Localization and Mapping for Autonomous Navigation in Outdoor Terrains : A Stereo Vision Approach

Motilal Agrawal, Kurt Konolige and Robert C. Bolles*
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025
{agrawal, konolige, bolles}@ai.sri.com

Abstract

We consider the problem of autonomous navigation in unstructured outdoor terrains using vision sensors. The goal is for a robot to come into a new environment, map it and move to a given goal at modest speeds (1 m/sec). The biggest challenges are in building good maps and keeping the robot well localized as it advances towards the goal. In this paper, we concentrate on showing how it is possible to build a consistent, globally correct map in real time, using efficient precise stereo algorithms for map making and visual odometry for localization. While we have made advances in both localization and mapping using stereo vision, it is the integration of the techniques that is the biggest contribution of the research. The validity of our approach is tested in blind experiments, where we submit our code to an independent testing group that runs and validates it on an outdoor robot.

1 Introduction

Recent advances in computing hardware coupled with the availability of different types of sensors have brought the dream of autonomous robots closer to reality now. Not surprisingly, good map making and localization are critical to the autonomous operation and navigation of these robots. In this paper, we concentrate on showing how it is possible to build a consistent, globally correct map for outdoor unstructured environments in real time using stereo vision as the primary sensor.

The goal is for a small outdoor robot to come into a new area, learn about and map its environment, and move to a given goal at modest speeds (1 m/sec). This problem is especially difficult in outdoor, off-road environments, where tall grass, shadows, deadfall, and other obstacles predominate. Although work in outdoor navigation has preferentially used laser rangefinders [11, 3, 5], we use stereo vision as the main sensor. Stereo vision is a good choice for several reasons: it is low power, low cost, and can register dense range information from close objects. More importantly, vision sensors allow us to use more distant objects as landmarks for navigation, and to learn and use color and texture models of the environment, in looking further ahead than is possible with range sensors alone. Our robot uses a combination of the following vision-based techniques to make globally consistent maps in real time.

- Efficient, precise stereo algorithms. Over the past 8 years, we have refined a set of stereo algorithms to take advantage of parallel-data instructions on PC hardware. We can perform stereo analysis on 512x384 images in less than 40 ms, enabling a faster system cycle time for real-time obstacle detection and avoidance.

- Visual odometry (VO) for fine registration of robot motion and corresponding obstacle maps. We have developed techniques that run at 15 Hz on standard PC hardware, and that provide 4% error over runs of 100 m. Our method can be integrated with information from inertial (IMU) and GPS devices, for robustness in difficult lighting or motion situations, and for overall global consistency. Visual odometry is compared to the combination of GPS and IMU, and shown to be superior in producing useable obstacle maps.

- A fast RANSAC method for finding the ground plane. The ground plane provides a solid base for obstacle detection algorithms in challenging outdoor terrain, and produces high-quality obstacle maps for a planning system.

- Sight-line analysis for longer-range inference. Stereo information on our robot is unreliable past 8m, but it

*Financial support from DARPA under the Learning Applied to Ground Robots (LAGR) program is gratefully acknowledged.

is possible to infer free space by finding "sight lines," directions in which it is likely there is freespace.

While we have made advances in many of the areas above, it is the integration of the techniques that is the biggest contribution of the research. The validity of our approach is tested in blind experiments, where we submit our code to an independent testing group that runs and validates it on an outdoor robot. In the most recent tests, we have finished first out of a group of eight teams, in some cases by a large margin.

1.1 System overview

Figure 1(a) shows our robot equipped with two stereo devices, each stereo device encompassing a 120 degree field of view, with a baseline of 12 cm. In the current setup, the robot is near-sighted: depth information for each stero device degrades rapidly after 6m. There is also an inertial unit (IMU) with angular drift of several degrees per minute, and a Garmin GPS. There are 4 Pentium-M 2 GHz computers, one for each stereo device, one for planning and mapmaking, and one for control of the robot and integration of GPS and IMU readings. In our setup, each stereo computer performs local map making and visual odometry, and sends registered local maps to the planner, where they are integrated into a global map. The planner is responsible for global planning and reactive control, sending commands to the controller.

1.2 Related work

There has been an explosion of work in mapping and localization (SLAM), most of it concentrating on indoor environments [6, 9]. The sensor of choice for indoor environments is a laser rangefinder, augmented with monocular or stereo vision. In much of this work, high-accuracy GPS is used to register sensor scans; exceptions are [5, 11]. In contrast, we forego laser rangefinders, and explicitly use image-based registration to build accurate maps. Other approaches to mapping with vision are [15, 16], although they are not oriented towards realtime implementations. Obstacle detection using stereo has also received some attention [15]. There have been a number of recent approaches to visual odometry [13, 14]. Our system is distinguished by realtime implementation and high accuracy using a small baseline in realistic terrain. Finally, [16] applies visual matching techniques to global consistency for maps, in a technique similar to the maximum likelihood method proposed here.

In the following sections, we first discuss local map creation from visual input. Then we examine visual odometry and registration in detail, and show how consistent global maps are created. Finally, we present performance results for several tests in Spring 2006.

2 Local map construction

The object of the local map algorithms is to determine, from the visual information, which areas are freespace and which are obstacles for the robot: the *local map*. From the stereo disparity image, we compute a nominal ground plane, which yields free space near the robot. We also analyze height differences from the ground to find obstacles. Finally, via the technique of sight lines we can infer freespace to more distant points.

2.1 Stereo analysis and ground plane extraction

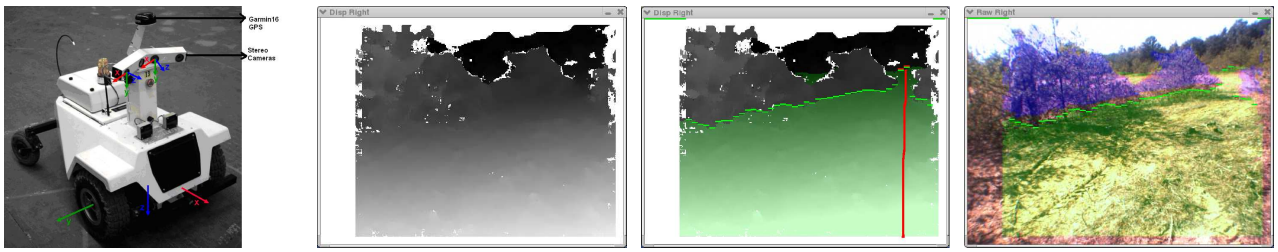
We use a fast stereo algorithm [8] to compute a disparity image at 512x384 resolution (Figure 1(b)). In typical outdoor scenes, it is possible to achieve very dense stereo results. The high resolution gives very detailed 3D information for finding the ground plane and obstacles. Each disparity image point $[u, v, d]$ corresponds to a 3D point in the robot's frame. We compute the 3D points using a 4x4 homography [2] $[X, Y, Z, w]^T \approx H[u, v, d, 1]^T$.

The most important geometric analysis is finding the ground plane. Although it is possible to detect obstacles using local variation in height, using a ground plane simplifies processing and yields more stable results. To extract a ground plane, we use a RANSAC technique [4]. Any three noncolinear 3D points define a plane hypothesis. We choose randomly over the set of points, biasing the choices towards points the are distributed over the first 5 meters near the robot, and rejecting hypotheses that are too slanted. Hypothesized planes are ranked by the number of points that are close to the plane.

This method is very robust, returning reasonable ground planes even in difficult situations, with clumps of grass or other distractors. Figure 1(d) shows an example of the extracted ground plane, with a green overlay indicating the inliers. Points that lie too high above the ground plane, but lower than the robot's height, are labeled as obstacles. This method is extremely simple, but has proven to work well in practice, even when the ground has modest dips and rises; one reason is that it only looks out to 6m around the robot. As the robot approaches a rise in the ground, for example, the ground plane will gradually assume the angle of the rise. A more sophisticated analysis would break the ground plane into several segments or model more complex shapes.

2.2 Sight lines

Because of the wide angle and short baseline of the stereo devices, depth information becomes very uncertain after 6 to 8m. Although we cannot precisely locate obstacles past this limit, we can determine if there is freespace,



(a) Our Robot (b) Disparity Image (c) Ground Plane (d) Obstacles

Figure 1. (a): Stereo sensors on our robot (b): Disparity image from the left view of the robot; closer pixels are lighter (c): Extracted ground plane, in green overlay. Limit of ground plane is shown by green bar; sight line has a red bar. (d): Ground plane overlayed on original image, in green. Obstacles are indicated in purple

using the following observation. Consider the interpreted image of Figure 1(c). There is a path that goes around the bushes and extends out a good distance. The ground plane extends over most of this area, and then ends in a distant line of trees. The trees are too far to place with any precision, but we can say that *there is no obstacle along the line of sight to the trees*. Given a conservative estimate for the distance of the trees, we can add freespace up to this estimate.

The computation of sight lines is most efficiently accomplished in the disparity space. We divide the disparity image into narrow columns, and for each column attempt to find a contiguous ground plane up to an obstacle. If the ground plane exists, and the obstacle is distant enough, we can add a sight line hypothesis to the local map. In Figure 1(c), the limits of the contiguous ground plane in a column are marked by green bars. Where the ground plane abuts a distant object, there is a vertical red line indicating a sight line. Note in the example that the sight line follows the obvious path out of the bushes.

3 Constructing consistent global maps

In this section we provide solutions to two problems: representing and fusing the information provided by visual analysis, and registering local maps into a consistent global map using visual odometry.

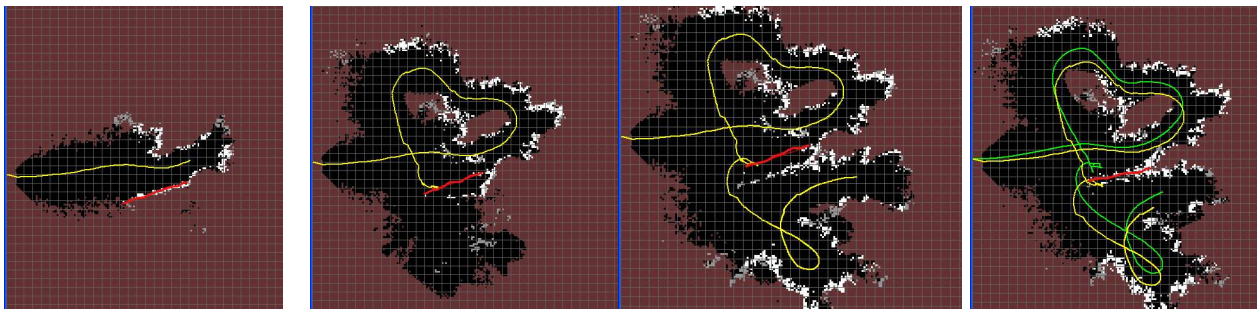
3.1 Map representation

For indoor work, a standard map representation is a 2D *occupancy grid* [12], which gives the probability of each cell in the map being occupied by an obstacle. Alternatives for outdoor environments include 2.5D elevation maps and full 3D voxel maps [7]. These representations can be used to determine allowable kinematic and dynamic paths for an outdoor robot in rough terrain. We choose to keep the simpler 2D occupancy grid, foregoing any complex calculation of the robot's interaction with the terrain. Instead, we abstract the geometrical characteristics of terrain into a set of categories, and fuse information from these categories to create a *cost* of movement.

We use a grid of 20cm x 20cm cells to represent the global map. Each cell has a probability of the belonging to the three categories derived from visual analysis (Section 2): obstacle, ground plane freespace and sight line freespace. Note that these categories are not mutually exclusive, since, for example, a cell under an overhanging branch could have both freespace and obstacle properties. We are interested in converting these probabilities into a cost of traversing the cell. If the probabilities were mutually exclusive, we would simply form the cost function: $c = \sum_i p_i c_i$, where c_i is the cost associated with category i. With non-exclusive categories, we chose a simple prioritization schedule to determine the cost. Obstacles have the highest priority, followed by ground plane, sight lines, and paths.

3.2 Registration and visual odometry

Our robot is equipped with a GPS that is accurate to within 3 to 10 meters in good situations. GPS information is filtered by the IMU and wheel encoders to produce a more stable position estimate. However, because GPS drifts and jumps over time, it is impossible using these devices to differentiate GPS errors from other errors such as wheel slippage, and the result is that local maps cannot be reconstructed accurately. Consider the situation of Figure 2. Here the robot goes through two loops of 10m diameter. There is a long linear feature (a low wall) that is seen as an obstacle at the beginning and end of the loops. Using the filtered GPS pose, the position of the wall shifts almost 2m during the run as is evident from 2(c). Our solution to the registration problem is to use *visual odometry* (VO) to ensure local consistency in map registration. Over larger regions, filtering VO with GPS information provides the necessary cor-



(a) First Stage (b) Second Stage (c) Third Stage (d) Third Stage using VO

Figure 2. (a),(b) & (c): Three stages during a run using GPS filtered pose. Obstacle points are shown in white, freespace in black, and the yellow line is the robot's path. The linear feature is marked by hand in red in all three maps, in its initial pose. (d): Map registration using VO in the same sequence. GPS filtered path is shown in yellow, VO filtered path is in green.

rections to keep errors from growing without bounds. We describe these techniques in the next two sections.

3.3 Visual Odometry

Our robot presents a challenging situation for visual odometry: wide FOV and short baseline make distance errors large, and a small offset from the ground plane makes it difficult to track points over longer distances. We have developed a robust visual odometry solution that functions well under these conditions; we describe it in some detail here. For a more detailed description of the visual odometry system please refer to our detailed paper [1].

Our visual odometry system uses feature tracks to estimate the relative incremental motion between two frames that are close in time. Corner feature points are detected in the left image of each stereo pair and tracked across consecutive frames. Figure 3(a) shows the tracked feature points over two consecutive frames. These feature points are then triangulated at each frame based on stereo correspondences. Three of these points are used to estimate the motion using absolute orientation. This motion is then scored using the pixel reprojection errors in both the cameras. We use the disparity space homography [2] to evaluate the inliers for the motion. In the end, the hypothesis with the best score (maximum number of inliers) is used as the starting point for a nonlinear minimization problem that minimizes the pixel reprojection errors in both the cameras simultaneously, resulting in a relative motion estimate between the two frames.

We have found that the approach outlined above is very efficient (> 15Hz) and works remarkably well, even for stereo rigs with a small baseline. The fact that we are triangulating the feature points for each frame, builds a firewall for error propagation. However, this also means that there will be a drift when the rig is stationary. In order to avoid this drift, we update the reference frame (the frame with reference to which the motion of the next frame is computed) only when the robot has moved some minimum distance (taken to be 5 cm in our implementation). Since, we are re-triangulating for every frame, it is important to calibrate the stereo cameras well. A standard plane based calibration step works well for all our experiments. The fundamental reason that our approach gives reliable motion estimates, even in small-baseline situations is due to the fact that we stick to image-based quantities and use both the left and right images symmetrically. The absolute orientation step used to generate the hypothesis uses the left and the right cameras symmetrically to generate the motion hypothesis. The hypothesis is evaluated and scored based on reprojection errors in both views, resulting in an accurate estimate of the motion. This estimate is then refined in the nonlinear minimization step which also uses the two cameras uniformly.

The IMU and the wheel encoders are also used to fill in the relative poses when visual odometry fails. This happens due to sudden lighting changes, fast turns of the robot or lack of good features in the scene (e.g. blank wall). Thus it complements the visual pose system. The relative motion between consecutive frames are chained together to obtain the absolute pose at each frame. Obviously, this is bound to result in accumulation of errors and drifting. We use GPS to correct the pose of the vehicle through a very simple filter and is described next.

3.4 Global Consistency

Relative motions between consecutive frames are chained together to obtain the absolute pose at each frame. Obviously, this is bound to result in accumulation of errors and drifting. We use GPS to correct the pose of the vehicle through a simple linear filter. Pose information is used when

the GPS receiver has at least a 3D position fix, and heading information is used only when the vehicle is travelling 0.5 m/s or faster, to limit the effect of velocity noise from GPS on the heading estimate. In addition, GPS measurements are used only if the robot has travelled a minimum distance from the last GPS measurement. The filter nudges the VO pose towards global consistency, while maintaining local consistency. Over larger loops, of course, the 3 m deviation of the GPS unit means that the map may not be consistent. In this case, other techniques such as wide-baseline image matching [10] would have to be employed.

The quality of the registration from filtered VO, shown in Figure 2(d), can be compared to the filtered GPS of Figure 2(c). The low wall, which moved almost 2m over the short loops when using GPS, is much more consistent when VO is employed. And in cases where GPS is blocked or degraded, such as under heavy tree cover in Figure 3(b), VO still produces maps that are locally consistent. It also allows us to determine wheel slips and stalls with almost no false positives – note the end of the run in Figure 3(b), where the robot was hung up and the wheels were slipping, and wheel odometry produced a large error.

3.5 Evaluation of Visual Odometry

We have implemented and tested our integrated pose system on several outdoor terrains. Since GPS is accurate to only about 3-4 meters, in order to validate our results, the robot was moved in a closed loop on a typical outdoor environment over 50–100 m, and used the error in start and end poses.

Table 1 compares this error for vehicle odometry (IMU + wheel odometry), visual odometry and the GPS integrated visual odometry for four loops. Except for the first loop, visual odometry outperforms the vehicle odometry, even without GPS filtering, and is comparable to the std of GPS (3m). VO substantially outperformed odometry in 3 of the 4 loops, mainly because of turns and wheel slippage during those runs. This is especially evident in loop 4, where the robot was slipping in mud for a substantial amount of time.

4 Results

The combined visual processing results in local maps that represent traversability with a high degree of fidelity. Figure 3(b) shows the results of an autonomous run of about 130m, over a span of 150 seconds. The first part of the run was along a mulch path under heavy tree cover, with mixed sunlight and deep shadows. For this run, we used an offline learning of mulch paths on a test site, then used the learned models on the autonomous run to recognize the mulch color. Cells categorized as path are shown in yellow;

Run Number	1	2	3	4
Distance(meters)	82.4	141.6	55.3	51.0
Method	Percentage Error			
Vehicle Odometry	1.3	11.4	11.0	31.0
Raw Visual Odometry	2.2	4.8	5.0	3.9
Visual Odometry & GPS	2.0	0.3	1.7	0.9

Table 1. Loop closure error in percentage.

	Test 12		Test 13	
	BL	**R**	**BL**	**R**
Run 1	5:25	1:46	5:21	2:28
Run 2	5:34	1:50	5:04	2:12
Run 3	5:18	1:52	4:45	2:12

Table 2. Run times for baseline (BL) and our robots(R).

black is freespace. Obstacles are indicated by purple (for absolute certainty), and white-to-gray for decreasing certainty. We did not use sight lines for this run.

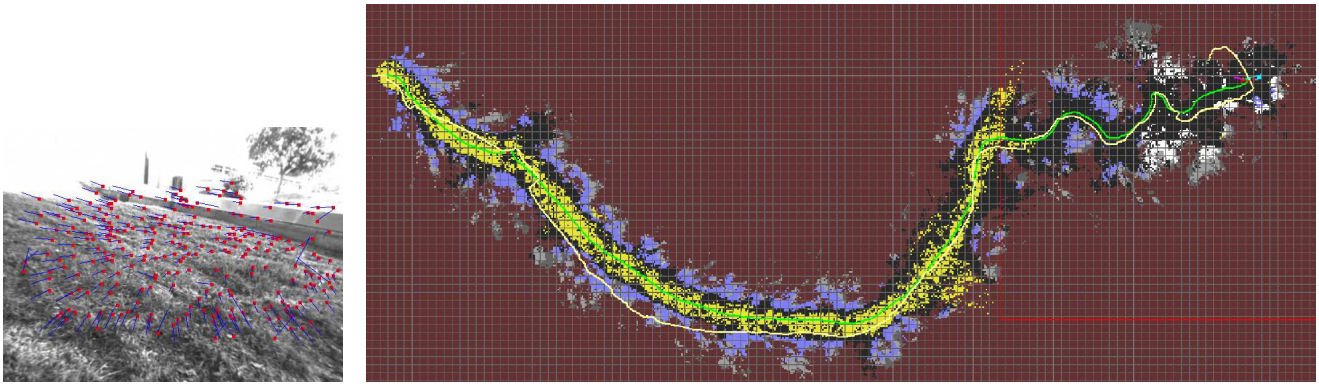
The path did not lead directly to the goal, and there were many opportunities for the robot to head cross-country. About two-thirds of the way through the run, no more paths were available, and the robot went through heavy grass and brush to the goal. The robot's pose, as estimated from filtered visual odometry is in green; the filtered GPS path is in yellow. Because of the tree cover, GPS suffered from high variance at times.

A benefit of using VO is that wheel slips and stalls are easily detected, with no false positives. For example, at the end of the run, the robot was hung up on a tree branch, and spun its wheels for a bit. The filtered GPS, depending on wheel odometry, moved far off the global pose, while the filtered VO stayed put.

4.1 Performance

One of the unique aspects of the project is independent experimental evaluation. An independent testing group ran monthly blind demos of the perception and control software developed by eight teams and compared their performance to a baseline system. Experiments were conducted every month, with project performers sending in their code on a flash disk to the evaluation team. The disks were inserted into a robot, which then was run over a test course. The last three demos (11, 12, and 13) were considered tests of performance from the first 18 months of the program.

Our team was first in the last two tests (Test 12 and 13), after being last in Test 11. Most of the problems in Test 11 were caused by using the baseline planner and controller (we wrote our own planner and controller for Tests 12 and 13). Table 2 shows the times for the runs in these tests. We achieved the short run times in Tests 12 and 13 through a



(a) Tracked Features

(b) Autonomous Run

Figure 3. (a): Example of visual odometry showing the motion of successfully tracked features (b): Reconstruction on a 130m autonomous run. Yellow is recognized path (learnt offline), black is freespace, and purple, gray and white are obstacles. The green line shows the robot's trajectory using VO and the yellow line shows the trajectory using IMU and GPS.

combination of precise map building and high-speed path planning. Our average speed was over 1.1 m/s, while the robot top speed was limited to 1.3 m/s. Map building relied on VO to provide good localization, ground-plane analysis to help detect obstacles, and sight lines to identify distant regions that are likely to be navigable.

5 Conclusion

We have demonstrated a complete system for off-road navigation in unstructured environments, using stereo vision as the main sensor. The system is very robust - we can typically give it a goal position several hundred meters away, and expect it to get there. But there are hazards that are not dealt with by the methods discussed in this paper: water and ditches are two robot-killers. Finally, we would like to use visual landmarks to augment GPS for global consistency, because it would give finer adjustment in the robot's position, which is critical for following routes that have already been found.

References

[1] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive gps. In *Proceedings International Conference on Pattern Recognition (ICPR)*, 2006.

[2] M. Agrawal, K. Konolige, and L. Iocchi. Real-time detection of independent motion using stereo. In *IEEE workshop on Motion*, 2005.

[3] P. Bellutta, R. Manduchi, L. Matthies, K. Owens, and A. Rankin. Terrain perception for DEMO III. In *Proc. of the IEEE Intelligent Vehicles Symp.*, 2000.

[4] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. ACM.*, 24:381–395, 1981.

[5] J. Guivant, E. Nebot, and S. Baiker. High accuracy navigation using laser range sensors in outdoor applications. In *ICRA*, 2000.

[6] J. S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *CIRA*, 1999.

[7] K. Iagnemma, F. Genot, and S. Dubowsky. Rapid physics-based rough-terrain rover planning with sensor and control uncertainty. In *ICRA*, 1999.

[8] K. Konolige. Small vision systems: hardware and implementation. In *Intl. Symp. on Robotics Research*, pages 111–116, 1997.

[9] J. J. Leonard and P. Newman. Consistent, convergent, and constant-time slam. In *IJCAI*, 2003.

[10] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. J. of Computer Vision*, 60(2):91–110, 2004.

[11] M. Montemerlo and S. Thrun. Large-scale robotic 3-d mapping of urban structures. In *ISER*, 2004.

[12] H. Moravec and A. Elfes. High resolution maps for wide angles sonar. In *ICRA*, 1985.

[13] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *CVPR*, 2004.

[14] C. F. Olson, L. H. Matthies, M. Schoppers, and M. W. Maimone. Robust stereo ego-motion for long distance navigation. In *CVPR*, 2000.

[15] A. Rankin, A. Huertas, and L. Matthies. Evaluation of stereo vision obstacle detection algorithms for off-road autonomous navigation. In *AUVSI Symp. on Unmanned Systems*, 2005.

[16] D. J. Spero and R. A. Jarvis. 3D vision for large-scale outdoor environments. In *Proc. of the Australasian Conf. on Robotics and Automation (ACRA)*, 2002.