Proceedings of the 2001 IEEE/RSJ
International Conference on Intelligent Robots and Systems
Maui, Hawaii, USA, Oct. 29 – Nov. 03, 2001

Estimating the Motion of an Underwater Robot from a Monocular Image Sequence

Rafael Garcia, Xevi Cufi and Marc Carreras

Computer Vision and Robotics Group
Institute of Informatics and Applications
University of Girona, E.P.S.
17071 Girona, Spain
e-mail: {rafa,xcuf,mcarreras}@eia.udg.es

Abstract

When underwater vehicles perform navigation close to the ocean floor, computer vision techniques can be applied to obtain quite accurate motion estimates. The most crucial step in the vision-based estimation of the vehicle motion consists on detecting matchings between image pairs. Here we propose the extensive use of texture analysis as a tool to ameliorate the correspondence problem in underwater images. Once a robust set of correspondences has been found, the three-dimensional motion of the vehicle can be computed with respect to the bed of the sea. Finally, motion estimates allow the construction of a map that could aid to the navigation of the robot.

1 Introduction

Quite often the underwater vehicles have to perform certain tasks alongside the ocean floor. The rich amount of visual information available when the submersible moves next to a static environment (bed of the sea, rocks, etc.) has been exploited in the last years to provide additional sensing to the robot. Many vision-based systems have been proposed (*i.e.* [1,2,3]), nevertheless, it is necessary to provide tools to improve the accuracy of those systems that have to work in an adverse environment. Unfortunately, underwater images are difficult to process due to the medium transmission characteristics [4]. These properties provoke a blurring of the elements of the image with high clutter in the regions of interest and lack of distinct features. Region-correlation techniques have been extensively used to search for correspondences between pairs of images [5,6], allowing, thereby, the detection of motion. Although these approaches lead to successful matchings in well-contrasted images, in some cases the lack of image features cause the matching procedure to fail. For this reason a new approach based on region matching and selective texture analysis is proposed in this paper. The extensive use of textural operators can highly improve the quality of the image correspondences, ameliorating the subsequent motion estimation. This is accomplished by equipping the URIS underwater robot (figure 1) with a color camera which acquires images of the bottom of the sea. As the vehicle moves, its 3D motion can be computed by making use of the intrinsic parameters of the camera and the detected correspondences. Finally, a visual map of the zone surveyed by the submersible can be constructed since the motion parameters are already known [7].

The paper is organized as follows: first, section 2 describes the general aspects of the textural operators that have been used in our study. Next, the motion detection problem will be tackled, analyzing those aspects that could affect the quality of the estimates. Afterwards, the construction of a 3D visual map is analyzed. Finally, results on real images are presented.

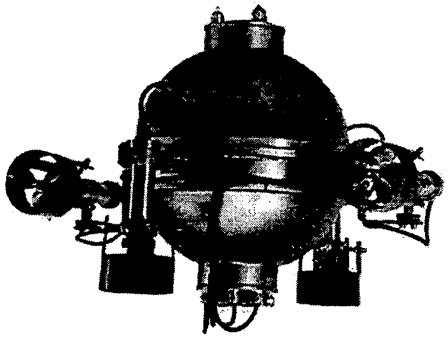

Figure 1: The URIS Underwater Vehicle, prototype developed at the University of Girona

0-7803-6612-3/01/$10.00©2001 IEEE

2 Texture analysis

We have tested the behavior of different texture operators in underwater images. Concretely, we searched for texture parameters that remain constant for the same scene patch for the whole image sequence. One of the operators that have been used are the *texture energy filters* [8], which are derived from the computation of a series of statistical measures on a pre-filtered image. This image is obtained by applying a set of masks (3×3 or 5×5) that define some textural properties of the image. In order to obtain these masks, a series of vectors defining some textural proprieties are combined. The typical vectors are *level*, *edge*, *spot*, *wave*, *ripple* and *oscillation* [8].

A second texture operator based on the spatial distribution of pixels in the image has been used: *Co-occurrence matrix* [9]. It takes into account the frequency of appearance of the pairs of pixels located at a distance *d* and an angle θ (co-occurrences). A set of statistics is computed for every co-occurrence matrix, obtaining the textural characteristics of the image.

Finally, since a textured region can be described by means of its texture spectrum –that is, a set of values called texture units– a set of 3×3 simple local patterns can be defined. The different texture units can be determined from these patterns, obtaining a texture measure of the considered region. This last texture operator, known as *Local Binary Pattern* [10], has also been used in our study.

It should be taken into account that the first two operators can generate several measurements, depending on the number of orientation angles, the distance of correlation and the size of the neighborhood. We have tested several configurations of these texture operators in order to find the most advantageous set-up for our application. Finally, we chose 4 different angles for the coocurrence matrix, taking only distances of 1-pixel; and 9 masks of the energy filter taking only a 3×3 neighborhood. From our experience, the use of larger neighborhoods provides little improvement at the expense of a higher computational cost.

3 Motion detection

The AUV motion estimation process is performed in several phases, as illustrated in figure 2. First, the radial distortion of the camera is corrected, then a set of matches is computed in order to find the motion from one image to the next. This motion is measured as a rotation **R** plus a translation **t** that relates the position of the vehicle in two consecutive time instants. Finally, the relative positions are mapped into a global frame, constructing a map of the surveyed zone.

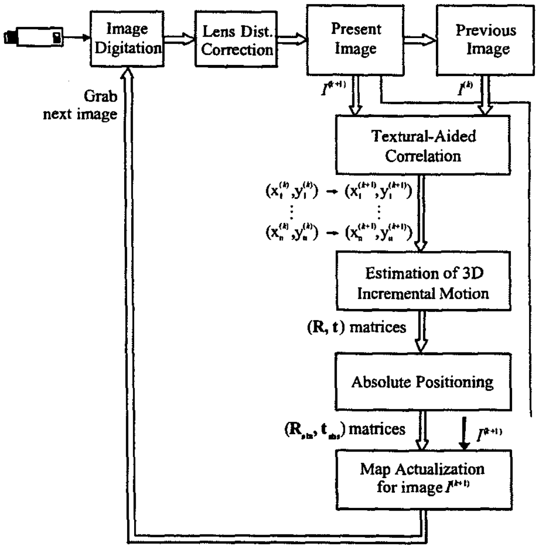
Image Digitation
Lens Dist. Correction
Present Image
Previous Image
Grab next image
$I^{(k+1)}$
$I^{(k)}$
Textural-Aided Correlation
$(x_1^{(k)},y_1^{(k)}) \rightarrow (x_1^{(k+1)},y_1^{(k+1)})$
$(x_n^{(k)},y_n^{(k)}) \rightarrow (x_n^{(k+1)},y_n^{(k+1)})$
Estimation of 3D Incremental Motion
(R, t) matrices
Absolute Positioning
$(\mathbf{R}_{abs}, \mathbf{t}_{abs})$ matrices
$I^{(k+1)}$
Map Actualization for image $I^{(k+1)}$

Figure 2: Dataflow of the motion-detection algorithm

3.1 Correction of geometric deformations

The physical lenses introduce a non-linear distortion in the observed image points. Moreover, when treating underwater images, the ray deflections at the water-camera housing and the air-camera housing interfaces introduce a second distortion [11]. For this reason, camera calibration has to be performed underwater.

We describe here a derivation of the Faugeras-Toscani algorithm to correct the radial distortion by means of the following equations, derived from [12]:

$$x_u = \left(\frac{x_d - x_0}{k_x}\right) + \left(\frac{x_d - x_0}{k_x}\right) \cdot k_1 \cdot r^2 + c_x \qquad (1)$$

$$y_u = \left(\frac{y_d - y_0}{k_y}\right) + \left(\frac{y_d - y_0}{k_y}\right) \cdot k_1 \cdot r^2 + c_y \qquad (2)$$

where (x_u, y_u) are the ideal undistorted coordinates of the measured distorted point (x_d, y_d), and (c_x, c_y) are the coordinates of the center of the image. The parameters k_x, k_y are the scaling factors in the x and y directions, respectively. They account for differences on the image axes scaling. The principal point of the image is defined by (x_0, y_0), and it represents the coordinates of the projection of the optical center of the camera on the image plane. k_1 is the first term of the radial correction series, and r is the squared distance of (x_d, y_d) from the center of the image, and accomplishes:

$$r=\sqrt{\left(\frac{x_d-x_0}{k_x}\right)^2+\left(\frac{y_d-y_0}{k_y}\right)^2} \qquad (3)$$

Once these parameters are known, image correction for radial distortion can be computed. In our implementation, the undistorted values are obtained from a Look Up Table that has been computed offline.

3.2 Finding correspondences

The matching algorithm is sequenced as following: first, the high gradient areas of the first image are detected through a detector of interest points (some sort of corner detector). Then, for every corner in the first image, a set of possible matchings is established in the second image. These matches are detected by means of area-correlation. Finally, a set of texture measures is taken for the area surrounding the original corner in the first image, and this texture set is compared to the textures computed at every possible matching on the second image. As will be shown in the results, the most similar texture patch corresponds normally to the correct match.

3.2.1 Detection of interest points

A very simple and fast detector of interest points has been implemented. First, a Canny edge detector is applied to the image [13], binarizing the output of the filter at a quite high threshold. Thus, an undersegmented image containing only the most relevant contours of the image is obtained. A pixel is considered to be an *interest point* if it is in the intersection between two straight lines, that is to say, if it has 3 or more neighbors also belonging to any edge.

3.2.2 Gray-level region-correlation

In order to establish correspondences between images a classical correlation technique can be applied. Since our images are acquired by a color camera, we have found that best results are obtained if the correlation is applied to the blue band of the image. This fact is related to the variation of the optical properties of different water bodies depending on the interaction between the light and the aquatic environment [4]. Given that the light suffers less absorption when it has a higher frequency, the blue component of the image provides higher contrast than the average of all frequencies, that is, the intensity component.

A similarity function measures whether a point in the second image is likely to be the right matching of a given interest point in the first image. We then define a minimum threshold for a matching point to be considered a possible correspondence of a given interest point. In this way, for each point in the first image, we thus have a set of p candidate matches in the second image. The number p of possible matches may be different from one interest point to another.

3.2.3 Texture extraction and similarity measure

Given an interest point, the problem now is to decide which is the right match among the p candidates selected by the correlation procedure. An $n\times n$ neighborhood is selected around the interest point. The texture operators defined in section 2 are computed in this area, sub-sampling the $n\times n$ window. In this way a texture vector is obtained for the neighborhood of the interest point. Every point of the neighborhood provides 9 measures of the energy filters, 4 of the co-occurrence matrix, and 1 of the Local Binary Pattern operator. Thus, the texture vector contains $n^2\times(9+4+1)=k$ texture values. The same operation is performed in the second image centering the $n\times n$ window on every one of the p possible matches. Then, the p texture vectors are compared with the texture vector of the interest point by computing the point-to-point Euclidean distance. The best match is selected as the one minimizing the following distance:

$$d(\vec{a},\vec{b}_j)=+\sqrt{\sum_{i=1}^{k}(a_i-b_i)^2}, \quad \forall j\in[1..p] \qquad (4)$$

where $\vec{a}$ is the texture vector of the interest point in the first image, and $\vec{b}_j$ stores the texture attributes of every candidate matching.

3.3 Determination of the motion parameters

The next problem to solve is the estimation of the motion of the submersible between two images $I^{(k)}$ and $I^{(k+1)}$. The motion is expressed in terms of a rotation **R** and a translation **t**. According to the epipolar geometry theory [14], a linear equation can be written relating the *essential matrix* **E** with the correspondences between imaged points in consecutive frames [15] (see eq. 5). The essential matrix expresses the position and orientation of a coordinate system (*i.e.* a camera) with respect to another. It is possible to arrange the elements of the 3×3 **E** matrix forming a 9×1 column vector ε, obtaining the following equation:

$$\mathbf{U}\cdot\varepsilon=0 \qquad (5)$$

where

$$\mathbf{U}=\begin{bmatrix} x_1^{(k)}x_1^{(k+1)} & y_1^{(k)}x_1^{(k+1)} & x_1^{(k+1)} & x_1^{(k)}y_1^{(k+1)} & y_1^{(k)}y_1^{(k+1)} & y_1^{(k+1)} & x_1^{(k)} & y_1^{(k)} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^{(k)}x_n^{(k+1)} & y_n^{(k)}x_n^{(k+1)} & x_n^{(k+1)} & x_n^{(k)}y_n^{(k+1)} & y_n^{(k)}y_n^{(k+1)} & y_n^{(k+1)} & x_n^{(k)} & y_n^{(k)} & 1 \end{bmatrix}$$

and $\varepsilon=[e_{11} \quad e_{12} \quad e_{13} \quad e_{21} \quad e_{22} \quad e_{23} \quad e_{31} \quad e_{32} \quad e_{33}]^T$ correspond to the elements of matrix **E**.

If the parameters in ε are available, the motion (**R**,**t**) could be computed from (see [15,16]):

$$\mathbf{E}^T\cdot\mathbf{t}=0 \qquad (6)$$

$$\mathbf{E}=\mathbf{t}_\times\cdot\mathbf{R} \qquad (7)$$

where $\mathbf{t}_\times$ is an antisymmetric matrix defined by **t** that accomplishes $\mathbf{t}_\times \cdot \mathbf{p} = \mathbf{t} \times \mathbf{p}$ for any 3D vector $\mathbf{p}$. Therefore, the first step consists on obtaining the elements of matrix **E**. A least squares solution can be applied to solve for ε in the following way:

$$\min_{\varepsilon} \|\mathbf{U} \cdot \varepsilon\|^2, \text{ subject to } \|\varepsilon\| = \sqrt{2} \tag{8}$$

Once matrix **E** has been found (re-arranging the elements of ε), translation **t** and rotation **R** can be obtained from the minimization of equations (9) and (10), respectively.

$$\min_{\mathbf{t}} \|\mathbf{E}^T \cdot \mathbf{t}\|^2, \text{ subject to } \|\mathbf{t}\| = 1 \tag{9}$$

$$\min_{\mathbf{R}} \|\mathbf{E} - \mathbf{t}_\times \cdot \mathbf{R}\|^2 \tag{10}$$

By performing the minimizations of equations (9) and (10) the incremental three-dimensional information on the motion of the robot is obtained for two consecutive time instants. For a more detailed description on how to find (**R**,**t**) from image correspondences the reader is addressed to [15,16].

4 Map construction

Once the translation **t** and rotation **R** parameters between images $I^{(k)}$ and $I^{(k+1)}$ have been found, the perspective projection matrix **P** can be obtained:

$$\mathbf{P}^{(k)} = \mathbf{A}[\mathbf{I} \ \ \mathbf{0}] \tag{11}$$

$$\mathbf{P}^{(k+1)} = \mathbf{A}[\mathbf{R} \ \ \mathbf{t}] \tag{12}$$

where **A** is the 3x3 matrix of intrinsic parameters of the camera, obtained through calibration [12,17]. In this way, the 3D coordinates M=(X,Y,Z) of any image point m=(x,y) can be obtained by means of:

$$s_1 \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{13}$$

where s_1 is a scaling factor, and m and M are expressed in homogeneous coordinates. In order to obtain more accurate estimations, when the robot is moving forward, three different views are used to compute the 3D of the coordinates of a point M, as illustrated in figure 3. The parameter α is selected on-line, depending on the velocity of the submersible. The estimation of M is performed by correlating a neighborhood of m_1 in the different images of the sequence. Sometimes, three-dimensional occlusions caused by irregularities in bathymetry provoke this estimation to fail [18]. For this reason it is important to exploit temporal redundancy by selecting only those features that have a high correlation score, discarding the others.

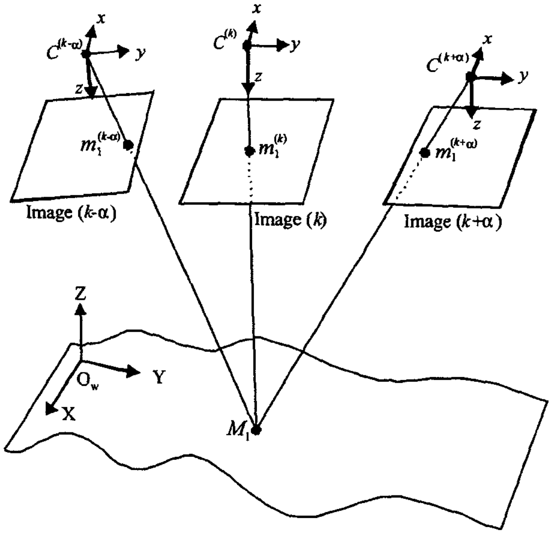

Figure 3: Integration of three frames taken from three different points of view of the same scene.

Once the 3D coordinates of the points projected on image $I^{(k+1)}$ are known, they can be mapped together with the pixels of the image $I^{(k)}$ to construct a composite image. Matrices (**R**,**t**) relate the 3D points of one image with those of the previous one. If a common reference frame is selected for all the images, then the relative increments given by (**R**,**t**) should be converted to absolute measures ($\mathbf{R}_{abs}$,$\mathbf{t}_{abs}$) expressed in terms of the world coordinate system (O_w in figure 3). The resulting map is stored in a matrix where every element stores four values: the 3D coordinates of that point (X,Y,Z) and its gray level as imaged in the image. Since several contributions from different images are available for the same 3D point, the system averages the values of the matrix as new values are available. A visualization tool for such a map is being developed. At the moment we can only evaluate the accuracy of the motion estimation process by means of the 2D mosaic-based visualization map described in [7].

5 Results

Several experiments have been performed in order to validate the texture-based matching strategy. Typical underwater situations have been used to test our system. The images have been acquired by a color camera carried by the URIS underwater robot while this was being teleoperated. The acquisition frame rate was set to 3 f.p.s. The images were stored to disk and have been processed offline.

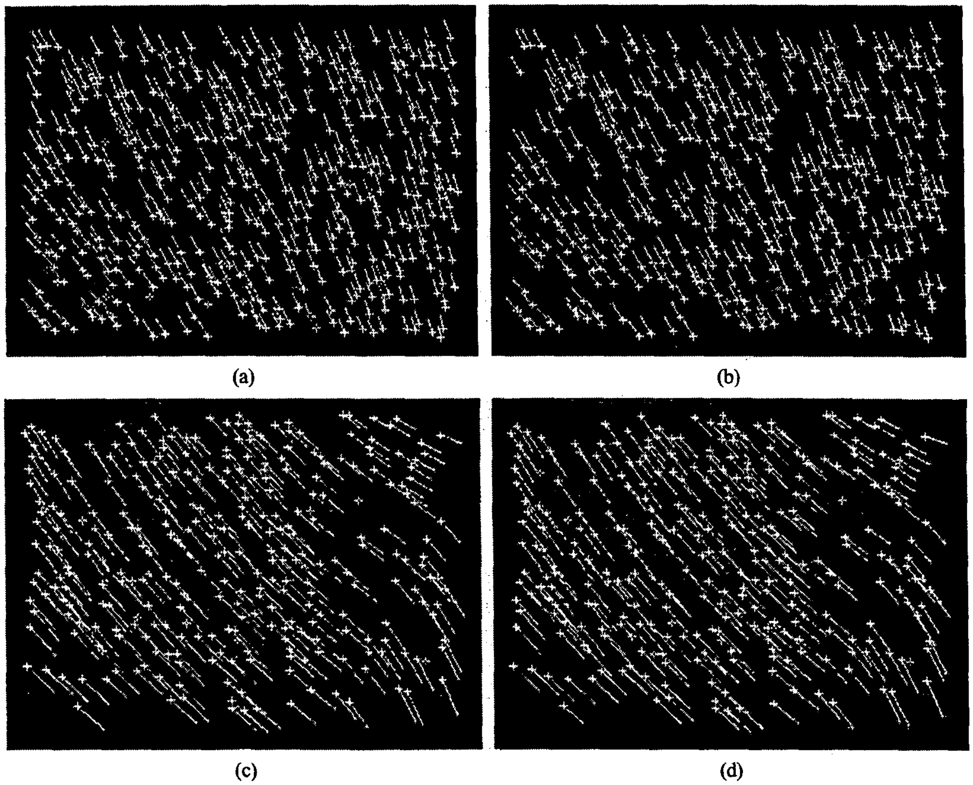

Figure 4: Comparison of the gray-level correlation results (left) with the use of textural analysis (right);
(a) vs. (b) 15 incorrect matches are found by the correlation, while texture analysis procedure detects 2 false matches.
(c) vs. (d) gray-level correlation finds 10 false matches for 8 of the texture operators.

In order to evaluate the goodness of the matching procedure we have used a two step approach. First, the interest points have been detected in the first image, and the region correlation has been applied to the second image. Next, a human operator has marked all the visually incorrect matches, in order to know which correspondences have been incorrectly established. This has been done for five different sequences, taking 10 pairs of images per sequence.

Figure 4 shows a sample of the automatic detection of matches by using the classical correlation procedure (left column), and the effect of applying the texture analysis to all the candidate matches. Two different situations are illustrated in this figure: fig. 4(a) and (b) present a scene containing rocks. These images exhibit considerable differences in depth along the image. In this case 434 interest point were detected by the algorithm. After the region correlation procedure 15 points were incorrectly matched. The texture analysis was then applied to all the candidate matches, producing 2 false matches. This high rate is due to the existence of clearly differentiated zones in the image with rather different textures.

Figures 4(c) and (d) show a rather planar floor of sand and algae, with an elevated rock on the bottom right side of the images. 329 points were correlated obtaining 10 false matches with the classical approach, for 8 wrong correspondences with the texture operators. The texture analysis corrected 5 matches that were incorrectly matched through correlation, but introduced 3 new false matches among the others.

From the total of 50 images that have been tested, in broad outline matching results resemble those of the images illustrated in figure 4. The amount of interest points ranged from 200 to 450 points, and a percentage of mismatched points similar to those presented above: ranging from 6 to 2% of false matches in the classic correlation; and from 4 to 0.5% for the texture analysis. Only in two cases textural analysis performed worst than classical correlation, but with a very small difference.

6 Conclusions

We have proposed in this paper a new method to improve image matching in underwater image sequences for estimating the motion of an underwater robot. The accuracy of the method is directly related to the exploitation of the local texture parameters in the image. Our approach has proved to perform better than the merely use of classical region correlation. However, in some situations, it may introduce some new false matches.

The construction of a visual map of the surveyed area has been proposed. Our approach does not suffer from the constraint of planar scene imposed by other methods, such as those based on homographies. However, although 3D estimation from correspondences works fine in irregular scenes with depth variation, it has serious limitations when all the matches are coplanar. A possible solution could be to use two alternative motion detection techniques. If we can detect when the points are close to a plane we could switch to the second methodology, and when depth changes appear the system should switch back. In order to know when to switch from one method to the other, a predicted image could be constructed (*e.g.* assuming coplanarity) to be comp ared with the real image in the next time instant.

On the other hand, the quality of the motion estimation could also take profit of a robust estimation technique (*i.e.* LMedS) in order to detect outliers among the correspondences.

Finally, it should be noticed that the increasing computational power of nowadays computers would allow URIS to construct real-time high-accuracy maps in the near future.

References

[1] X. Xu and S. Negahdaripour, "Vision-based motion sensing from underwater navigation and mosaicing of ocean floor images", in *Proc. of the MTS/IEEE OCEANS Conference*, vol.2, pp. 1412–1417, 1997.

[2] N. Gracias and J. Santos-Victor, "Underwater Video Mosaics as Visual Navigation Maps", *Computer Vision and Image Understanding*, vol. 79, no. 1, pp. 66–91, 2000.

[3] R. Marks, S. Rock, and M. Lee, "Real-time video mosaicking of the ocean floor", *IEEE Journal of Oceanic Engineering*, vol. 20, no. 3, pp. 229-241, 1995.

[4] C. J. Funk, S.B. Bryant, P.J. Beckman Jr., "Handbook of underwater imaging system design", Ocean Technology Department, Naval Undersea Center, 1972.

[5] D.M. Mount, N.S. Netanyahu, J. Le Moigne, "Efficient algorithms for robust feature matching", *Pattern Recognition*, no. 32, pp. 17–38, 1999.

[6] A. Giachetti, "Matching techniques to compute image motion", *Image and Vision Computing*, no. 18, pp. 247–260, 2000.

[7] R. Garcia, J. Batlle, X. Cufi, and J. Amat, "Positioning an Underwater Vehicle through Image Mosaicking", in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, vol. 3, pp. 2779–2784, 2001.

[8] K.I. Laws, "Textured Image Segmentation", Ph.D. *Thesis*, Processing Institute, University of Southern California, Los Angeles, 1980.

[9] R.M. Haralick, K. Shanguman, and I. Dinstein, "Textural Features for image classification", *IEEE Trans. on Systems, Man and Cybernetics*, vol. 3, pp. 610–621, 1973.

[10] T. Ojala, M. Pietikainen, and D. Harwood, "A comparative Study of Texture Measures with Classification Based on Feature Distribution", *Pattern Recognition*, vol. 29, pp. 51–59, 1996.

[11] X. Xu, "Vision-Based ROV System", *Ph.D. Thesis*, University of Miami, 2000.

[12] O.D. Faugeras, and G. Toscani, "The calibration problem for stereo". Proc. of the *IEEE Computer Vision and Pattern Recognition*, pp. 15-20, 1986.

[13] J. Canny, "A Computational Approach to Edge Detection", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986.

[14] H. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections", Nature, no. 293, pp. 133-135, 1981.

[15] O. Faugeras, "Three-dimensional computer vision: a geometric viewpoint", MIT Press, 1993.

[16] Z. Zhang, "A New Multistage Approach to Motion and Structure Estimation: From Essential Parameters to Euclidean Motion Via Fundamental Matrix", *INRIA Research Report n.* 2910, June 1996.

[17] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses", *IEEE Journal on Robotics and Automation*, vol. RA-3, pp. 323-344, August 1987.

[18] S. Tiwari, "Mosaicking of the Ocean Floor in the Presence of Three-Dimensional Occlusions in Visual and Side-Scan Sonar Images", n *Proceedings of the OES/IEEE Symposium on Autonomous Underwater Vehicle Technology*, pp. 308-314, June 1996.