1. **Perspective Projection [2pts]**

Consider a perspective projection where a point

\[ P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]

is projected onto an image plane \( \Pi' \) represented by \( k=f' \) as shown in the following figure.

![Perspective Projection Diagram](image)

The first, second, and third coordinate axes are denoted by \( i, j, \) and \( k \) respectively. Consider the projection of a ray in the world coordinate system

\[ Q = \begin{bmatrix} 5 \\ 7 \\ 0 \end{bmatrix} + t \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \]

where \(-\infty \leq t \leq -1\). Calculate the coordinates of the endpoints of the projection of the ray onto the image plane.

2. **Rigid Transformation [2pts]**

Consider a rigid transformation where a point \( A \) is rotated about the \( j \)-axis by \( \pi/6 \) radians and translated by

\[ \begin{bmatrix} 0 \\ 3 \\ 1 \end{bmatrix} \]

to another point \( A' \). When the two points \( A \) and \( A' \) are represented in the homogeneous coordinate system by

\[ A = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{and} \quad A' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \]
respectively. Write a 4x4 matrix \( M \) such that
\[
A' = MA
\]

3. **Projective Transformation [2pts]**

A rectangle under a projective transformation can be mapped to an arbitrary quadrilateral. Find the 3x3 matrix mapping the unit square with homogeneous coordinates for its four corners
\[
(0, 0, 1), (1, 0, 1), (1, 1, 1), (0, 1, 1)
\]
to the quadrilateral with corners
\[
(7, 8, 1), (5, 3, 1), (7, 3, 1), (9.5, 7, 1)
\]
respectively. Normalize so that the last entry of the matrix is 1 (i.e. the entry in the third column and third row).

4. **Thin Lens Equation [2pts]**

An illuminated arrow forms a real inverted image of itself at a distance \( w = 50 \text{cm} \), measured along the optic axis of a convex thin lens (see Figure 1). The image is just half the size of the object.

![Figure 1: Problem 3 Setup](image)

(a) How far from the object must the lens be placed?
(b) What is the focal length of the lens?
(c) Suppose the arrow moves 5cm to the right while the lens and image plane remain fixed. This will result in an out of focus image; what is the radius of the corresponding blur circle formed from the tip of the arrow on the image plane assuming the diameter of the lens is \( d \)?

5. **Affine Projection [2pts]**

Consider an affine camera and a line in 3D space. Consider three points \((A, B, and C)\) on that line, and the image of those three points \((a, b and c)\). Now consider the distance between \(a\) and \(b\) and the distance between \(a\) and \(c\). Show that the ratio of the distance is independent of the direction of the line.
6. Affine Pose Estimation [2pts]

An affine camera transforms 3D (homogeneous) points \( \mathbf{x} = [x, y, z, 1]^T \) according to,

\[
x' = M \mathbf{x}
\]

where

\[
M = \begin{bmatrix}
a & b & c & d \\
e & f & g & h \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

is the camera matrix and \( \mathbf{x}' = [x', y', 1]^T \) is the mapping onto the image plane.

Given a set of corresponding points \( \mathbf{x} \) and \( \mathbf{x}' \), your task is to recover the affine camera parameters.

(a) What is the minimum number of corresponding points needed to recover the camera parameters?

(b) Are there any conditions on the positions of these corresponding points for successful recovery of \( M \)? List two degenerate configurations. Can you think of a necessary condition?

(c) Running `house_model.m` will display an image of an affine transformed house. A set of 8 corresponding points are contained in the Matlab variables \( \text{Xcorr} \) (original points) and \( \text{Xcorr}_a \) (affine transformed points). Write a Matlab function that takes as input \( \text{Xcorr} \) and \( \text{Xcorr}_a \) and returns the affine transformation matrix, \( M \). What is the recovered affine transformation matrix?

(d) Repeat part (c) using only correspondence points 1,3,4, and 6. What happens when you try to estimate \( M \)? Why?

**What to turn in**: Answers to the questions and a print out of your code from problem 6c.

7. Programming Assignment: Image Warping and Mosaicing [10pts]

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 1 matrix). All data necessary for this assignment is available on the course web page.

**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 2 and 4 show examples of images taken by different camera views.

Given two camera views, 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.

---

\(^1\)This assignment is based on an assignment by Prof. Jean Ponce at the University of Illinois at Urbana-Champaign and Prof. Martial Hebert at CMU.
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 3 and 5 depict a typical mosaicing results. Alternatively, if the scene has a range of depths (not a plane), but the camera rotates about its optical center, the images can be stitched together into a mosaic.

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.

**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 $p = (u_1, v_1, 1)^\top$ in image 1 and $q = (u_2, v_2, 1)^\top$ in image 2.
Fact 1 There exists a unique (up to scale) 3x3 matrix \( H \) such that, for any point \( P \):
\[
q \equiv Hp
\]
(Here denotes 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 C2 from the image of the point in camera C1 without knowing its three-dimensional location. Such an \( H \) 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 the optical center. Do you see why? This will even work if the zoom of the camera changes between images.

**Estimating Transformations From the Image Points**

Given a set of points \( \{(u_1, v_1^i)\}, i = 1...N \) in image 1 and the corresponding set of points \( \{(u_2, v_2^i)\}, i = 1...N \) in image 2, show that \( H \) can be recovered from two sets of image points using homogeneous linear least squares (\( H \) is only defined up to scale).

Write a program which computes the matrix \( H \) using the method you just derived. Input arrays should be 2xN matrices, listing a single image points 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 \( H \) to them (checkout the function cpselect in Matlabs Image Processing Toolbox for help selecting corresponding points). 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 equal to 1. Also remember that \( q \) and \( Hp \) are only proportional to each other, or equivalently we have \( q'Hp = 0 \).

Here are a few implementation tips:

- Use the images img4.jpg and img5.jpg on the web page as test images. You can read these images using the function imread in Matlab. (There are six images there, img1.jpg to img6.jpg, but you only need test your method on these 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 -1 to 1 as opposed to from 1 to 800. 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.

**Image Warping and Mosaicing**

Write a program which takes as input an image \( I_{in} \), a reference image \( I_{ref} \), and a 3x3 homography \( H \) and returns two images as outputs. The first image is \( I_{warp} \), which is the input image \( I_{in} \) warped according to \( H \) to be in the frame of the reference image \( I_{ref} \). The second output image is \( I_{merge} \), 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,

---

2 Hartley normalization is a well known procedure for doing this. Given a set of homogeneous 2D points \( \{x_1, \ldots, x_n\} \), the Hartley normalized point set is given by \( \hat{x} = Ax_i \), where \( A \) is a 3x3 linear transformation such that \( \{\hat{x}_1, \ldots, \hat{x}_n\} \) has zero mean and unit variance. In the context of homography estimation, we can write \( q = Hp \Rightarrow A_qq = A_qHA_p^{-1}A_pHp \Rightarrow \hat{q} = H\hat{p} \). Thus, we can solve for \( H \) as \( H = A_q^{-1}H A_p \).
consider how to bring together the information from both images into the mosaiced image.

Finally, apply your method to all six images, using img3.jpg as the reference frame, to produce a single mosaic, akin to Figure 3. Repeat this process with at least three images of the blackboard to produce a mosaic akin to figure 5.

While not necessary for this assignment, you can take out your own camera to take a collection of images and create your own mosaic. Here are some issues that might arise outside of the lab setting: Note that you don’t want to use a wide-angle lens or zoom too far out. Wide-angle lenses often display barrel distortion in which straight lines become curved (bowed out). This type of distortion has to be corrected before applying the technique of this assignment. Also, when an image is shot on automatic mode, the exposure may vary between images, and so it may be necessary to adjust the brightness values to avoid a visible seam between images. On further complication is vignetting; the image may be darker toward the outside edges of the image. If so, it may be necessary to estimate the vignetting and compensate for it (you don’t need to do this for the assignment).

**What to turn in:**
Submit a hardcopy on the due date with the following:

(a) Your code for estimating H, image warping, and image merging.
(b) An image showing the set of corresponding points you selected plotted on the original images.
(c) An image showing the result of warping applied to image 5.
(d) An image showing the result of combining images 5 and 4.
(e) An image showing the mosaic of all six images using image 3 as the reference frame.
(f) An image showing a mosaic of at least four blackboard images as well as the original images.

In addition, please email to aziegler@cs.ucsd.edu a copy of your code and figures. Please send as a zip or tar file. In the subject line, please put the string CSE252HW1.