1 Introduction

This is a programming assignment, which should be done in Matlab as many of the necessary numerical routines are readily available (e.g., eig for the computation of the eigenvalues and eigenvectors of a matrix). All data necessary for this assignment is available on the course web page.\footnote{This assignment is courtesy of Prof. Jean Ponce at the University of Illinois at Urbana-Champaign and Prof. Martial Hebert at CMU. Data courtesy of Dr. Theo Papadopoulo at INRIA and RealViz.}

2 Image Warping

2.1 Introduction

In this assignment, 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 \textit{Mosaicing}.

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 course web page.

2.2 Projective transformations of the plane

To begin, we consider the projection of planes in images. Imagine two cameras $C_1$ and $C_2$ looking at a plane $\pi$ in the world. Consider a point $P$ on the plane $\pi$ and its projections $p = (u_1, v_1, 1)^T$ in \textit{image}1 and $q = (u_2, v_2, 1)^T$ in \textit{image}2.

\textbf{Fact:} There exists a unique (up to scale) $3 \times 3$ matrix $H$ such that, for any point $P$:

$$q \equiv Hp$$

(Here $\equiv$ 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.
Figure 1: Views from five cameras

Figure 2: Mosaic image.
The interesting thing about this result is that by using $\mathbf{H}$ we can compute the image of $\mathbf{P}$ 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 $\mathbf{H}$ is a projective transformation of the plane, also referred to as a homography.

Perhaps even more interesting, a homography can also be used to represent the transformation between images of a genuine 3-D scene for multiple images taken by camera that rotates in 3-D about a fixed point. Do you see why?

### 2.3 Estimating transformations from the image points

Given a set of points $\{(u_1^i, v_1^i), i = 1...N\}$ in image1, and the corresponding set of points $\{(u_2^i, v_2^i), i = 1...N\}$ in image2, show that $\mathbf{H}$ can be recovered from two sets of image points using homogeneous linear least squares (since $\mathbf{H}$ is only defined up to scale).

Write a program which computes the matrix $\mathbf{H}$ 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 two of the provided images, and applying the transformation $\mathbf{H}$ to them. (Checkout the function `cpselect` in Matlab’s Image Processing Toolbox for help selecting corresponding points.)

**Important hints**: Recall that $\mathbf{H}$ 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 $\mathbf{H}$ to be equal to 1. Also remember that $\mathbf{q}$ and $\mathbf{H}\mathbf{p}$ are only proportional to each other, or equivalently we have $\mathbf{q} \times \mathbf{H}\mathbf{p} = 0$.

Here are a few implementation tips:

- Use the images `piscine1.gif` and `piscine2.gif` on the web page 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 coordinates 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 $\mathbf{I}_{\text{in}}$, a reference image $\mathbf{I}_{\text{ref}}$, and a $3 \times 3$ homography, and returns 2 images as outputs. The first image is $\mathbf{I}_{\text{warp}}$, which is the input image $\mathbf{I}_{\text{in}}$ warped according to $\mathbf{H}$ to be in the frame of the reference image $\mathbf{I}_{\text{ref}}$. The second output image is $\mathbf{I}_{\text{merge}}$, a single 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. Also, consider how to bring together the information from both images into the mosaiced image.

Finally, apply your method to all five images to produce a single mosaic, akin to Figure 2.

### 3 What to submit ?

Submit a hardcopy at the due date with the following.
1. Derivation of the algorithm for estimating $H$.

2. Your code for: estimating $H$, image warping, and image merging.

3. An image showing the set of corresponding points you selected plotted on the original images.

4. An image showing the result of warping applied to image 1.

5. An image showing the result of combining images 1 and 2.

6. An image showing the mosaic of all five images.

In addition, please email to kriegman@cs.ucsd.edu a copy of your code. If it is a single file, just send the .m file. If it is contained in multiple files, send a zip or tar file. In the subject line, please put the string: CSE252HW1