Mobile Robot Localization with Sparse Landmarks

Nathaniel Fairfield[a] and Bruce Maxwell[b]

[a]Bluefin Robotics
[b]Swarthmore College

ABSTRACT

This paper describes a mobile robot system designed to explore and map an area such as is encountered in urban search and rescue simulations. The robot uses homogeneous artificial landmarks for localization as it builds metric and topological maps, determining landmark distance and bearing with groundplane calculations from a single camera and using Kalman filtering techniques to perform localization. When implemented on a Magellan II mobile robot, the localization technique correctly localized the robot while exploring and mapping.

Keywords: localization, Kalman, landmark, mobile robot

1. INTRODUCTION

One of the most common approaches to mapping is to collect sensor information into an evidence grid.[1] A nice feature of evidence grids is that the geometry of the evidence grid map corresponds directly to the metrical geometry of the world, providing a basis estimate of the robot's location for the purposes of localization and navigation.[2] The problem with constructing these maps is that it requires the integration of many different sensor readings that have been collected while the robot was moving around from place to place. In order to be sure that the readings all line up, the robot must keep very careful track of its current location. This is the problem of localization.

There have been a broad range of approaches to the problem of how to maintain accurate position information about a mobile agent. The most obvious (and expensive) include perfectionist techniques which try to use absolute coordinates generated either internally by odometry, or externally by a differential GPS or some other high-accuracy tracking system. The first is limited by wheel slippage and the resolution of the wheel encoders, and the second is cumbersome and expensive. Other approaches require parallel or perpendicular walls with low amounts of clutter which can be easily recognized and oriented to.[2] Under ideal circumstances these systems work quite well, but the idea is to get out of the lab and into the less-than-ideal world.

Many approaches depend on *a priori* knowledge of the environment, usually in the form of a hand-crafted CAD map.[3] One approach which uses either a pre-programmed map or carefully generated long-term maps (see 4) for localization is to construct short-term sonar or laser range finder maps which are gathered quickly enough that there isn't much odometry error and so are metrically consistent.[3] The short-term maps are then transformed to best fit the existing long-term map, and the inverse of the transformation is applied to the robot's localization odometry. This approach is computationally intensive and is difficult to adapt to a dynamic environment, though good results have been achieved (for example 4, 5). An advantage of this type of mapping is that it does allow fairly continuous localization in that the robot's odometry is corrected regularly and frequently, keeping individual corrections small.[6] On the down side, sonar and laser range finders are easily confused in dynamic environments, may be confused by specular surfaces, and have limited range.

A Markov model can be used to relate proximate nodes in the topological map so that the robot can disambiguate two locations with near-identical feature vectors based on history.[3, 7] Markov localization implements the common-sense intuition that changes in robot location, due either to robot motion or to localization corrections, must be fairly small and only to areas roughly adjacent to the current position estimation. Especially in cases where localization is completely dependent on visual landmarks (and odometry information is unavailable or ignored), the robot must have some way to decide whether a transition from one place to another is statistically likely.[7] By using Markov localization, robot systems can actually localize themselves within a known map without knowing their starting position.[3]

Although visual navigation can be achieved using optical flow,[8] vision-based mobile robot navigation usually focuses on the detection of landmarks, either natural or artificial.[9] The camera on a robot is able to visually locate landmarks relative to the robot's position and use the landmarks' stored positions to derive the robot's current location. Ideally, the robot should be able to pick out naturally occurring landmarks and use them to correct its odometry. Unfortunately, it is very difficult to get a robot to wisely choose its own landmarks, robustly recognize them from any angle, and accurately estimate their position. On the other hand, there exist excellent real-time algorithms for detecting custom artificial landmarks, such as the self-similar patterns described in 10. Artificial landmarks can be constructed to be easy to identify and to yield extra coded data, such as a unique id number. Such approaches have two drawbacks. First, the patterns used are easily occluded (except in the case where the landmarks are placed on the ceiling, such as 11), and second they are too large and complex to be deployed autonomously. In other words, their use is restricted to engineered environments where human handlers can lay out landmarks for the robot. This is clearly a problem in any situation where human planners cannot be present to place the landmarks, such as an urban search and rescue situation.

Other approaches to landmarks have assumed that multiple landmarks, usually three, can be observed simultaneously and used to triangulate the robot's position in linear time.[12] This approach only requires that the bearing to the three landmarks be known and not any distance information, though it does need a hand-made metric map of the area with marked landmarks. Constructing maps using a triangulation approach is a very cumbersome process (though it obviously works; cartographers used to do everything that way!).

Crowley[13] uses binocular vision to calculate angle and distance to a landmark, or just angle, or just distance, and uses Kalman techniques for localization. The binocular vision approach obviously has great promise, but in the interim, monocular vision is more common and much less complex.

2. IMPLEMENTATION

2.1. The Magellan II

The Real World Interfaces (RWI) Magellan II is a small round robot with symmetrically opposed wheels which allow it to rotate on its axis (see Fig. 1). The basic sensor array consists of three rings of 16 bump (contact), sonar, and IR sensors mounted around the sides of the robot. In addition the Magellan II used in this paper has a Canon VC-C4 pan-tilt-zoom camera. The on-board computer is a Pentium III running RedHat Linux, which communicates with the robot's rFlex controller over a 9600 baud serial line. We could connect to the robot over a wireless ethernet connection and forward XWindows sessions. This greatly facilitated debugging and run-time monitoring of the robot's state. All code was written in C.

2.2. Vision System

On the mobile robot platform the vision system was only one of several different modules, including information collecting modules which read in sensor data and placed abstracted results into shared memory, analysis modules which took the available information and made decisions and plans, and output modules which made the robot move or speak. All of these different modules have to share the common resources of the robot's computer. The vision module is by far the most processor intensive so it must be carefully designed not to hog computing resources. And yet vision is also one of the richest sources of information for the robot so the vision module must make full use of the resources it is allowed. We took two basic steps to address these issues. First, we modularized and threaded the various vision tasks (motion detection, landmark identification, etc) such that a call to each task would cost roughly an equal amount of computing power.* Secondly, we implemented a stochastic scheduler, which allowed us to set a maximum total number of tasks to be run while maintaining the ability to prioritize those tasks appropriately depending on what the robot is looking for.

*For example, in the case of motion detection, a fairly complex task requiring a sequence of several images, this meant that it took several calls to the motion detection task to complete a cycle.

Figure 1. The Magellan II with axially mounted camera. The robot is about half a meter wide.

Figure 2. A robot's eye view of several of the rings which were used as landmarks. During an experimental run, they are deployed about three meters apart.

2.3. Evidence Grid

To construct the evidence grid, the robot used a version of the original code used by Moravec in 1 which had been placed in a wrapper by Bruce Maxwell. As each range sensor value is entered in to the evidence grid at the appropriate location and orientation, a map of the environment emerges. A cell size of five centimeters per pixel was used during the experiments, which allowed large maps (up to thirty meters square) to be constructed, while retaining sufficient accuracy to allow the robot to perceive navigationally significant details.

One of the most effective measurements of the occupation of a cell in the evidence grid is the robot itself. Any space that the robot currently occupies is almost certainly empty. For this reason, the current location of the robot was entered into the grid as a robot-shaped blob of certainty. Since otherwise the robot was hesitant to commit on the complete emptiness of other areas for which it only had sonar data, this dark path of robot locations helped to trace the route that the robot traversed during a particular test.

2.4. Visual Identification

The approach described here uses the camera to visually identify and locate small green plastic rings that have been placed as landmarks (see Fig. 2). All of the landmarks are identical and contain no encoded information. The bright shade of green for the rings was selected because it is a fairly unique color in most environments, although in the robot lab where some of the test runs were performed, there were certain ethernet cables with exactly the same shade of green. In order to avoid false identifications, the robot ignored a green object if its estimated position was more than a certain distance away from the expected position of a landmark. This naive tactic was very effective at weeding out false positives, but upon occasions when the robot became seriously disoriented could cause it to ignore real landmarks. The size of the rings made it feasible to design an automatic deployment mechanism, and in fact a simple prototype of such a mechanism was constructed and tested, although it was not used in the final tests due to time constraints. The rings were deployed as the robot explored according to the Drop Landmark schema described above.

The first step in calculating the position of the ring relative to the robot was to locate the ring in the image. Because of the unique color of the rings, the image was thresholded for bright green and then a two-pass segmentation algorithm was used to find the centroids of green blobs of made up of at least five pixels. A minimum of five pixels was chosen because under certain lighting conditions specularities may create tiny spots of green (or any other color). The minimum size threshold also automatically discarded images which

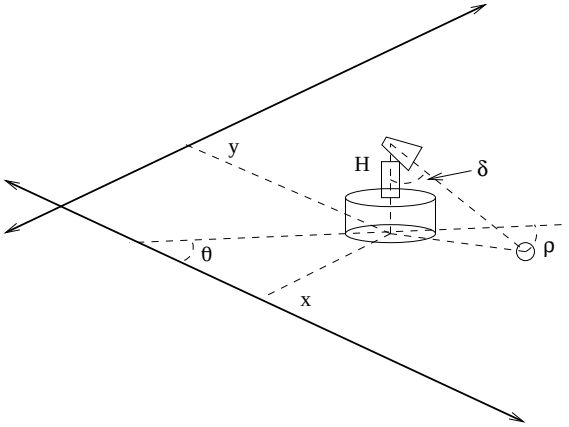

Figure 3. A ring's position is calculated relative to the robot's position and orientation.

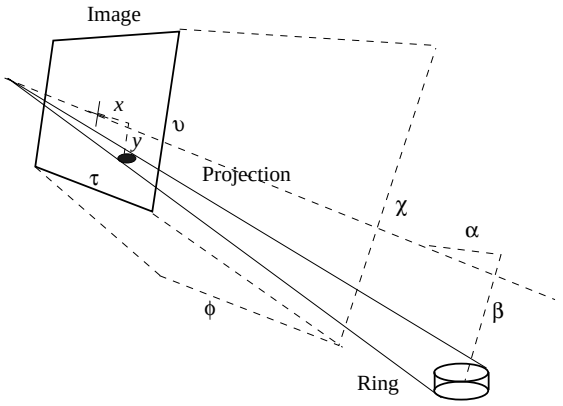

Figure 4. A ring is projected onto the image plane of the camera.

contained no rings at all. The segmentation algorithm returned the coordinates $(x; y)$ of the blobs in the image. Due to the deliberate spacing between landmarks (usually around three meters), it was assumed that only one landmark would be visible at a time, so only one blob (the largest) was used for localization purposes. Because the horizontal and vertical fields of view of the camera and the horizontal and vertical dimensions of the image were known (see Fig. 4), the horizontal and and vertical angles of the ring relative to the camera orientation could be calculated as follows:

$$= x - \frac{}{2} \times \frac{\tau}{2}$$

$$= y - \frac{}{2} \times \frac{\upsilon}{2}$$

Since the height of the camera H and its angles of declination and rotation and the robot's global position $x; y$ and orientation are known (see Fig. 3), the perceived distance to the ring relative to the robot's current position is calculated as

$$distance = tan(\ \ +\ \ ) \times H$$

$$ringY = x + sin(\ \ +\ \ +\ \ ) \times distance$$

$$ringX = x + cos(\ \ +\ \ +\ \ ) \times distance$$

After having calculated the perceived position of the ring, the robot searched through the list of known ring positions and found the stored location with the smallest Euclidean distance from the perceived ring location. If the closest match was further than a threshold, in this case somewhat arbitrarily chosen to be a meter, the ring identification was discarded as false.

If a good match was found, the problem remained of how the robot's current position should be corrected to take this information into account. In effect there are three pieces of relevant information: the known ring position, the perceived distance to the ring, and the robot's current location. Each of the three is subject to error. The known ring position may be inaccurate because of odometry error before the ring was deployed. There is little to be done about the error of the stored ring location. The perceived distance may be inaccurate because the camera has limited resolution and may have picked a bad centroid to judge the distance to the landmark (remember that at a distance of two meters, a difference of a few pixels can result in an error of several centimeters). Multiple observations of a ring can increase the overall trustworthiness of the perceived ring location. Finally, the robot's estimate of its current location may be erroneous due to the odometry errors that accumulate over time.

Since the overall error of our three pieces of location information increases gradually (due to the odometry error), it seemed reasonable to attempt to make the minimum correction possible. To find the minimum

correction, the slope of a line from the current robot position to the known ring position was calculated. Then the point was found on the circle around the known ring position with a radius of the perceived distance to the ring that intersected this line. That point was the robot's best guess for its current location. These corrections were then applied to the robot's $x; y$ and odometry (see Fig. **??**).

2.5. Kalman Filtering

The landmark position yielded by process described above needed to be used to correct the odometry of the robot. Unfortunately, the visual estimates of heading and distance, from which the positions of the landmark was calculated were far from accurate. Variations in lighting conditions, ring orientation (or robot orientation relative to the ring), and the fluctuations common in all frame-grabbers made ring identification tricky and far from precise.

Unfortunately, the position estimates of the rings were only as accurate as the images captured by the camera allowed, and in order to maintain a high frame-rate the images were captured at a fairly low resolution (160x120 pixels). This meant that for distant rings, the difference of a few pixels in the image could result in the displacement of tens of centimeters of our position estimate. Ideally, the certainty of the ring's location would be integrated into the Kalman calculation.

The simplest approach is to completely trust the visual landmark estimation and to jump to the nearest point and orientation on the circle defined by the distance to the landmark as determined visually. This approach is very simple and quick, but as the visual readings are noisy, a certain amount of jitter is inevitable.

A more sophisticated approach using a Kalman filter takes into account the expected error in the visual estimate of landmark position. The Kalman filter also factors in the information we already have from odometry, weighing it by the estimated accuracy of the encoders which decays over time. Because the robot only has these two sources of information and the model of the robot's position is the same as the odometry, a very simplified version of the Kalman filter could be used (for a good explanation of the Kalman filter, see[14]). In the first implementation of the Kalman filter, the robot's odometry was corrected by a fraction of the amount suggested by each positive landmark identification. This fraction is called the Kalman gain and was experimentally tuned to reflect the average error in the visual identification. Briefly,

$$K = \frac{P}{P + R};$$

where K is the Kalman gain, P the accumulated odometry error, and R the sensor error. The new position x_t is calculated from the old position x_{t-1} by

$$x_t = x_{t-1} + K(z);$$

where z is the correction calculated from the landmark.[†]

Several variations of these parameters were used during testing. In the simple "jump" technique, $P = 1$ and $K = 1$. In the first implementation used during testing, $P = 1$ and $K =$ a constant (usually around 0.3). In the second implementation, $P = 1$ and K is scaled depending on the expected error of the visual landmark identification as predicted by its distance (in other words, K decreased with the distance of the landmark).

In a more complex implementation more true to the principles of the Kalman filter, P would increase arithmetically over time in accordance with accumulated odometry error (such a test was not run). Another untested approach might use a combination of these techniques. Instead of setting the Kalman gain as a single function of the distance of the landmark, it could be separated into $x; y$ and components. The Kalman gain of the $x; y$ component would decrease with distance, but the component would increase with the distance of the landmark. This would take advantage of the fact that orientation estimates are better with further landmarks, but position estimates are better with close landmarks.

[†]Crowley 13 describes a mathematical model used to estimate uncertainty in position and orientation, and also a good overview of the use of Kalman filters in robot navigation.

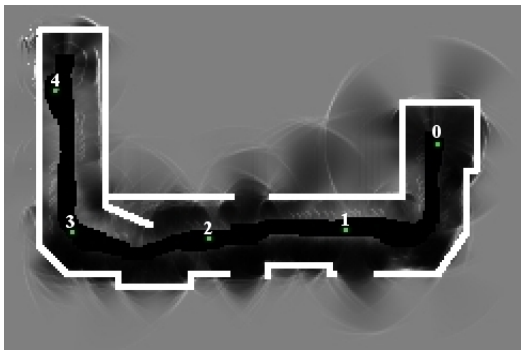

Figure 5. The floor plan for the part of Hicks Hall which was used during mapping tests, with the five landmark locations labeled. The hallway is about ten meters long.

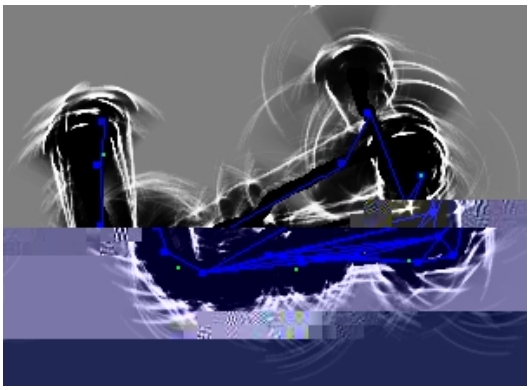

Figure 6. A map with $K = 0$ (no localization) after three passes.

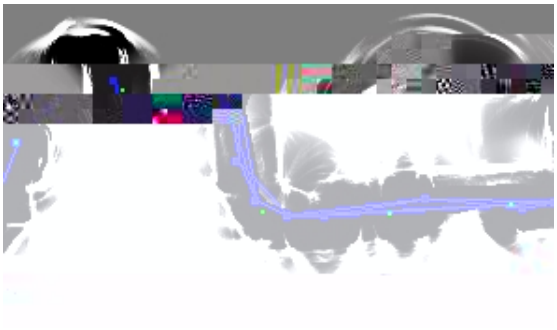

Figure 7. A map with $K_{x,y} = 0.3$ and $K_\theta = 0.1$ after three passes.

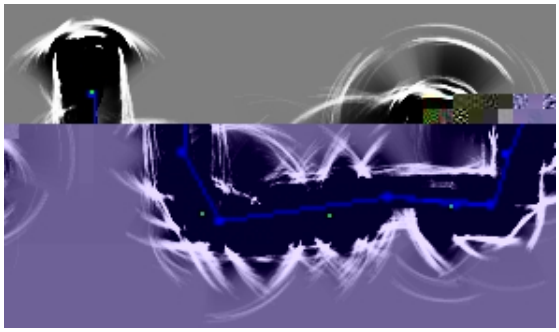

Figure 8. A map with $K_{x,y} = 0.3$ and $K_\theta = 0.3$ after three passes.

3. RESULTS

All of the tests were run in the hallways of Swarthmore College's Hicks Hall. Figure 5 shows a plan of the area mapped in Hicks. In the tests, the robot was started in the upper right corner area and then made several passes (out and back) along the corridor and into the other room. Note the three open doors in the hallway. Depending on the time of day, different doors were alternatively open or closed–not all maps were constructed with the same configuration of doors. A very obvious characteristic of the maps in Hicks is that the walls were quite specular and so the evidence grids tended to be affected by false echos, which created artifacts behind walls.

As a base case, several maps were constructed with no localization. Figure 6, is the result of three passes without localization. In addition to the overall darkening of the map due to more sonar readings (and false echos), the effects of the accumulation of odometry error is clearly obvious in the unaligned wall lines and gradual counter-clockwise shift.

The next set of maps (see Figures 7, 8, 9, and 10) were made with values of $K_{x,y,\theta}$ ranging from 0.1 to 0.5 (all maps were made with three passes). These maps are much better than the maps with no localization, but they clearly suffer from the spurious echos and false readings of the sonars. Regardless, for low values of $K_{x,y}$ and K_θ the robot was correctly recognizing the landmarks and localizing itself, as is demonstrated by the fact the the walls of the hallway were super-imposed on successive passes. However, as K_θ grew past 0.2, the robot began to over-correct to the point where in Figure 10 with $K_{x,y} = 0.5$ and $K_\theta = 0.5$, the robot became completely lost and the map is as bad or worse than the maps constructed with no localization at all.

Figure 9. A map with $K_{x,y} = 0.5$ and $K_\theta = 0.3$ after three passes.

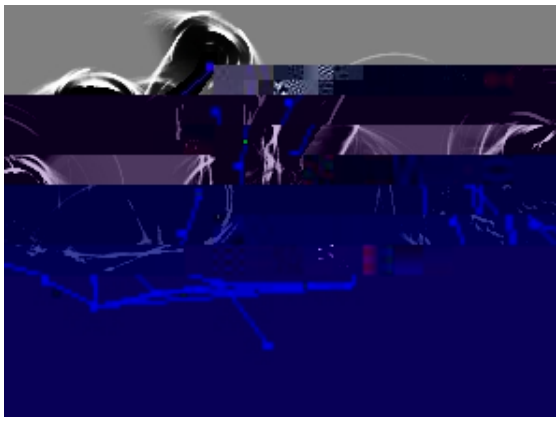

Figure 10. A map with $K_{x,y} = 0.5$ and $K_\theta = 0.5$ after three passes.

4. DISCUSSION

4.1. Landmark Localization

The simple landmark technique yielded satisfactory results and provided frequent small corrections to the odometry which allowed the robot to correct odometry error and maintain correct localization. In turn, this allowed the robot to construct accurate evidence grid maps and topological navigation maps. The Kalman gain was found to be optimal somewhere in between 0.1 and 0.3.

Identical landmarks were easy to deploy and identify, and because of the deliberate spatial distance it was very difficult to misidentify a landmark. At the beginning of this project, the green rings were thought of as merely as a starting point for a more generic system, which would use naturally occurring landmarks. Many of the techniques developed for artificial landmarks could easily be adapted for use with a system which picked out its own landmarks from "interesting" visual features.

Regardless, the simple type of landmark used in this experiment, which is small and simple to deploy is an excellent intermediary between a system which uses highly engineered landmarks (like the self-similar patterns[10]) and a system which dynamically identifies landmarks, a much more difficult proposition.

4.1.1. Problems

The landmarks we used are easily moved, either by third party agents within the environment or even the robot itself. In the actual tests, the markers were taped down because the rooms were often in use by other people. A simple improvement would be to use flat or disc-shaped landmarks which would also present a more uniform surface (and hence color) to the robot, increasing the robustness of the recognition algorithm and the accuracy of the position estimation and reducing the probability that the landmark would be disturbed.

When a landmark was finally sighted after the robot had not sighted a landmark for a relatively long period of time, the localization corrections were occasionally large because of the accumulated odometry error. This caused abrupt skips and other artifacts in the evidence grid map. Although landmark sightings were fairly frequent, the odometry error wasn't corrected continuously and so any small errors that crept in between sightings were be incorporated into the sensor readings as they were entered into the evidence grid.

4.1.2. Further Work

It would have been good to methodically and rigorously calculate the best Kalman gain factors, as well as the proper ratio in relation to distance. The values used for the odometry error were also largely guesses and could also have been refined. On the other hand, even with rough estimates of these values, the robot was able to use the landmarks to localize itself.

Originally, the landmarks were not designed to provide localization for mapping. Rather, they were supposed to be used by the robot to find its way back out of complex urban search and rescue situations. While the current system obviously keeps track of which landmark the robot is currently looking at, it doesn't have any way to deal with getting lost. In other words the system cannot recover from large odometry error (the sort of thing that is common enough in urban search and rescue situations). A dual Markov-Kalman approach seems to be well suited to the problem of recovering the robot's position from multiple sightings after a catastrophic odometry error.

5. CONCLUSIONS

The mobile robot implementation has shown the feasibility of simple transparent localization by fusing information from sparse landmarks and odometric data using a Kalman filter. Although most values of the Kalman gain yielded an improvement over an odometry-only approach, a range from 0.1 to 0.3 was found to be best for accurate mapping.

ACKNOWLEDGMENTS

This paper would not have been possible without the advice of Bruce Maxwell, Gil Jones' probing questions, and the excellent comments of Martin Krafft and Andrew Stout. Many thanks.

REFERENCES

1. M. Martin and H. Moravec, "Robot evidence grids," Tech. Rep. CMU-RI-TR-96-06, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 1996.
2. S. Thrun and A. Bucken, "Integrating grid-based and topological maps for mobile robot navigation," in *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, AAAI, August 1996.
3. J. Gutmann, W. Burgard, D. Fox, and K. Konolige, "An experimental comparison of localization methods," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98)*, 1998.
4. B. Yamauchi, A. Schultz, and W. Adams, "Mobile robot exploration and map building with continuous localization," in *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, IEEE, May 1998.
5. D. Fox, W. Burgard, and S. Thrun, "Probabilistic methods for mobile robot mapping," in *Proceedings of the IJCAI-99 Workshop on Adaptive Spatial Representations of Dynamic Environments*, IJCAI, 1999.
6. A. Schultz, W. Adams, and B. Yamauchi, "Integrating exploration, localization, navigation and planning with a common representation," *Autonomous Robots* **6**, June 2000.
7. H. Shatkay and L. Kaelbling, "Learning topological maps with weak local odometric information," in *IJCAI (2)*, pp. 920–929, 1997.
8. C. Madsen and C. Andersen, "Optimal landmark selection for triangulation of robot position," *J. Robotics and Autonomous Systems* **13**(4), pp. 277–292, 1998.
9. A. Briggs, D. Scharsctein, and S. Abbott, "Reliable mobile robot navigation from unreliable visual cues," in *Fourth International Workshop on Algorithmic Foundations of Robatics*, WAFR, 2000.
10. D. Scharstein and A. Briggs, "Real-time recognition of self-similar landmarks," tech. rep., Middlebury College, 1999.
11. C. Becker, J. Salas, K. Tokusei, and J. Latombe, "Reliable navigation using landmarks," in *Proceedings of thee IEEE International Conference on Robotics and Automation*, pp. 401–406, June 1995.
12. M. Betke and K. Gurvits, "Mobile robot localization using landmarks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, **2**, pp. 135–142, 1994.
13. J. Crowley, "Mathematical foundations of navigation an perception for an autonomous mobile robot." In L. Dorst, editor, Reasoning with Uncertainty in Robotics. Springer Verlag, 1996.
14. G. Welch and G. Bishop, "An introduction to the kalman filter," Tech. Rep. TR 95-041, University of North Carolina at Chapel Hill, Chapel Hill, NC, 1999.