CMPT 412
Assignment 3 Report

Maoshun Shang

301280479

Introduction

This assignment mainly focuses on the technique studied in lectures, including basics about perspective projection, camera intrinsic and extrinsic, triangulation, and stereo etc. The programming language used is MATLAB, which helps with reducing the complexity of heavy computation on matrix and vector calculations.

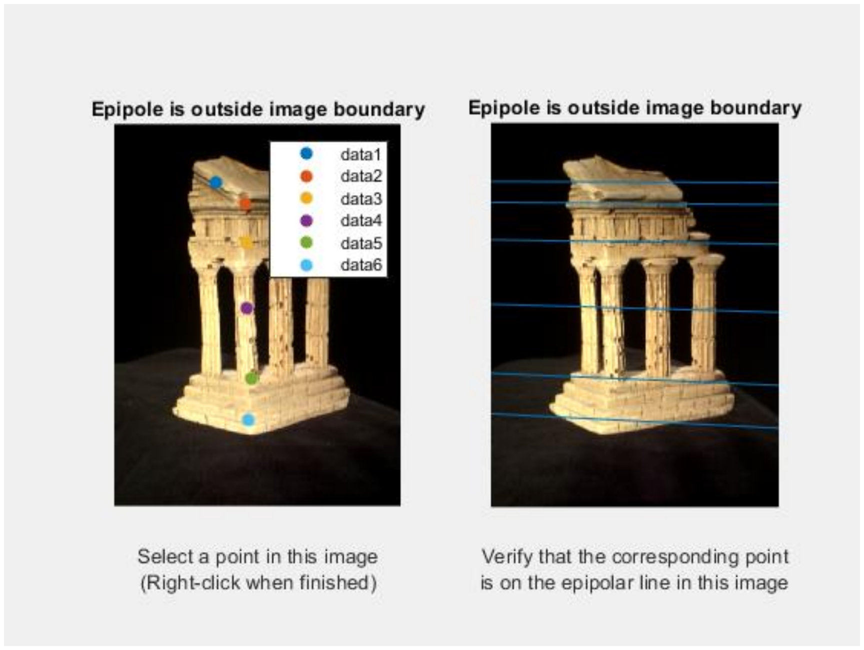
Part 1: Sparse Reconstruction

3.1.1: 8-points Algorithm

With the provided temple images and some corresponding points, the calculated fundamental matrix (with refined process) is:

$$F = \begin{bmatrix} -1.0838 * 10^{-9} & 1.7544 * 10^{-7} & -1.1706 * 10^{-5} \\ 8.0295 * 10^{-8} & 5.4399 * 10^{-9} & -0.0015 \\ -1.4504 * 10^{-5} & 0.0015 & 0.0064 \end{bmatrix}$$

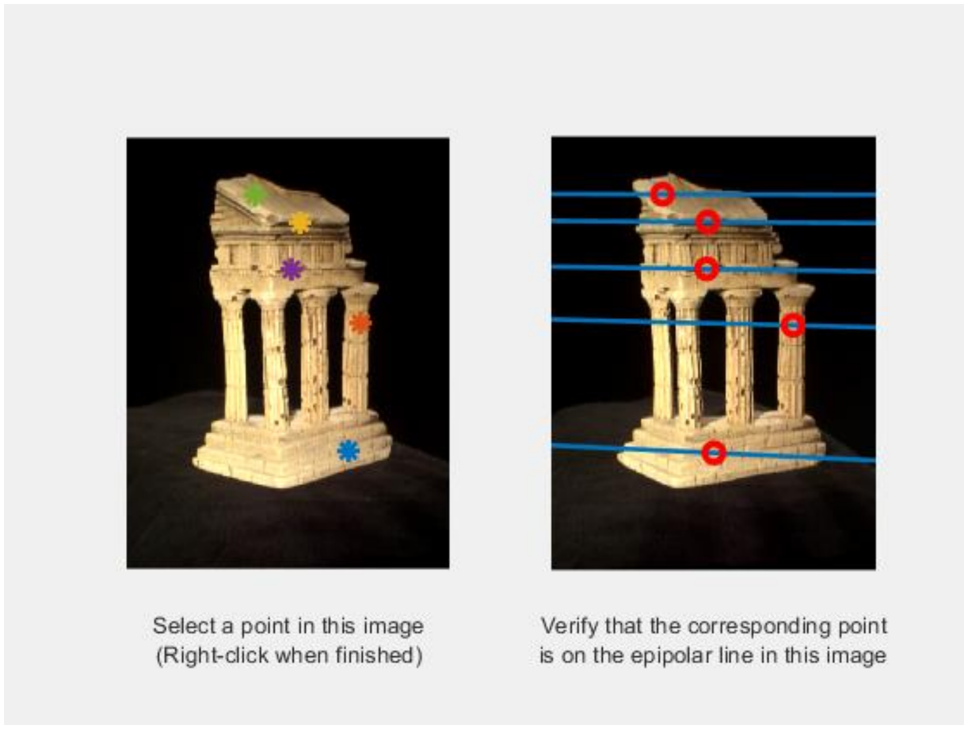
Here is the graph for visualizing some epipolar lines from the script "displayEpipolarF.m":


Epipole is outside image boundary
data1
data2
data3
data4
data5
data6
Epipole is outside image boundary
Select a point in this image
(Right-click when finished)
Verify that the corresponding point
is on the epipolar line in this image

3.1.2: Find Epipolar Correspondences

The similarity function I decide to use is sum of squared difference (Euclidean distance) between two windows. Please note the sum function needs MATLAB R2018b or later version to run.

The result of running "epipolarMatchGui.m" is shown below, the performance on easy points are good: the corners ,dots and most edges can be matched correctly. However, there are points that fail consistently. For example, the blue point on the stairs (left image) is matched to a point a little bit toward left. This is due to all the stairs have similar texture or pattern. That is, if we take a point, P1, on the stair and another point, P2, on a different area of the stair along the same epipolar line, the window around P1 is very similar to that around P2 so that the algorithm wrongly regards P1 and P2 as a pair of same point (but in fact they are different points).


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

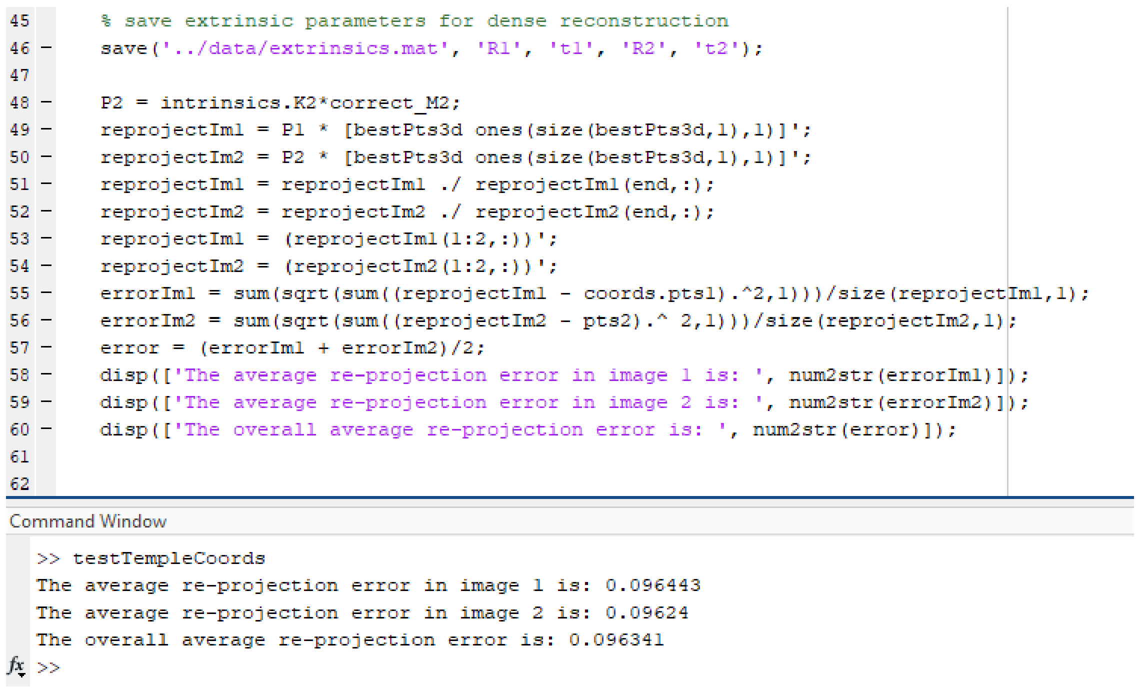
3.1.3: Compute the Essential Matrix

$$E = \begin{bmatrix} -0.0025 & 0.4070 & 0.0476 \\ 0.1863 & 0.0127 & -2.2833 \\ 0.0076 & 2.3114 & 0.0026 \end{bmatrix}$$

3.1.4: Implement Triangulation

We retrieve 4 possible extrinsic matrices for camera2, for getting the correct matrix, I keep a 3D array containing all projected 3D points, where the size of the 3rd dimension is 4 (for 4 possible matrices). We count the number of negative depth points, meaning the points have negative z-value, along its 3rd dimension. Then select the matrix producing the minimum amounts of negative-depth 3D points as optimum extrinsic matrix.

We re-project the 3D point back to the 2D images, then calculate the mean Euclidean distance. For image1, the re-projection error is 0.096443, while the image 2 has average re-projection error 0.09624. The overall re-projection error is 0.096341 over all points in image1 and image2. (See the screenshot below)



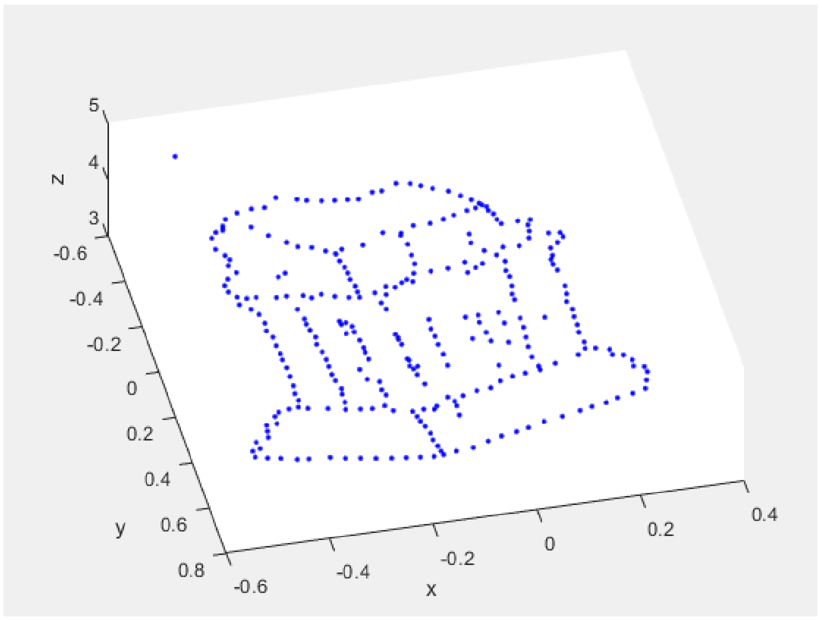
```
% save extrinsic parameters for dense reconstruction
save('../data/extrinsics.mat', 'R1', 't1', 'R2', 't2');

P2 = intrinsics.K2*correct_M2;
reprojectIm1 = P1 * [bestPts3d ones(size(bestPts3d,1),1)]';
reprojectIm2 = P2 * [bestPts3d ones(size(bestPts3d,1),1)]';
reprojectIm1 = reprojectIm1 ./ reprojectIm1(end,:);
reprojectIm2 = reprojectIm2 ./ reprojectIm2(end,:);
reprojectIm1 = (reprojectIm1(1:2,:))';
reprojectIm2 = (reprojectIm2(1:2,:))';
errorIm1 = sum(sqrt(sum((reprojectIm1 - coords.pts1).^2,1)))/size(reprojectIm1,1);
errorIm2 = sum(sqrt(sum((reprojectIm2 - pts2).^ 2,1)))/size(reprojectIm2,1);
error = (errorIm1 + errorIm2)/2;
disp(['The average re-projection error in image 1 is: ', num2str(errorIm1)]);
disp(['The average re-projection error in image 2 is: ', num2str(errorIm2)]);
disp(['The overall average re-projection error is: ', num2str(error)]);
```

Command Window

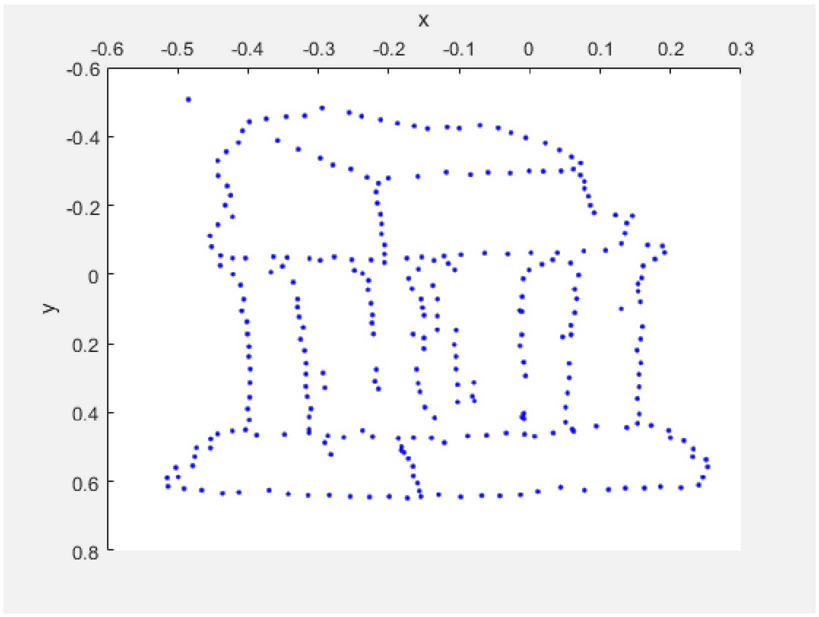
```
>> testTempleCoords
The average re-projection error in image 1 is: 0.096443
The average re-projection error in image 2 is: 0.09624
The overall average re-projection error is: 0.096341
>>
```

3.1.5: Test TempleCoords

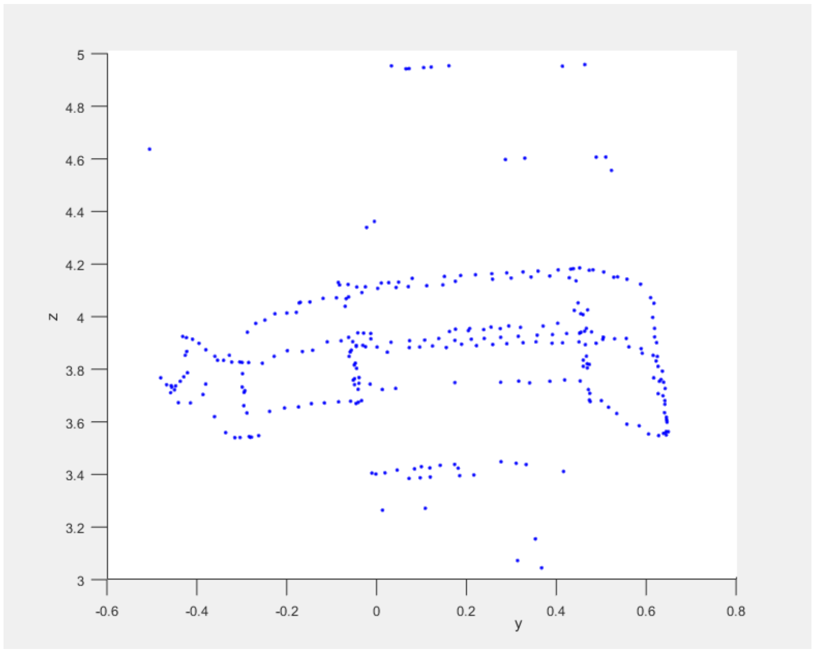
The following are 3 images of the final reconstruction of the templeCoords points from different angles.



A 3-D view



A view from z-axis direction

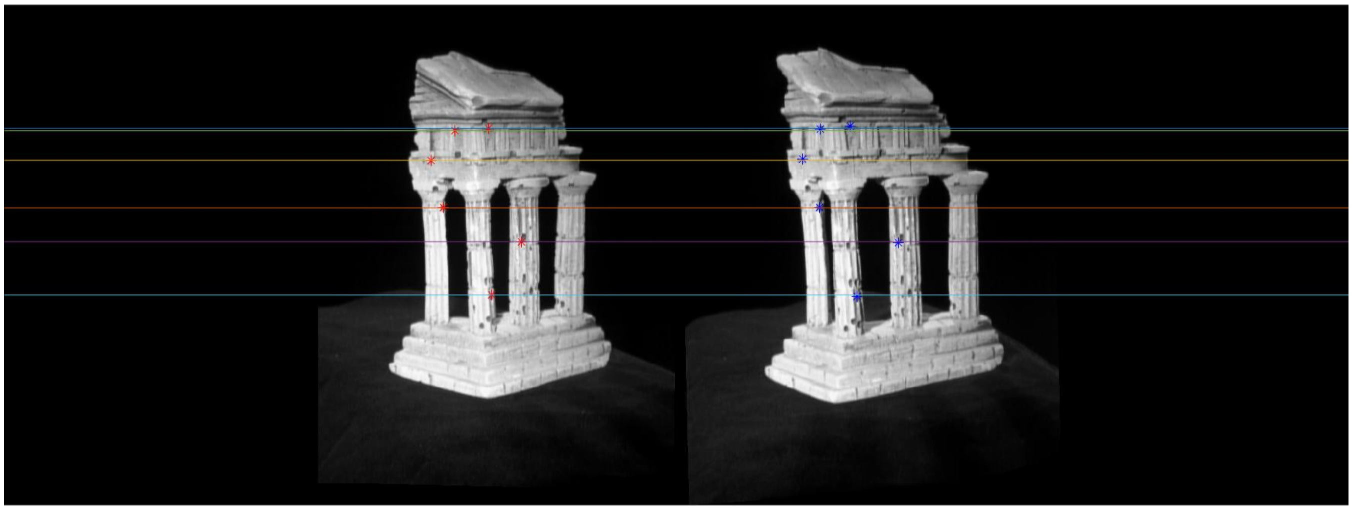


A view from x-axis direction

Part 2: Dense Reconstruction

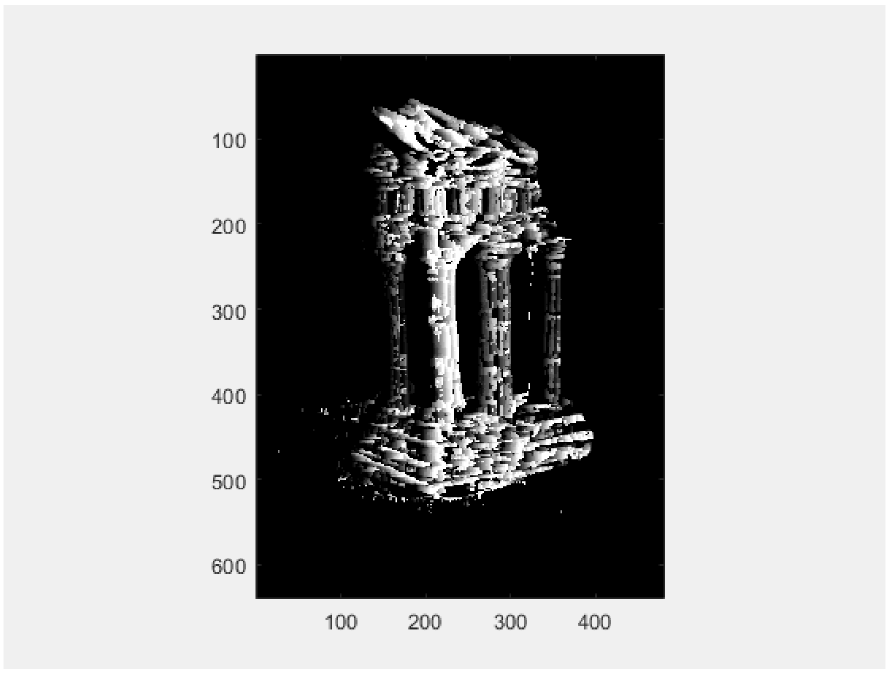
3.2.1: Image Rectification

The following is a sample output of running “testRectify.m”. The results show 6 perfectly horizontal epipolar lines and 6 pairs of corresponding points on both images lying on the same epipolar line.



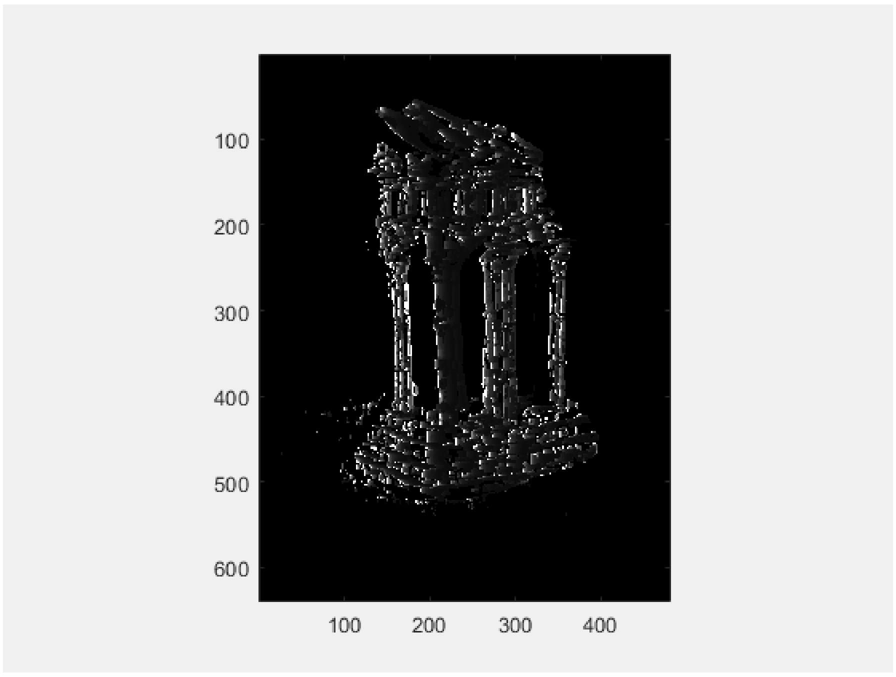
3.2.2: Dense Window Matching

The two images below are the outputs of running “testDepth.m”. The first image shows disparity map of the temple.



3.2.3: Depth Map

The image below is the depth map of the temple.


100
200
300
400
500
600
100
200
300
400