Sonar Resolution-Based Environment Mapping

Leyla Cahut*, Kimon P. Valavanis

Robotics and Automation Laboratory
Center for Advanced Computer Studies
University of Southwestern Louisiana
Lafayette, Louisiana 70504, USA

Hakan Deliç

Signal and Image Processing Laboratory
Department of Electrical Engineering
Boğaziçi University
Bebek 80815 Istanbul, Turkey

Abstract

Ultrasonic range sensors are used to obtain the information required for collision–free navigation of a mobile robot in a semi-structured or unstructured environment. A set of range readings from a ring of sonars are correlated to acquire a 2-D map of the robot's environment. The map is continuously enhanced via novel matching and update algorithms as new data are collected while the robot is in motion. The algorithm utilizes confidence measures that are directly obtained from the sonar's resolution or accuracy. There are no additional modeling assumptions and the approach is robust. The algorithms are experimentally tested on the Nomad 200 mobile robot and Nomad's Cognos Software Development Package.

1 Introduction

Feature-based environment mapping has been preferred for collision-free navigation and localization purposes due to its accurate representation of objects [1, 6]. Particularly, systems that rely solely on ultrasonic range sensors (sonars) have become popular since sonar offers a low-cost and relatively accurate alternative to other sensing types [7]. For instance, in [3], the environment map is a two-dimensional depiction where surfaces and objects are represented as connected sequences of line segments. The uncertainty in data is taken into account by maintaining a confidence measure with each line segment.

Unfortunately, the tolerances employed for matching line segments in [3] were set in an ad hoc manner. Variance and covariance expressions derived through Kalman filtering of the robot's position were used in [4]. However, as experimental observations indicate [5, 8], sensor noise and errors can be accurately modeled by a class of probability distributions, calling for robust statistical procedures, whereas Kalman filtering assumes strict Gaussianity for both measurement and system noise.

In this paper, we consider a mapping algorithm where the distance information returned by the sonar sensors are transformed into points in the Cartesian coordinate system. The points are grouped into linear clusters, and recursive line fitting is subsequently applied to each cluster. Once an initial map is thus obtained, an enhanced map of the environment is maintained through continuous, joint matching and update processes as new data are collected. Two consecutive sonar readings are said to "match" if they originate from the same location. Then a proper "update" procedure fuses the readings in order to enhance the available map.

We propose sonar accuracy-based tests for matching line segments. That is, the test tolerances become functions of sensor specifications and error performance rather than ad hoc settings. In particular, the Polaroid 6500 sensor, used extensively in the experiments, has $\pm 1\%$ accuracy over its entire range [9], which we take advantage of to define a ball of uncertainty around each sonar reading. The resulting map enhancement procedure is adaptive and makes no a priori modeling assumptions.

2 Sonar Uncertainty

Ignoring errors caused by scattering, the sonar beamwidth and resolution are the sources of uncertainty. The Polaroid 6500 sonar manual claims $\pm 1\%$ resolution (accuracy). However, in experiments and simulations, errors of magnitude up to 3% were observed for slightly tilted objects [2] owing to more pronounced scattering. Assuming that a resolution of $\pm \xi\%$ creates a linear uncertainty of $\pm \xi D_1/100$ along the signal direction, where D_1 is the range read-

*L. Cahut is now with Profilo Communications Technologies-PROTEK Inc., Mecidiyeköy 80384 Istanbul, Turkey.

ing, a second range reading D_2, obtained by perhaps some other sonar on the ring, should lie at most $\xi(D_1 + D_2)/100$ apart from the first point if the difference is to be attributed to sonar resolution. Thus, we define a resolution-based uncertainty region as a ball of radius $\xi D_1/100$ around the point-coordinate $(x_1,\ y_1)$ corresponding to the range reading D_1:

$$B_{D_1} = \{(x,\ y):\ \sqrt{(x - x_1)^2 + (y - y_1)^2} \leq \xi D_1/100\}.$$

The ball B_{D_1} may be centered at any point on the arc of uncertainty due to the 25° beamwidth of the Polaroid 6500 sonar. The uncertainty region U_1 is the union of balls:

$$U_1 = \bigcup_{\{B_{D_1}:\ (x_1,\ y_1)\ \text{is on the arc}\}} B_{D_1}.$$

If the uncertainty region of two consecutive readings, D_1 and D_2 intersect, then any discrepancy in the point coordinates can be accounted for by the resolution or beamwidth of the Polaroid 6500 sonar, and hence we can assert that there is a match between the two points. Defining I_{D_1,D_2} to be the indicator function of the event of a match between the points corresponding to the range readings D_1 and D_2, the test can be described as follows:

$$I_{D_1,D_2} = \begin{cases} 1, & \text{if } U_1 \bigcap U_2 \neq \emptyset \\ 0, & \text{if } U_1 \bigcap U_2 = \emptyset. \end{cases}$$

The inherent challenge in resolution-based matching described above is the enormous amount of computation required to search for ball intersections corresponding to each center point located on the arcs of uncertainty. On the other hand, there is no satisfactory recipe for extracting the exact point of reflection on the 25°–arc from the range reading information. In fact, we strongly believe that issues such as arc uncertainty and backscattering must be tackled at the sonar signal processing level where raw data can be manipulated.

Given the observations in the preceding paragraph, we shall assume in this paper that any sonar reading comes from the middle of the 25°–arc. The map errors that might be caused by this simplistic approach will be temporary since more reliable data will be collected as the robot moves in the direction of the source of the erroneous reading.

Since midpoints of arcs are assumed as sources of reflection, the matching operation reduces to searching for the intersection of two balls, each centered at the points that correspond to two consecutive range readings. That is, the test becomes

$$I_{D_1,D_2} = \begin{cases} 1, & \text{if } B_{D_1} \bigcap B_{D_2} \neq \emptyset \\ 0, & \text{if } B_{D_1} \bigcap B_{D_2} = \emptyset. \end{cases}$$

This sonar accuracy-based approach, to test if two range readings emanate from the same point on the same object with $\xi = 3$, will form the basis for developing matching procedures between line segments in the next section.

3 Sonar Resolution-Based Map Generation

The process of generating and improving a sonar map consists of three stages: 1) clustering or grouping the sonar data; 2) processing the grouped information, and; 3) map enhancement.

3.1 Clustering the Sonar Data

The range readings received from the sonars are checked against the threshold d_{max}, where d_{max} is chosen to be 255 inches [2], [9]. Any depth reading that exceeds this threshold is discarded due to the fact that the signal–to–noise ratio (SNR) becomes unacceptable at such distances. The range readings which are below the threshold are transformed to Cartesian coordinates:

$$x_k = D\cos(\alpha_k \pm \beta + \gamma) + r\cos(\alpha_k + \gamma) + R_x,$$

$$y_k = D\sin(\alpha_k \pm \beta + \gamma) + r\sin(\alpha_k + \gamma) + R_y,$$

where D is the depth reading received from a sonar, β is an angle within the sonar's beam where the echo is received ($|\beta| \leq 12.5°$), r is the robot's radius, α_k is sonar k's orientation with respect to sonar 0 ($\alpha_k = 22.5k$), γ is sonar 0's angular position on the sonar ring with respect to the x-axis, R_x and R_y denote robot's position in the world coordinate frame. For the reasons stated in Section 2, β is taken to be 0°.

Once the range readings are converted to Cartesian coordinates, they are clustered such that each group represents a single face of an object. An outermost point of an object face is found when the distance between two neighboring points is greater than a preset threshold $t_{\mathtt{gap}}$. If no gap is detected ($t_{\mathtt{gap}}$ is not exceeded), the distance between the next pair of neighboring points is checked. The clustering operation continues until all available points are exhausted.

3.2 Processing the Cluster Information

The clustered points are processed through recursive line fitting procedure. This part of the proposed methodology follows the same path as Crowley's [3].

3.3 Map Enhancement

The initial environmental map created needs to be improved as the robot continues navigating in the environment. The update of the map is done by fusing the new information with the existing knowledge and involves three steps. Matching process: Each new line segment is compared to the existing line segments to determine if there is a match. If no match is found, the new line segment is either dismissed, or it is kept as a new object face. Update process: If a match is found between two line segments, the higher confidence end-points are employed to form the line equation of the updated segment. The procedure is finalized by projecting the outermost of the four points (from the two lines segments) onto the updated line equation and extending the line segment to the projected point. Correlation process: If the matching process determines that a new line segment represents a new object face, this process checks to see if the line segments might be treated as extensions of one another. The correlation process smoothens the map but does not add any crucial information for navigation purposes. Therefore, we only furnish the procedure in the next section without providing experimental results. For results, see [2].

3.3.1 Matching Process

Let l_e and l_n denote line segments in the existing and new maps, respectively, that are candidates for a match. Two line segments are declared to have a match if all of the following three tests are satisfied in sequence.

- Orientation test: Difference between slopes of l_e and l_n should be small. This translates to requiring the difference in angular orientation with respect to some reference axis to be less than some threshold. Typically, this threshold is chosen to be 15° [3] .
- Colinearity test: In addition to the orientation test, the distance, d, from the mid-point of l_n to the line equation of l_e should be smaller than the threshold defined below:

$$|d| \leq 0.03D_1 + 0.015(D_2 + D_3).$$

The first and second terms on the right hand side of the equation account for the accuracy of the range reading D_1 and the mid-point between D_2 and D_3 respectively.

- Overlap test: This stage of the matching process consists of two steps: Mid-point overlap check (MPC), and end-point overlap check (EPC). MPC is satisfied if the projection of the mid-point of l_n intersects l_e. This check is repeated for all the existing lines until a match is found. If MPC fails with all the available lines, then we seek to find out if there is any overlap at all by initiating EPC, which is a check for partial overlap. EPC passes if one of the end-points of l_n intersects l_e when projected onto l_e.

The first two tests jointly ensure that the line segments are aligned along a common line equation. Failure of the first test implies mismatch in angular orientation. If the orientations match but the second test fails, then the two line segments are sufficiently "parallel" to one another (in the sense of the test) but too distant. Once the orientation and alignment match is confirmed, the third test checks for overlap. The possibilities are as follows:

1. MPC passes: The new line segment is about parallel to and near an existing line segment, and the two lines have at least 50% overlap. Thus a "good" match is achieved. If MPC fails for an existing line segment, we repeat the test for the neighboring line segments in the map next until a match is found. If MPC fails for all the existing line segments, the matching process is continued with the EPC test between the existing line segments possessing the outermost points of the object face and the new line segment.
2. MPC fails, EPC passes: There is "partial" match with one of the existing lines. Since this match accounts for less than 50% overlap, we declare that the new line segment is an extension of the existing one.
3. MPC and EPC both fail: There are two approaches that can be taken. Either the new line segment is considered as a newly detected object or object face and plotted on the map as an entirely separate line, or its proximity to the existing lines is checked and a decision is made whether the new line segment is to be connected to the nearest existing line segment.

If any one of the three tests fails, then the new line segment is assumed to represent a new object face. The

only exception occurs when the orientation test fails but the colinearity and overlap tests pass for any of the existing line segments. This anomaly might stem from erroneous data and hence the new line segment is dismissed. It is now clear that the matching process first seeks for a match, then for dismissal. Only after it is neither matched nor dismissed, is a new line segment kept as a new object face.

The matching process will work only if the distance traveled between two consecutive snapshots is small enough that the signal-to-noise ratios of the corresponding readings in the two data sets are comparable. Otherwise, a false inconsistency will occur due to the significant improvement in the quality of the range readings as the robot moves much closer to the same object, or vice versa. In our experiments, consistently accurate matches were found for distances up to 10 inches traveled between two consecutive snapshots.

3.3.2 Update Process

Once all three tests pass, updating the line segment will be performed between two consecutive snapshots. There are two goals that we try to achieve simultaneously: given the four points, draw the longest possible line segment satisfying the line equation given by the two highest-confidence end-points. To that end, between the two sets of matching end-points, the ones with the higher confidence are kept to determine the equation of the updated line segment, l_u. In other words, the confidence in a point is inversely proportional to its corresponding range reading. If any of the lower-confidence points lies outside the segment bounded by the higher-confidence points, then the former point is projected to intersect l_u, and subsequently, l_u is extended to the point of intersection. This way a worst case scenario is presented to the navigation unit, to avoid collision with an obstacle that is not yet well-observed [2].

Let an existing line segment be defined between the left end-point (x_o^L, y_o^L) and the right end-point (x_o^R, y_o^R). Likewise, suppose the new line segment be between the left and right end-points (x_n^L, y_n^L) and (x_n^R, y_n^R), respectively. The update procedure takes the following course depending on the outcomes of MPC and/or EPC:

1. MPC passes: First the higher confidence end-points are determined. Let $D[(\cdot, \cdot)]$ be the range reading corresponding to a point. Then the end points (x_u^L, y_u^L) and (y_u^R, y_u^R) of the updated line equation are such that

$$D[(x_u^L, y_u^L)] = \min\{D[(x_o^L, y_o^L])], D[(x_n^L, y_n^L)]\},$$

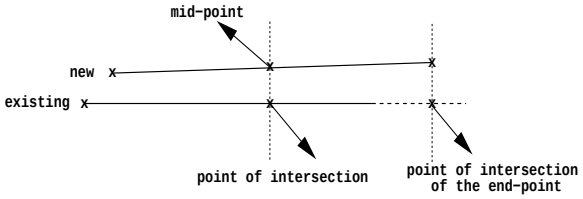

Figure 1: Projection of one of the endpoints lying outside the old line segment.

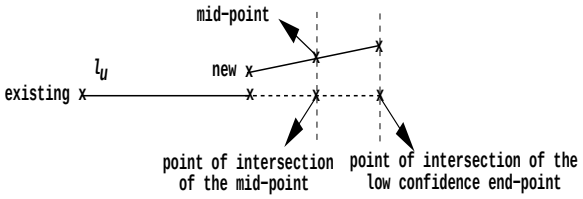

Figure 2: Projection of the right end-point of the new line segment on the updated line segment.

$$D[(x_u^R, y_u^R)] = \min\{D[(x_o^R, y_o^R])], D[(x_n^R, y_n^R)]\},$$

respectively. The points (x_u^L, y_u^L) and (x_u^R, y_u^R) determine the line equation l_u will lie on. Next we project the lower confidence end-points onto the line equation of l_u. If a projection does not intersect the updated line segment, then a new line is drawn between the updated end-points and the point of intersection (POI) of that projection and the line equation (see Figure 1). Otherwise, (x_u^L, y_u^L)–(x_u^R, y_u^R) defines the updated line segment. Thus the longest possible line segment satisfying the highest-confidence line equation is obtained given the four points (x_o^L, y_o^L), (x_o^R, y_o^R), (x_n^L, y_n^L) and (x_n^R, y_n^R).

2. MPC fails, EPC passes: Without loss of generality, suppose that EPC passed for the left end-point only. Then, we keep either the left end-point of the new line segment or the right end-point of the existing line segment as a middle-point (x_u^M, y_u^M) of l_u such that

$$D[(x_u^M, y_u^M)] = \min\{D[(x_o^R, y_o^R)], D[(x_n^L, y_n^L)]\}.$$

Of the two end-points (x_o^L, y_o^L) and (x_n^R, y_n^R), the one with the higher confidence is kept as the end-point of l_u, and the one with the lower confidence is projected on the line equation of l_u formed by (x_u^M, y_u^M) and the higher confidence end-point (see Figure 2). Thus,

$$(x_u^L, y_u^L) = \begin{cases} (x_o^L, y_o^L) \text{ if } D[(x_o^L, y_o^L)] \leq D[(x_n^R, y_n^R)] \\ \text{POI of } (x_o^L, y_o^L) \text{ and } l_u \text{ if else,} \end{cases}$$

$$(x_u^R, y_u^R) = \begin{cases} (x_n^R, y_n^R) \text{ if } D[(x_n^R, y_n^R)] \leq D[(x_o^L, y_o^L)] \\ \text{POI of } (x_n^R, y_n^R) \text{ and } l_u \text{ if else.} \end{cases}$$

As a result of update, the object face is represented by a set of existing and new points. Using these points, recursive line fitting is applied again to refine the sonar profile of the object face.

3.3.3 Correlation Process

One approach, if there is no match, is to assume that the new line segment could be representing another face of some object previously detected. The correlation algorithm described below is applied to append the new unmatched face to the existing object shape.

The correlation is performed when the minimum distance between the end-points of the new line segments and existing line segments is found to be less than or equal to some preset threshold $t_{\mathtt{gap}}$. Define $L_{\mathtt{existing}} = \{l_1, \ldots, l_p\}$ and $L_{\mathtt{new}} = \{l_{p+1}, \ldots, l_{p+q}\}$ as the set of existing line segments belonging to all the detected object faces and the set of new lines forming some new object face, respectively. Let $d_i(L_{\mathtt{existing}}, l_i)$, $l_i \in L_{\mathtt{new}}$, denote the minimum distance from any end-point of l_i to any of the end-points of the lines in $L_{\mathtt{existing}}$. Furthermore, define:

$$d_{min} = \min_{l_i \in L_{\mathtt{new}}} d_i(L_{\mathtt{existing}}, l_i).$$

If $d_{min} \leq t_{\mathtt{gap}}$, then the set of line segments in $L_{\mathtt{new}}$ are appended to the line

$$l^* = \arg \min_{l_i \in L_{\mathtt{new}}} d_i(L_{\mathtt{existing}}, l_i).$$

If the minimum d_{min} exists, it is determined that the lines in $L_{\mathtt{new}}$ represent another face of some existing object; in particular the new line whose end-point is closest to l^* should form a corner with l^*. If a corner is not formed, the end-points closest to each other, and the lines that these points are associated with are interpolated. This process is repeated until all the unmatched new object faces in $L_{\mathtt{new}}$ above are exhausted. The new line obtained as a result of interpolation forms a corner between the closest two lines on different faces of same object (see Figure 3).

If the line segments in $L_{\mathtt{new}}$ neither match nor correlate with the existing line segments, then $L_{\mathtt{new}}$ is distinguished to be a distinct object face.

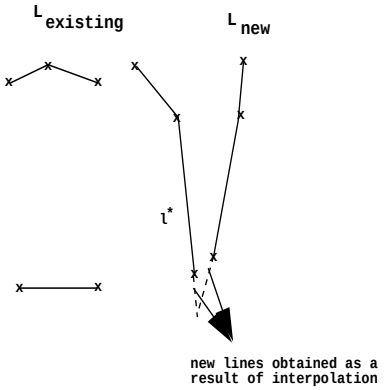

Figure 3: Correlating a new line segment with an existing line segment.

4 Simulation Results And Experiments

The developed algorithms have been tested on the Nomad 200 mobile robot which is endowed with a ring of sixteen sonar sensors. The robot was placed in a rectangular room of 305×154 inches, called the "Blue Room" because of the blue styrofoam walls forming the borders of the test area. The minimum gap size, $t_{\mathtt{gap}}$, is taken to be twice the robot's diameter including the bumper, ensuring for passage through an opening under the worst case scenario where the neighboring sonars give identical readings and the reflections occur from the opposing ends of the neighboring sonar arcs [2]. For the Nomad 200, $t_{\mathtt{gap}} = 2(2r + 3) = 42$ inches (3 inches account for the bumper circling Nomad 200). When both leftmost and rightmost vertices are found, the leftmost point, the intermediate points and the rightmost point are grouped together to constitute a face of the object.

In Figures 4 and 5, the robot is depicted within the mapped 2–D contours of the Blue Room. In the experiments, the robot was first positioned at various locations inside the Blue Room. Then, a snapshot of sixteen sonar readings were collected. The robot was moved 5 inches to the right, and a second set of data were recorded from the sonars. Each time a snapshot was obtained, a map was drawn using the recursive line fitting procedure. In the end, the two sets of points were fused through the matching and update algorithms of Section 3. As can be seen in Figure 4, as the robot moves to the right, it sees more of the left wall accounting for the greater coverage in the latter map. The line segments with points of higher confidence are kept (see Figure 5). Notice that the upper wall remains at about the same distance as the robot moves to the right, and both snapshots give identical line segments.

A case where the sonar returns unreliable data is shown in Figure 6 (upper right corner of the room). The mapping algorithm successfully draws the best

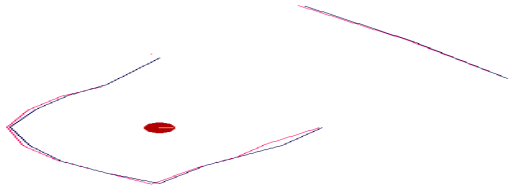

Figure 4: Environment maps derived from the first and second snapshots using recursive line fitting. The Nomad 200 mobile robot moved 5 inches to the right between the snapshots. (Red: first snapshot. Blue: second snapshot.)

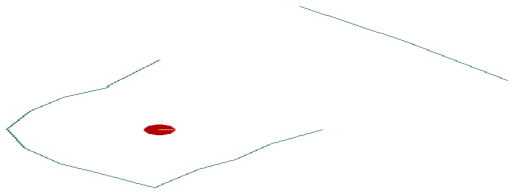

Figure 5: Enhanced environment map obtained by fusing the first and second snapshots in Figure 4.

contours based on the available data as seen in Figure 7.

A common observation from these experiments is that in a large room, the sonar ring is not capable of detecting all the walls. Some sections of the Blue Room were either too far from the mobile robot, or the sonic beams were scattered due to the severe angle of incidence yielding the walls undetectable. In such instances, the robot has to navigate to different locations within the room for maximum coverage.

The algorithms developed in this paper work even when the robot is closely surrounded by objects (e.g. a tight room with a tiny opening). As seen in Figures 8 and 9, a complete map is successfully drawn and updated. More complex floor plans were considered using Nomad's Cognos Software Development Package in [2].

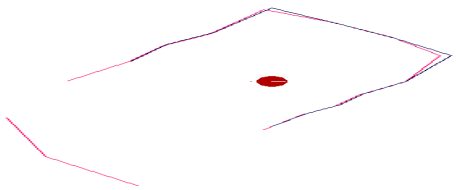

Figure 6: Environment maps derived from the first and second snapshots using recursive line fitting. The Nomad 200 mobile robot moved 5 inches to the right between the snapshots. (Red: first snapshot. Blue: second snapshot.)

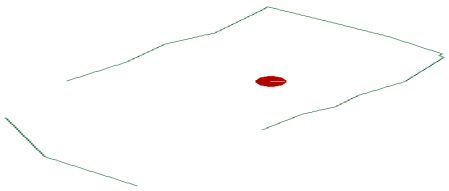

Figure 7: Enhanced environment map obtained by fusing the first and second snapshots.

Figure 8: Environment maps derived from the first and second snapshots using recursive line fitting, superimposed on the floor plan. The simulated robot, which is surrounded all around by objects, moved 5 inches to the right between the snapshots. Nomad's Cognos Software Development Package was used in this experiment. (Red: first snapshot. Blue: second snapshot.)

5 Discussion and Conclusions

In this paper, we introduced a novel matching and update processes which are based on the sonar accuracy or resolution. When consecutive snapshots are taken within 1–5 inches difference between robot's previous and current location, it is expected that the readings should fall within a ball of radius of $\pm 1 - 3\%$ times the previous range reading by virtue of the expected sonar accuracy specifications. Any discrepancy beyond this is attributed to erroneous data or the detection of new object faces. Two-dimensional representations obtained as a result of resolution-based mapping are reliable since the thresholds are based on the accuracy range. Another advantage of tying decision thresholds to sonar resolution is that confidence in the sensor readings becomes a function of the robot-to-object distance. This quality is intuitively pleasing since signal attenuation increases with distance.

The mapping and enhancement procedures developed in this paper have a number of advantages. The initial map generated by the first set of sonar data serves as the local model of the environment, and no a priori information is needed. This approach makes the method robust because in semi-structured and unstructured environments, where the local model based on a floor plan may not always be relevant. A typical example is an office room with people walking around.

Figure 9: Enhanced environment map obtained by fusing the first and second snapshots after the floor plan in Figure 8 is removed.

Furthermore, the confidence thresholds derived this way are adaptive, depending on the sonar-to-object distance.

The limitation of the matching and update processes embedded in the mapping algorithm is that it requires continuous data acquisition with a robot displacement of at most 10 inches between two consecutive snapshots. With the advent of high-speed microprocessors, the data acquisition and processing rates are already fast enough to satisfy this constraint.

The algorithms described in this paper have no provision to deal with erroneous data resulting from scattering or sensor uncertainties due to the 25° beamwidth of the Polaroid 6500 transducer. In order to overcome these problems and achieve a more accurate map of the environment, sonar signal processing of the raw data is necessary prior to data fusion. The actual TOF readings should be adjusted depending on the type of surfaces (i.e., diffracting or reflecting). Furthermore, multiple-sensor integration of visual, x-ray, tactile, etc. sensors will robustify the fusion process by compensating for the inherent weakness of each type of sensors with the information provided by the others. Our method is universal since it can be applied to range readings obtained from any kind of sensors.

Acknowledgment

This work has been partially supported by NSF Research Grants BES-9506771 and BES-9712565.

References

[1] J. Borenstein, H. R. Everett, and L. Feng, *Navigating Mobile Robots: Systems and Techniques*, Wellesley, Massachussetts: A K Peters, 1996.

[2] L. Cahut, "Feature-Based Environment Mapping Using Sonar Sensors", M.S. Thesis, University of Southwestern Louisiana, Lafayette, Louisiana, December 1997.

[3] J. L. Crowley, "Navigation for an Intelligent Mobile Robot", *IEEE Journal of Robotics and Automation*, vol. 1, 1985, pp. 31-41.

[4] J. L. Crowley, "World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging", *Proceedings of the IEEE International Conference on Robotics and Automation*, Scottsdale, Arizona, May 1989, pp. 674-680.

[5] H. F. Durrant-Whyte, "Sensor Models and Multisensor Integration", *International Journal of Robotics Research*, vol. 7, 1988, pp. 73-89.

[6] D. Lee, *The Map-Building and Exploration Strategies of a Simple Sonar-Equipped Mobile Robot*, Cambridge, United Kingdom: Cambridge University Press, 1996.

[7] J. J. Leonard, H. F. Durrant-Whyte, *Directed Sonar Sensing for Mobile Robot Navigation*, Boston, Massachussetts: Kluwer Academic Publishers, 1992.

[8] R. Mandelbaum, G. Kamberova, and M. Mintz, "Statistical Decision Theory for Mobile Robotics: Theory and Application", *Proceedings of the Conference on Multisensor Fusion and Integration*, Washington, DC, December 1985, pp. 17-24.

[9] The Nomad 200 User's Manual, Nomadic Technologies Inc., Mountain View, California, 1996.