

CS 4495/7495 Computer Vision  
**Assignment 3: Creating a Cubic VR Movie**

13th October 2005

## **Introduction**

In this assignment you will use Sift features and RANSAC to automatically create 4 to 6 mosaics that can be used to create your very own Cubic VR movie. As seen in class, RANSAC is an algorithm to select inlier correspondences from a large set of putative correspondences. It does so by randomly sampling a minimum set of correspondences required for the model fitting, fitting a model to this set, and checking for the consensus on this model among the remaining putative correspondences. Thus, models containing outliers are rejected since they do not generate sufficient consensus.

The rest of this document describes the process in greater detail. Each section also has a piece at the end titled 'Deliverables' that contains the thing you should turn in related to that section.

## **1 Capturing a Panorama Sequence**

To capture the full cubic VR movie, you will need to produce 4 or 6 square mosaics corresponding to the cube faces, depending on whether you choose to go for a 360 degree panorama or the full viewing sphere. The latter is way cooler, of course, but is not possible in all environments, and is also more cumbersome to capture. As the auto-calibration code will accumulate errors quickly when estimating rotations via a chain of homographies, y

You will need to capture one image per face whose viewing direction is exactly perpendicular to the face: that is then the "anchor image" for that face. Then the other images can be registered to the anchor images to create the mosaics without knowledge of the rotation matrices, just like the example code that was demonstrated in class.

### **1.1 Deliverables**

1. By Friday 10/21, make the image sequences available on the class swiki. Also provide the focal length While you can now again upload all the images to the swiki, if you have a way to make the images available online that would be better.

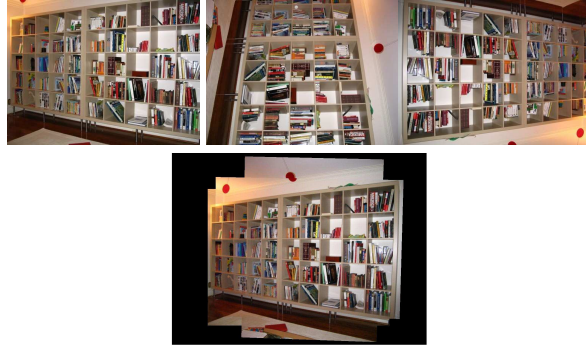


Figure 1: Mosaicking of three images to produce a single wide angle image

## 2 Computing Homographies

A homography is a point to point mapping between two spaces. For example, it can represent the correspondence between two images of the same scene. Homographies between pairs of images have to be computed in order to perform image mosaicking where multiple images taken from a rotating camera with fixed camera center can be stitched together into a panoramic view.

In 2D, a homography is defined by a  $3 \times 3$  matrix  $H$ , which relates the points  $p$  in one image (for our purposes, we can consider the 2D spaces to be images) to corresponding points  $p'$  in the second image.

$$wp' = Hp$$

where  $w$  is a scale parameter. Since the homography is only defined up to a scale, it only has 8 degrees of freedom and hence, can be estimated using a minimum of 4 correspondences between the two images.

Let  $(x, y, 1)$  and  $(x', y', 1)$  be a pair of corresponding points. Then, we have

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

from which we can eliminate  $w$  to get the two equations

$$\begin{aligned} h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' - x' &= 0 \\ h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy' - y' &= 0 \end{aligned}$$

Hence, every correspondence yields two equations, so that with four correspondences we can get a linear system with eight equations in eight variables. If there are more correspondences available, the system becomes over-determined and the least-squares solution can be extracted. This linear system can be written as  $Ah = 0$ , where,

assuming  $n$  correspondences,  $A$  is a  $2n \times 9$  matrix given by

$$A = \begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 & -y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 & -x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y'_2 & -y_2y'_2 & -y'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nx'_n & -y_nx'_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_ny'_n & -y_ny'_n & -y'_n \end{pmatrix}$$

and  $h$  is the  $9 \times 1$  vector containing all the elements of the homography  $H$ . Now, we can simply obtain  $H$  by performing the SVD of  $A$  and taking the eigenvector corresponding to the smallest eigenvalue of  $A$ .

To make this process insensitive to translation and scaling of the measurements, we need to perform a normalization similar to the one in the last assignment.

- Center the measurements at origin by pre-multiplying each measurement in homogeneous coordinates by  $T = \begin{pmatrix} 1 & 0 & -m_x \\ 0 & 1 & -m_y \\ 0 & 0 & 1 \end{pmatrix}$  where  $(m_x, m_y)$  is the mean of the measurements.
- Scale the modified measurements to have mean length  $\sqrt{2}$  by pre-multiplying with the matrix  $S = \begin{pmatrix} \frac{\sqrt{2}}{d} & 0 & 0 \\ 0 & \frac{\sqrt{2}}{d} & 0 \\ 0 & 0 & 1 \end{pmatrix}$ , where  $d$  is the average length of the measurements after the above step.
- After obtaining the homography using SVD, we have to de-normalize it. This can be done by the step  $H' = T_1^{-1}S_1^{-1}HS_2T_2$ , where  $T_1$  and  $S_1$  are the normalization matrices for the points in the first image,  $T_2$  and  $S_2$  are the corresponding matrices for the second image.  $H'$  is the true homography obtained after the de-normalization.

## 2.1 Deliverables

You should implement the SVD-based algorithm as described above in MATLAB and test it using the library images we showed in class. There are three test images, library[123].jpg, available in the file `a3-matlab.tar.gz` on the class website. You should fit homographies  $H_2$  from library1.jpg to library2.jpg, and  $H_3$  from library1.jpg to library2.jpg using the procedure above, and use them in the script `mosaic03.m` to produce the resulting mosaic.

1. Turn in the **code** and the **values** for the homography, along with a **short writeup** to Grant (grant.schindler@cc.gatech.edu).
2. Make the mosaic image available as deliverable 2 on the wiki.

## 3 Automatically Fitting a Homography using RANSAC

### 3.1 The RANSAC algorithm

A good description of the RANSAC algorithm is available from the book, the lecture slides, and many other sources. A summary of the algorithm is given here. Assume that the following are true -

- The model can be estimated from  $s$  data items
- There are  $N$  data items in total

Then the RANSAC algorithm consists of the following steps -

1. Selects  $s$  data items at random
2. Estimate model  $M$  using these  $s$  data items
3. Calculate how many data items (of  $N$ ) fit the model  $M$  within a user given tolerance. Call this  $K$ .
4. If  $K$  is big enough, accept fit and exit with success.
5. Repeat the above steps  $I$  times
6. Return the best model found so far

The main parameters in the algorithm are the user defined threshold in step 3 and the number of samples to be drawn,  $I$ .  $I$  can be found if the proportion of the inliers in the data  $\epsilon$  is known and the desired confidence of the correct model  $\eta$  is provided as 
$$I = \frac{\log(1-\eta)}{\log(1-\epsilon^s)}.$$

### 3.2 Fitting a homography using RANSAC

To fit a homography using RANSAC, the minimum sample set that we need is a set of four corresponding points between two images. We can select this set of four correspondences randomly from a set of putative correspondences obtained using SIFT feature matching. We can then fit a homography using the four correspondences as described in Section 2. To check if the remaining correspondences agree with the fitted homography, the difference  $p' - Hp$  can be found for each pair  $(p, p')$  of corresponding points. The threshold on this difference tells whether we consider the correspondence to be an inlier or an outlier.

### 3.3 Creating the Cube Face Mosaics

Each face of the cube should cover a viewing angle of 90 degrees in both horizontal and vertical direction. Each face mosaic should be a square image, mosaicked out of the anchor images and other images that are its direct neighbors (assuming that is enough to cover the 90 degree FOV). You can use the homographies found above, and create

with them a set of homographies from each image to the mosaic image coordinate frame. To see how, look at mosaic01-04.m available on the web-site. Hint: you will need to compose homographies.

These 4 to 6 face images can then be used to create a Quicktime VR movie. Frank made one and put it on <http://www.cc.gatech.edu/~dellaert/personal.html>, but he did not use enough images to cover the entire viewing sphere. Your job is to produce the mosaic cube images such that the mosaic is seamless all around.

Optionally, you can also make the Quicktime VR movie yourself and also put a link on the swiki. It is very easy: just drag and drop the images onto the MakeCubic tool, available from Apple (a link is on the web-site).

### 3.4 Deliverables

You should implement the RANSAC algorithm as described above in MATLAB, and create 6 (or 4) mosaics, one for each face of the cube. In the writeup (see below), also describe how you determined the homographies (from the mosaic to the images) to obtain the 90 degree field of view. Note the link on the web-site to a Quicktime VR authoring page, where various calculators are available.

1. Turn in the **code** and along with a **short writeup** to Grant ([grant.schindler@cc.gatech.edu](mailto:grant.schindler@cc.gatech.edu)).
2. Make the (4 to 6) mosaiced cube face images available as deliverable 3 on the swiki. Grant will convert them into a cubic VR movie for you.
3. Optional: use the makeCubic tool to create your own Quicktime VR movie.

## 4 How and when to turn in

This image sequences with sift features are due on 10/21, while the rest of the assignment is due on 10/28 (Friday) before class. As described above, email the deliverables to Grant. You should also hand in during class a hard copy of your writeup.