An Image Mosaicking Method based on the Integration of Grey Level and Textural Features

R. Garcia, X. Cufí, X. Muñoz, L. Pacheco, J. Batlle
Computer Vision and Robotics Group
IIiA, University of Girona, E.P.S. (P-II)
17003 Girona, Spain
e-mail: {rafa,xcuf,xmunoz,jbatlle}@eia.udg.es

Abstract

This paper presents a feature-based mosaicking system that exploits the characteristics of image texture. The creation of the mosaic is accomplished in four stages: first, a corner detector is used to localise prominent features in the image. Next, a similarity measure is performed against the next image of the sequence, but it is only applied on the zones selected by the corner detector. This measure is carried out in two ways: a grey-level block-matching strategy obtains a first similarity measure, and a texture-based correlation procures a second measure. Both measures are merged on a reliability factor that helps in the selection of the correct correspondence. Then, a variant of the LMedS robust estimator is used to eliminate the false matches. Finally, the image registration parameters are computed. Experimental results on real images have been tested on a lab sequence, showing the effectiveness of the proposed method.

Keywords: mosaic construction, texture extraction, planar transformations.

1 Introduction

A mosaic is a large composite image which results of the alignment of several images of a video sequence. Image mosaics can be used for many different applications [1], such as construction of visual maps for underwater robots navigation [2,3], recovery of camera and object motions, image stabilisation, structure from motion, map construction from aerial or satellite photographs, video compression, etc. Two main approaches have been exploited in the literature to construct image mosaics: feature-less and feature-based methods. In both cases the aim is to find a set of 2D planar transformations *registering* every input image with the coordinates system of the mosaic. This transformation, also known as homography [1], can be expressed as a matrix multiplication in homogeneous coordinates:

$$\tilde{\mathbf{x}}_i^{'} = \mathbf{H} \cdot \tilde{\mathbf{x}}_i \quad \text{or} \quad \begin{bmatrix} x_i^{'} \\ y_i^{'} \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{1}$$

where $\tilde{\mathbf{x}}' = (\mathrm{x}_1^{'}, \mathrm{x}_2^{'}, \mathrm{x}_3^{'})$ are the homogeneous coordinates of a 2D point $\mathbf{x}'$ defined in the present image (*I'*), being $\mathbf{x}' = (u', v') = (\mathrm{x}_1^{'}/\mathrm{x}_3^{'}, \mathrm{x}_2^{'}/\mathrm{x}_3^{'})$ its corresponding Cartesian coordinates; and $\tilde{\mathbf{x}} = (\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3)$ are the homogeneous coordinates of the same point defined in the mosaic reference frame[1]. The symbol $\cong$ indicates equality up to scale, and $h_{11}, h_{12}, ..., h_{32}$ are 8 parameters that determine the 2D projective transform. The image *registration* process consists in finding this 8-parameter set. In the case of the *feature-less* methods, it is achieved through an iterative process by minimising the sum of the squared intensity errors over all corresponding pairs of pixels which are present in both images [4]. A commonly used method to solve this iterative non-linear minimisation is the Levenberg-Marquardt algorithm [5]. Although feature-less methods are quite accurate, they suffer from the slowness of computation, need good initial values for a good convergence, and can get stuck at local minima.

For all the reasons mentioned above we have chosen a *feature-based* approach. Equation (1) is computed from the set of features detected in the image *I'* and their matchings in the previous image. If more than 4 feature/matching pairs are available the system can be solved by a least squares technique. Once the best transformation **H** has been found, the new image can be warped into the reference frame of the mosaic.

The flow of processing of our system is the following: first, prominent features are detected in the input image, and matched with the next one (section 2). Then, a strategy to reject false matches is described (section 3), providing a reliable list of points and their correspondences to compute the registration parameters (section 4).

2 Feature detection

The searching for feature correspondences is performed through a two-step approach. First, the zones of the two consecutive images presenting high spatial gradient information are detected by means of a corner detector [6]. Next, two correlation measures are obtained: (a) a grey-level block-matching strategy is applied to find in *I'* a set of possible matches of every corner in *I*; and (b) the textural characteristics of these areas of image *I* are used as a matching vector to be correlated with the high gradient zones of the next image *I'*. On the one hand, the *mean square error* (MSE) is used as a similarity measure to evaluate the block-matching goodness [7]. On the other hand, the texture parameters

---

[1] Some works find the transformation from the present image (I') to the previous one (I), and then relate the frame of the mosaic with I' through the inverse of the product of the homographies relating every 2 images of the sequence.

that are taken into consideration are: *Coocurrence matrix* [8], *Energy filters* [9], and *Local Binary Patterns* [10]. We should take into account that the first two methods can generate several measurements, depending on the number of orientation angles, the distance of correlation and the size of the neighbourhood. In our application we chose 8 different angles for the coocurrence matrix, taking only distances of 1-pixel; and 9 masks of the energy filter taking only a 3x3 neighbourhood. From our experience, the use of larger neighbourhoods provides little improvement at the expense of a higher computational cost. Thereupon, the texture vector contains 18-positions, namely: 8 measurements of coocurrence matrix, 9 measures of energy, and 1 of local binary patterns. This texture vector is mapped onto an 18-dimentional space, where it is compared with the texture vector of the original point in the previous image. The Euclidean distance is then computed, obtaining a texture similarity measure for every possible correspondence.

After this process, a set of correspondences in image *I'* are obtained from every corner of image *I,* and every match has two measures of similarity: block-matching and texture. By averaging these two values, the reliability (r) of every match is obtained. Taking into account the reliability value, we have devised a method (the *disambiguate algorithm*) to correctly choose the right correspondence among the whole set of matchings. Suppose that we want to find the best correspondence of the j^{th} corner $\mathbf{m}_j$, and that it has *q* possible matches in *I'*, the disambiguate algorithm can be formulated as shown in figure 1.

1. Compute the centroid of the set of matches, weighting every coordinate depending on the reliability measure (r_{jx}):

$$c_{xy} = \left(\bar{x}'_j, \bar{y}'_j\right) = \left(\frac{\sum_{i=1}^{q} x'_{ji} \cdot r'_{ji}}{q}, \frac{\sum_{i=1}^{q} y'_{ji} \cdot r'_{ji}}{q} \right)$$

2. From the centroid $\mathbf{c}_{xy}$, fix a radius of size $\boldsymbol{R}$, and eliminate those matches that are out of the circle defined by $\boldsymbol{R}$.
3. Select the matching with the highest reliability as the correct correspondence.
4. If no matching appears inside the circle, then eliminate the corner.

Figure 1: Scheme of the *disambiguate algorithm* to be applied to every corner of *I*.

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

3 Outlier rejection

After the correspondences have been found, 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. 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 aims at finding the matrix **H** which minimises the median of the squared errors. The minimisation is performed by searching in the parameter space, and the error is defined by the distance of a point to the projection of its correspondence. A brief description of the LMedS algorithm is given below, but a more detailed description of this method can be found in [11].

The principle of our implementation of the LMedS is the following: given the regression problem of computing the matrix **H** from a set of data points, where 4 is the minimum number of data points which determine a solution, 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. The median of the squared residuals M_{err} is defined by:

$$M_{err} = \underset{j}{med}\left(d^2\left(\tilde{\mathbf{m}}_j, \mathbf{H}\tilde{\mathbf{m}}_j^{'}\right)\right) + \left(d^2\left(\tilde{\mathbf{m}}_j^{'}, \mathbf{H}^{-1}\tilde{\mathbf{m}}_j\right)\right) \tag{3}$$

where $\tilde{\mathbf{m}} = (\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3)$ are the homogeneous coordinates of a 2D point $\mathbf{m}$ defined in the image plane *I*, being $\mathbf{m} = (x_i, y_i) = (\mathrm{x}_1/\mathrm{x}_3, \mathrm{x}_2/\mathrm{x}_3)$ 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 $\tilde{\mathbf{m}}$, defined on image *I*, to the projection on the same image plane of its correspondence $\tilde{\mathbf{m}}^{'}$.

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 [11] 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.

4 Feature-based registration

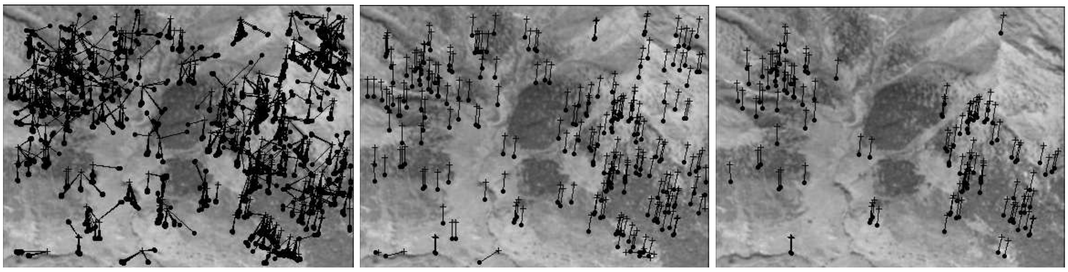
Once a robust list of correspondences is available, the first image in the sequence is selected as a base frame. The mosaic coordinate system is placed at the origin of this reference frame. When a new image has to be added to the mosaic, a 2D planar transformation **H** provides its best fitting with respect to the previous image (or to the mosaic image). In section 1, the most general projective transformation was introduced. However, if the sort of camera motion is known beforehand, the projective model may contain more free parameters than necessary. Therefore, a more adequate motion model can be chosen. The simplest transformation is pure translation, followed by translation and rotation (rigid or euclidean model), and introducing scaling (similarity model). More complex transformations are obtained with the affine and projective models (translation, rotation, scaling, and shear or perspective deformation, respectively).

After the motion model has been chosen, every new image is added to the mosaic, through the mapping provided by matrix **H**. For every image, 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 followed strategy the mosaic is only updated within the regions where 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.

5 Experimental results

The experimental mosaics presented in this paper were created in the lab taking a sequence of images of a cartographic map. The motion of the camera simulated an underwater vehicle mapping the ocean floor, which is one of the potential applications of mosaicking in undersea robotics [12,13].

Figure 2 shows the different phases accomplished by the mosaicking system. The crosses represent the position of every corner in the first image, while the possible matchings in the next image are represented with a small circle. The obtained corners of *I* and the possible matchings in *I'* are shown in Figure 2a.



(a) (b) (c)

Figure 2: (a) corners detected in image *I* and possible matches in *I'*; the corners are represented with the cross, and the matchings are drawn with a small circle; (b) result of the *disambiguate* algorithm; (c) result of LMedS, eliminating the points having a distance to their epipolar line larger than the median distance.

Figure 2b shows how the *disambiguate* algorithm described above eliminates most of the incorrect matches, leaving only one matching for every corner. This figure exhibits how the algorithm is not able to choose the correct matches in a small number of cases. However, it reduces the computational burden of the subsequent steps by providing a single match for every corner. The result of our particular implementation of LMedS is demonstrated in Figure 2c. This algorithm is able to eliminate all the incorrect matches, although some of the correct ones are also deleted. A mosaic example is illustrated on Figure 3. The most recent information (or "last in" strategy) has been used.

6 Conclusions and Further Work

In this paper we have presented a mosaicking algorithm that is able to exploit the advantages of feature-based block-matching and texture analysis for solving the correspondence problem. The viability of the system has been demonstrated with real world experiments. Textures can highly improve the matching process, and robust estimation techniques further ameliorate the dominant motion estimation. However, further research is needed to determine which other texture parameters could also fit the matching process. In the future we plan to use this technique to create visual map for positioning an underwater robot during a mission.

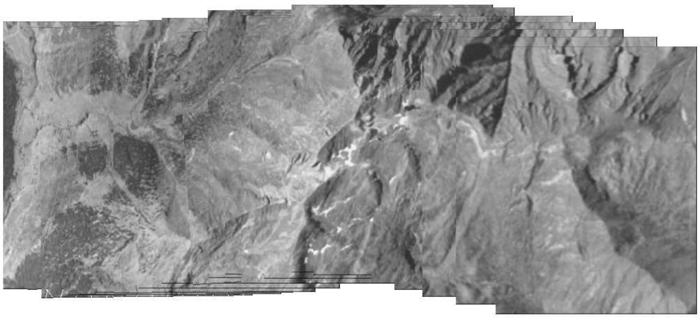


Figure 3: Mosaic example using the rigid transformation model

References

[1] R. Szeliski, “Image mosaicing for tele-reality applications”, in Proceedings of the IEEE Workshop on Applications of Computer Vision, pp. 44–53, 1994.

[2] R.L. Marks, S.M. Rock, and M.J. Lee, “Real-time video mosaicking of the Ocean floor”, IEEE Journal of Oceanic Engineering, vol. 20, no.3, pp. 229–241, 1995.

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

[4] R. Szeliski and S. B. Kang, “Direct methods for visual scene reconstruction”, in *Proceedings of the IEEE Workshop on Representations of Visual Scenes*, pp. 26-33, Cambridge, Massachusetts, June 1995.

[5] W. H. Press, B.P. Flannery, S. A. Teukolsky, and W. T. Vetterling, “Numerical Recipes in C: the art of scientific computing”, Cambridge University Press, Second Edition, 1992.

[6] L. Kitchen, A. Rosenfeld, “Gray-Level corner detection”, Pattern Recognition Letters, vol. 1, no. 2,pp. 95-102, 1982.

[7] R. Klette, K. Schüns, and A. Koschan, “Computer Vision, Three-dimensional Data from Images”, Springer-Verlag, 1998.

[8] R.M. Haralick, K. Shanguman, and I. Dinstein, "Textural Features for image classification", IEEE Transactions on Systems, Man and Cybernetics, vol. 3, pp. 610-621, November, 1973.

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

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

[12] R. Haywood, “Acquisition of a micro scale photographic survey using an autonomous submersible”, in Proceedings of the OCEANS Conference, vol. 5, pp. 1423–1426, 1986

[13] N. Gracias and J. Santos-Victor “Automatic mosaic creation of the ocean floor”, in Proceedings of the OCEANS Conference, vol. 1, pp. 257–262, 1998.