Project 3
3D Reconstruction

Due date: 23:59 Thursday 12/10 (2020)

Simon Fraser University

CMPT 412 Fall 2020

By: Saqib Aziz Dhuka

Student ID: 301338669

Due: 10th December 2020

Saqib Aziz Dhuka
301338669
CMPT 412 Fall 2020

Project 5

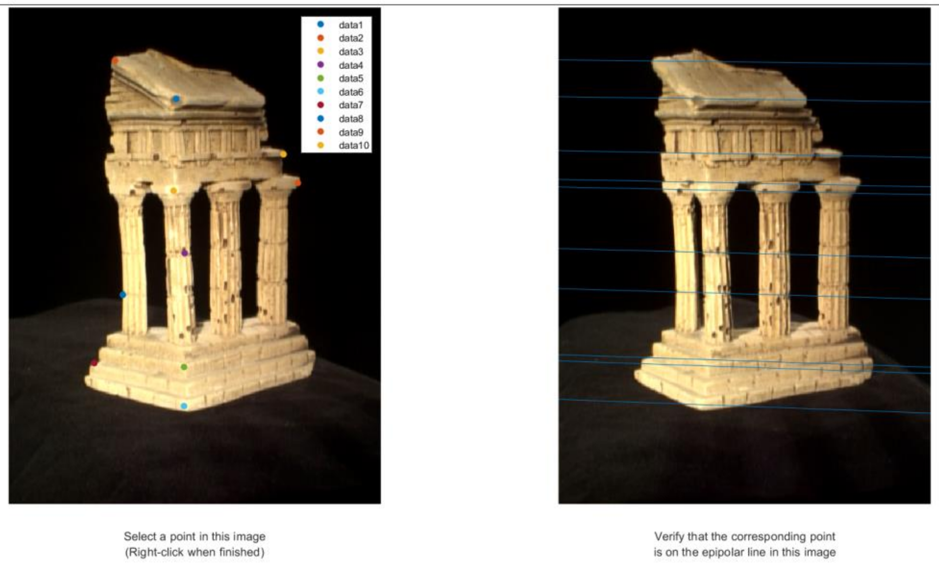
3.1

3.1.1:

Matrix F:

```
0.0000   -0.0000   -0.0000
-0.0000   -0.0000    0.0010
0.0000   -0.0010   -0.0042
```

Note: The lines are hard to spot. Please zoom in to view them better.



Select a point in this image
(Right-click when finished)

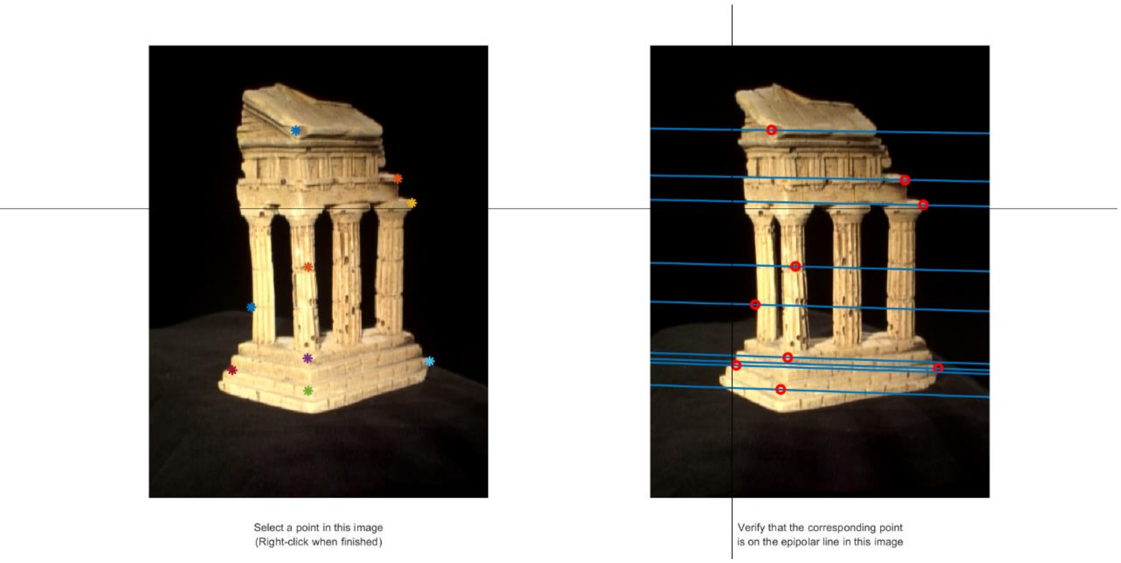
Verify that the corresponding point
is on the epipolar line in this image

Saqib Aziz Dhuka
301338669
CMPT 412 Fall 2020

Project 5

3.1.2:

I utilized a window of size 5 around the corresponding point and then computing the distance for each of those points. I then chose the best point with minimum error and then filled up the returned pts2 variable. The output of some points is shown below. Although it is not shown below, the algorithm fails for some points when the calculation is too fine and it ends up getting the wrong point on the other image since the other image is at a different angle. Also, there is noise which causes minor errors.


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:

Esssential Matrix

```
   0.0052   -0.1315   -0.0318
  -0.2604   -0.0031    1.4784
  -0.0038   -1.4965   -0.0017
```

3.1.4

I chose the correct Projection matrix by running triangulate function on each extrinsic matrix returned by camera2.m. I made sure to multiply both P1 and P2 by K1 and K2 respectively to get the projection matrix. I used a max variable that would count the number of positive (including zero) elements in the Z column and the one with most positive elements is the correct projection matrix. Another condition was also that the extrinsic matrix's Z for 3,3 should also be positive so that viewing the dots on the plot is easier. The second condition was for testTempleCoords where we show the image, rather than for run_3_1_4 where we show the errors. The code for this section can be found in run_3_1_3.m, where triangulate is called. The errors came out to be:

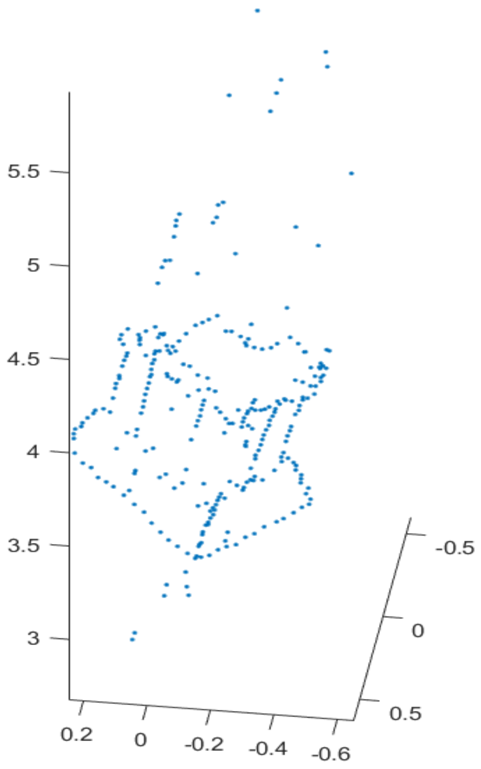
- error 1: 0.2371; error 2: 0.2374

Saqib Aziz Dhuka
301338669
CMPT 412 Fall 2020

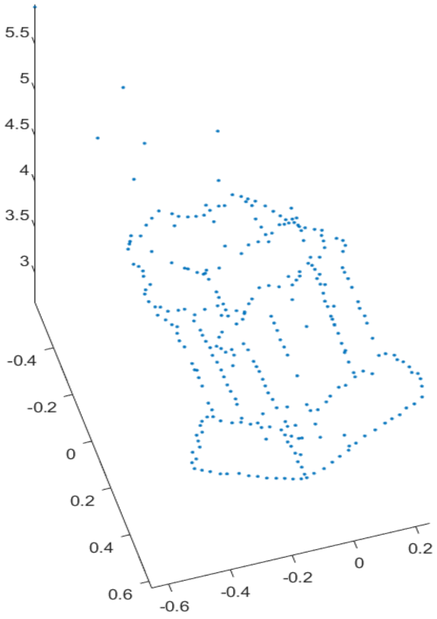
Project 5

3.1.5

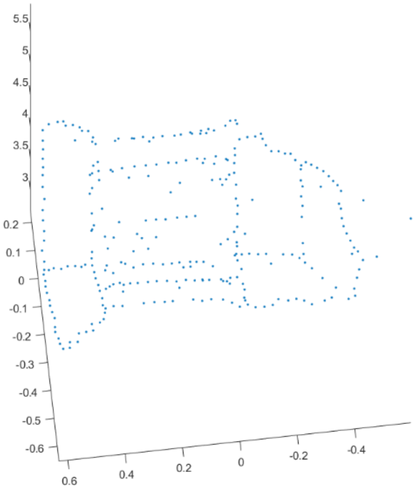
Top-left at an angle



Right side at an angle



Side view

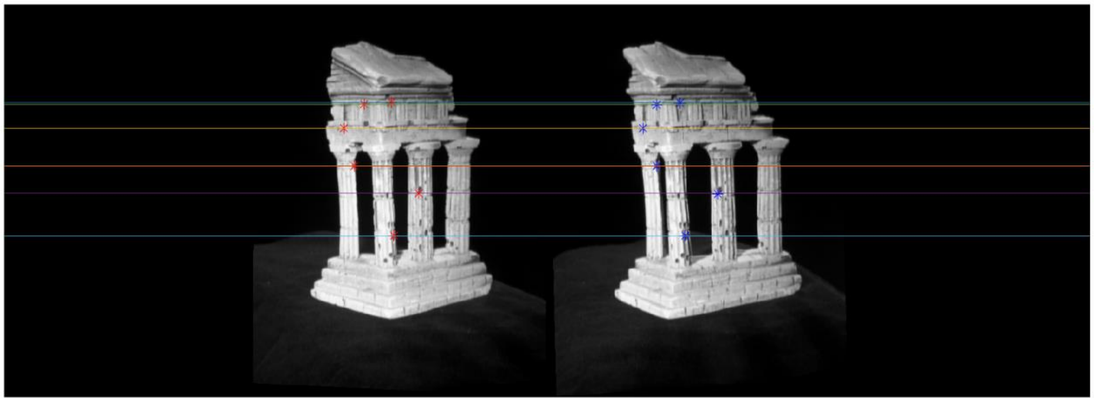


Saqib Aziz Dhuka
301338669
CMPT 412 Fall 2020

Project 5

3.2

3.2.1



3.2.2/3.2.3

Window size was changed to 5. Individual pictures can be found inside results folder.

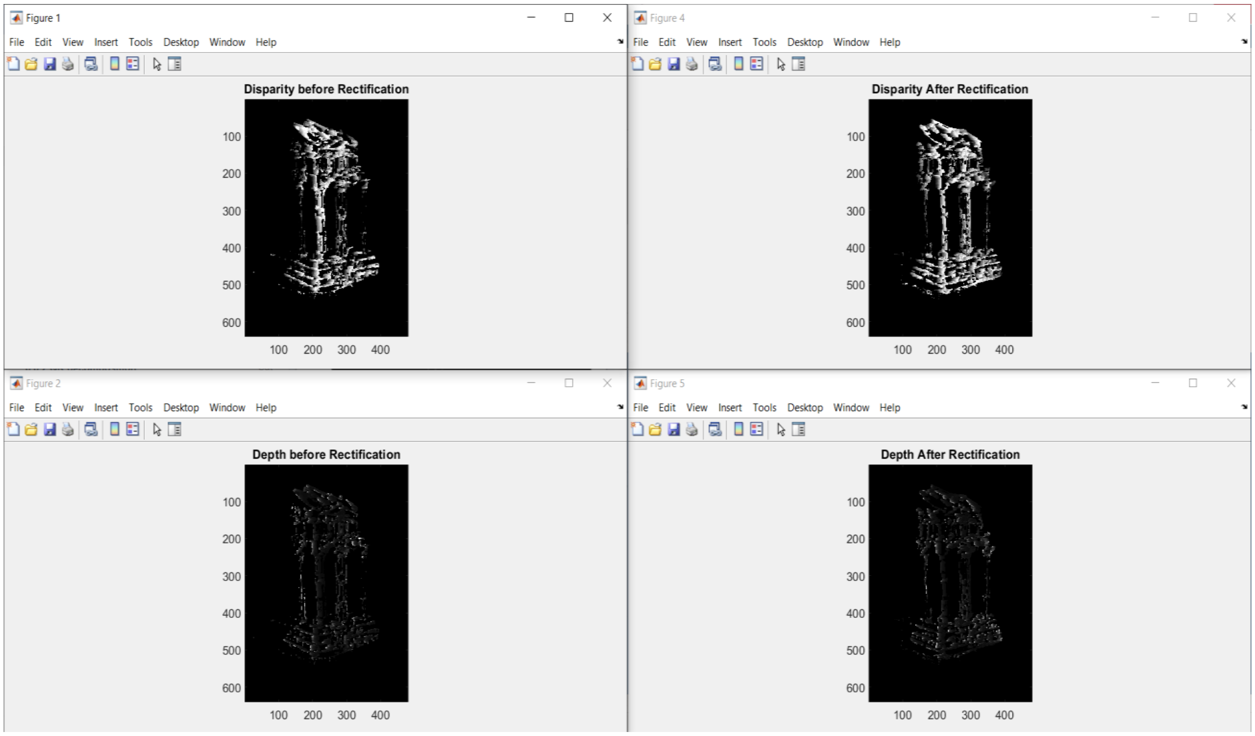


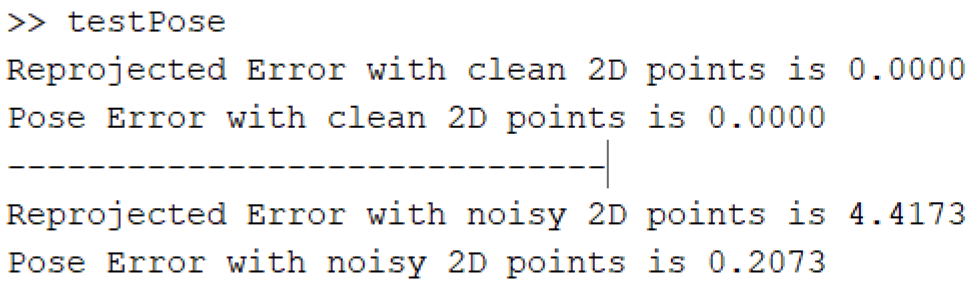
Figure 1
File Edit View Insert Tools Desktop Window Help
Disparity before Rectification
100 200 300 400 500 600
100 200 300 400
Figure 4
File Edit View Insert Tools Desktop Window Help
Disparity After Rectification
100 200 300 400 500 600
100 200 300 400
Figure 2
File Edit View Insert Tools Desktop Window Help
Depth before Rectification
100 200 300 400 500 600
100 200 300 400
Figure 5
File Edit View Insert Tools Desktop Window Help
Depth After Rectification
100 200 300 400 500 600
100 200 300 400

Saqib Aziz Dhuka
301338669
CMPT 412 Fall 2020

Project 5

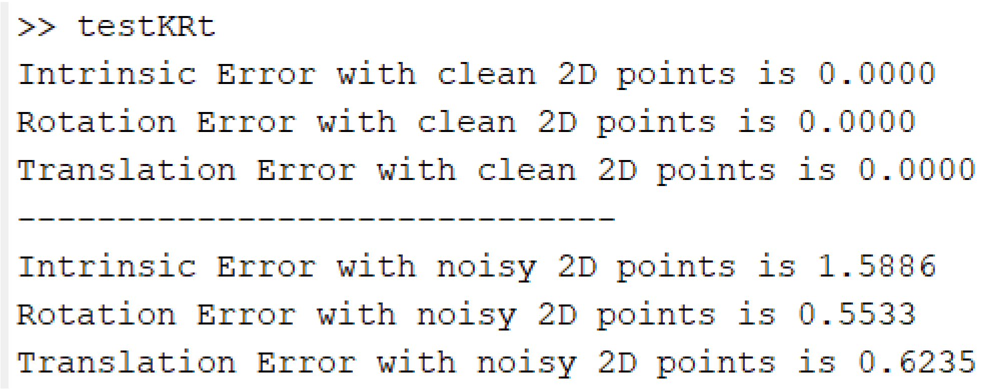
3.3

3.3.1



```
>> testPose
Reprojected Error with clean 2D points is 0.0000
Pose Error with clean 2D points is 0.0000
------------------------------
Reprojected Error with noisy 2D points is 4.4173
Pose Error with noisy 2D points is 0.2073
```

3.3.2

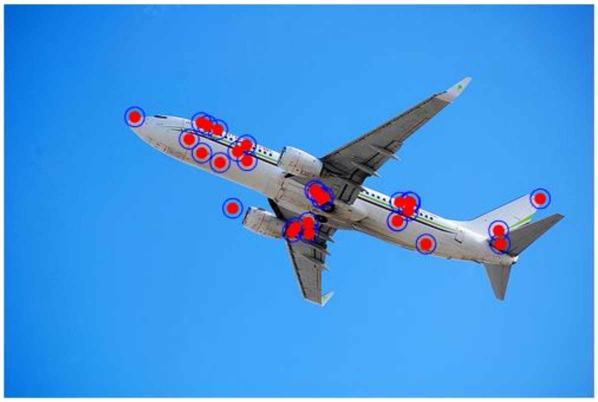


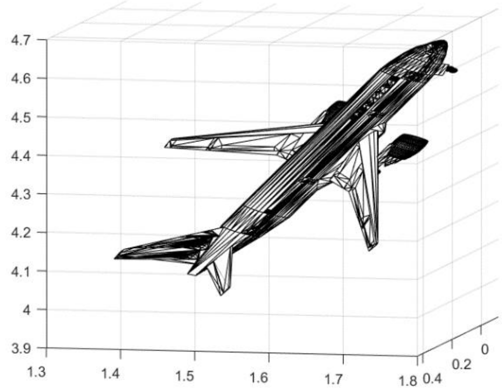
```
>> testKRt
Intrinsic Error with clean 2D points is 0.0000
Rotation Error with clean 2D points is 0.0000
Translation Error with clean 2D points is 0.0000
------------------------------
Intrinsic Error with noisy 2D points is 1.5886
Rotation Error with noisy 2D points is 0.5533
Translation Error with noisy 2D points is 0.6235
```

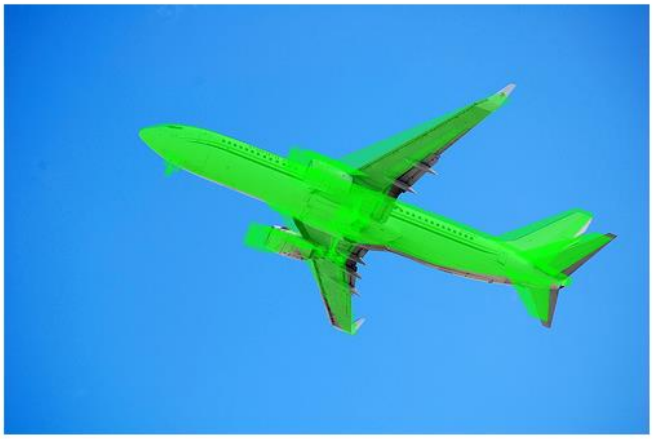
Saqib Aziz Dhuka
301338669
CMPT 412 Fall 2020

Project 5

3.3.3







Color one just for fun: