Project 5 - 3D Reconstruction

Student Name: Wenbin Li

Student ID: 301563887

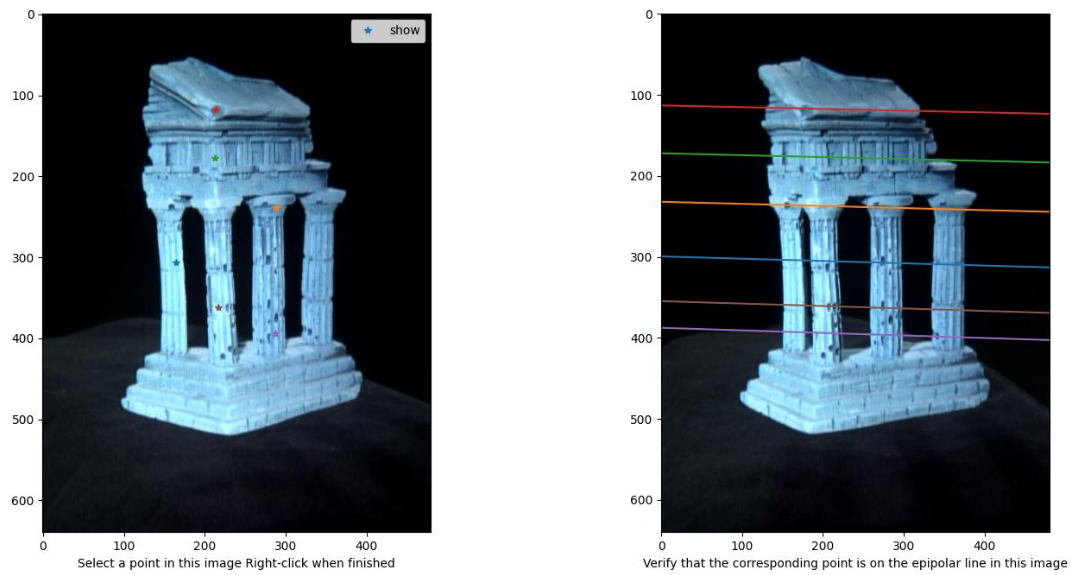
Late day use: 1 day

3.1.1

The Recovered Fundamental Matrix is:

```
PS C:\Users\96552\Desktop\cmptcvlab5\project5_package\python> python eightpoint.py
Estimated Fundamental Matrix (F):
[[ 1.77324726e-09 -1.70876561e-08 -8.77582202e-06]
 [-6.63558809e-08 -4.05296608e-10  4.95571220e-04]
 [ 1.69307908e-05 -4.75954394e-04 -2.06189981e-03]]
PS C:\Users\96552\Desktop\cmptcvlab5\project5_package\python>
```

Following is the visualization of epipolar lines:



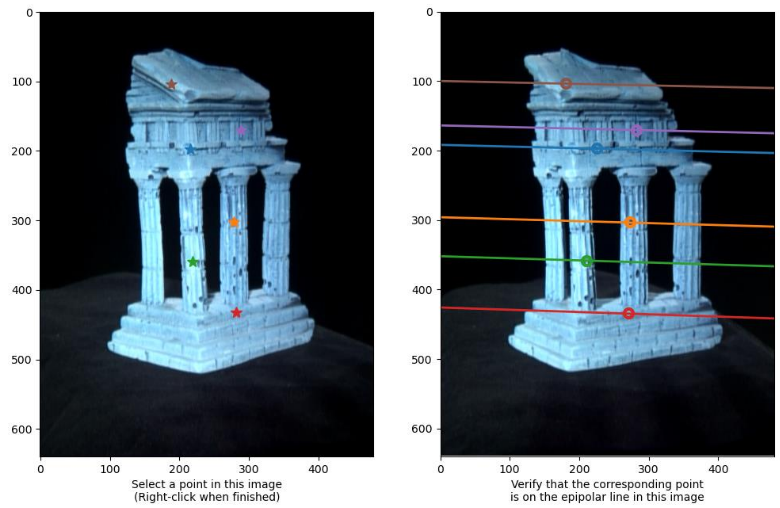
3.1.2

The similarity metric I used is Euclidean distance. This metric is particularly useful when dealing with pixel coordinates in images, as it directly relates to the spatial distance between corresponding points.

The screenshot of the terminal indicates the selected point in Image 1 and Image 2.

```
leslie@Freya python % python3 main.py
Selected Points in Image 1: [[215.71703334 197.21551356]
 [278.42995006 302.11711972]
 [219.13773789 359.12886219]
 [281.85065461 432.10389256]
 [288.69206371 169.84987718]
 [188.35139695 103.7162559 ]]
Corresponding Points in Image 2: [[225 197]
 [273 303]
 [210 358]
 [271 434]
 [282 170]
 [181 103]]
```

The following is the visualization.



The blue pillar and its corresponding point seem to be not well-related. One explanation can be that illuminations are changed within these two images. Variations in lighting between the two images can change the appearance of the same point, making it difficult to match based on intensity alone.

3.1.3

The estimated essential matrix is:

```
PS C:\Users\96552\Desktop\cmptcvlab5\project5_package\python> python main.py
Computed Essential Matrix (E):
[[ 0.00409907 -0.03964299 -0.01894139]
 [-0.15394421 -0.00094368  0.72542881]
 [ 0.00165055 -0.73429419 -0.00084402]]
```

3.1.4

The correct extrinsic matrix was identified through a two-step verification process.

The correct P2(matrix) should triangulate the points such that the majority of them lie in front of both cameras (positive z-values in both camera coordinates).

Additionally, the correct P2 should minimize the reprojection error, i.e., when the 3D points are projected back onto the images, the 2D points should be as close as possible to the original pts1 and pts2, which can suggest the matrix that yielded the lowest mean Euclidean error.

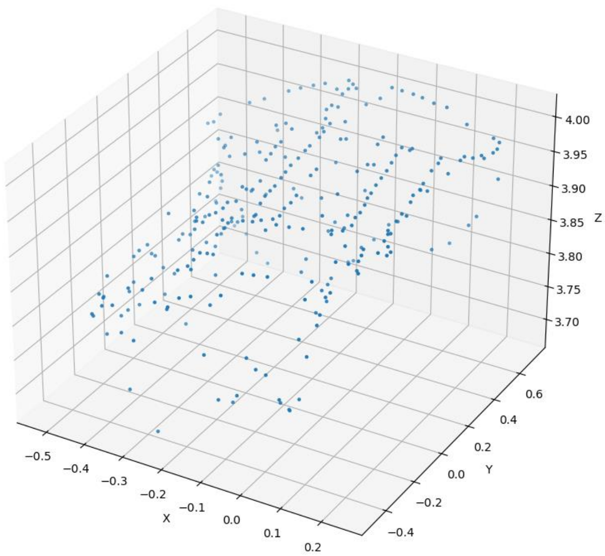
This matrix not only satisfied the geometric constraint of the camera's perspective but also provided the best fit to the observed data.

The reprojection error calculated using mean Euclidean distance is respectively shown in the following screenshot.

```
PS C:\Users\96552\Desktop\cmptcvlab5\project5_package\python> python test_error.py
Best error for points in image 1: 0.5013867435038208
Best error for points in image 2: 0.5055608095300058
```

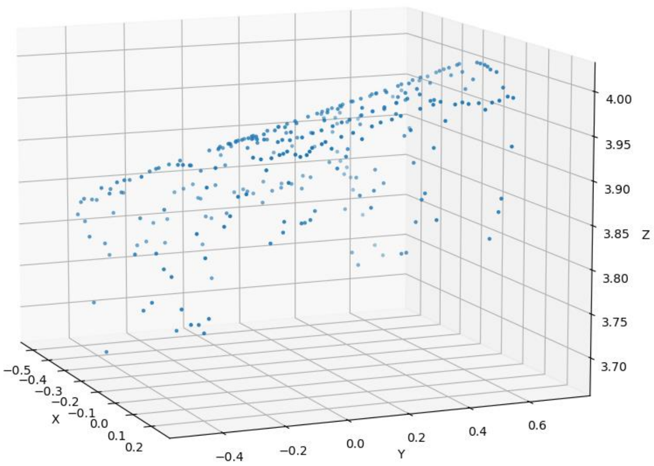
3.1.5

Front view

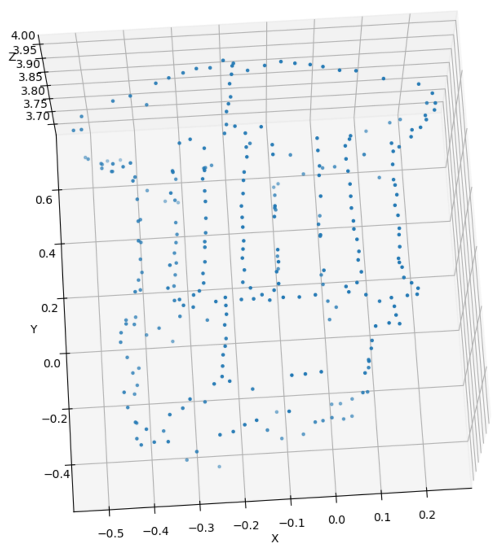


WENBIN LI
SFUID: 301563887

Side View

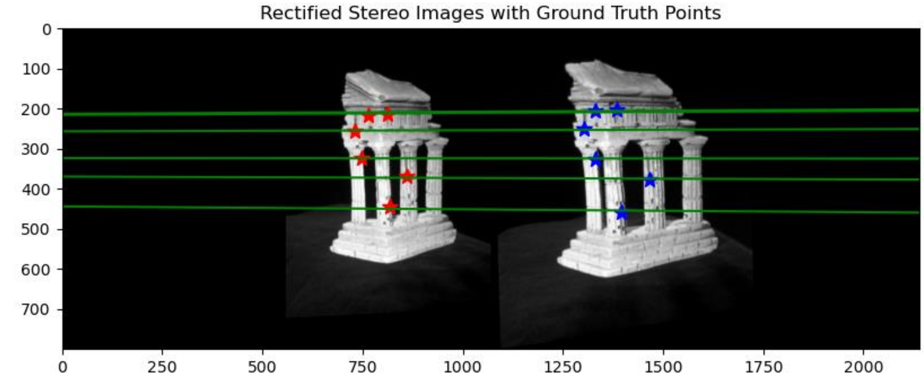


Bottom-up view



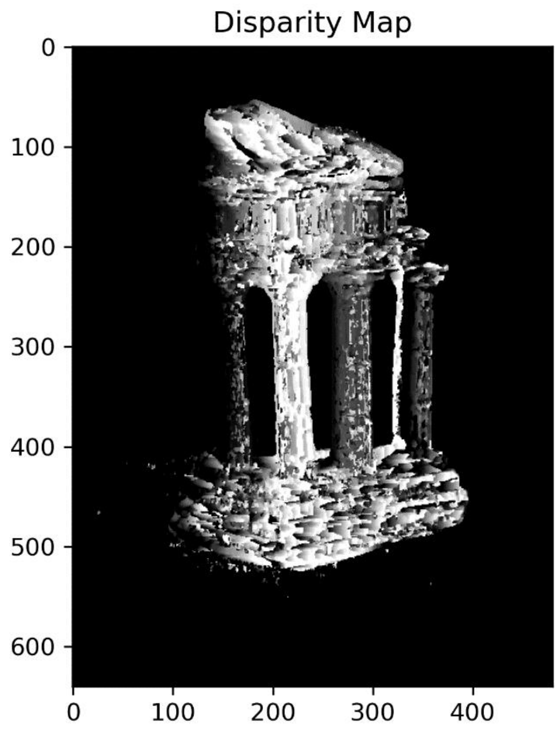
WENBIN LI
SFUID: 301563887

3.2.1



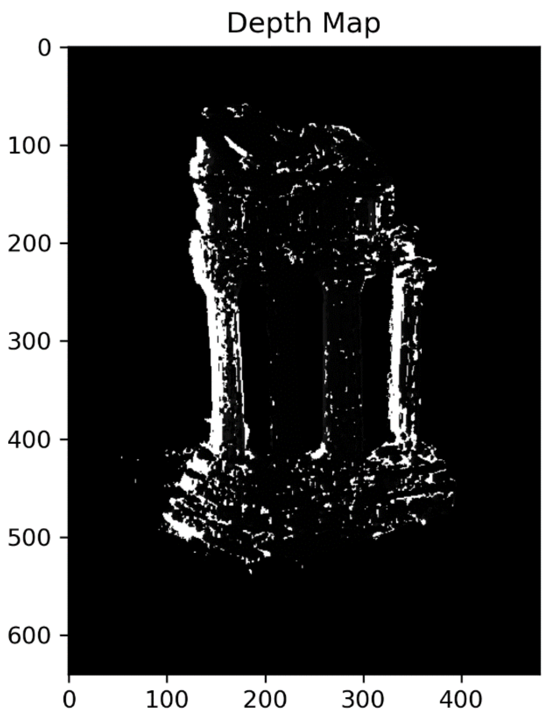
3.2.2

Disparity Map



3.2.3

Depth Map



3.3.1

```
PS C:\Users\96552\Desktop\cmptcvlab5\project5_package\python> python testPose.py
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 3.1034
Pose Error with noisy 2D points is 0.1454
```

Reprojected Error: When dealing with noisy 2D points, a reprojected error of 3.1034 is not unusual. Noise in the data can come from measurement inaccuracies, quantization error in pixel values, etc. This error indicates that there is some discrepancy between the estimated 2D points and the original noisy points, which is expected.

Pose Error: A pose error of 0.1454 with noisy 2D points is relatively low. It suggests the pose estimation is quite robust to noise, although there's a slight deviation from the true pose.

3.3.2

```
PS C:\Users\96552\Desktop\cmptcvlab5\project5_package\python> python testKRt.py
Intrinsic Error with clean 2D points is 0.0000
Rotation Error with clean 2D points is 2.7901
Translation Error with clean 2D points is 2.6131
------------------------------
Intrinsic Error with noisy 2D points is 3.0328
Rotation Error with noisy 2D points is 2.0264
Translation Error with noisy 2D points is 2.0590
```

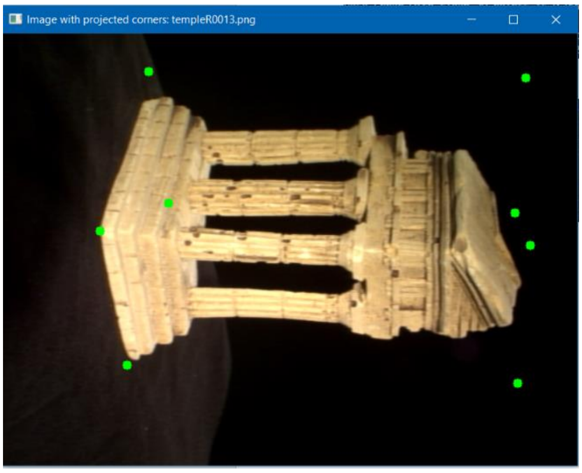
General insights for testKRt errors:

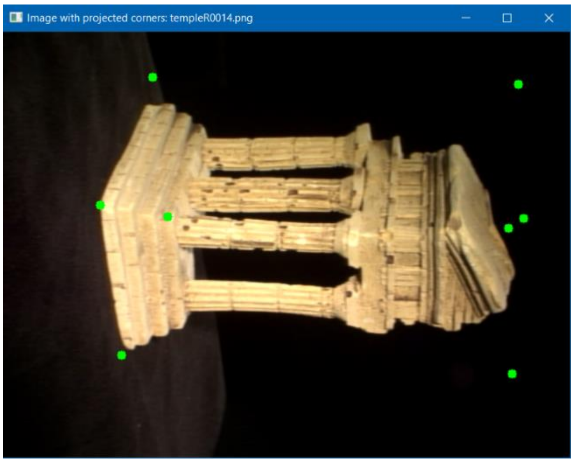
It's clear that the presence of noise affects the errors. This is expected in any real-world scenario.

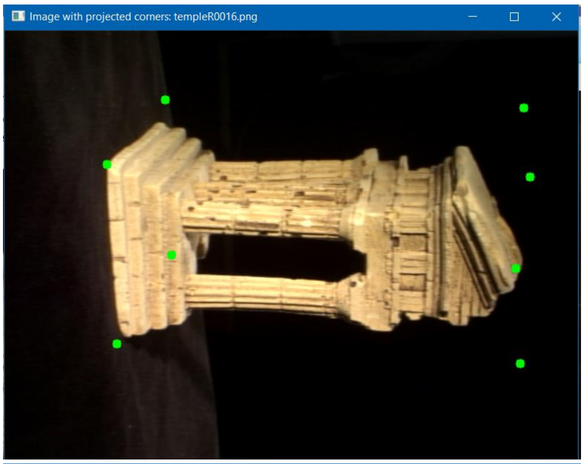
The key is that the errors should be within a reasonable range, which seems to be the case here.

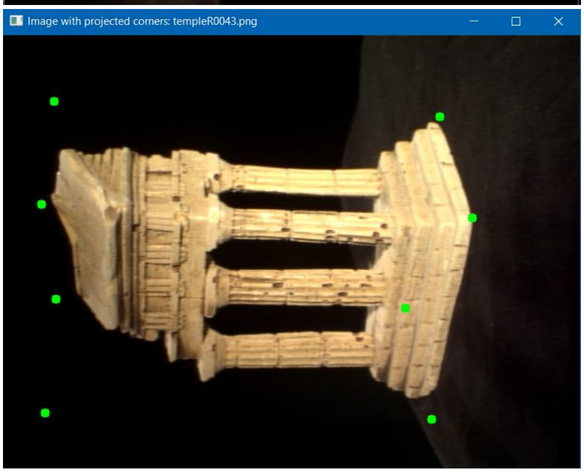
3.4.1

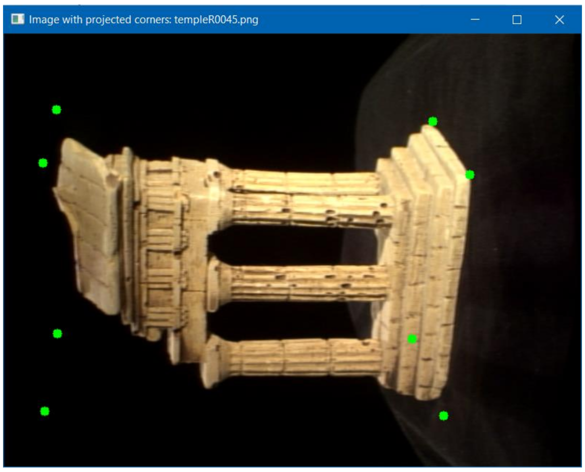
By creating bounding box over three axes, I successfully project the corners to all the images and here are the visualizations.











3.4.2

The following is the function I computed for getting the depth map:

read_camera_parameters(file_path): Reads camera parameters from a *_par.txt file and returns a dictionary with parameters for each image.

camera_params: Use the function to populate the camera_params dictionary with parameters for each image.

construct_projection_matrix(k_values, r_values, t_values): Constructs a 3x4 projection matrix from intrinsic and extrinsic camera parameters.

load_images(image_file_names): Loads images given a list of image filenames and returns a dictionary of image data.

project_point(point_3d, p_matrix): Projects a 3D point to 2D space using a projection matrix.

compute_3d_point(x, y, depth, p_matrix): Computes the 3D coordinates for a point at a given depth and pixel coordinates in the reference image.

within_bounds(point, shape): Checks if a point is within the bounds of an image shape.

compute_consistency(i0, i1, x, p_matrix_i0, p_matrix_i1): Computes the consistency score between two images for a set of 3D points.

compute_depth_map(ref_image, other_images, p_matrices, bbox_corners, depth_range): Main function that orchestrates the computation of the depth map for the reference image.

NCC.compute(c0, c1): Computes the normalized cross-correlation between two sets of pixel color data.

Due to the time limitation, I only get the best_depth and best_score as -0.09194 and 0.99546284395141 respectively.

The depth map visualization is yet to be complete.