COS429 Homework 6

Due: Tuesday, December 2, 2008

1 Introduction

This is a Matlab programming assignment. All data necessary for this assignment are available at http://www.cs.princeton.edu/courses/archive/fall08/cos429/hw6/hw6data.zip This assignment is courtesy of Prof. Martial Hebert at CMU. Data courtesy of Dr. Theo Papadopoulo at INRIA and RealViz.

2 Image Warping

2.1 Introduction

In this problem, we consider a vision application in which a scene is being observed by multiple cameras at various locations and orientations. Figure 1 shows examples of images taken by five different cameras.

Given two cameras, a natural thing to do would be to compute how the scene currently seen from camera 1 would appear from camera 2's point of view. In particular, this would allow one to paste together multiple images which have overlapping regions, even if those images were obtained from different locations. This is also called *Mosaicing*.

Figure 1: Views from five cameras

Figure 2: Mosaic image.

An image mosaic is created by first choosing one camera as the reference frame and its associated image as the reference image. The task then consists of mapping all other images onto the reference frame so that all images can be displayed together with the reference image. In the most general case, there would be no constraints on the scene geometry, making the problem quite hard to solve. If, however, the scene can be approximated by a plane in 3D, a solution can be formulated much more easily – even *without* the knowledge of camera calibration parameters. Figure 2 depicts a typical mosaicing result.

To solve this section of the homework, you will first derive the transformation that maps one image onto another in the planar scene case. Then you will write a program to find this warping and apply it to a pair of test images, which are provided on the assignments web page.

2.2 Projective transformations of the plane

To begin, we consider the projection transformations of planes in images. Imagine two cameras C_1 and C_2 looking at a plane π in the world. Consider a point P on the plane π and its projections $p=(u_1,v_1,\ 1)^T$ in $image1$and $q=(u_2,v_2,\ 1)^T$ in $image2$.

Fact: There exists a unique (up to scale) 3 × 3matrix H such that, for any point P:

$$q \equiv Hp$$

(Here ≡ denotes the equality in homogeneous coordinates, meaning that the left and right hand side are proportional.) Note that H only depends on the plane and the projection matrices of the two cameras.

The interesting thing about this result is that by using H we can compute the image of P that would be seen in camera C_2 from the image of the point in camera C_1 without knowing its three-dimensional location. Such an H is a *projective* transformation of the plane, also referred to as a *homography*.

2.3 Estimating transformations from the image points

Given a set of points $\{(u_{1}^{i}, v1^{i})\}$, $i = 1...N$ in *image*1, and the corresponding set of points $\{(u_{2}^{i}, v2^{i})\}$, $i = 1...N$ in *image*2, show that H can be recovered from two sets of image points using *homogeneous* linear least squares (since H is only defined up to scale). This should be very similar to the linear camera calibration technique.

Write a program which computes the matrix H using the method you just derived. Input arrays should be $2 \times N$ matrices, listing a single image point's coordinates per column. Verify your results by manually selecting two sets of corresponding image points on the images provided above, and applying the trans-formation H to them. (Checkout the function *cpselect* in Matlab's Image Processing Toolbox for help selecting corresponding points if you are using Matlab.) Note that, throughout this estimation procedure, camera projection matrices did not come into play at all! Hence, no calibration was necessary.

Important hints: Recall that H is a projective transformation matrix and hence, defined only *up to a scale*. A good way to enforce this is by constraining the squared Frobenius norm (sum of the squared entries) of the matrix H to be 1. Also remember that q and Hp are only *proportional* to each other, or equivalently we have $q \times Hp = 0$.

Here are a few implementation tips:

- ←Use the images *piscine1.gif* and *piscine2.gif* in the assignment directory as test images. You can read these images using the function *imread* in Matlab. (There are five images there, *piscine1.gif* to *piscine5.gif* but you only need test your method on the first two.)
- ←Image coordinates of the points are the corresponding row and column indices of the image array.
- ←Beware of numerical ill-conditions: Your estimation procedure may perform better if image coordi-nates range from 0 to 2 as opposed to from 1 to 200. Consider scaling your measurements to avoid numerical issues.
- ←For the estimation to work well, a sufficient number of points should be provided. You should select the corresponding points in the two test images manually (or with the help of *cpselect*) and give all your results with your own point set.

2.4 Image warping and mosaicing

Write a program which takes as input an image *Iin*, areference image *Iref*,and a 3×3 homography, and returns 2 images as outputs. The first image is *Iwarp*, which is the input image *Iin* warped according to H to be in the frame of the reference image *Iref*. The second output image is *Imerge*, asingle mosaic image with a larger field of view containing both the input images. In order to avoid aliasing and sub-sampling effects, consider producing the output by solving for each pixel of the *destination* image rather than mapping each pixel in the input image to a point in the destination image (which will leave holes). Also note that the input and output images will be of different dimensions.

3 Submission

Please submit the following items to Blackboard (https://blackboard.princeton.edu/).

1. Derivation of the algorithm for estimating H. (For this item ONLY, you may instead submit a hardcopy to the homework box outside room 418a of the CS building before 11:59pm, December 2).
2. Your results on piscine1.gif (image1) and piscine2.gif (image2):
    a. An image showing the set of corresponding points you selected plotted on the original images.
    b. An image showing the result of warping applied to image1.
    c. An image showing the result of combining images 1 and 2.
3. Matlab code
    a. estimate_transform.m and image_warp.m