropriate "selfish" output set on the THEN side. The center-of-gravity defuzzification [7], [8] combines all the fired rules' effects into a coherent decision on the output commands issued to the robot. Simple rules provide a broad coverage of the input space. This effectively reduces the overall number of rules required, and provides coarse-grain approximation which is sufficient for our purposes.

An interesting thing to note is that the initial design rationale of separating the controller into two behaviors made the rules easier to write. For example, since the reactive behavior does not take into account the goal location, the rules implementing it did not need to match the goal location sets in the IF side. Similarly, rules implementing the deliberative goal-seeking behavior had little to do with the state of the sensor readings.

4. RESULTS AND DISCUSSION

The system described in this report has been completely implemented on the robot hardware, and the experiments used to tune and modify the rules were all carried out in the real-world environment in which YODA/F is to operate. An analysis of the controller and its performance shows that several strategies combined together in making a successful solution.

The fuzzy behavior-based approach and its unified treatment of reactive and deliberative behaviors of the controller offers an alternative to the behavior arbitration scheme offered by [3], and used in the earlier version of YODA [10], [11]. Instead of choosing a single behavior which would be appropriate in a given situation, the fuzzy approach allows all behaviors to influence the robots actions in parallel, each to the degree to which it is appropriate. This structure results in an emergent cooperation between the behaviors as they together control the robot's actions. The resulting behavior of YODA/F imitates biological creatures to some degree. Different resulting behaviors may be achieved by changing the weights of the obstacle-avoidance and goal-seeking behaviors: If the weights of the former are increased, the robot acts as if afraid of passing in narrow places. If the weights of the latter are increased instead, the robot is "hungrier" and bolder in attempting to reach the goal.

The approach also shows how reactive and deliberative behaviors may be conceptually and practically unified. Reactive behaviors respond directly to the external environment as perceived by the sensors and do not maintain an internal model of the world to guide their behavior. Deliberative behaviors present the other extreme - they reason using an internal model of the world and do not consider sensory input - they respond to the internal environment, composed of beliefs and goals. The fuzzy controller constructed treats these two types of behaviors equally. The only difference between them is in the source of the input they use - external (sensors) or internal (beliefs and goals).

Although conceptually the above ideas may be attractive in themselves, their practical implementation as fuzzy rule-bases was an important factor in the resulting structure and design of the actual controller. The potential exponential explosion of the number of rules used to control the robot with its large variable space was a major motivation for moving from a single flat rule-base to a hierarchical system, employing two components working in sequence, rather than parallel. There are theoretical results of the benefits of hierarchical systems, and the optimal structure of such systems [9]. But restructuring the flat controller system for YODA/F as a two-level hierarchy proved to be beneficial both in terms of computation (which is to be expected from the theory) as well as design; because the abstractions taken corresponded to the abstractions made naturally by the human designer, the rules were easier to maintain. However, we cannot expect every system to have this design characteristic.

We also have managed to reduce the number of rules used in the second rule-base by writing rules with very large coverage of the variable space. Such rules, which provide coarse-grain approximation of the behavior function, may not be appropriate for systems were the required behavior is intricate and complex, since this will require finer-grain approximation.

5. FUTURE WORK

We plan to continue our efforts in addressing current issues emerging from the controller architecture as well as future extensions. We hope that learning will improve performance and allow planning to emerge as a property of the architecture, rather than an arbitrary structure. A major source of concern is the time of matching all the rules against the various inputs. We are investigating ways to address this issue in the future.

Acknowledgments

We wish to thank Dr. Nevatia of USC and ISI's robot-fans group for their support. Special thanks to Jafar Adibi, and the LittleStones.

References

[1] Arkin R. C., "Reactive Robotic Systems", In *The Handbook of Brain Theory and Neural Networks*, Arbib M. A., Ed., MIT Press, pp. 793-796, 1995.

[2] Arkin R. C., "The impact of cybernetics on the design of a mobile robot system: A case study", *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE Systems, Man, and Cybernetics Society, Vol. 20 (6), pp. 1245-1257, 1990.

[3] Brooks R., "A robust layered control system for a mobile robot", *IEEE Journal of Robotics and Automation*, Vol. 2, pp. 14-23, 1986.

[4] Denning Robot Manual, *Denning MRV-3 Product Manual*, Denning Mobile Robotics, Inc., 1989.

[5] Fikes, R. E., and Nilsson N. J., "STRIPS: a new approach to the application of theorem proving to problem solving" *Artificial Intelligence*, Vol. 2 (3), pp. 189-208, 1971.

[6] Gat E., "Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots", In *Proceedings of the Tenth National Conference on Artificial Intelligence*, San Jose, California, pp. 809-815, 1992.

[7] Kosko B., "Optimal Fuzzy Rules Cover Extrema" *International Journal of Intelligent Systems*, John Wiley & Sons, Inc., Vol. 10 pp. 249-255, 1995.

[8] Kosko B., "Additive Fuzzy Systems: From Function Approximation to Learning", In *Fuzzy Logic and Neural Network Handbook*, Chen C. H., Ed., McGraw-Hill, Inc., pp. 9.1 - 9.22, 1996.

[9] Raju G. V. S. and Zhou J., "Adaptive Hierarchical Fuzzy Controller" *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE Systems, Man, and Cybernetics Society, Vol. 23 (4), pp. 973-980, 1993.

[10] Shen W., Adibi J., Cho B., Kaminka G., Kim J., Salemi B., and Tejada S., "YODA: The Young Observant Discovery Agent", *In Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Portland, Oregon, Vol. 2, pp. 1357, 1996.

[11] Shen W., Adibi J., Cho B., Kaminka G., Kim J., Salemi B., and Tejada S., "YODA: The Young Observant Discovery Agent", *AI Magazine*, Spring 1997 (To Appear).