CSE 152: Introduction to Computer Vision
Homework 3

Instructions:

- Total points: 100
- **Due: 11:59 pm, Monday, Nov 26, 2018**
- For the implementation part, we've provided you with a number of functions and scripts in the hopes of alleviating some tedious or error-prone sections of the implementation. **Please implement your part under the comment line "YOUR CODE HERE," and stick to the headers, variable names, and file conventions provided.**
- For the theory problems, please justify your solutions using necessary derivations or explanations.
- When you're finished, compress your code and writeup into a single file named **YourPID_hw3.zip** and submit it to descartes.ucsd.edu. **Also submit your writeup (pdf) to Gradescope.**
- A complete submission consists of the following files:
  - `Q2_1.m`
  - `Q2_2.m`
  - `Q2_3.m`
  - `eightpoint.m`
  - `epipolarCorrespondence.m`
  - `camera2.m`
  - `displayEpipolarF.m`
  - `epipolarMatchGUI.m`
  - `triangulate.m`
  - `Q2_1.mat`
  - `Q2_3.mat`
  - A writeup (PDF format)

  Make sure that we can run your code without any modification, otherwise you are at risk of losing points.
- **Start early!** Please familiarize yourself with MATLAB and allow enough time for debugging.

1 Theory Questions: Epipolar Geometry [29 points]

1. **(14 points.)** Consider the case of two cameras viewing an object such that the second camera differs from the first by a pure translation parallel to the image plane. Show that the epipolar lines in the two cameras are parallel.

2. **(15 points.)** Suppose two cameras fixate on a point P (see figure 1) in space such that their principal axes intersect at that point. Show that if the image coordinates are normalized so that the coordinate origin $(0, 0)$ coincides with the principal point, the $\mathbf{F}_{33}$ element of the fundamental matrix is zero.

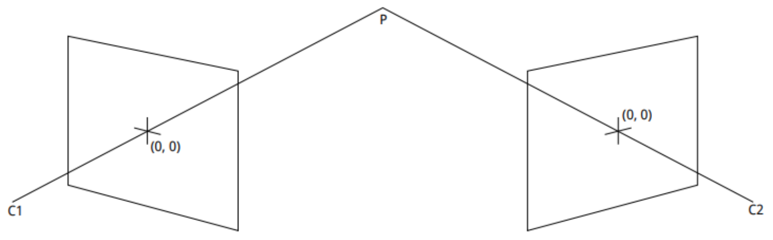

Figure 1: $C1$ and $C2$ are the optical centers. The principal axes intersect at point P.

2 3D Reconstruction [70 points]

In this problem, you will estimate the fundamental matrix from a pair of images. Then you will compute the camera matrices and triangulate the 2D points to obtain the 3D scene structure. The data for this problem can be found in the `temple/` directory, where there are two images of a temple.

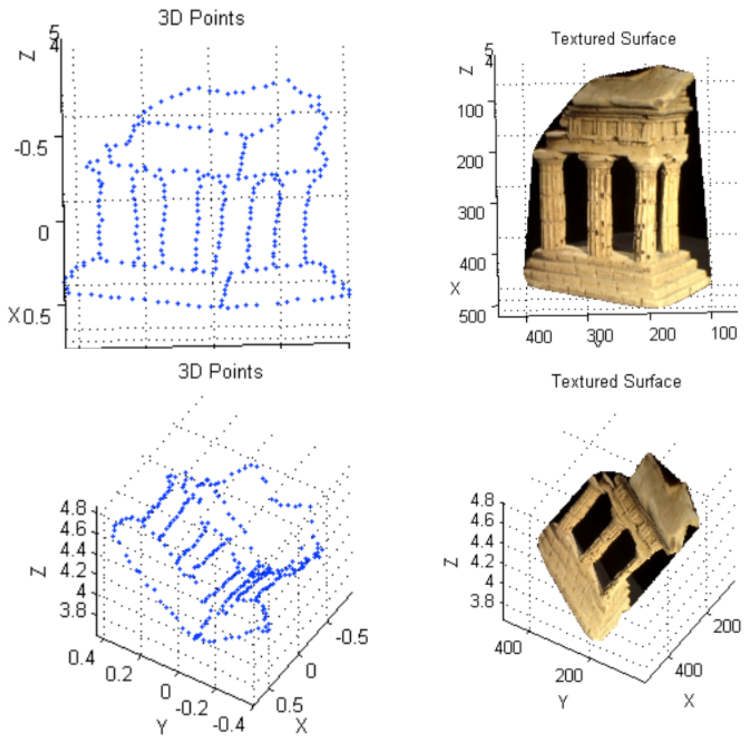

Figure 2: Two novel views of an object reconstructed in 3D from two images.

2.1 The Eight Point Algorithm [20 points]

FILE TO EDIT: `eightpoint.m`
FILE TO RUN: `Q2_1.m`
ADDITIONAL DELIVERABLES: `Q2_1.mat`, screenshot(s) in writeup

The 8-point algorithm discussed in class and outlined in Section 10.1 of Forsyth & Ponce is arguably the simplest method for estimating the fundamental matrix. For this section, you can use the provided correspondences from `temple/some_corresp.mat`. Pass them to your implementation of the 8-point algorithm.

For this portion of the assignment, fill in the function in `eightpoint.m` with the following signature:

```
function F = eightpoint(X,Y,M);
```

where X and Y are each $N \times 2$ matrices with coordinates that constitute correspondences in the first and second image respectively (the format returned by `cpselect`). M is the scaling factor. Remember that the first coordinate of a point in the image is its column entry, and the second coordinate is the row entry. Note: 8-point is just a figurative name. Your algorithm should use an overdetermined system.

You should scale the data by dividing each coordinate by M, the largest image dimension. After computing F, you will have to "unscale" the fundamental matrix.

To visualize the correctness of your estimated F, use the provided function `displayEpipolarF`. **In your writeup, include a screenshot of the output of `displayEpipolarF` showing three points and the epipolar lines in the other image, demonstrating that the corresponding points are on the epipolar lines in the second image. This image should look like the one below (Figure 3).**

Also save F, M, and your points as `Q2_1.mat`.

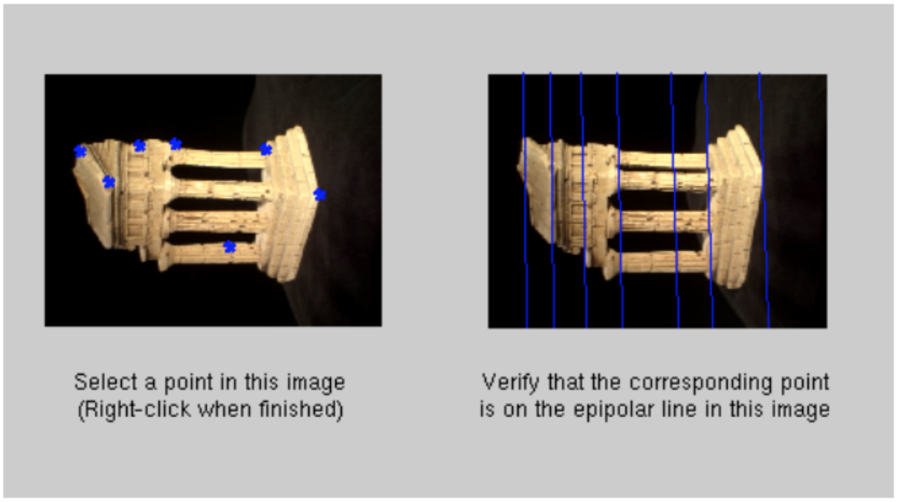
Select a point in this image
(Right-click when finished)
Verify that the corresponding point
is on the epipolar line in this image

Figure 3: Points on the epipolar lines.

2.2 Metric Reconstruction [20 points]

FILE TO EDIT AND RUN: `Q2_2.m`
ADDITIONAL DELIVERABLES: screenshot(s) in writeup

The fundamental matrix can be used to determine the camera matrices of the two views used to generate F. In general, M_1 and M_2 will be projective camera matrices, i.e., they are related to each other by a projective transform. For details on recovering M_2, see 7.2 in Szeliski. Thankfully, we have provided you with the function `camera2(F, K1, K2, pts1, pts2)` that will find M_2 given F , K_1, and K_2 and a number of correspondence points. Load and use the correspondence points provided in `some_corresp.mat`. Load and use the intrinsic matrices K_i from `temple/intrinsics.mat`.

We have also provided a function to triangulate a set of 2D coordinates in the image to a set of 3D points with the signature:

```
function P = triangulate(M1, pts 1, M2, pts 2)
```

where `pts_1` and `pts_2` are the 2D image coordinates and P is a matrix with the corresponding 3D points, per row. Make sure to multiply your M1, M2 by the camera matrices!

Included in the homework folder is a file `many_corresp.mat` which contains 288 hand-selected points from `im1` saved in the variables `x1` and `y1`.

Now, we can determine the 3D locations of these point correspondences using the `triangulate` function. These 3D point locations can then be plotted using the MATLAB function `scatter3`. The resulting figure can be rotated using the Rotate 3D tool, which can be accessed through the figure menubar.

Please take six screenshots showing different views of the 3D visualization, and include them in your writeup.

2.3 3D Correspondence [30 points]

FILE TO EDIT: `epipolarCorrespondence.m`
FILE TO RUN: `Q2_3.m`
ADDITIONAL DELIVERABLE: `Q2_3.mat`, screenshot(s) in writeup

To conclude this project, you will calculate correspondences using 3D information via epipolar geometry.

In `epipolarCorrespondence.m`, fill in the function

$$\texttt{function}[\texttt{x2}, \texttt{y2}] = \texttt{epipolarCorrespondence}(\texttt{im1}, \texttt{im2}, \texttt{F}, \texttt{x1}, \texttt{y1})$$

and associated helper functions `getWindow` and `computeDifference`.

`epipolarCorrespondence` takes in the x and y coordinates of a pixel on `im1`, and your fundamental matrix **F**, and returns the coordinates of the pixel on `im2` which correspond to the input point. Fortunately, we know the fundamental matrix **F**. Therefore, instead of searching for our matching point at every possible location in `im2`, we can simply search over the set of pixels that lie along the epipolar line (recall that the epipolar line passes through a single point in `im2` which corresponds to the point (x1, y1) in `im1`). Slide a window along the epipolar line and find the one that matches most closely to the window around the point in `im1`. There are various ways to compute the window similarity. For this assignment, simple methods such as the Euclidean or Manhattan distances between the intensity of pixels should suffice. Implementation hints:

- Experiment with various window sizes.
- It may help to use a Gaussian weighting of the window, so that the center has greater influence than the periphery.
- Since the two images only differ by a small amount, it might be beneficial to consider matches for which the distance from (x1, y1) to (x2, y2) is small.

To test your function `epipolarCorrespondence`, we have included a GUI:

$$\texttt{function}[\texttt{coordsIM1}, \texttt{coordsIM2}] = \texttt{epipolarMatchGUI}(\texttt{im1}, \texttt{im2}, \texttt{F})$$

This script allows you to click on a point in `im1`, and will use your function to display the corresponding point in `im2`. The process repeats until you right-click in the figure, and the sets of matches will be returned. Use your function `epipolarCorrespondence` to calculate the corresponding points in `im2` from chosen points in `im1`. Save your points and **F** to `Q2_3.mat`.

Also include as part of your writeup a screenshot of the output of `epipolarMatchGUI` demonstrating three identified correspondences across the two temple images.

3 Questionnaire [1 point]

Approximately how many hours did you spend on this homework? Please answer this question in your writeup.