Combining Multiple Goals in a Behavior-Based Architecture

Julio K. Rosenblatt Charles E. Thorpe

Robotics Institute
Carnegie Mellon University
Pittsburgh PA 15213
{jkr,cet}@cmu.edu

Abstract

Our experience over the years with different architectures and planning systems for mobile robots has led us to a distributed approach where an arbiter receives votes for and against commands from each subsystem and decides upon the course of action which best satisfies the current goals and constraints of the system. Centralized arbitration of votes from distributed, independent decision-making processes provides coherent, rational, goal-directed behavior while preserving real-time responsiveness to its immediate physical environment.

The Distributed Architecture for Mobile Navigation (DAMN) has been successfully used to integrate various independently developed subsystems, providing systems that perform road following, cross-country navigation, or teleoperation while avoiding obstacles and meeting mission objectives. Examples of implemented systems are given. Further research will seek to more rigorously define the behavior of the system.

Keywords: mobile robots, architecture, behaviors, distributed, voting, arbitration

1 Introduction

In order to function in unstructured, unknown, or dynamic environments, a mobile robot must be able to perceive its surroundings and generate actions that are appropriate for that environment and for the goals of the robotic system. Mobile robots need to combine information from several different sources. For example, the CMU Navlab vehicles are equipped with sensors such as video cameras, laser range finders, sonars, and inertial navigation systems, which are variously used by subsystems that follow roads, track paths, avoid obstacles and rough terrain, seek goals, and perform teleoperation. Because of the disparate nature of the raw sensor data and internal representations used by these subsystems, combining them into one coherent system which combines all their capabilities has proven to be very difficult.

To function effectively, an architectural framework for these sensing and reasoning processes must be imposed to provide a structure with which the system may be developed, tested, debugged, and understood. However, the architecture should serve as an aid, not a burden, in the integration of modules that have been developed independently, so it must not be overly restrictive. It must allow for purposeful goal-oriented behavior yet retain the ability to respond to potentially dangerous situations in real-time while maintaining enough speed to be useful.

Deliberative planning and reactive control are equally important for mobile robot navigation; when used appropriately, each complements the other and compensates for the other's deficiencies. Reactive components provide the basic capabilities which enable the robot to achieve low-level tasks without injury to itself or its environment, while deliberative components provide the ability to achieve higher level goals and avoid mistakes which could lead to inefficiencies or even mission failure.

Hierarchical approaches allow slower abstract reasoning at the higher levels and faster numerical computations at the lower levels, thus allowing varying trade-offs between responsiveness and optimality as appropriate at each level [11] [1]. While such an approach provides aspects of both deliberative planning and reactive control, the top-down nature of hierarchical structures tends to overly restrict the lower levels [13]. In hierarchical architectures, each layer controls the layer beneath it and assumes that its commands will be executed as expected; this introduces a need to monitor the progress of desired actions and to report failures as they occur [20], thus increasing complexity.

Rather than imposing a top-down structure to achieve this desired symbiosis of deliberative and reactive elements, the Distributed Architecture for Mobile Navigation takes an approach where multiple modules concurrently share control of the robot by sending votes to be combined rather than commands to be selected [16].

2 The Distributed Architecture for Mobile Navigation

Figure 1 shows the organization of the Distributed Architecture for Mobile Navigation (DAMN), in which individual behaviors such as road-following or obstacle-avoidance send votes to the command arbitration module; these inputs are combined and the resulting command is sent to the vehicle controller. Each action-producing module, or *behavior*, is responsible for a particular aspect of vehicle control or for achieving some particular task; it operates asynchronously and in parallel with other behaviors, sending its outputs to the arbiter at whatever rate is appropriate for that particular function. Each behavior is assigned a weight reflecting its relative priority in controlling the vehicle. A mode manager may also be used to vary these weights during the course of a mission based on knowledge of which behaviors would be most relevant and reliable in a given situation.

Like other distributed architectures, DAMN enjoys several advantages over a centralized one, including greater reactivity, flexibility, and robustness. While all votes must pass through the command arbiter before an action is taken, the function provided by the arbiter is fairly simple and does not represent the centralized bottleneck of more traditional systems. However, unlike reactive systems such as the Subsumption Architecture [3], the perception and planning components of a behavior are not prohibited from maintaining complex internal representations of the world. It has been argued that "the world is its own best model" [4], but this assumes that the vehicle's sensors and the algorithms which process them are essentially free of harmful noise and that they can not benefit from evidence combination between consecutive scenes. In addition, disallowing the use of internal representations requires that all environmental features of immediate interest be visible to the vehicle sensors at all times, unnecessarily constraining and reducing the flexibility of the overall system.

2.1 Command Arbitration

In a distributed architecture, it is necessary to decide which behaviors should be controlling the vehicle at any given time. In some architectures, this is achieved by having priorities assigned to each behavior; of all the behaviors issuing commands, the one with the highest priority is in control and the rest are ignored [3] [18]. In order to allow multiple considerations to affect vehicle actions concurrently, DAMN instead uses a scheme where each behavior votes for or against each of a set of possible vehicle actions [17]. An arbiter then performs *command fusion* to select the most appropriate action. While the Motor Schema framework [2] also offers a means of fusing commands from multiple behaviors, it suffers from the well known problem of local minima in potential fields. Another, perhaps more serious problem, is that arbitration via vector addition can result in a command which is not satisfactory to any of the contributing behaviors. DAMN arbiters do not average commands, but rather select the command which has the most votes from the behaviors.

Each behavior generates a vote between -1 and +1 for each possible action; in the case of the turn arbiter, the possible actions are a discrete set of steering commands. At each iteration, the arbiter collects any new votes that the behaviors may have sent since the last cycle, computes a normalized weighted sum of the votes, and determines which command has the maximum vote value. In order to avoid problems with discretization such as biasing and "bang-bang" control (i.e., alternating between discrete values in order to achieve an intermediate value), the arbiter performs sub-pixel interpolation. This is done by performing Gaussian smoothing, selecting the command option with the highest value, and then fitting a parabola to that value and the ones on either side; the peak of that parabola is used as the command to be issued to the controller. This process, which is repeated at a rate of 10 Hz, is illustrated in Figure 2, where the votes from two behaviors (a & b) are linearly combined (c), and then smoothed and interpolated to produce the resulting command (d). This is similar to defuzzification in Fuzzy

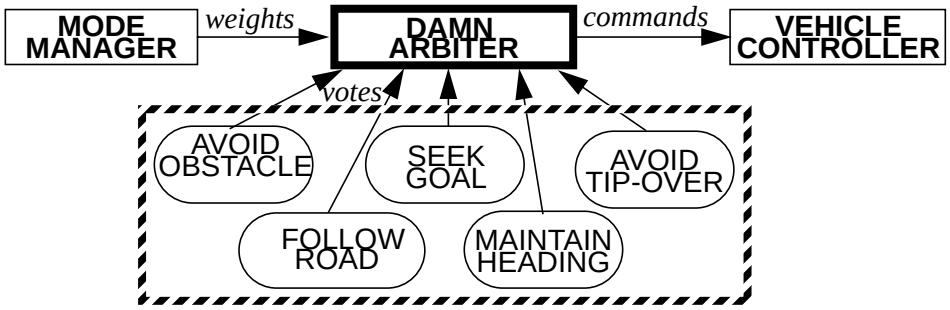

Figure 1: Behaviors sending votes to arbiter

Logic systems [9]; indeed an architecture has been implemented which recasts DAMN into a Fuzzy Logic framework [24].

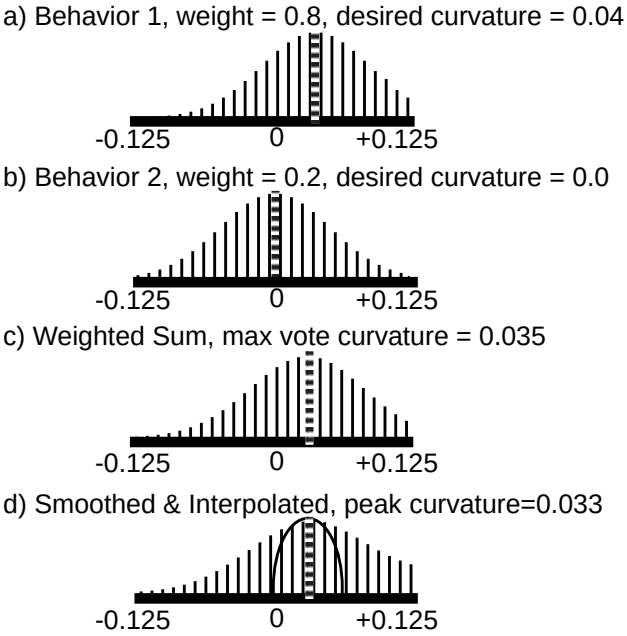

Figure 2: Command fusion process

A field of regard arbiter and its associated behaviors have also been recently implemented and used for the control of a pair of stereo cameras on a pan/tilt platform. Behaviors vote for different possible field of regard polygons (the camera field of view mapped on to the ground plane); the arbiter maintains a local map of these votes and transforms them as the vehicle moves. At each iteration, these votes are mapped into a pan-tilt space, and then smoothed and interpolated as described above. Figure 3 shows the votes for field of regard polygons generated by an behavior which votes against looking in the direction of known obstacles since travelling in that direction is impossible. Behaviors also vote based on considerations such as looking toward the goal and looking at a region contiguous to already mapped areas. The darkest polygon in the figure corresponds to the pan and tilt angles selected by the arbiter.

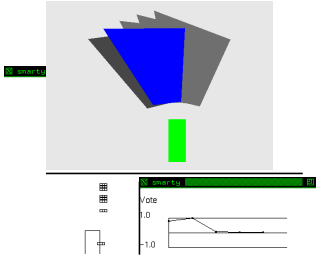

Figure 3: Field of regard voting and arbitration

2.2 Behaviors

Within the framework of DAMN, behaviors must be defined to provide the task-specific knowledge for controlling the vehicle. Since behaviors may be arbitrarily deliberative or reflexive, DAMN is designed so that behaviors can issue votes at any rate; for example, a reflexive behavior may operate at 10 Hz, another behavior may maintain local state and operate at 1 Hz, while a global planner may issue votes at a rate of 0.1 Hz. The use of distributed shared control allows multiple levels of planning to be used in decision-making without the need for an hierarchical structure. However, higher-level reasoning modules may still exert meta-level control within DAMN by modifying the voting weights assigned to behaviors.

Safety Behaviors. A basic need for any mobile robot system is the ability to avoid situations hazardous to itself or to other objects in its environment. Therefore, an important part of DAMN is its "first level of competence"[3], which consists of behaviors designed to provide vehicle safety. In contrast to priority-based architectures which only allow one behavior to be effective at any given moment, the structure of DAMN and its arbitration scheme allow the function of these safety behaviors to be preserved as additional levels of competence are added.

The *Obstacle Avoidance* behavior receives a list of current obstacles in vehicle-centered coordinates and evaluates each of the possible command options, as illustrated in Figure 4. The source of these obstacles may be intraversable regions of terrain determined by range image processing or stereo vision, by sonar detection of objects above the ground plane, or any other means of obstacle detection as appropriate to the current task and environment [5][8].

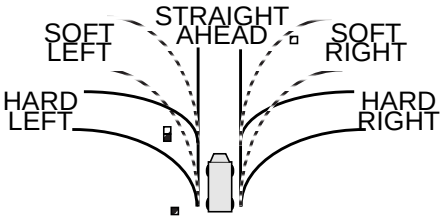

Figure 4: Arc evaluation in the *Obstacle Avoidance* behavior

Another vital aspect of vehicle safety is insuring that the commanded speed and turn stay within the dynamic constraints of the vehicle as it travels over varying terrain conditions. The *Limit Turn* behavior sends votes to the arbiter that reflect these constraints, voting against commands that violate them.

Road Following. Once vehicle safety has been assured by the obstacle avoidance and dynamic constraint behaviors, it is desirable to integrate task-level modules such as the ALVINN road following system [15]. In the case of ALVINN, creating a behavior that independently evaluated each arc was relatively straightforward. The output units of a neural network represent evaluations of particular turn commands; these units are simply resampled to the DAMN turn vote space, using a Gaussian of the appropriate width. This process is illustrated in Figure 5.

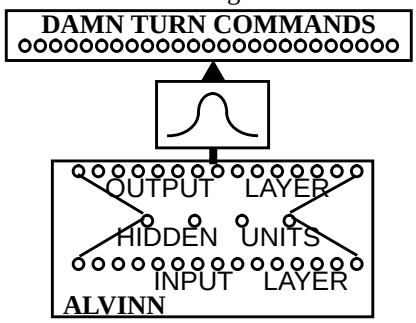

Figure 5: Resampling of ALVINN output layer

Teleoperation. The STRIPE teleoperation system [6] allows an operator to designate waypoints for the vehicle; STRIPE determines an appropriate steering commands and sends a set of votes representing a Gaussian centered on the desired command. This allows the dynamic constraints and obstacle avoidance behaviors to be used in conjunction with STRIPE so that the safety of the vehicle is still assured.

Goal-Directed Behaviors. While the lower-level behaviors operate at a high rate to ensure safety and provide functions such as road following and obstacle avoidance, higher-level behaviors are free to process information at a slower rate and periodically issue votes to the arbiter that guide the robot towards the current goal.

The *Goal Seeking* behavior is one way to provide this capability. This simple behavior directs the vehicle toward a series of goal points specified in global coordinates either by the user [8] or by a map-based planner [7]. The desired turn radius is transformed into a series of votes by applying a Gaussian whose peak is at the desired turn radius and which tapers off as the difference between this turn radius and a prospective turn command increases.

Some more sophisticated map-based planning techniques based on the A* search algorithm [10] have also been integrated and used within the DAMN framework. These planners use dynamic programming techniques to determine an optimal global path; however, an important point is that they do not hand a plan down to a lower level planner for execution, but rather maintain an internal representation that allows them to participate directly in the control of the vehicle based on its current state.

For example, the D* planner [21] creates a grid with information on how best to reach the goal from any location in the map. The global plan is integrated into DAMN as a behavior by determining, for each possible turn command, the distance to the goal from a point along that arc a fixed distance ahead (as shown in Figure 6).

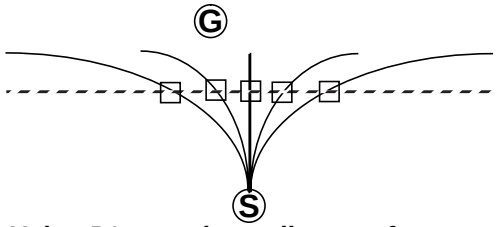

Figure 6: Using D* to evaluate distance from goal

2.3 Evolutionary Development and Integration

DAMN is designed so that various behaviors can be easily added or removed from the system, depending on the current task at hand. Although the modules described above all use very different paradigms and representations, it has been relatively straightforward to integrate each and every one of them into the framework of DAMN. Sensor fusion is not necessary since the command fusion process in DAMN preserves the information that is critical to decision-making, yielding the capability to concurrently satisfy multiple objectives without the need for centralized bottlenecks. A detailed description of an implemented system and the experimental results achieved can be found in [8].

The voting strengths, or weights, of each behavior are specified by the user or by a mission planning module. DAMN is fairly insensitive to the values of these weights and the system performs well without a need to tweak these parameters. For example, the *Obstacle Avoidance* behavior and the *Seek Goal* behavior have been used with weights of 0.75 and 0.25 and of 0.9 and 0.1, respectively, and in both cases the robot successfully reached its goal while avoiding obstacles. The voting weights of each behavior can also be modified by a mode manager module such as the Annotated Maps system, which was integrated with DAMN to provide this capability [23].

All of the behaviors described in Section 2.2 have been used in conjunction with each other in various configurations, yielding systems that were more capable than they would have been otherwise. Conceptually, three levels of competence have been implemented in DAMN thus far, as shown in Figure 7. These levels of competence are convenient for describing the incremental manner in which the system's capabilities evolve; however, it is important to note that all behaviors co-exist at the same

level of planning. The importance of a behavior's decisions is reflected by the weighting factor for its votes, and is in no way affected by the level of competence in which it is described.

The safety behaviors are used as a first level of competence upon which other levels can be added. Movement is the second level of competence that has been implemented; road following, cross-country navigation, and teleoperation behaviors have all been run together with the obstacle avoidance behavior to provide various forms of generating purposeful movement while maintaining safety [23]. The third level of competence is comprised of the various map-based goal-seeking behaviors. Cross-country behaviors have been combined with goal-oriented behaviors to produce directed off-road navigation [7] [13] [21].

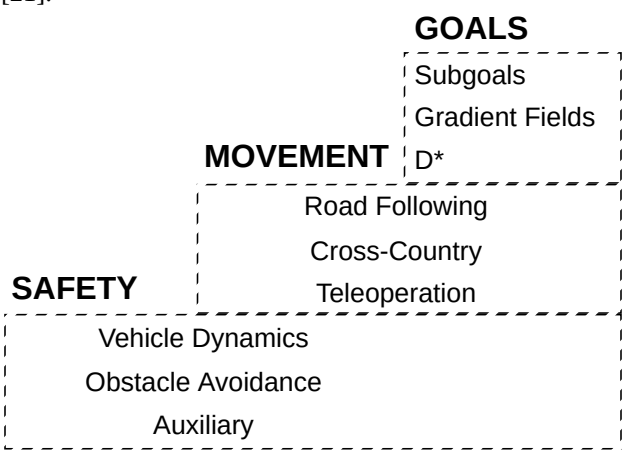

Figure 7: Evolutionary development in DAMN

2.4 Future Work

DAMN has proven to be very effective as an architecture which greatly facilitates the integration of a wide variety of different vehicle navigation subsystems; however, as DAMN is used for increasingly complex tasks where many behaviors may be issuing votes concurrently, there will be a greater need to have the semantics of the voting process defined more carefully. By explicitly representing and reasoning about uncertainty within the decision-making processes, a system can be created whose effects are well-defined and well-behaved.

We have decided to pursue utility theory as a means of defining the semantics of the voting and arbitration processes. Because we are attempting to decide which among a set of possible actions to take, it is natural to make judgments on the usefulness of each action based on its consequences. If we assign a utility measure *U(c)* for each possible consequence of an action, then the *expected utility* for an action *a* is:

$$U(a) = \sum_{c} U(c) \cdot P(c \mid a, e)$$

where *P(c|a,e)* is the probability distribution of the consequence configuration *c*, conditioned upon selecting action *a* and observing evidence *e* [14]. Thus, if we can define these utilities and probabilities, we can then apply the *Maximum Expected Utility* (MEU) criterion to select the optimal action based on our current information.

If utility theory is to be applied to the problem of evidence combination and action selection for mobile navigation tasks, a means for defining the utility functions must be provided. Behaviors are defined in order to achieve some task, so it is fair to assume that there must be at least an implicit measure of "goodness" or utility with respect to that task. For example, an obstacle avoidance behavior's task is to maximize distance to obstacles, so that the distance from the vehicle to the nearest obstacle could be used as a measure of goodness. Likewise, proximity to the current goal could be used for the goal-based behaviors, as proximity to the center of a road (or lane) could be used by road following modules.

We will also be investigating command fusion using a local map for turn and speed arbitration, as was done for the field of regard arbiter. This will allow the arbiter to ensure consistency in the voting process by transforming votes as the vehicle moves. This will also allow behaviors to vote on where to travel rather than how to steer; however, complications due to the non-holonomic constraints of the vehicle will need to be addressed.

Another issue that needs to be addressed is the coupling between turn and speed commands. Currently, the turn and speed arbiters operate independently. Turn behaviors can use vehicle speed as one of their inputs and vice versa; however, there is currently no mechanism which allows behaviors to vote for a combination of turn and speed. Research is ongoing to determine a command representation which allows both coupled and independent voting, and is still efficient enough for real-time purposes.

3 Conclusion

The Distributed Architecture for Mobile Navigation has been successfully used to create systems which safely follow roads or traverse cross-country terrain while avoiding obstacles and pursuing mission goals. DAMN imposes no constraints on the nature of the information or the processing within a behavior, only on the interface between the behavior and the command arbiter. Furthermore, the behaviors are not subject to timing constraints; each behavior operates asynchronously and in parallel. In addition to its use on the CMU Navlab vehicles, DAMN has also been used at Martin Marietta, Hughes

Research Labs, and Georgia Institute of Technology.

Like centralized or hierarchical architectures, DAMN is able to assimilate and use high-level information. Non-reactive behaviors may use plans to achieve goals and coordinate with other agents. However, like other behavior-based architectures, it also avoids sensory and decision-making bottlenecks and is therefore able to respond in real-time to external and internal events. Finally, unlike architectures with strictly prioritized modules, DAMN's vote arbitration scheme allows multiple goals and constraints to be fulfilled simultaneously, thus providing goal-oriented behavior without sacrificing real-time reactiveness.

Acknowledgments

The UGV project is supported by ARPA under contracts DACA76-89-C-0014 and DAAE07-90-C-R059, and by the National Science Foundation under NSF Contract BCS-9120655. Julio Rosenblatt is supported by a Hughes Research Fellowship. The authors would like to acknowledge the shared efforts and technical support of the Navlab group at Carnegie Mellon University, as well as the support and guidance of Dave Payton at the Hughes Research Labs.

References

[1] J. Albus, H. McCain, R. Lumia, NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM), NBS Tech. Note 1235, Gaithersburg, MD, 1987.

[2] R. Arkin, *Motor Schema Based Navigation for a Mobile Robot: An Approach to Programming by Behavior*, ICRA 1987.

[3] R. Brooks, *A Robust Layered Control System for a Mobile Robot*, IEEE Journal of Robotics and Automation vol. RA-2, no. 1, pp. 14-23, April 1986.

[4] R. Brooks, *Intelligence Without Reason*. Proc. IJCAI'93. 1993.

[5] M. Daily, Harris, J., Keirsey, D., Olin, K., Payton, D., Reiser, K., Rosenblatt, J., Tseng, D. and Wong, V., *Autonomous Cross-Country Navigation with the ALV*, in IEEE Conference on Robotics and Automation, Philadelphia, PA, April, 1988. (Also appears in DARPA Knowledge Based Planning Workshop, December, 1987 pp 20-1 to 20-10).

[6] J.S. Kay, C.E. Thorpe. *STRIPE Supervised Telerobotics Using Incremental Polygonal Earth Geometry.* In Proc. Intelligent Autonomous Systems Conference, 1993.

[7] Keirsey, D.M., Payton, D.W. and Rosenblatt, J.K., *Autonomous Navigation in Cross-Country Terrain*, in Proceedings of Image Understanding Workshop, Cambridge, MA, April, 1988.

[8] D. Langer, J.K. Rosenblatt, and M. Hebert, *A Behavior-Based System For Off-Road Navigation*, in IEEE Journal of Robotics and Automation, vol. 10, no. 6, pp. 776-782, December 1994; and *An Integrated System For Autonomous Off-Road Navigation*, in the Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, May 1994.

[9] C. Lee, *Fuzzy Logic in Control Systems: Fuzzy Logic Controller -- Parts I & II*, IEEE Transactions on Systems, Man and Cybernetics, Vol 20 No 2, March/April 1990.

[10]N. Nilsson, *Shakey the Robot*, SRI Tech. Note 323, Menlo Park, Calif., 1984.

[11]D.W. Payton, *An Architecture for Reflexive Autonomous Vehicle Control*, in IEEE International Conference on Robotics and Automation, San Francisco, CA, April 7-10, 1986, pp. 1838-1845. (invited)

[12]D.W. Payton, *Internalized Plans: A Representation for Action Resources*, *Robotics and Autonomous Systems*, 6(1), 1990, pp. 89-103. (Also in Designing Autonomous Agents, ed. Pattie Maes, MIT Press, Cambridge, Mass. 1991, pp. 89-103.)

[13]D.W. Payton, J.K. Rosenblatt, D.M. Keirsey. *Plan Guided Reaction*. IEEE Transactions on Systems Man and Cybernetics, 20(6), pp. 1370-1382, 1990.

[14]J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers, 1988.

[15]D. Pomerleau, *Neural Network Perception for Mobile Robot Guidance*, Ph.D. dissertation, Carnegie-Mellon Technical Report CMU-CS-92-115, 1992

[16]Rosenblatt, J. (1995), DAMN: A Distributed Architecture for Mobile Navigation, in proceedings of the 1995 AAAI Spring Symposium on Lessons Learned from Implemented Software Architectures for Physical Agents, H. Hexmoor & D. Kortenkamp (Eds.), Menlo Park, CA: AAAI Press.

[17]J.K. Rosenblatt and D.W. Payton. *A Fine-Grained Alternative to the Subsumption Architecture for Mobile Robot Control*. in Proc. of the IEEE/INNS International Joint Conference on Neural Networks, Washington DC, vol. 2, pp. 317-324, June 1989.

[18]S.J. Rosenschein and L.P. Kaelbling. *The Synthesis of Digital Machines with Provable Epistemic Properties*. In Proc. Theoretical Aspects of Reasoning about Knowledge. pp 83-98. 1986.

[19]G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, 1976.

[20]R. Simmons, L.J. Lin, C. Fedor, *Autonomous Task Control for Mobile Robots*, Proc. IEEE Symposium on Intelligent Control, Philadelphia, PA, September 1990.

[21]A. Stentz, *Optimal and Efficient Path Planning for Unknown and Dynamic Environments*, Carnegie-Mellon Technical Report CMU-RI-TR-93-20, 1993.

[22]A. Stentz, M. Hebert, *A Complete Navigation System for Goal Acquisition in Unknown Environments*, Carnegie-Mellon Technical Report CMU-RI-TR-94-7, 1994.

[23]C. Thorpe, O. Amidi, J. Gowdy, M. Hebert, D. Pomerleau, *Integrating Position Measurement and Image Understanding for Autonomous Vehicle Navigation*. Proc. Workshop on High Precision Navigation, Springer-Verlag Publisher, 1991.

[24]J. Yen, N. Pfluger, *A Fuzzy Logic Based Robot Navigation System*, AAAI Fall Symposium, 1992.