Proceedings of the 1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems

Optimal Wall Pose Determination in a Shared-Memory Multi-Tasking Control Architecture

Mohamed Dekhil and Thomas C. Henderson
Department of Computer Science
University of Utah
Salt Lake City, UT 84112

Abstract

In previous work, we proposed a new technique for the recovery of planar surfaces using two beam-spread sonar readings [9, 10, 11]. In this paper, we present an implementation of this technique in a wall-following algorithm using two sonar readings. This program is used as a client in a sensor-based distributed control scheme for mobile robots.

1 Introduction

In any closed-loop control system, sensors are used to provide the feedback information that represents the current status of the system and the environmental uncertainties. Sonar sensors in common use today (e.g., the Polaroid sensor) produce with reasonable accuracy the range to the nearest surface, but the direction to that surface is not explicitly determined; rather the surface is known to lie within a certain angle spread centered about the line of direction of the sensor (e.g., 22.5° for the Polaroid sensor). (See Figure 1.)

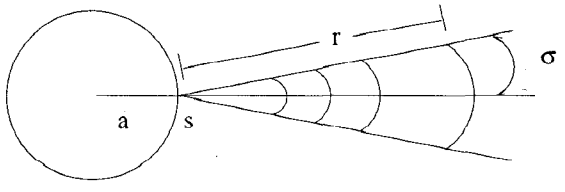

Figure 1: Beamspread of Sonar Sensor

Multiple sonar readings are required to disambiguate the location (pose) of the reflecting surface. Several researchers have investigated the use of sonar in mobile robotics [3, 6, 8, 16, 17], and others have directly addressed the problem of wall detection [1, 2, 13, 18], or have shown the minimum number and arrangement of sonar sensors to detect obstacles [14, 15]. We have previously addressed the optimal pose recovery of planar surfaces in sonar data [9, 10, 11]. In that work we posed the k-wall/m-sonar (kWmS) problem and solved 1W1S)

> **Problem:** Given m sonar transmitter/receiver sensors situated on a circular ring placed in a k wall enclosure, what is the optimal sensing strategy to determine the pose of the k walls?

The sonar sensor is assumed to have a non-zero beam spread (e.g., 22.5 degrees for a Polaroid sensor), and optimal is defined in terms of the recovery of the wall's pose with the minimum number of sensors used and moves made.

We used this technique to implement two simple wall-following programs. These programs are used as clients in a shared-memory multi-tasking architecture for controlling mobile robots. In the following sections we describe the control architecture and the incorporation of our technique in the control system.

2 Shared-Memory Multi-Tasking Control Scheme

This technique was used to implement several algorithms that require wall pose determination as part of a larger project for designing a distributed control scheme for mobile robots. In this scheme, several controllers (clients) work in parallel, competing for the server. The server selects the command to be executed based on a pre-defined priority scheme. Each of these clients has a certain task, and may use the sensor readings to achieve its goal. A special client with the task of avoiding obstacles is assigned the highest priority. The clients may also aquire the current state of the system and the command history to update their control strategy. This control scheme is similar to Brooks' subsumption architecture [4, 5] in that the controller is decomposed into parallel task achieving behaviors rather than information processing modules.

The logical sensor approach [12, 20], which we used to model the sensory system, allows flexible and modular

This work was supported in part by the Advanced Research Projects agency under Army Research Office grants number DAAH04-93-G-0420 and by NSF grant CDA 9024721.

0-7803-3700-X/96 $5.00 ©1996 IEEE

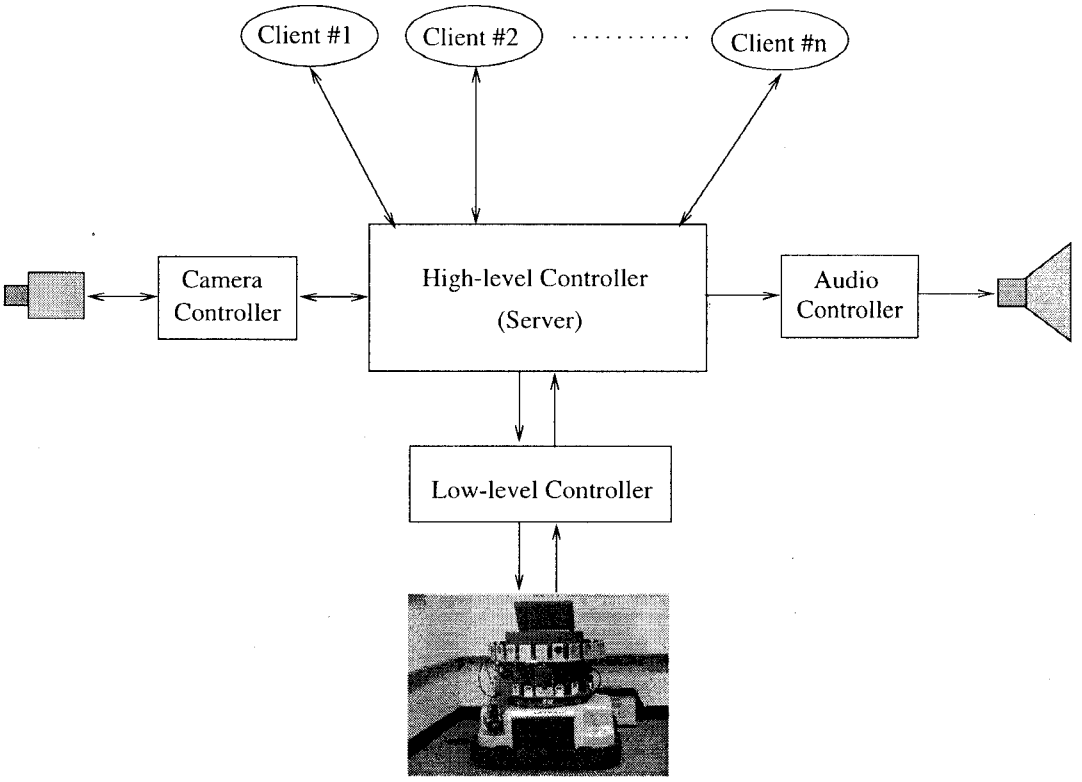

Figure 2: The proposed control scheme.

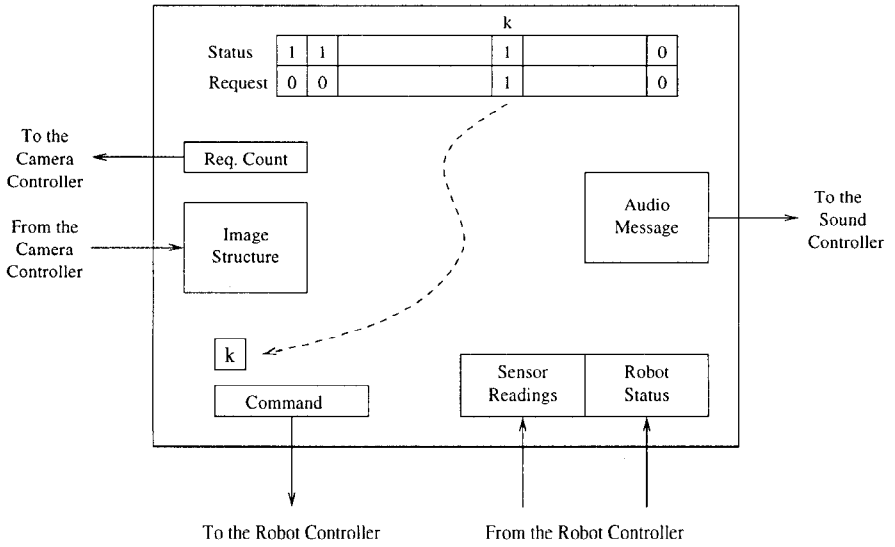

Figure 3: The shared memory structure in the server.

design of the controllers. It also provides several levels of data abstraction and tolerance analysis based on the sensor type and the required task. This approach is used to build high-level requests which may be used by the application programs.

Figure 2 shows a schematic view of the proposed control scheme. In this figure, there are three main resources the clients may compete for. These resources are the robot (Egor), the camera, and the speakers. Each of these resources has a low-level controller that carries out the required commands and returns the required information.

The task of the server is to manage the resources and to handle clients' requests. We use shared memory to perform the communication between the server and the clients. Figure 3 shows the shared memory structure used in the server.

2.1 Communication Protocols

In this control scheme, there are several clients competing to send commands to the server. Also, some of the clients need to know the sensor reading and the current status of the robot (position, orientation, etc.) which is available to the server. Therefore, we need to define a communication protocol to organize these commands, and to set a priority scheme for selecting the controlling client.

The protocol used can be described as follows:

1. A client sends a request to the server whenever it needs to issue commands to the robot.
2. If the priority of that client is higher than the client in control (if any), the control is passed to the requesting client. Otherwise the request is denied.
3. Once the client gains control, it starts sending commands to the server which directs these commands to the robot.
4. When the client finishes sending commands, or if a client with higher priority is requesting control, the current client relinquishes the control to the server.

Figure 4 shows a state diagram of the communication protocol.

Reading the status of the robot or the sensor values does not require any waiting. This information is part of the shared memory structure and is accessible by all clients at any time. This improves the speed of queries and reduces the bandwidth of requests to the server.

3 Experiments and Results

This control architecture was implemented and tested on a mobile robot called "LABMATE" designed by Transitions Research Corporation [22] to check the applicability and validity of the proposed control scheme, and the results were very encouraging [7].

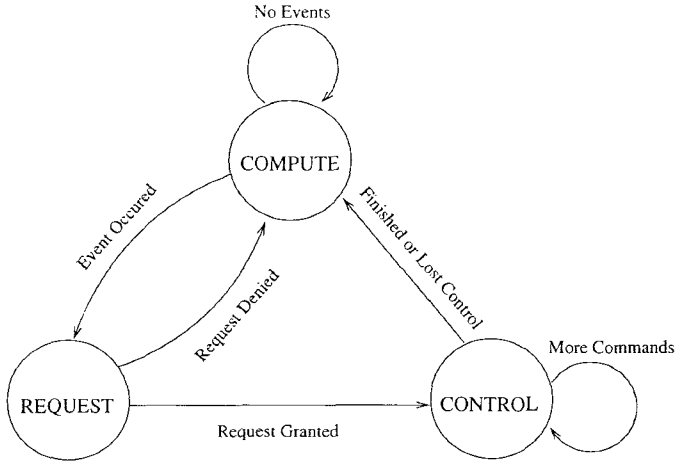

Figure 4: State diagram of the communication protocol.

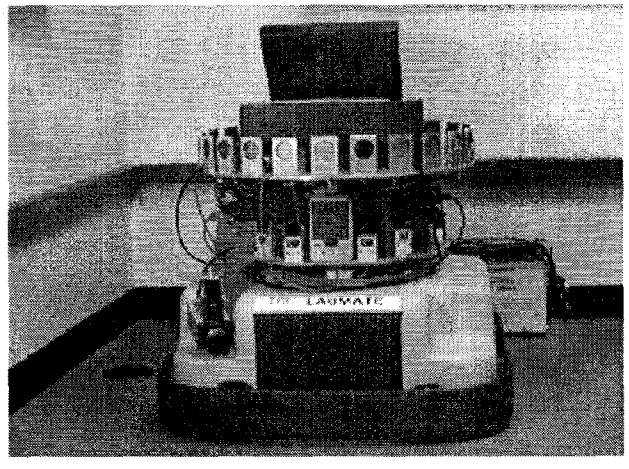

Figure 5: The LABMATE robot with its equipments.

The LABMATE was used for several experiments at the Department of Computer Science, University of Utah. It also was entered in the 1994 AAAI Robot Competition [19, 21]. For that purpose, the LABMATE was equipped with 24 sonar sensors, eight infrared sensors, a camera and a speaker. [1] Figure 5 shows the LABMATE with its equipment.

The hardware setup used in these experiments consisted of a PC running Linux to provide parallel processing capabilities, and an RS232 serial cable connecting the PC to the LABMATE. No special hardware was used and this setup was inexpensive and easy to use.

[1] The LABMATE preparations, the sensory equipments, and the software and hardware controllers were done by L. Schenkat and L. Veigel at the Department of Computer Science, University of Utah.

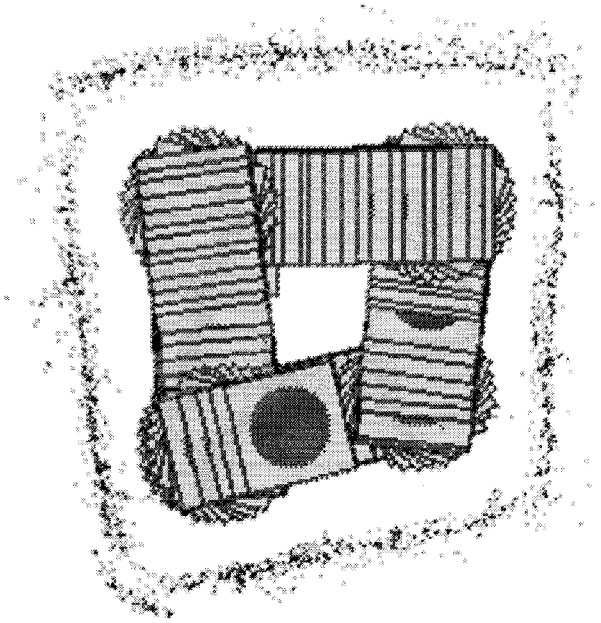

Figure 6: Simple wall following program using two sonar readings.

3.1 Experiment (1): Simple Wall-Following Client

Using the control scheme described above, we implemented a simple wall following client that used only two sonar readings to determine the wall orientation. In this experiment, there are two clients running in parallel, the "follow-wall" client, and the "go" client. The "go" client just sends commands to move the robot forward. The follow-wall client has a higher priority than the "go" client. Whenever there is a wall in front of the robot, the "follow-wall" client takes command and applies the proposed technique to determine the orientation of the wall, then it sends commands to the robot to rotate until it is parallel to the wall. After that it releases the control and the "go" client starts sending commands to move forward.

In this algorithm, we use only the front three sonars to detect if a wall is within a certain distance from the robot, then we use two out of these three sonars to determine the wall orientation. Once the robot is oriented parallel to the wall, the algorithm does not adjust for changes in the orientation of the wall as long as it is not in front of the robot. Figure 6 shows the robot path during this experiment.

To measure the accuracy and repeatability of this algorithm, we let the robot run for several rounds in the same enclosure, and we measured the distance to the wall to which it is parallel to. The results are shown in Figure 7. In this figure, the peaks represent transitions between walls. Between the peaks we can see the distance slightly changing due to the error in estimating the wall position. This error arises from the noise in the sonar reading since we are not using any averaging or filtering of data. Figure 8 on the other hand demonstrates the repeatability of the calculated angle. The figure shows the heading of the robot for about four repeated cycles, each cycle composed of four transitions.

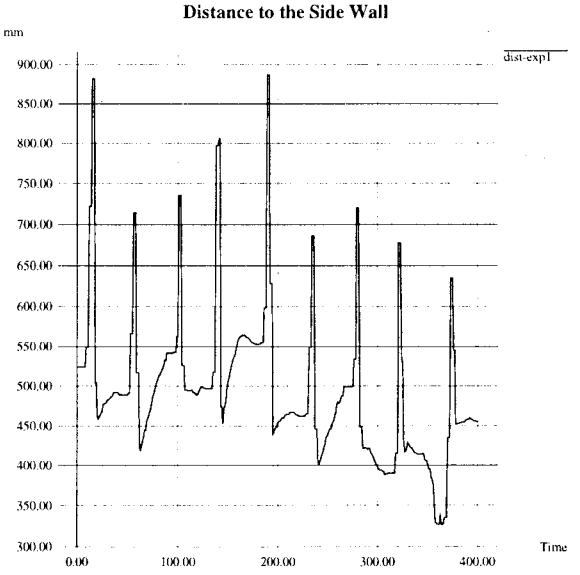

Figure 7: The distance between the robot and the wall it follows.

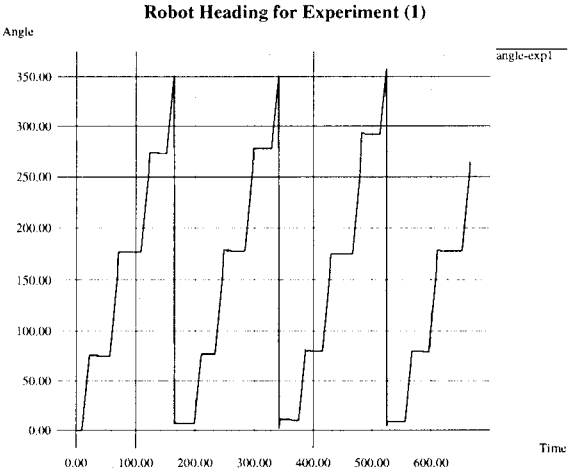

Figure 8: The robot heading for experiment (1).

3.2 Experiment (2): Adaptive Wall-Following Client

In the second experiment we added another client "follow-wall-2" which uses three sonars on the side of the robot close to a wall, and determine the angle required to correct the robot's heading to be parallel to that wall. The priority of this client is lower than that of the "follow-wall" client, but higher than the "go" client.

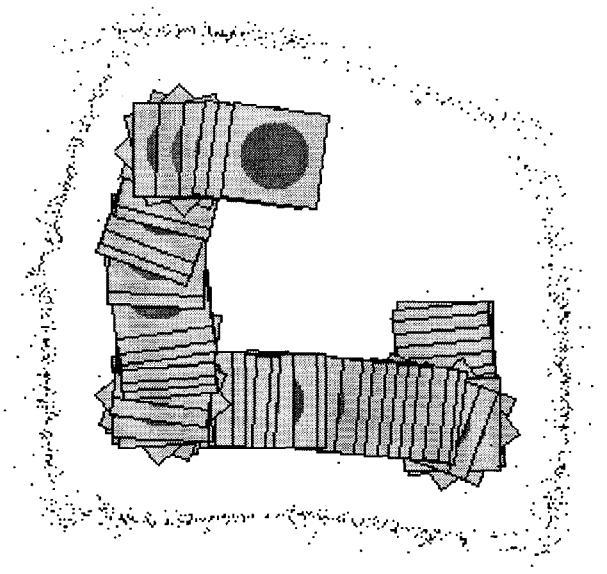

Figure 9: Adaptive wall following program.

The results of this experiment are shown in Figure 9.

As in the previous experiment, we tested the accuracy and repeatability of the algorithm. Figure 10 shows the distance between the robot and the wall it is following. Here we can see some variations in the distance due to the direction correction done by the "follow-wall-2" client. Figure 11 shows the heading of the robot for several cycles. Note the small corrections in the heading between transitions. However, these small corrections may cause unstable behavior of the robot heading since the sonar readings are not accurate while the robot is moving.

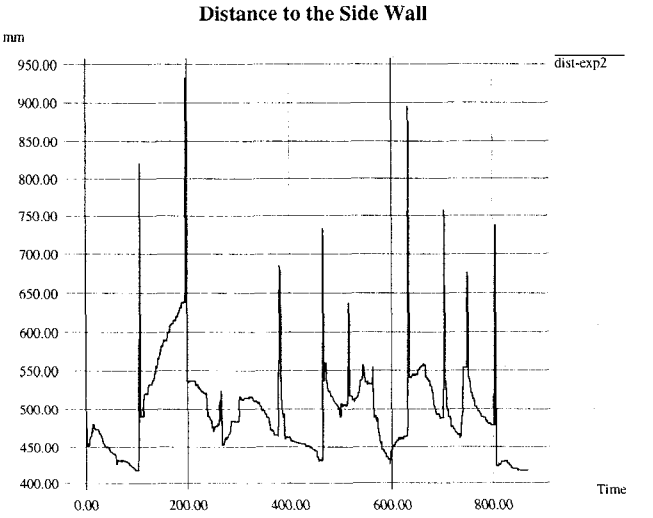

Figure 10: The distance between the robot and the wall it follows.

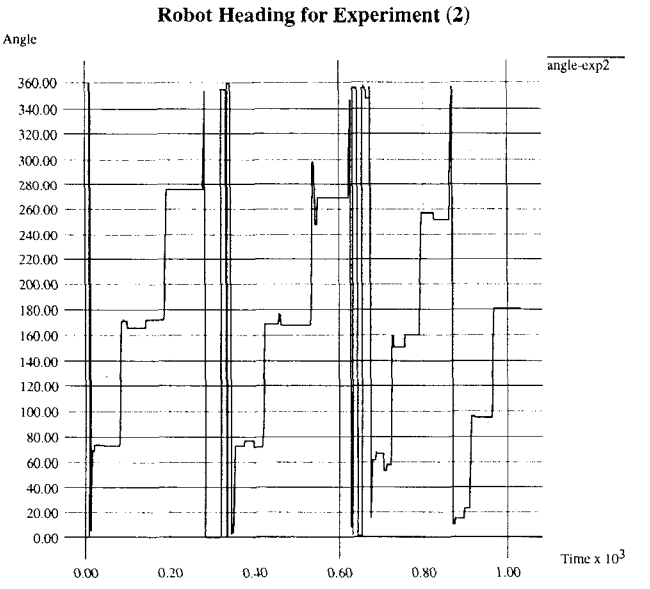

Figure 11: The robot heading for experiment (2).

4 Conclusion

The analysis and application of an optimal sonar sensing strategy is given for the use of two sonar readings for the recovery of wall positions in the environment. This technique can be used to generate hypotheses of wall surfaces which helps define precise strategies to take more data which corroborate or disconfirm the hypotheses. The underlying geometrical arguments may also be relevant to other kinds of sensors with similar beam spread physics. Simulation results and experiments on a mobile robot show the applicability and the accuracy of this technique.

The proposed technique has been demonstrated in a distributed control scheme on a mobile robot, in particular, in the implementation of the clients that require accurate determination of wall orientation and following. This demonstrates the utility of a simple, yet robust, use of an optimal sensing approach to the use of sonar data.

References

[1] BORENSTEIN, J., AND KOREN, Y. Error eliminating rapid ultrasonic firing mobile robot obstacle avoidance. *IEEE Journal of Robotics and Automation 11* (1995), 132–138.

[2] BOZMA, B., AND KUC, R. Differentiating sonar reflections from corners and planes by employing an intelligent sensor. *IEEE Trans. Pattern Analysis and Machine Intelligence 12*, 6 (June 1990), pp. 560–569.

[3] BOZMA, O., AND KUC, R. Building a sonar map in a specular environment using a single mobile sensor. *IEEE Trans. Pattern Analysis and Machine Intelligence 13*, 12 (December 1991), pp. 1260–1269.

[4] BROOKS, R. A. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation RA-2*, 1 (March 1986), pp. 14–23.

[5] BROOKS, R. A. A hardware retargetable distributed layered architecture for mobile robot control. In *IEEE Int. Conf. Robotics and Automation* (1987), pp. 106–110.

[6] CROWLEY, J. Navigation for an intelligent robot. *IEEE Journal of Robotics and Automation RA-1*, 1 (March 1985), pp. 31–41.

[7] DEKHIL, M., SOBH, T. M., AND EFROS, A. A. Sensor-based distributed control scheme for mobile robots,. In *IEEE International Symposium on Intelligent Control (ISIC 95), Monterey, California* (August 1995).

[8] ELFES, A. Sonar-based real-world modeling and navigation. *IEEE Trans. Robotics and Automation RA-3*, 3 (June 1987), pp. 249–265.

[9] HENDERSON, T. C., BRUDERLIN, B., DEKHIL, M., SCHENKAT, L., AND VEIGEL, L. Sonar sensing strategies. In *IEEE Int. Conf. Robotics and Automation* (April 1996), pp. 341–346.

[10] HENDERSON, T. C., DEKHIL, M., BRUDERLIN, B., SCHENKAT, L., AND VEIGEL, L. Flat surface recovery from sonar data. Tech. Rep. UUCS-96-003, University of Utah, March 1996.

[11] HENDERSON, T. C., DEKHIL, M., BRUDERLIN, B., SCHENKAT, L., AND VEIGEL, L. Flat surface recovery from sonar data. In *DARPA Image Understanding Workshop* (February 1996), pp. 995–1000.

[12] HENDERSON, T. C., AND SHILCRAT, E. Logical sensor systems. *Journal of Robotic Systems* (Mar. 1984), pp. 169–193.

[13] KLEEMAN, L., AND KUC, R. An optimal sonar array for target localization and classification. *IEEE Trans. Robotics and Automation 14*, 4 (August 1995), pp. 295–318.

[14] KUC, R. A spatial sampling criterion for sonar obstacle detection. *IEEE Trans. Pattern Analysis and Machine Intelligence 12*, 7 (July 1990), pp. 686–690.

[15] KUC, R. A physically based navigation strategy for sonar guided vehicles. *Int. J. Robotics Research 10*, 2 (April 1991), pp. 75–87.

[16] LEONARD, J. J., AND DURRANT-WHYTE, H. F. Mobile robot localization by tracking geometric beacons. *IEEE Journal of Robotics and Automation 7*, 3 (1991), 376–382.

[17] MATTHIES, L., AND ELFES, A. Integration of sonar and stereo range data using a grid-based representation. In *IEEE Int. Conf. Robotics and Automation* (Apr. 1988), pp. pp. 727–733.

[18] PEREMANS, H., AUDENAERT, K., AND CAMPENHOUT, J. V. A high-resolution sensor based on tri-aural perception. *IEEE Trans. Robotics and Automation 9*, 1 (February 1993), pp. 36–48.

[19] SCHENKAT, L., VEIGEL, L., AND HENDERSON, T. C. Egor: Design, development, implementation – an entry in the 1994 AAAI robot competition. Tech. Rep. UUCS-94-034, University of Utah, Dec. 1994.

[20] SHILCRAT, E. D. Logical sensor systems. Master's thesis, University of Utah, August 1984.

[21] SIMMONS, R. The 1994 AAAI robot competition and exhibition. *AI Magazine 16*, 2 (Summer 1995), pp. 19–30.

[22] TRC TRANSITION RESEARCH CORPORATION. *LABMATE user manual, version 5.21L-f.*, 1991.