A System to Evaluate the Accuracy of a Visual Mosaicking Methodology

R. Garcia, J. Batlle and X. Cufi

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

Abstract –**When underwater vehicles navigate close to the ocean floor, computer vision techniques can be applied to obtain motion estimates. A complete system to create visual mosaics of the seabed is described in this paper. Unfortunately, the accuracy of the constructed mosaic is difficult to evaluate. The use of a laboratory setup to obtain an accurate error measurement is proposed. The system consists on a robot arm carrying a downward looking camera. A pattern formed by a white background and a matrix of black dots uniformly distributed along the surveyed scene is used to find the exact image registration parameters. When the robot executes a trajectory (simulating the motion of a submersible), an image sequence is acquired by the camera. The estimated motion computed from the encoders of the robot is refined by detecting, to subpixel accuracy, the black dots of the image sequence, and computing the 2D projective transform which relates two consecutive images. The pattern is then substituted by a poster of the sea floor and the trajectory is executed again, acquiring the image sequence used to test the accuracy of the mosaicking system.**

I. INTRODUCTION

Over the past few years, visual mosaics have greatly advanced as a tool for undersea exploration and navigation of underwater vehicles. The position and orientation of the submersible can be calculated by integrating the apparent motion of the images which form the mosaic. Several strategies have been presented in the literature to recover the vehicle motion by means of visual mosaics, *e.g.* [1,2,3]. These visual positioning systems allow the vehicle to localize itself on the mosaic map as it is being constructed (known as Concurrent Mapping and Localization). Once the map has been constructed, the mosaic can be used to plan the path of the vehicle during the execution of the mission. These visual sensors are gradually substituting other positioning sensing technologies, such as acoustic transponder networks [4], for some specific tasks. While these sonic beacon systems require the vehicle to move within the area they are covering, visual mosaicking does not restrict the autonomous capabilities of the submersible to a limited area. However, mosaicking systems can only be used when the vehicle is performing tasks near the ocean floor and require a reasonable visibility in the working area. In other respects, vision systems are much less costly when compared to setting and calibrating a network of sonar-based transponders covering the site of interest.

Unfortunately, the accuracy of the constructed mosaic is difficult to evaluate. A possible option is to measure the correctness of the mosaic by comparing its estimations with the information provided by other on-board sensors [5]. When the experiments are performed in indoor water tanks, an overhead camera can take absolute position measurements which are compared with those provided by the mosaicking system [6]. This second approach can only be applied when the vehicle moves on the surface. In both cases, the estimation of the true trajectory is subject to small biases. We propose the use of a laboratory setup to reduce this bias, obtaining an accurate error measurement.

This paper is structured as follows: section II describes the algorithm we are using to construct underwater mosaics. Section III details the laboratory setup used to evaluate the mosaicking system. Next, some of the experiments that have been performed to quantify the errors across the mosaic are shown in section IV. Finally, section V provides a brief conclusion and outlines future work.

II. THE MOSAICKING SYSTEM

The creation of the mosaic is accomplished in the following stages: First, a correction of lens distortion is performed. A detector of *interest points* then selects the most reliable features of the undistorted image and the correspondences of these features are matched in the next image of the sequence. Next, the system identifies the points which describe the dominant motion of the image by means of a robust outlier-detection algorithm. Once the pairs of features describing the dominant motion have been selected, a 2D projective transformation matrix relating the coordinates of both images is computed. Finally, the registered images are merged onto a composite mosaic image [7].

A. Correction of Lens Distortion

Correcting the distortion produced by the camera lenses and the ray diffraction at the water-camera housing and the air-camera housing interfaces requires the estimation of a number of intrinsic camera parameters [8]. A simplification of the Faugeras-Toscani algorithm has been implemented to correct uniquely radial distortion, instead of performing full camera calibration [9]:

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

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

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 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.

B. Selection of Interest Points

The next step of the mosaicking algorithm consists of the selection of adequate interest points in the present image to be matched in the next frame. These candidate features should be adequate in the sense of being easy to match in the next image. Therefore, the selection of robust interest points depends, to a large extent, on the technique used to detect correspondences. Normally, small windows containing high frequencies are quite adequate since they are located in the border of different image textures. For this reason, our interest point detector searches for small zones presenting high spatial gradient information in more than one direction. To do this, the image is convolved with two directional high-pass filters (in the x and y directions). The areas with the highest value in both directions are selected. In fact, this strategy is quite similar to that followed by some corner detectors [10,11] or feature trackers [12]. When a feature is selected, the algorithm goes on to search for any other selected features in its neighborhood. If a higher-valued feature exists in this neighborhood, only the best feature is selected as an interest point. This avoids the selection of other features in the same neighborhood and ensures a reasonable distribution of the interest points within the image.

C. Region Matching and Texture Characterization

Finding correspondences between images is not an easy task in computer vision, and even less in underwater imaging. On that account we pay special attention to the matching process, carrying out a two step approach. First, a block-matching strategy is applied to the gray-level images [13], selecting a set of candidate matches for a given interest point. Then a texture characterization of the points is used for selecting the best correspondence. For every interest point in the present image I, a correlation score is computed in the next image I'. This is performed by comparing a small $n \times n$ window centered at the interest point $\mathbf{m} = [x, y]$ with all the possible locations of the feature $\mathbf{m}' = [x', y']$ in the next image, as shown in (4). These possible locations of the feature $\mathbf{m}'$ are limited to a window of I', centered at the coordinates of $\mathbf{m}$ in the first image. The size of this window depends on the motion between consecutive images.

$$corr(\mathbf{m}, \mathbf{m}') = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left[I(x+i, y+j) - \overline{I(x,y)}\right]\cdot\left[I'(x'+i, y'+j) - \overline{I'(x',y')}\right]}{n^2\sqrt{\sigma^2(I)\cdot\sigma^2(I')}} \qquad (4)$$

where $\overline{I(x,y)}$ is the average of the gray-levels in the $n \times n$ neighborhood, and $\sigma^2(I)$ is the standard deviation of the image I in the $n \times n$ window centered at the interest point $\mathbf{m}$, which is given by:

$$\sigma^2(I) = +\sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left[I(x,y)\right]^2}{n^2} - \overline{I(x,y)}^2} \qquad (5)$$

In this way, it is possible to find in I' a set of possible correspondences $\mathbf{m}'_i$ of every interest point in I. To decide which of these matches is the right one, the textural characteristics of these areas of image I are used as a matching vector to be correlated with the selected matches of the next image I'. An extensive study has been carried out in order to compare different texture operators. We have implemented and tested different configurations of some statistical-based texture operators, *i.e.* co-occurrence matrix [14], energy filters [15], local binary patterns [16], contrast features [16] and Markov random field models

[17], among others. Two texture parameters have been selected because of their excellent performance in helping to solve the correspondence problem, as well as their fast computation times: *Energy filters* [15] and *Contrast features* [16,18].

Texture energy filters consist of pre-filtering the image with a set of 3×3 and 5×5 masks, and then computing a series of statistical measures for every mask (in our case standard deviation and positive/negative mean):

$$\sigma = +\sqrt{\sum_{i=1}^{n}\frac{(c_i-\mu)^2}{n}}, \text{ with } \mu = \frac{\sum_{i=1}^{n} c_i}{n} \quad (6)$$

$$\text{positive mean} = \mu^+ = \frac{\sum_{i=1}^{n} c_i}{n}, \text{ with } c_i >= 0 \quad (7)$$

$$\text{negative mean} = \mu^- = \frac{\sum_{i=1}^{n} c_i}{n}, \text{ with } c_i < 0 \quad (8)$$

where n is the size of the vector which stores the neighboring pixels, and c_i is the ith element of this vector.

We have extended the definition of *contrast feature* given by Ojala and Pietikäinen [18]. The authors proposed the use of a 3×3 neighborhood to be used jointly with *Local Binary Patterns*. We also consider the 5×5 and 7×7 neighborhoods of the selected point as the region to analyze. The contrast operator consists of performing a gray-scale differentiation in the region which is being considered. The neighboring pixels are compared with the selected point, computing the average of those neighbors with a gray-value higher than that of the center pixel. A second average is computed with the neighbors with an intensity value below the selected pixel. Then, the difference of both averages is computed. This value is known as **contrast** of the texture. Different values of contrast are obtained depending on the size of the selected neighborhood.

Both texture operators result in a vector of texture values characterizing every interest point of the present image *I*, namely:

- Energy L3L3 Standard Deviation 3×3
- Energy E3E3 Positive Average 3×3
- Energy E3E3 Negative Average 3×3
- Energy L5S5 Positive Average 3×3
- Energy E5L5 Standard Deviation 3×3
- Energy E5S5 Negative Average 3×3
- Contrast 3×3
- Contrast 5×5
- Contrast 7×7

For a more detailed description of how these texture operators are derived see [15,16]. Once this vector of 9 parameters has been computed for a given interest point of the first image *I*, it is then computed for every candidate match in I'. After a process of normalization, the texture vector of the interest point is compared with the textural properties of all the possible matches by means of the weighted Euclidean distance. A texture similarity measure is then obtained for every possible correspondence.

After this process, every candidate match has two measures of similarity: (i) a block-matching correlation score obtained through (4); and (ii) a texture score produced by feature characterization. By averaging these two values, the best correspondence is selected.

Once this procedure has been accomplished, for every interest point in image *I* a unique match is obtained in image I'.

D. *Estimating the Dominant Motion through Outlier Rejection*

After the correspondences have been solved, a set of displacement vectors relating the features of two images of the sequence is obtained. Every vector relates the coordinates of the same feature in both images. Our aim is now to recover the apparent motion of the camera from these features. This can be done by computing a 2D transformation matrix **H** which relates the coordinates of a feature in a frame with its coordinates in the previous one:

$$\tilde{\mathbf{m}} = \mathbf{H}\cdot\tilde{\mathbf{m}}' \text{ or } \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \quad (9)$$

where $\tilde{\mathbf{m}} = (x_i, y_i, 1)^T$ and $\tilde{\mathbf{m}}' = (x_i', y_i', 1)^T$ denote a correspondence point in two consecutive images; the symbol ~ indicates that the points are expressed in homogeneous coordinates, and ≅ expresses equality up to scale. The matrix that performs this transformation is known as "homography" [19], and can be computed by SVD if 4 or more pairs of matchings are available, as shown in (10).

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1\cdot x_1' & -y_1\cdot x_1' & -x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1\cdot y_1' & -y_1\cdot y_1' & -y_1' \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_n\cdot x_n' & -y_n\cdot x_n' & -x_n' \\ 0 & 0 & 1 & x_n & y_n & 1 & -x_n\cdot y_n' & -y_n\cdot y_n' & -y_n' \end{bmatrix} \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0 \quad (10)$$

Although an accurate texture analysis is devoted to the matching procedure, some false matches (known as *outliers*) could still appear among the right correspondences. For this reason, a robust estimation method has to be applied. The *Least Median of Squares* (LMedS) algorithm can be used for finding the matrix **H** which minimises the median of the squared residuals M_{err} :

$$M_{err} = \underset{j}{med}\left(d^2\left(\tilde{\mathbf{m}}_j, \mathbf{H}\tilde{\mathbf{m}}'_j\right)\right) + \left(d^2\left(\tilde{\mathbf{m}}'_j, \mathbf{H}^{-1}\tilde{\mathbf{m}}_j\right)\right) \quad (11)$$

where $\tilde{\mathbf{m}} = (x_1, x_2, x_3)$ are the homogeneous coordinates of a 2D point $\mathbf{m}$ defined in the image plane *I*, being $\mathbf{m} = (x_i, y_i) = (x_1/x_3, x_2/x_3)$ its corresponding Cartesian coordinates; and $d^2\left(\tilde{\mathbf{m}}_j, \mathbf{H}\tilde{\mathbf{m}}'_j\right)$ is the square distance from a point $\tilde{\mathbf{m}}_j$, defined on image *I*, to the projection on the same image plane of its correspondence $\tilde{\mathbf{m}}'_j$. Hence, the error is defined by the distance of a point to the projection of its correspondence [20].

The LMedS algorithm works as follows: given the regression problem of computing the matrix **H** from a set of data points, compute a candidate solution based on a randomly chosen 4-tuple from the data. Then, estimate the fit of this solution to all the data, defined as the median of the squared residuals.

Once the best solution has been found, a minimal median is obtained. As from the median, the *mean* and the *standard deviation* can be computed (see [20] for details). Therefore, in our implementation, those points at a distance larger than the median are eliminated, and matrix **H** is recomputed with the remaining points.

E. Mosaic Construction

The process of mosaic construction is described below. First, the initial image in the sequence is selected as a base frame. The mosaic coordinate system is placed at the origin of this reference frame. Then, when image (*k*+1) has to be added to the mosaic, a 2D planar transformation ${}^{k}\mathbf{H}_{k+1}$ provides its best fitting with respect to the previous image. In order to obtain a global registration from image (*k*+1) to the mosaic reference frame, the following matrix product has to be performed [21]:

$${}^{1}\mathbf{H}_{k+1} = \prod_{i=1..k} {}^{i}\mathbf{H}_{i+1} \quad (12)$$

where ${}^{1}\mathbf{H}_{k+1}$ is the homography that produces the coordinates of a point in the mosaic image, from the coordinates of the same point in image (*k*+1).

For every image added to the common frame, the mosaic can be updated according to four different strategies: (a) *first in*; (b) *last in*; (c) temporal mean; or (d) temporal median. Depending on the strategy followed the mosaic is only updated within the regions in which no information existed before (a); or every new image is completely added to the mosaic (b); or a temporal filter is applied to render the mosaic image on the overlapping regions (c) and (d). The temporal filters are very useful to remove transient data from the mosaic, keeping only the background.

III. EXPERIMENTAL SETUP

As the mosaic increases in size, small errors in the estimation of dominant motion between consecutive frames provoke an accumulated error. It is possible to reduce this error by periodically registering the current frame with the mosaic image. In this work we want to be able to evaluate the nature of error propagation in the resulting mosaic quantitatively. A laboratory setup to obtain an accurate error measurement is proposed. The system consists on a robot arm carrying a down-looking camera (see Fig. 1). This robot has limited accuracy, but good repeatability. The accuracy parameter is defined as the distance between an arbitrarily prescribed location and the one that has actually been achieved, while the repeatability is measured as the radius of the sphere which contains the points reached after positioning the tool in the same place repeatedly [22].

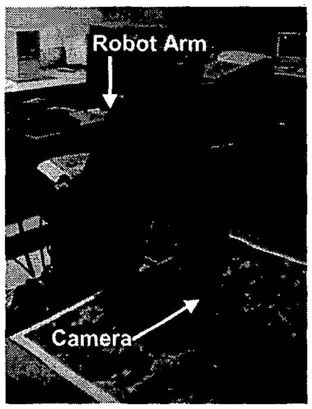


Fig. 1. Experimental setup. A robot arm carries a down-looking camera and takes images of a poster simulating the sea floor.

The robot arm is required to execute the same pre-defined trajectory twice. A calibration pattern formed by a white background and a matrix of black dots uniformly distributed along the surveyed scene is initially placed under the robot, covering the working area. It will be used to detect the exact image registration parameters. When the robot executes a trajectory (simulating the motion of a submersible), an image sequence is acquired by the camera. The radial distortion produced by the lenses is corrected for the whole sequence. The first image of the sequence is warped to a reference frame, which represents the ideal transform of the calibration pattern onto a virtual mosaic frame, *i.e.*, the distance in pixels from any black dot to one of its horizontal and vertical neighbors is always constant. This initial 2D transformation from the first image to the reference frame would be the identity matrix if the initial position of the camera had the image plane perfectly parallel to the scene.

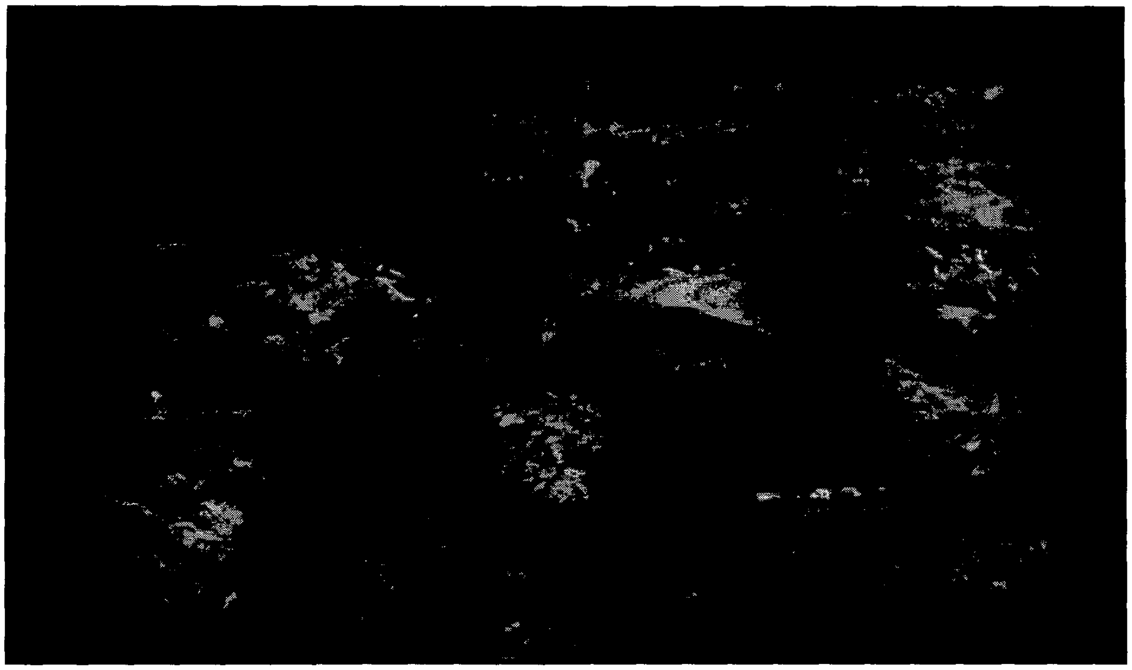


Fig. 2. Mosaic created from a sequence of 208 images. The size of the mosaic is 2700 × 1600 pixels. The individual images are 384 × 288 pixels

The estimated motion computed from the encoders of the robot serves as an initial estimate of the actual motion. Then, this estimate is refined by automatically detecting, to subpixel accuracy, the black dots of the calibration pattern in the image sequence. An initial estimate of the position of the black dots in the image is predicted from the information provided by the robot. When these dots are detected in the image the error can be corrected. From the new position of the calibration dots, a 2D projective transform which relates every pixel to the virtual mosaic image is computed. Next, the pattern is substituted by a poster of the sea floor and the trajectory is executed again. This second time, the acquired image sequence is used to test the accuracy of the mosaicking system.

IV. RESULTS

We have performed several experiments to compare the accuracy of our mosaicking algorithm against the real values. Normally, as the mosaic increases in size, drift error is expected to increase. We have performed several tests with different parameterization of the mosaicking system which have proved this fact, even though not always happens. Fig. 2 shows an example of one of the mosaics that has been automatically created, by means of the algorithm described in section II. In this case, the trajectory described by the camera starts at the lower part of the picture and moves up. If we analyze this mosaic, it appears as "visually correct" within the whole area, except a small misalignment between the first images and the first time the camera crosses an already visited zone, in the lower part of the image. In the second loop, when the camera passes through its way for a second time, no misalignments are visible. It is not possible to quantify the distortion of the mosaic in any other area of the image by means of a visual inspection. However, if we plot this path against the real one, other drift errors can be detected, as shown in Fig. 3.

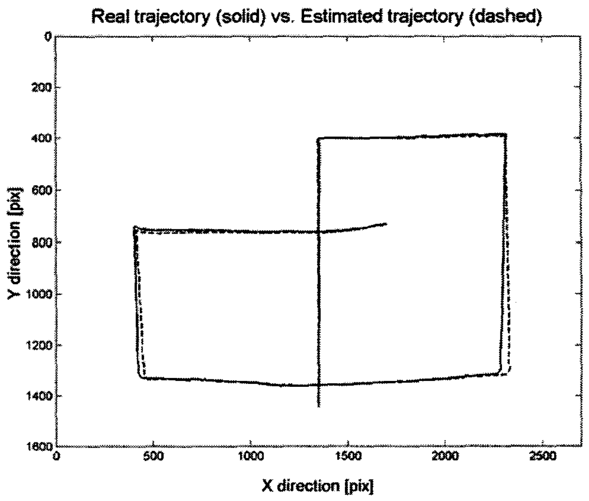


Fig. 3. Results of experiments for mosaic construction. Estimated (dashed) and real (solid) trajectories followed by the robot arm.

The mosaicking algorithm has estimated this trajectory by computing the motion between every pair of consecutive images in the sequence. Fig. 2 represents the position of the pixel located at the center of the images. Since there exists a considerable overlap between every image and the next one, the same area is common to several images. In this case, an alternative would be the selection of a reference image, and the estimation of motion between this image and the next few frames. This second approach yields to a mosaic presenting less drift (because incremental errors increase more slowly), but usually with a worst visual appearance. On the contrary, the methodology that has been used to create the mosaic of Fig. 2 generates maps with a good visual appearance, but is less adequate to estimate the position of an underwater vehicle.

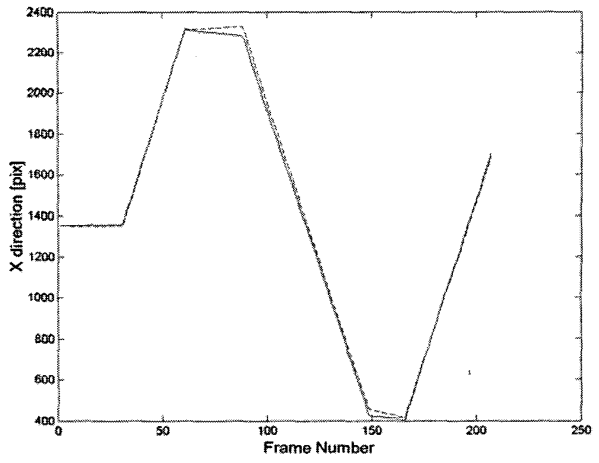
The lower right corner of Fig. 2 presents a drift of 46 pixels. If we want to use the mosaicking system to position an underwater vehicle, this drift would suppose an error in the order of 25 to 50 cm depending on the altitude of navigation of the submersible. This drift is kept approximately constant along the horizontal path in the lower part of the figure. This can be more easily observed in Fig. 4. Almost at the end of the path (in the second cross over), the drift has been reduced to 6 pixels. In this way, the visual analysis of the mosaic looks quite good, while considerable errors in the estimation of the trajectory have occurred.

V. CONCLUSIONS AND FUTURE WORK

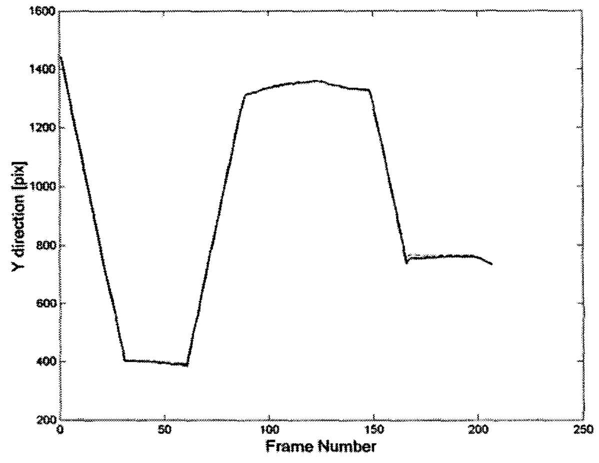
The construction of visual mosaics of the ocean floor can provide accurate position estimates for local navigation of underwater vehicles. We have presented an approach to quantify the distortion across visual mosaics. A solution to the problem of measuring error propagation in the construction of a mosaic has been proposed. This solution is valid for laboratory testing only, but it has also proved to be helpful in testing and tuning the different parameters of our mosaicking system,· *e.g.* selection of texture operators, number of interest points, comparative of different motion models, etc.

The construction of a mosaic for robot navigation suffers from drift errors, which are not distributed uniformly across the mosaic, but rather appear in certain areas. This is probably due to the lack of adequate information in the images acquired on these areas to produce good motion estimates.

Our approach to construct underwater mosaics has been validated by means of the experimental set-up. Further experiments will be carried out to enhance the performance of the mosaicking algorithm. In this work we have only dealt with planar scenes. In the future, perspective projection of non-planar objects should be studied in the construction of mosaics. The use of this validation tool will suppose a step forward towards the creation of a robust mosaicking methodology to be applied in real missions.



(a)



(b)

Fig. 4. Temporal evolution of the estimated (dashed) and real (solid) trajectories for the X and Y coordinates, respectively (a) and (b).

REFERENCES

[1] 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.

[2] S. Negahdaripour, X. Xu and A. Khamene, "Applications of direct 3D motion estimation for underwater machine vision systems," in *Proc. MTS/IEEE OCEANS*, vol. 1, pp. 51–55, 1998.

[3] N. Gracias and J. Santos-Victor, "Automatic mosaic creation of the ocean floor," in *Proc. MTS/IEEE OCEANS*, vol. 1, pp. 257–262, 1998.

[4] E.M. Geyer, P.M. Creamer, J.A. D'Appolito and R.G. Gains, "Characteristics and capabilities of navigation systems for unmanned untethered submersibles," in *Proc. Intl. Symp. on Unmanned Untethered Submersible Technology*, pp. 320-327, 1987.

[5] H. Singh, J. Howland, D. Yoerger and L. Whitcomb, "Quantitative photomosaicing of underwater imaging," in *Proc. MTS/IEEE OCEANS*, vol. 1, pp. 263–266, September 1998.

[6] S. Negahdaripour, X. Xu, A. Khamene and Z. Awan, "3D motion and depth estimation from sea-floor images for mosaic-based station-keeping and navigation of ROVs/AUVs and high-resolution sea-floor mapping," in *Proc. IEEE Workshop on Autonomous Underwater Robots*, pp. 191–200, 1998.

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

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

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

[10] C.G. Harris and M.J. Stephens, "A combined corner and edge detector," in Proceedings of the Fourth Alvey Vision Conference, Manchester, pp. 147–151, 1988.

[11] L. Kitchen and A. Rosenfeld, "Gray-Level corner detection," *Pattern Recognition Letters*, vol. 1, no. 2,pp. 95-102, 1982.

[12] J. Shi and C. Tomasi "Good features to track," in *Proc. of the IEEE Conf. on Computer Vision and Patt. Rec.*, pp. 593–600, 1994.

[13] Z. Zhang, R Deriche,. O. Faugeras, Q.T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," INRIA RR-2273, 1994.

[14] 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, November, 1973.

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

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

[17] B.S. Manjunath and R. Chellapa, "Unsupervised texture segmentation using Markov random field models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 478–482, 1991.

[18] T. Ojala and M. Pietikäinen, "Unsupervised texture segmentation using feature distributions," *Pattern Recognition*, vol. 32, pp. 477–486, 1996.

[19] J.G. Semple and G.T. Kneebone, "Algebraic projective geometry," Oxford University Press, 1952.

[20] P. Rousseeuw and A. Leroy, "Robust Regression and Outlier Detection," John Wiley & Sons, New York, 1987.

[21] 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.

[22] R.J. Schilling, *Fundamentals of Robotics: Analysis and Control*, Prentice-Hall International, 1990.