3D Localization and Tracking in Unknown Environments

P. Saeedi, D. G. Lowe and P. D. Lawrence
Department of Electrical and Computer Engineering, Department of Computer Science
University of British Columbia
Vancouver, BC, V6T 1Z4, Canada
parvanes@ece.ubc.ca, lowe@cs.ubc.ca, peterl@ece.ubc.ca

Abstract

This paper describes a vision-based system for 3D localization and tracking of a mobile robot in an unmodified environment. The system includes a mountable head with three on-board stereo CCD cameras that can be installed on the robot. Here the main emphasis is on the ability to estimate the geometric information of the robot independently from any prior scene knowledge, landmark or extra sensory device. Distinctive scene features are identified using a novel algorithm and their 3D locations are estimated with a stereo algorithm. Using multi-stage feature tracking and motion estimation in a symbiotic manner, precise motion vectors are obtained. The 3D position of the scene features are updated by a Kalman filtering process. Experimental results show that robust tracking and localization can be achieved using our vision system.

1 Introduction

Real-time localization and motion tracking of a mobile robot relative to its environment have been subjects of interest for many years. Such interest has resulted in a variety of methods that vary based upon the environment, prior knowledge about the environment, sensor, cost, accuracy and the tracking approach. Many systems are implemented for confined environments by utilizing artificial landmarks [2]. Such systems are highly dependent on their modified surroundings and can not function under beacon-free condition.

Several other approaches use maps of their environments that are either supplied or self generated in a learning phase. For example, MINERVA [16] is a tour-guide robot that uses camera mosaics of the ceiling along with encoders and sonar sensors for its localization.

Sim and Dudeck [14] introduced a landmark-map based method in which the position of the camera was estimated by a linear position interpolation of some match correspondences. These correspondences were chosen from a set of images acquired from xy grid locations in the learning phase. Although this method is scene dependent and has a limited accuracy, it does not suffer from long-term drift.

Ayache and Faugeras [1] presented a vision-based navigation system by extracting chain of edges that were later approximated by linear segments. Using trinocular stereo and triplets of homologous segments a local 3D map was created at each frame. This map was used in the next frame to predict new matches and refine the motion between the two frames and to create and update a global 3D map. Although the system is scene independent, it fails where there are no significant number of edges present in the scene. Further more there is no report on the long term drift and the cost.

Harris [6] introduced a 3D vision system by measuring the visual motion of the images' features. The position of the camera and the 3D locations of the features in the scene were updated over time by means of Kalman filters. This system has the advantage of being scene independent, but tolerable motion range is limited to small amounts between frames.

This paper describes our on-going research [11] at the University of British Columbia on the problem of real-time purely vision-based 3D motion tracking. We assume neither prior knowledge of the environment nor specific landmarks in the scene. The scene is assumed to be mostly rigid, although some non-rigid portions of the scene can be detected and ignored. The motion of the robot is assumed to be limited in acceleration. This allows the feature search techniques to work on a predictable range of possible matches. Features such as a fast binary corner detector, multi-stage tracking, and incorporation of large numbers of features from the sides of the view field have increased the robustness and accuracy of our system.

Our approach consists of several phases that are executed as depicted in Figure 1.

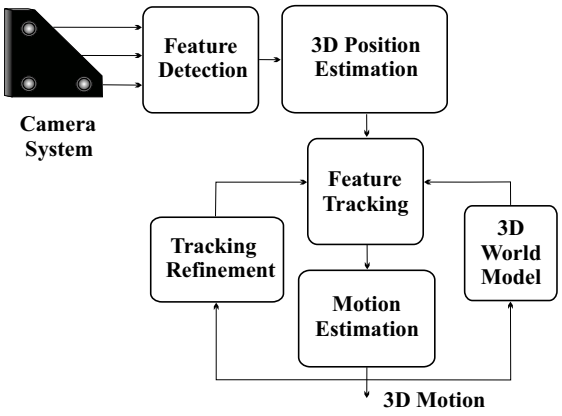

Figure 1: System overview.

- **Feature Detection:** meaningful features are detected in the images that can be tracked over a sequence of frames.
- **3D Position Estimation:** a 3D representation of the extracted features within the scene is obtained from a trinocular set of stereo images.
- **Match Tracking:** the features are matched using a multi-stage matching process.
- **Motion Estimation:** the relative motion of the camera is estimated in an absolute reference frame.
- **Tracking Refinement:** using an iterative process and the estimated motion, the match tracking is refined resulting in a more accurate motion estimation.
- **3D World Model Refinement:** the 3D world feature locations are refined by combining all the previous geometric measurements of the same features.

Details of each process are presented in the following sections and is followed by the presentation of the experimental results and conclusions.

2 Feature Detection

Although globally all the points in a scene convey some information about the motion, locally not all the pixel correspondences on the scene image carry valuable motion information. For example, edges, occlusions or areas of uniform intensity, can at best locally convey partial information about the motion. For this reason, we have chosen to work with discrete points of the scene that have maximum information content that are paritally invariant with respect to scale and rotation. In our previous work [11], the Harris corner detector [4] was used. Although it delivered a good localization with high stability, it was computationally expensive. A faster corner detector can lead to a more accurate and/or faster motion estimation since the changes between consecutive frames are less. We developed a binary corner detector [12], inspired by [15] that performs 1.8 times faster than Harris' method. The faster performance is achieved by exploiting binary images and substituting arithmetic operations with logicals. To generate a binary image that contains a good low-level information content, a Gaussian filter is first applied with a σ of 0.8. Next, the Laplacian is computed at each point of the intensity image. We approximate the image Laplacian value by:

$$\frac{\partial^2 I}{\partial x^2}+\frac{\partial^2 I}{\partial y^2} \approx (I_{i-1,j}+I_{i,j-1}+I_{i+1,j}+I_{i,j+1}-4I_{i,j}) \quad (1)$$

$I_{i,j}$ stands for the image intensity value at row i and column j. Such an approximation for the 2D Laplacian is separable and is implemented efficiently by logical operations. The binary image is then generated by the invariance of the sign of the Laplacian value at each point. At this point a circular mask with a diameter of 7 pixels is placed on each point of the binary image. The binary value of each point inside the mask is compared with that of the central point.

$$C(p_0,p) = \begin{cases} 1 & \text{if } L(p) = L(p_0), \\ 0 & \text{if } L(p) \neq L(p_0). \end{cases} \quad (2)$$

$L(p)$ represents the binary image value at location $p(x,y)$. A total running sum n is generated from the output of $C(p_0,p)$.

$$n(p_0) \quad = \quad \sum_{w} C(p_0,p) \quad (3)$$

n represents the area of the mask where the sign of the Laplacian of the image is the same as that of the central point. For each pixel to be considered a potential corner, the value of n should be smaller than at least half the size of the mask w. This value is shown as t in the corner response equation (4).

$$R(p_0) = \begin{cases} n(p_0) & \text{if } n(p_0) < t, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

For each candidate with $R(p_0) > 0$, a center of gravity $G(p_0)$ is computed.

$$G(p_0) = \sqrt{g(x_0)^2 + g(y_0)^2} \quad (5)$$

where

$$g(x_0) = \frac{\sum_w (x_0 - x)}{n(p_0)} \quad , \quad g(y_0) = \frac{\sum_w (y_0 - y)}{n(p_0)} \quad (6)$$

The center of gravity, G, provides the corner direction, as well as a condition to eliminate points with random distributions. Randomly distributed binary patches tend to have a center of gravity fairly close to the center of the patch. All points with close center of gravities are filtered out from the remaining process.

$$G(p_0) > |r_g| \quad (7)$$

The two conditions in (4) and (7) do not provide enough stability by themselves. A third inspection is performed by computing the directional derivative along the corner direction for the remaining candidates. Once again points with small directional intensity variations are eliminated. This condition is shown by:

$$|I(p_0) - I(p)| > I_t \quad (8)$$

I_t represents the brightness variation threshold. On average, the Binary method produces 80% of the number of corners that the Harris method finds. Our method detects corners on a 240×320 pixel image with sub-pixel accuracy in 23.293 msec.

3 3D Position Estimation

The 3D coordinates (X,Y,Z) relative to the robot for each feature is computed using a stereo algorithm. Our camera system captures a set of three images which are precisely aligned horizontally and vertically [10]. Candidate feature correspondences for the overlapping regions in the three stereo images are found and the measure of Normalized Sum of Square Differences are computed for each pair of match candidates. Then, the best match candidate is found by disparity sum minimization using the multiple-baseline algorithm [8]. In addition to the epipolar constraint, the agreement between the horizontal and vertical disparities is employed. This constraint eliminates unstable features, particularly those due to shadows. For the areas of the reference image that are common in either the horizontal or vertical stereo images, the Fua [3] method is employed. This method enforces the consistency of the matching process by incorporating a validity check along the epipolar lines. The Z value is computed using the average values of the two similar horizontal and vertical disparities.

4 Multi-Stage Feature Tracking

Corresponding $3D$ features are tracked from one frame (at time=t) to the next frame (at time=$t+\Delta t$). There is no prediction about the value or direction of the motion. Therefore a wide search scope is required to cover the range of possible motions. The displacement of the features between frames is affected by the feature to camera distance, the rotation and/or the translation that may have occured. This wide search scope increases the number of match candidates, elevating the possibility of false matches. In order to correct this problem, the feature tracking is performed in two iterative steps.

I. First, a large search window of 70×70 pixels is used around each feature point in the previous frame. This window provides a search boundary for the correspondence on the current frame. All the match candidates that fall inside these boundaries are chosen. Next, the Normalized Sum of Squared Differences for windows of 13×13 pixels around each corner and its candidates is used to select the most similar feature. The match correspondences between the two frames are used to estimate the motion. Although this first stage has limited accuracy, these results serve as a good initial estimate of the exact motion.

II. Second, using the rough motion estimation, all the features of the previous frame are transfered to the coordinates of the current frame. Regardless of the motion type or the distance of the features from the coordinate center, features with a persistent 3D location will end up on a very close neighborhood (we allow up to 4 pixels from their true correspondence in the current frame). By employing a constraint on the distance of the match features, correspondences are found very quickly with high accuracy. Since the feature space is sparse, there is usually only one candidate for each feature in the neighborhood. The Normalized Sum of Squared Differences is used to find the true match when there is more than one candidate.

Figure 2 shows match correspondences in the two steps. Not only is the number of false matches decreased when the prior motion information is used, but the total number of correct matches is increased by 30%. These matches were missed in the initial stage due to ambiguity with the large search window.

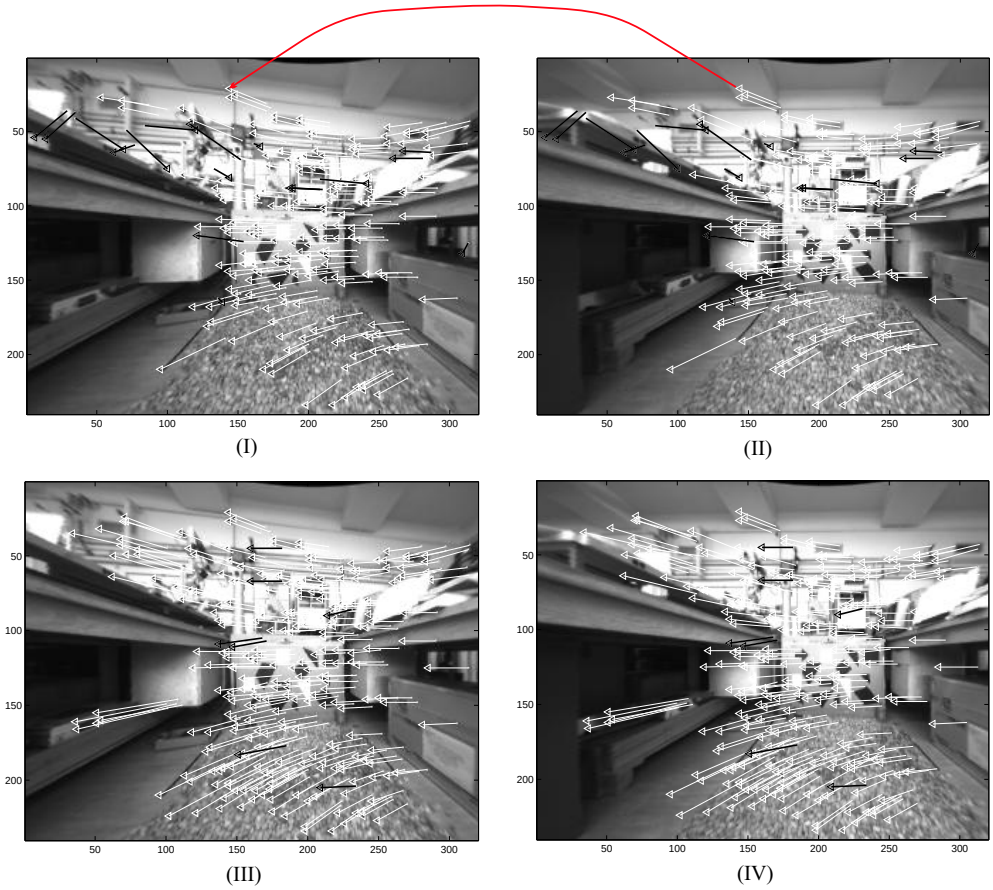

Figure 2: (I) and (II) Feature tracking with no motion knowledge. (III) and (IV) Feature tracking using the rough motion estimation. Rejected false matches are shown by black lines. The red arrow shows one feature's motion from first frame to next.

5 Motion Estimation

Having a set of corresponding features in two consecutive frames, the motion estimation becomes a problem of optimizing a 3D transformation. This transformation projects the world features that are constructed from the first image onto the second image. Although the 3D construction of 2D features is a non-linear function, the problem of motion estimation is still well-posed. This fact occurs because 3D motion includes rotations and translations. Rotations are functions of the cosine of the rotation angles. Translations parallel to the image plane are almost linear. Translations toward or away from the camera introduce a perspective distortion as a function of the inverse of the distance from the camera. The problem of 3D motion estimation is therefore a promising candidate for the application of Newton's method, which is based on the assumption that the function is locally linear. To minimize the probability of converging to a false local minimum, outliers are eliminated during the iteration process.

In our method, at each iteration, a correction vector x is computed that is subtracted from the current estimate resulting in a new estimate. If $P^{(i)}$ is the vector of image coordinates (u, v) for iteration i, then

$$P^{(i+1)} = P^{(i)} - x \tag{9}$$

Given a vector of error measurements between the world $3D$ features and their projections, we find the x that eliminates (minimizes) this error. The effect of each element of correction vector x_i on error measurement e_i, is the multiplication of the partial derivative of the error with respect to that parameter. This parameter is found by considering the main assumption of local linearity of the function

$$WJx = We \qquad \text{where} \qquad J_{ij} = \frac{\partial e_i}{\partial x_j} \tag{10}$$

J is the Jacobian matrix and e_i represents the error between the predicted location of the object and the actual position of the match found in image coordinates. W is a diagonal weighting matrix and its

components represent the uncertainty of each feature over time. This uncertainty decreases after a feature has been viewed repeatedly. More details about the weight coefficients are provided in the next section. Each row of the matrix from Equation 10 states that one measured error, e_i, should be equal to the sum of all changes in that error resulting from the parameter correction [7]. Equation 10 is usually over-determined and no unique solution exists, therefore we find a vector x that minimizes the 2-norm of the residual.

$$min\|WJx - We\|^2 \tag{11}$$

Equation 11 has the same solution as the normal equation

$$x = [(WJ)^T WJ]^{-1}(WJ)^T We \tag{12}$$

In each iteration of Newton's method, we solve the normal Equation 12 for x using LU decomposition [9].

6 3D Model Refinement

Several parameters can impact the system accuracy. Sensor noise and quantization associated with the image can each introduce a slight displacement at the feature locations within the image. Furthermore, such inaccuracies can lead to faulty match correspondences between frames. As the mobile robot navigates in its environment, most of the features fall into the camera field of view for a period of several frames. Detection of a feature in each frame by itself provides additional information about that feature. Also, as the camera becomes closer to a feature or as the features move from the image sides to its center, the 3D accuracy of the feature can improve dramatically. The second improvement is due to the fact that camera images are more distorted near the corners of the image as compared to the center of the image. Therefore by combining the measurement for a feature with all the previous information associated with the same feature, the uncertainty of that feature will reduce.

A positional covariance is associated with each observed feature using a Kalman filter. Each filter is updated using new information for the same feature over time. The location covariance of each feature, w_i in Equation 13, is used as the coefficient of the weight matrix in Equation 10. Features that are either close to the camera or are seen over a few frames or have stable 3D locations, have high weights in the least-squares minimization (Equation 11). The Kalman filter implementation is achieved in a similar fashion to Shapiro's method [13].

$$w_i^{-1} = w_{i-1}^{-1} + V_i^{-1} \tag{13}$$

$$x_i = x_{i-1} + k_i(b_i - x_{i-1}) \tag{14}$$

$$k_i = w_i V_i^{-1} \tag{15}$$

In these equations i represents the frame number. w_i is the uncertainty in the estimation of x corresponding to frame i, k_i denotes the filter gain, b_i indicates the current measurement of the feature, V_i represents the covariance matrix of the errors. To prevent bias from distant features, we work in the disparity space with axes that are the current image plane coordinates and the corresponding feature disparity [5]. We keep track of the features for a while even if they move out of camera's field of view. However if a feature is not seen in the last 6 consecutive frames it will be retired.

7 Experimental Results

The performance of the system has been evaluated through several experiments. The camera system captures gray scale images of 480×640 pixels. These images are rectified to a size of 240×320 pixels on a 1.14 GHz AMD Athlon^{TM} processor. In order to reduce the ambiguity between the pitch rotation and sideways movements, a set of wide angle lenses with a 104^o field of view is used. These lenses incorporate information from the sides of the images that behave differently under translational and rotational movements. The integration of the multi-stage feature tracking and motion estimation, allow a larger motion range between frames.

A graphical interface was created that produces a 3D model of the camera motion as it is being estimated. Two of these experiments are presented in this section. The first experiment is designed to measure the resulting drift using a closed path. For this purpose, the camera starts moving from point A on an arbitrary route and then returns to its starting point (Figure 3). Along this path 72 consecutive frames are processed. Table 1 represents the existing drift in this example. In this table Motion Range represents the absolute amount of the motion along each coordinate variable. The motion range along the Z axis is 2.94 m, along the X axis is 1.25 m, and around the Y axis has a rotational range of 10^o. To show the improvement resulting from Kalman filtering, the results are presented once with the Kalman filter and once without.

The second experiment is intended to investigate the localization accuracy. The camera is moved along a route from point A to the known point B. A number of 32 frames are processed along this route. The

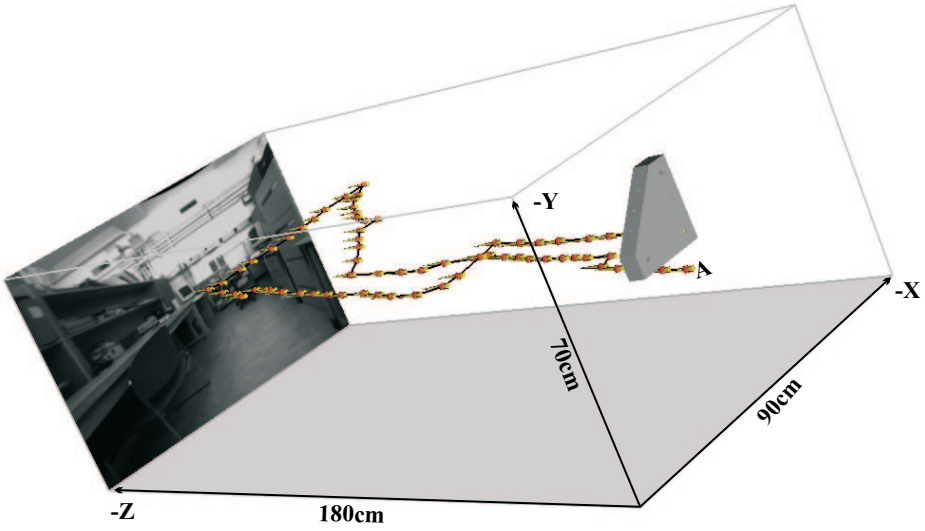

Figure 3: A graphical user interface displays the 3D position and orientation of the camera in a virtual environment.

Table 1: System drift for the first experiment.

Kalman Filter	Motion Range X(cm), Y(cm), Z(cm), yaw(^{o}), pitch(^{o}), roll(^{o})	System Drift X(cm), Y(cm), Z(cm), yaw(^{o}), pitch(^{o}), roll(^{o})
On	$125, 0, 294,$ $0^o, 10^o, 0^o$	$-1.77, -7.57, -0.45$ $1.1^o, 0.22^o, -0.06^o$
Off	$125, 0, 294,$ $0^o, 10^o, 0^o$	$13.68, -8.06, 4.04$ $0.20^o, -2.43^o, 0.24^o$

coordinates of point B are measured manually in a coordinate system with center A, Figure 4. Table 2 provides the estimated location and compares it with the actual location of point B. The path in this experiment is roughly L-shaped and the motion includes translation as well as rotation. Th motion range includes the overall translation of 159 cm along Z, 45 cm along X and the pitch rotation of 45^o. From these results, the drift of the system is limited to only a few centimeters.

Table 2: Localization error for the second experiment.

Actual Location X(cm), Y(cm), Z(cm)	Estimated Location X(cm), Y(cm), Z(cm)
$0, 0, -158.9$	$0.21, -1.09, -154.85$

The system performance has a speed of 2.8Hz for gray scale images of 240×320 pixels on a 1.14 GHz AMD Athlon^{TM} processor. The majority of the total computation time is spent on the correspondence matching routine (48.9%) in the stereo and the feature tracking processes.

8 Conclusions and Future Work

In this paper, we described a $3D$ vision-based location and motion tracking system for unknown environment to be used for the control of a mobile robot. This system effectively reduces the computational cost by employing a fast binary feature detector. The 3D performance is achieved by a set of triple stereo cameras with wide fields of view. Features such as sub-pixel resolution for feature detection and stereo, multi-stage feature tracking and motion estimation, and Kalman filtering have improved the accuracy and robustness of the system.

Currently, we are looking at the problem of minimizing the noise that is introduced in the rectification process. We have planned to incorporate a multi-scale match correspondence process to speed up the performance.

9 Acknowledgments

This work was supported by the Canadian IRIS/PRECARN Network of Centres of Excellence.

References

[1] N. Ayache, O.D. Faugeras, F. Lustman, and Z. Zhang. Visual Navigation of a Mobile Robot. *IEEE Interna-*

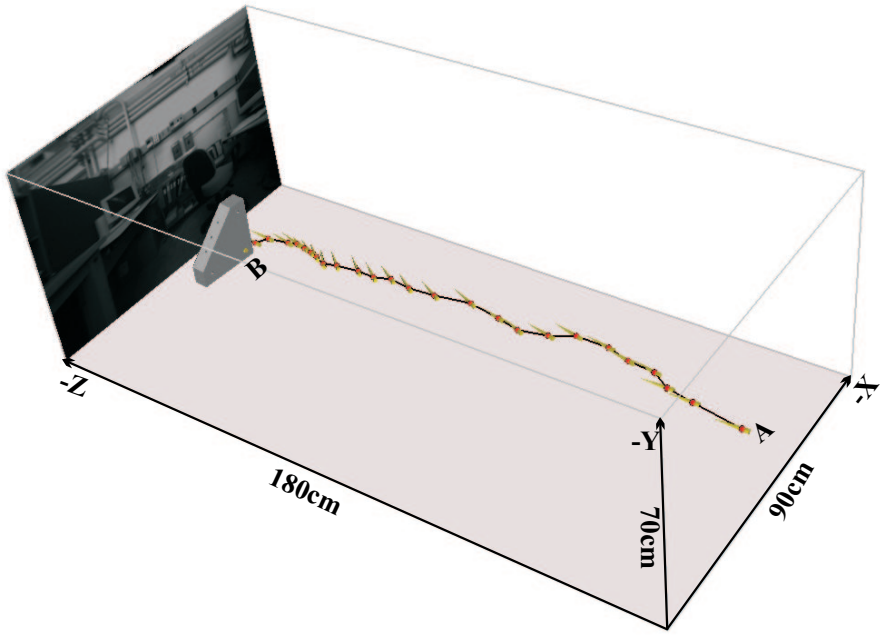

Figure 4: The 3D motion of the camera from point A to point B. The orientation at each frame is shown with a 3D arrow along the path. As shown, the camera rotates as well as translates.

tional Workshop on Intelligent Robots,, pages 651–658, 1988.

[2] J. Borenstein, H.R. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques.* AK Peters Wellesley, Massachusetts, 1996.

[3] Pascal Fua. *A parallel stereo algorithm that produces dense depth maps and preserves image features.* Machine Vision and Applications, Springer-Verlag, 1993.

[4] C. Harris and M. Stephens. A combined corner and edge detector. *Proceeding 4'th Alvey Vision Conference*, pages 147–151, 1988.

[5] C.G. Harris and J.M. Pike. 3D positional integration from image sequences. *Image and Vision Computing*, pages 87–90, 1988.

[6] Chris Harris. *Geometry from Visual Motion.* Active Vision, MIT Press, Cambridge, 1992.

[7] David G. Lowe. Fitting Parameterized Three-dimensional Models to Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 441–450, 1991.

[8] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 353–363, 1993.

[9] W.H. Press, S.A. Teukolsk, W.T. Vetterling, and B.P. Flannery. Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, 1992.

[10] Point Grey Research. Triclops Stereo Vision System. Technical report, Department of Computer Science, University of British Columbia, Vancouver, www.ptgrey.com, 1997.

[11] P. Saeedi, P. Lawrence, and D. Lowe. 3D motion tracking of a mobile robot in a natural environment. *IEEE International Conference on Robotics and Automation*, pages 1682–1687, 2000.

[12] P. Saeedi, D. Lowe, and P. Lawrence. An efficient binary corner detector. *The Seventh International Conference on Control, Automation, Robotics and Vision*, 2002.

[13] L.S. Shapiro, A. Zisserman, and M. Brady. 3D Motion Recovery via Affine Epipolar Geometry. *International Journal of Computer Vision, Vol 16*, pages 147–182, 1995.

[14] R. Sim and G. Dudek. Mobile Robot Localization from Learned Landmarks. *IEEE International Conference on Intelligent Robots and Systems*, pages 1060–1065, 1998.

[15] S. M. Smith and J. M. Brady. SUSAN- A new approach to low level image processing. *International Journal of Computer Vision*, pages 45–78, 1997.

[16] S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA: A second generation mobile tour-guide robot. *IEEE International Conference on Robotics and Automation*, pages 1999 –2005, 1999.