THE USE OF GEOMETRIC HASHING FOR AUTOMATIC IMAGE MOSAICING

Udhav Bhosle, Subhasis Chaudhuri and Sumantra Dutta Roy

Department of Electrical Engineering
Indian Institute of Technology Bombay
Powai, Mumbai - 400 076.
{udhav, sc, sumantra}@ee.iitb.ac.in

ABSTRACT

A camera typically has a very limited field of view. Image mosaicing involves stitching together images taken at different camera viewpoints, in order to have a wider field of view. Thus, automation of the above process is an important issue. This paper proposes a new method for automatic generation of mosaics, using Geometric Hashing. This speeds up the matching process in addition to automating it. We show the application of our method on two important cases namely, one of rigid planar camera motion, and panoramic mosaics. We provide experimental results in support of our proposed method.

1. INTRODUCTION

A camera typically has a limited field of view. A lens with a wide field of view (such as a fish-eye lens) incurs substantial distortion. In addition, capturing the entire scene with the limited camera resolution compromises the image quality. Hardware-based methods (*e.g.*, quick time VR, Surround Video) impose a strong limitation on the imaging conditions. Image mosaicing algorithms register or stitch a sequence of images into a composite image [1, 2, 3].

Image mosaicing involves the following:

1. *Image alignment:* One has to determine the transformation that aligns images to be combined into a mosaic. Registration or alignment methods can be loosely divided into following classes - algorithms that use the pixel values directly *i.e.*, correlation method; algorithms that use frequency domain method *i.e.*, fast Fourier transform; algorithms that use low level features such as corners or edge *i.e.*, feature based algorithms and algorithms that use relation between features *i.e.*, graph theoretical methods [4, 5].

2. *Image cut and paste:* Most regions in a mosaic are overlapping and are covered by more than one images. There are two ways to determine the region. (a) Combining the aligned images by a suitable function such as median, average, etc, and (b) Selecting a region from one of the images. Method (a) requires an accurate alignment over entire image area, otherwise the resulting mosaic will be blurred. The method (b) requires alignment only along the seams. This is more useful in cases where camera motion, scene geometry and imaging conditions are challenging [6].

3. *Image blending:* It is used to overcome intensity difference between the images, differences that are present even when images are perfectly aligned. These are created by dynamically changing camera gain [1].

2. GEOMETRIC HASHING

Image alignment requires matching M points in one image with N points in another. As such, this process has an exponential time complexity, $O(M^N)$. Lamdan *et al.* [7] propose Geometric

Hashing as a fast method for 2-D object recognition, where M object points are to be matched to N image points, *restricted to an affine framework*. We generalize this idea for image alignment (the first step in image mosaicing), according to the specific transformation between two images – Euclidean, Affine, or the most general Projective case.

A 2-D transformation requires K basis points ($K = 3$ for Euclidean and Affine, 4 for Projective). We can select ordered pairs of K basis points from the first image in $\binom{M}{K} \times K!$ ways (this is $\mathcal{O}(M^K)$). For each such basis, we compute the coordinates of the remaining $M - K$ ($\mathcal{O}(M)$) points. A *hash table* stores these coordinates, indexed by the basis points. We repeat the process for the second image. Matching rows of coordinates between hash tables of the two images has *quadratic* time complexity. We can reduce this to *linear* is we sort each row in the hash tables. Hence, the problem of matching image features reduces to $\mathcal{O}(M^{K+1}N^{K+1}) \times$ the row matching time. This is has polynomial time complexity, an improvement over the exponential time complexity required for a naive feature match. *It is important to note that the relative change of successive camera positions is often kept small to maximize the number of corresponding points between images.* We show the application of Geometric Hashing to two important cases of mosaicing. In each case, we use the above idea to further reduce the time complexity of image alignment.

3. MOSAICS FOR PLANAR RIGID CAMERA MOTION

Two camera positions are related by a 3-D Euclidean (rigid-body) transformation:

$$\mathbf{P}' = \mathcal{R}\mathbf{P} + \mathcal{T} \tag{1}$$

Here, $\mathbf{P} = [X\ Y\ Z]^T$ and $\mathbf{P}' = [X'\ Y'\ Z']^T$ represent the (non-homogeneous) 3-D coordinates of a point viewed by the two camera stations, and $\mathbf{p} = [x\ y\ 1]^T$ and $\mathbf{p}' = [x'\ y'\ 1]^T$ are the corresponding image points. For a planar rigid transformation (say in the XY-plane), $\mathcal{T} = [T_x\ T_y\ 0]^T$ and

$$\mathcal{R} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The 2-D image points and 3-D points in the camera coordinate system are related by

$$\lambda\mathbf{p} = \mathbf{A}\mathbf{P} \qquad \text{and} \qquad \lambda\mathbf{p}' = \mathbf{A}'\mathbf{P}' \tag{3}$$

where $\mathbf{A}$ and $\mathbf{A}'$ represent the matrix of internal camera parameters – the focal lengths in the x- and y- directions f_x, f_y, and a skew factor s. The internal parameter matrix is of the form [8]:

$$\begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Using the above equations, we can show that the 2-D coordinates of the corresponding points $\mathbf{p}$ and $\mathbf{p}'$ are related by a 2-D affine transformation with 6 parameters:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}. \tag{4}$$

We need at the correspondence of 3 or more points to estimate the affine parameters.

3 points are needed to form an affine basis. Hence, a hash table for an image with M points will have $\binom{M}{3} \times 3!$ rows, each with $M - 3$ *affine* coordinates of the remaining non-basis points. We can calculate the time complexity of the matching step as in Section 2. As mentioned there, the relative camera motion between frames is often small. Hence for many practical cases, we may make an additional assumption to speed up alignment.

Algorithm 1:
(1) Represent the reference frame by a set of corner points.
(2) For every non-collinear triplet of points, find the angle (θ) formed by two linearly independent vectors and the length (l) between the end points. We use these as parameters in the hash table. In this way, we have $\binom{M}{3}$ values of θ and l.
(3) For the second frame of the scene, find the angle θ and length l for every non-collinear triplet as shown in Figure 1. So we have $\binom{N}{3}$ values of θ and l for hash table comparison. (4) For every basis

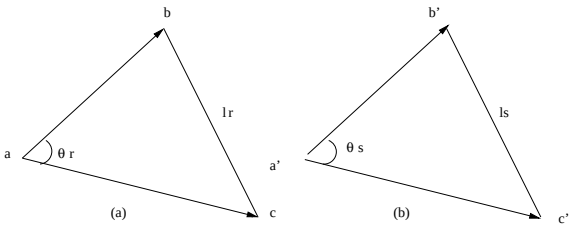

Figure 1: (a, b, c) triplet in the ref. image (left) and (a', b', c') triplet in the second image (right)

triplet in second image, find the difference between angle $\theta s_{(j)}$ and angle $\theta r_{(i)}$ of all basis triplet in the reference image.

$$\delta\theta_{(i,j)} = \mid \theta s_{(j)} - \theta r_{(i)} \mid$$

where $i = 1, 2, 3 \ldots \binom{M}{3}$; $j = 1, 2, 3 \ldots \binom{N}{3}$. Similarly, calculate the difference in length as

$$\delta l_{(i,j)} = \mid ls_{(j)} - lr_{(i)} \mid$$

Out of $\binom{M}{3} \times \binom{N}{3}$ combinations, few most likely to be correct pairs can be retained for further consideration. We discard the basis triplets which give angle difference more than a threshold. In this pass many pairs are expected to be disqualified. Then select those triplets for which δl is less than some small threshold. The idea of doing this is to reduce the length of the hash table, so that one has to compute only a few candidate matching triplets between the two image pairs. Since we are looking for correspondences between interest points detected in for separate images, only those triplets which preserve the shape and size in the two images are considered for possible matching. It should be noted that θ and l are *not* affine invariants. However, we may often make this assumption as motion of the camera is often kept very small to generate good quality mosaics.

Based on this correspondence, the transformation can be found from a pair of matched triplet or estimated from more matched triplets by least square error estimation (LSE) method. Select the Q points (we have considered $Q = 20$) around the basis triplet in reference image and the second image. Let these points be U_j and V_j. The required transformation can be obtained as solution of LSE estimation which minimizes the LSE measure

$$\delta = \sum_{j=1}^{p} \mid TU_j - V_j \mid^2 \tag{5}$$

with respect to unknown motion parameters. The minimum value of δ gives the correct transformation. The pixels in the overlapping part are taken from the single image or by averaging of pixels.

Here we are using the Harris corner detector [9] to detect interest points, which are used as feature points. The first row of Figure 2 shows two images taken by such an imaging setup. The image at

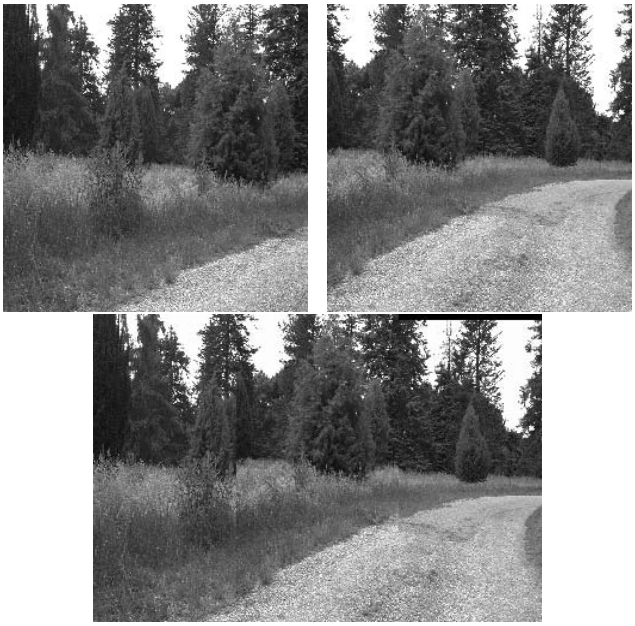

Figure 2: Two sample images and their resultant mosaic (bottom row): Details in text

the bottom shows the resultant mosaic. We show another example of this case in Figure 3.

4. PANORAMIC IMAGE MOSAICING

We now consider a camera rotating about its optical centre. Such images when stitched together constitute a panoramic mosaic. A commonly used camera model is [8]:

$$\lambda \mathbf{p} = \mathbf{A} \left[\begin{array}{c|c} \mathbf{R} & \mathbf{T} \end{array} \right] \mathbf{P_w} \tag{6}$$

relating the coordinates of a 3-D point in the world coordinate system $\mathbf{P_w} = [X \ Y \ Z \ 1]^T$ to its image

Figure 3: Resultant mosaic

point $[x\ y\ 1]^T$. λ is a projective constant. Here $\mathbf{R}$ denote a rotation matrix and $\mathbf{T}$, a translation vector. We can relate the image coordinates to the (non-homogeneous) coordinates of the 3-D points in the camera coordinate systems using $\lambda\mathbf{p} = \mathbf{AP}$ and $\lambda'\mathbf{p}' = \mathbf{A}'\mathbf{P}'$. For two cameras looking at the same point 3-D point $\mathbf{P_w}$

$$\mathbf{P}' = \mathcal{R}\mathbf{P} + \mathcal{T} \tag{7}$$

For panoramic image mosaicing, $\mathcal{T} = 0$. So $\lambda'\mathbf{A}'^{-1}\mathbf{p}' = \lambda\mathcal{R}\mathbf{A}^{-1}\mathbf{p}$. Hence, we have

$$\mu\mathbf{p}' = \mathbf{Hp} \tag{8}$$

H is a 3×3 invertible, non-singular homography matrix.

The above homography matrix represents a 2-D to 2-D projective transformation. Therefore, we use a projective basis for our geometric hashing-based scheme. We consider projective bases defined by pairs of four non-collinear projective points, using the canonical frame construction of [11]. This method considers mappings from the four non-collinear points to the corners of a unit square. Thus, we have $\binom{m}{4} \times m!$ possible choices for the basis vectors. We repeat the procedure of Section 2 for $k = 4$ here. However, as in Section 3, we can make a similar assumption here, to simplify the image alignment computation.

Algorithm 2:
(1) Represent the reference image by the sets of corners.
(2) For every quadruplet (of which three must be non-collinear), find the angles (θ_1, θ_2) formed by two linearly independent vectors and lengths(l_1, l_2) between two end points as shown in Figure 4.
(3) For the second frame of the scene, for every quadruplet find the corresponding (θ, l) values.
(4) for every quadruplet in the second image, find the difference between angle $\theta s1_{(j)}$ and angle $\theta r1_{(i)}$ and difference between $\theta s2_{(j)}$ and angle $\theta r2_{(i)}$ of all quadruplet in the reference image:

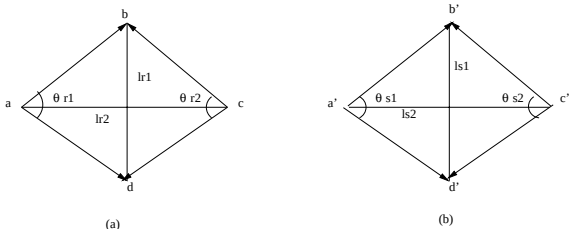

Figure 4: (a, b, c, d) basis quadruplet in reference image (left) and (a', b', c', d') basis quadruplet in second image(right).

$$\delta\theta1_{(i,j)} = \mid \theta s1_{(j)} - \theta r1_{(i)} \mid,\ \delta\theta2_{(i,j)} = \mid \theta s2_{(j)} - \theta r2_{(i)} \mid$$

Similarly, calculate the difference in lengths as

$$\delta l1_{(i,j)} = \mid ls1_{(j)} - lr1_{(i)} \mid,\ \delta l2_{(i,j)} = \mid ls2_{(j)} - lr2_{(i)} \mid$$

where $i = 1, 2, 3...\ \binom{M}{4}$; $j = 1, 2, 3...\ \binom{N}{4}$. out of $\binom{M}{4} \times \binom{N}{4}$ combinations, few most likely correct pairs can be identified through two passes. We can discard the quadruplets which gives angle difference more than threshold. The pairs of quadruplets with small difference in $\theta1$ and $\theta2$ will be considered for comparison based on lengths. By sorting based on $\delta l1$ and $\delta l2$, choose pairs with minimum value of $\delta l1$ and $\delta l2$. So, the pair with least values of $\delta\theta1, \delta\theta2, \delta l1, \delta12$, considered as a right candidate. Even in the absence of any invariance in parameters θ and l, the above constraints can be safely used as the relative change in these parameters is very small due to dense time sampling of images. The required transformation can be recovered from a pair of matched quadruplets, or estimated from more matched quadruplets by using least squares estimation method.

Figure 5: A panoramic mosaic created from a set of 30 frames of the Hiranandani complex, Powai

By finding transformation between two frames, the second frame is transformed with respect to first one and they are combined to form mosaic. Here, reference image is selected and all other images are registered with respect to the reference image, and mosaic is created. In this case, the region in the overlapping area is taken form one of the images, so there is no effect of blurring in the mosaic image. For the mosaic in Figure 5, we have considered a set of 30 images taken by a camera rotating by approximately 300^o.

5. CONCLUSION

This paper presents a new method for automatic generation of mosaic. Our method is based on Geometric Hashing. This gets over the problem of exponential time complexity in matching features across images. Additionally, the entire process does not require human intervention. We show results in the support of proposed strategies.

REFERENCES

[1] S. Peleg and B. Rousso, "Universal Mosaicing Using Pipe Projection," in *Proc. IEEE International Conference on Image Processing (ICIP)*, March 1998, pp. 123 – 142.

[2] S. Peleg and J. Herman, "Panormaic Mosaic by Manifold Projection," in *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, April 1997, pp. 338 – 343.

[3] P. Dani and S. Chaudhuri, "Automated Assembly of Images:Image Montage Preparation," *Pattern Recognition*, vol. 28, no. 3, pp. 431 – 445, March 1995.

[4] B. S. Reddy and B. N. Chatterji, "An FFT Based Technique for Translation ,Rotation and Scale Invariant Image Registration," *IEEE Trans. on Image Processing*, , no. 8, pp. 1266 – 1271, Augest 1996.

[5] L. G. Brown, "A Survey of Image Registration Techniques," *ACM Computing Surveys*, vol. 4, pp. 325 – 376, March 1992.

[6] A. Zisserman and D. Capel, "Automated Mosaicing with Super-resolution Zoom," in *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998, pp. 120 – 128.

[7] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "Object Recogination by Affine Invariant Matching," *Pattern Recognition*, pp. 335 – 344, June 1998.

[8] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[9] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," in *Proc. 4th Alvey Vision Conf.*, 1988, pp. 147 – 151.

[10] I. Zoghlami and R. Deriche, "Using Geometric Corners to Build a 2D Mosaic from a Set of Images," in *Proc. IEEE International Conference on Image Processing (ICIP)*, March 1997, pp. 420 – 425.

[11] C. A. Rothwell, *Recognition using Projective Invariance*, Ph.D. thesis, University of Oxford, 1993.