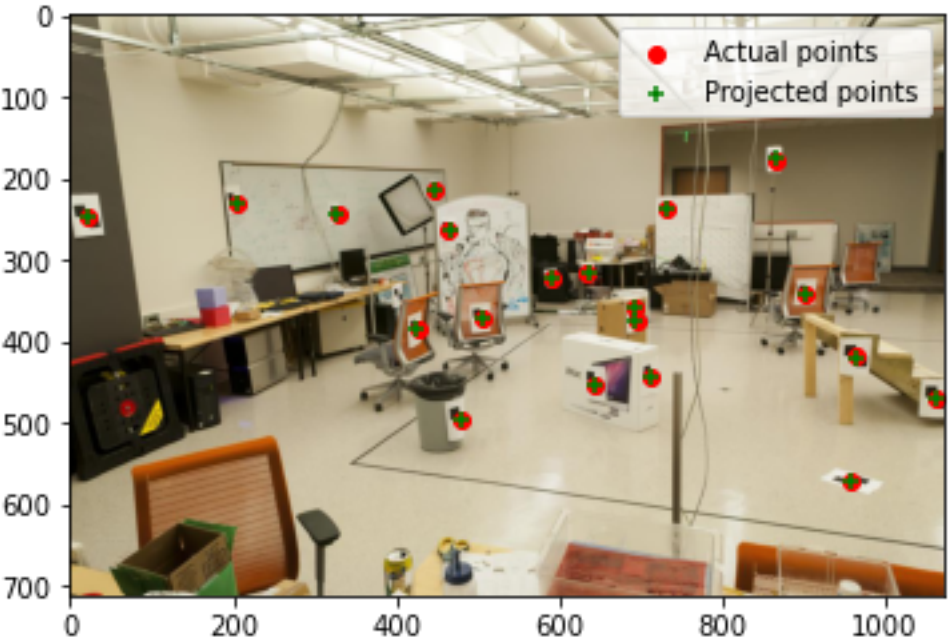
CS 6476 Project 4

Sandeep Dasari
sdasari38@gatech.edu
903540744

Part 1: Projection Matrix



The total residual is 14.711455

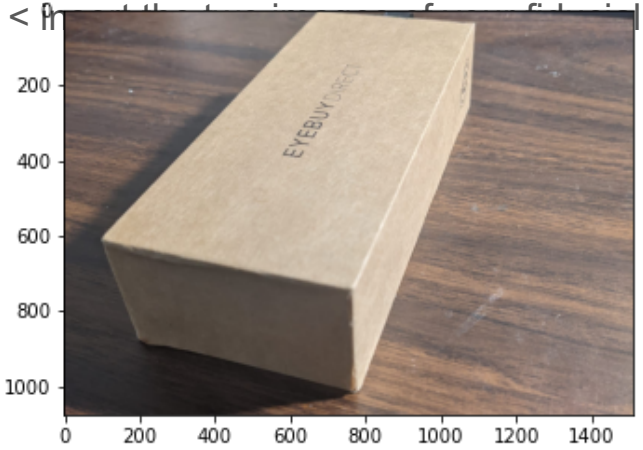
Part 1: Projection Matrix

At least how many 3D-2D point correspondences do you need to estimate the projection matrix? Why?

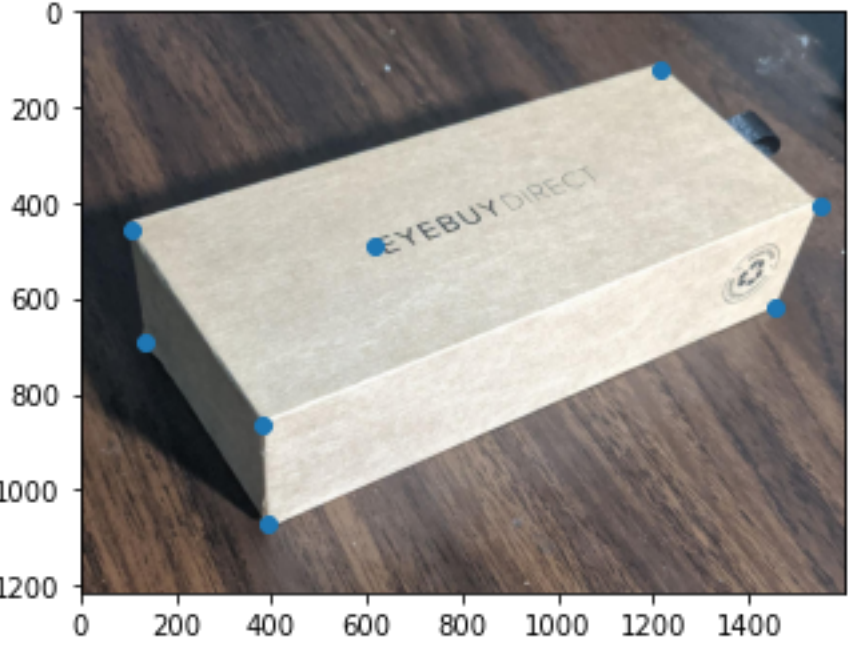
Projection matrix P of size 3 x 4 has 11 degrees of freedom since we fix P34 = 1. Each 3D-2D point correspondence gives us 2 equations while we need 11 equations to solve for 11 DOF. Thus we need atleast 6 3D-2D point correspondences to estimate the projection matrix.

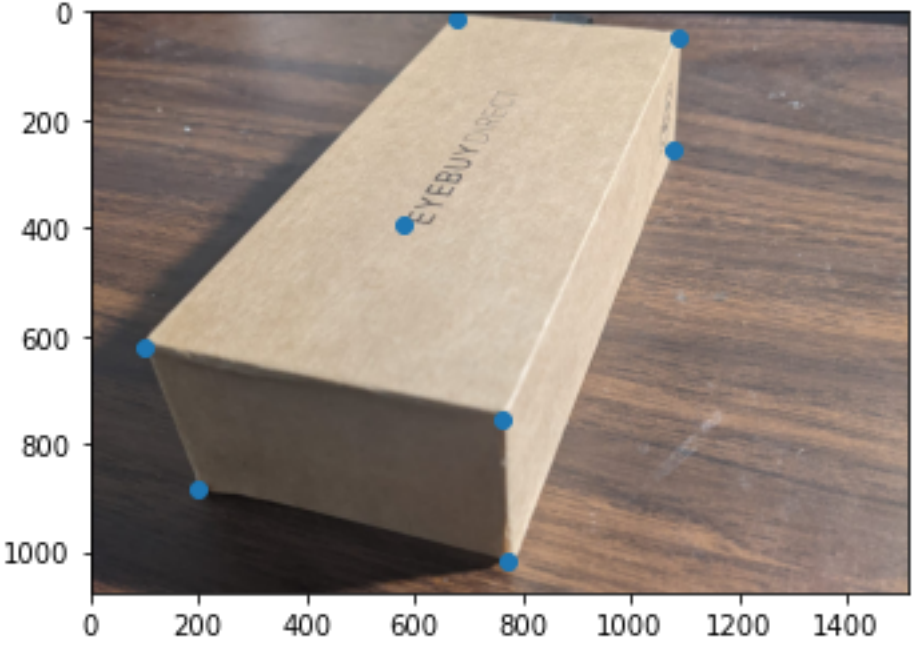
Part 2: Estimation with Your Own Images

< I object here>

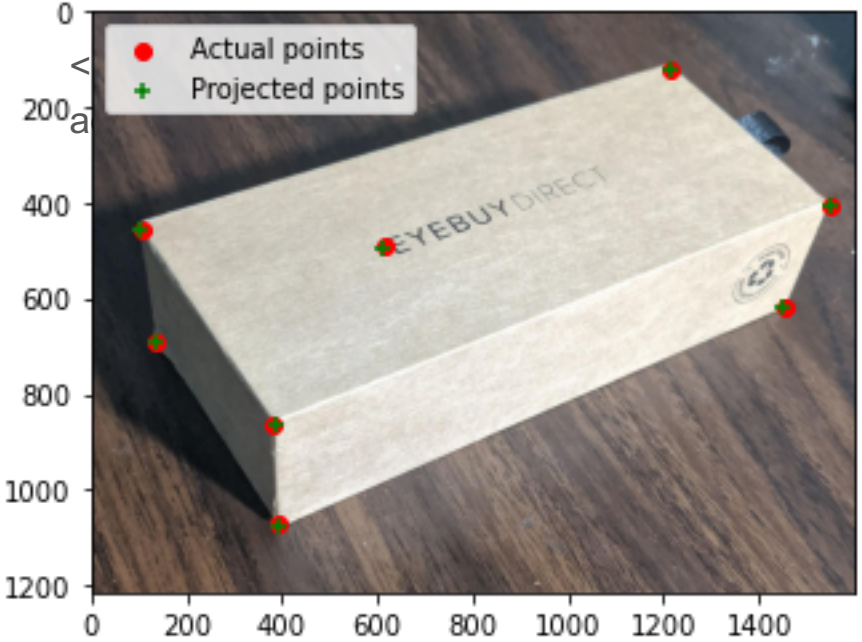


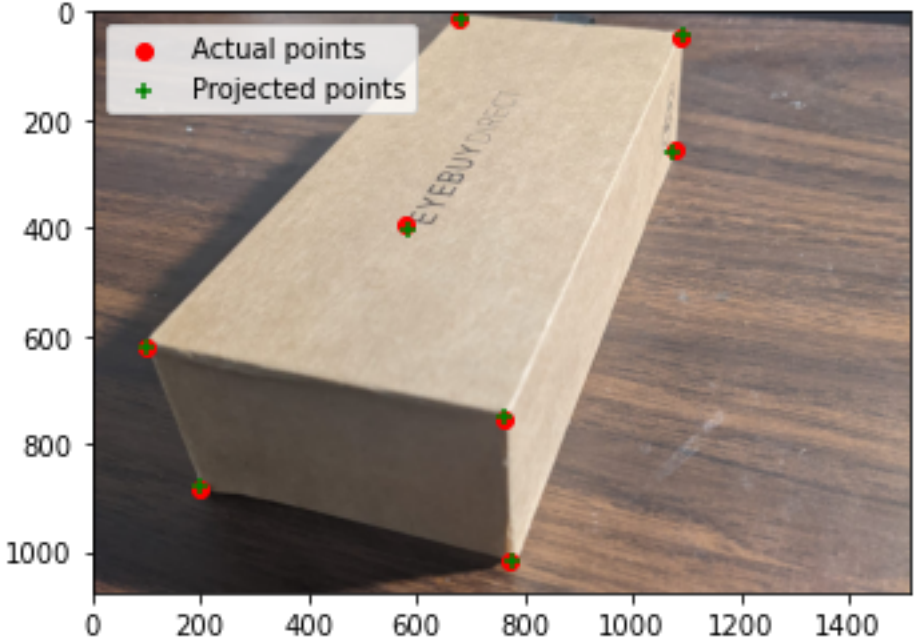
Part 2: Estimation with Your Own Images



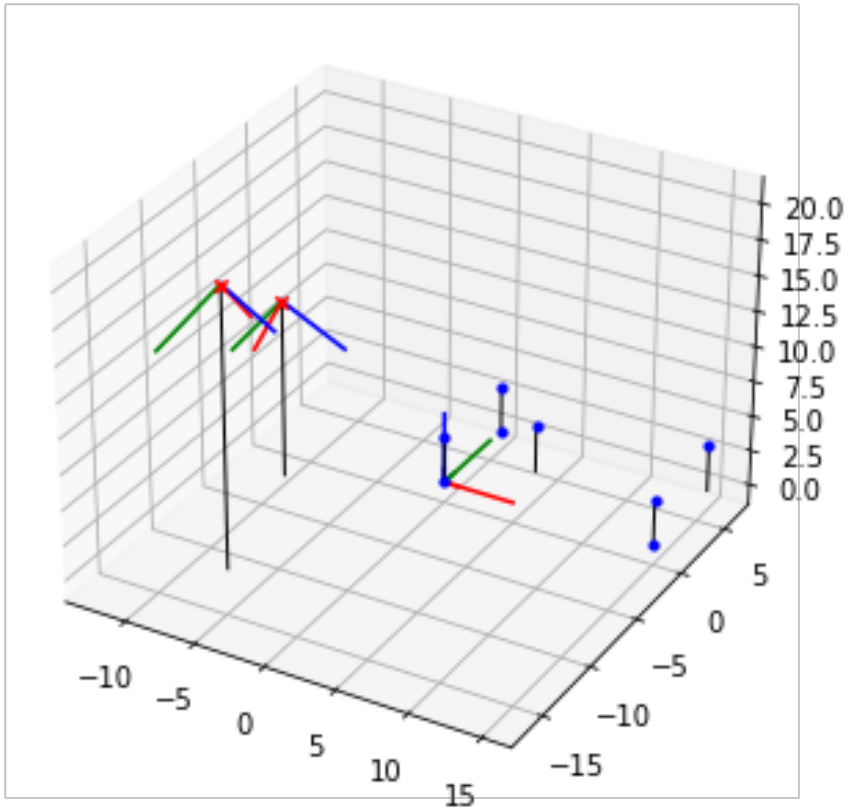


Part 2: Estimation with Your Own Images





Part 2: Estimation with Your Own Images



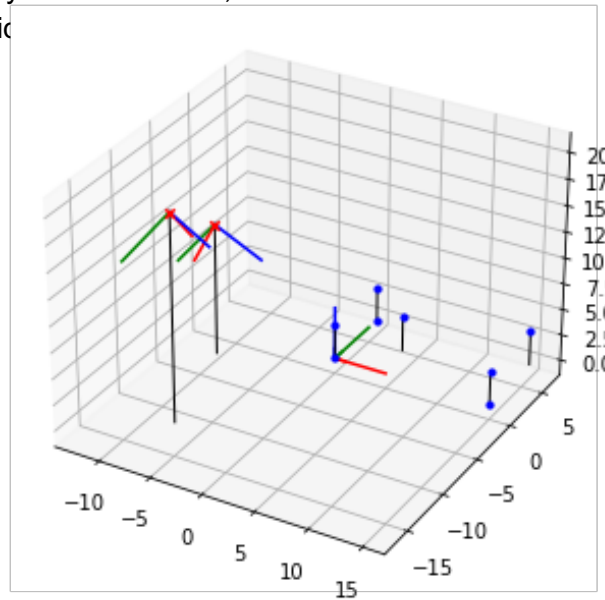
Part 2: Estimation with Your Own Images

- What would happen to the projected points if you increased/decreased the x coordinate, or the other coordinates of the camera center t? Write down a description of your expectations in the appropriate part of your writeup submission.

Upon changing our initial guess of camera center t from the given values, the least squares optimization algorithm has to work harder to optimize the projection matrix but we should still see the same poses of the cameras at the end of our experiment

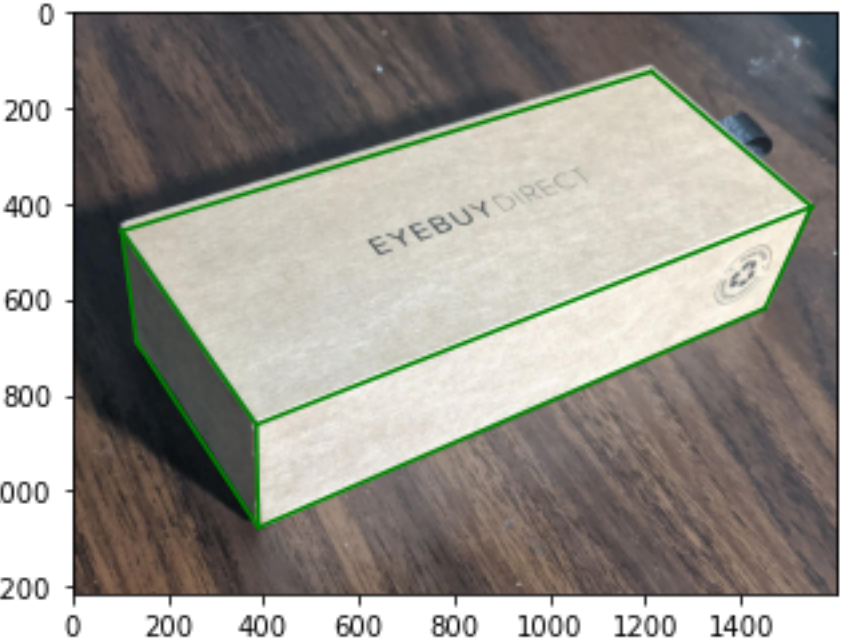
- Perform this shift for each of the camera coordinates and then recompose the projection matrix and visualize the result in the Jupyter notebook. Was the visualized result what you expected?

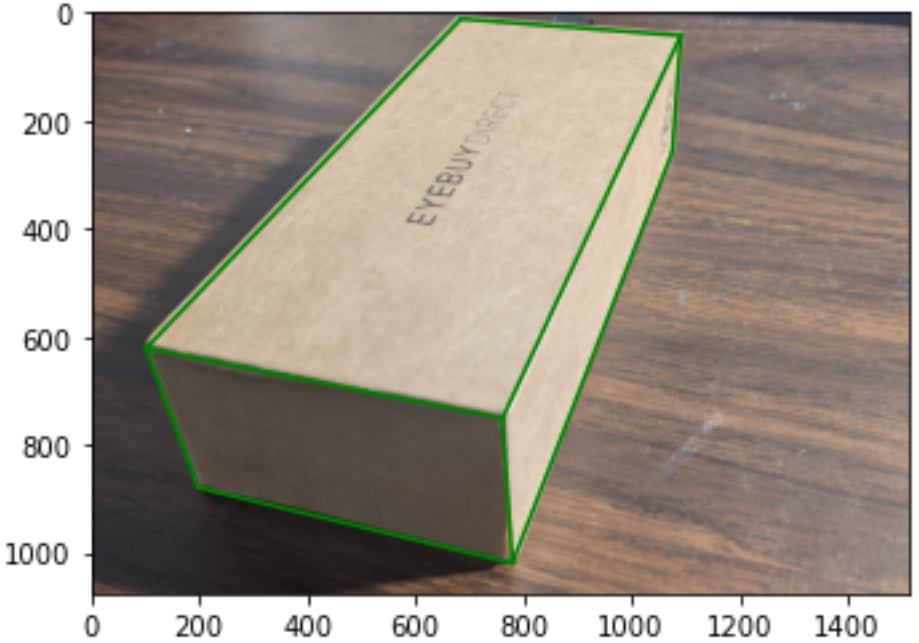
Yes, upon tweaking the camera center by multiples of 10 in all x, y and z directions, the final result was the same as previous slic



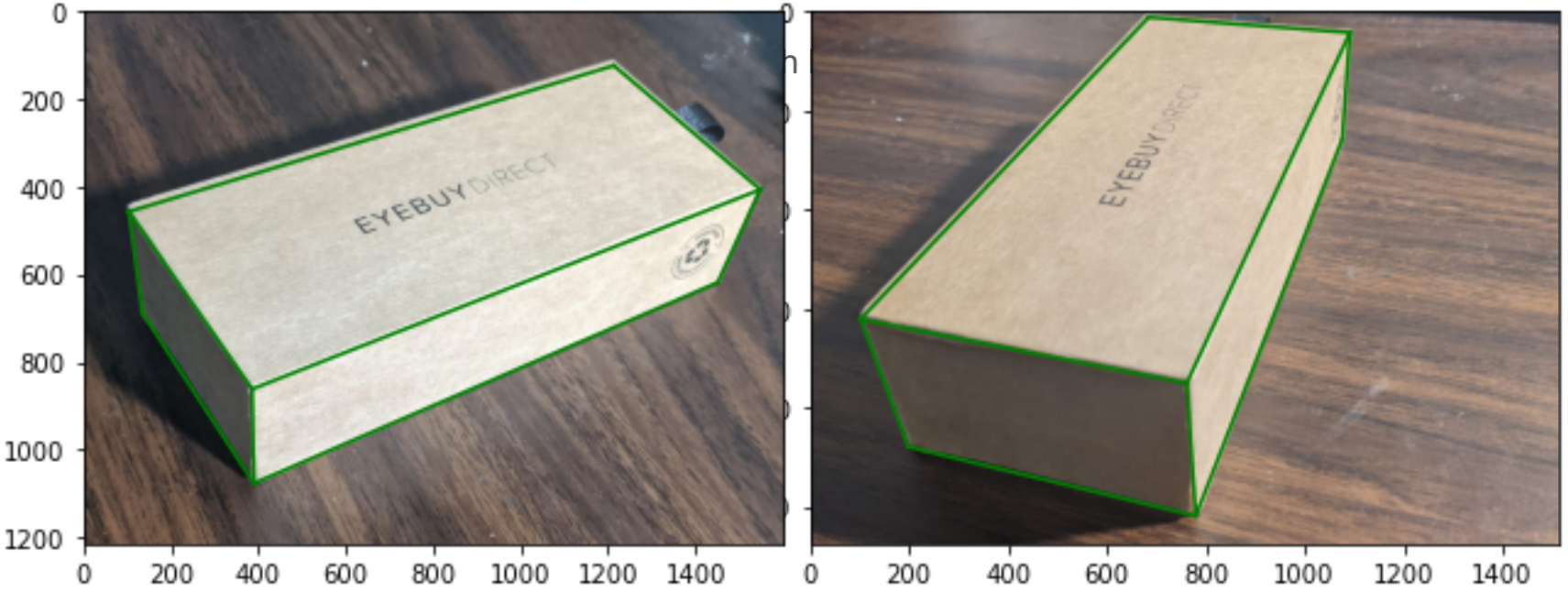
Part 2: Estimation with Your Own Images

<insert both images you take with bounding boxes around your fiducial object>





Part 3: Camera Coordinates



Part 3: Camera Coordinates

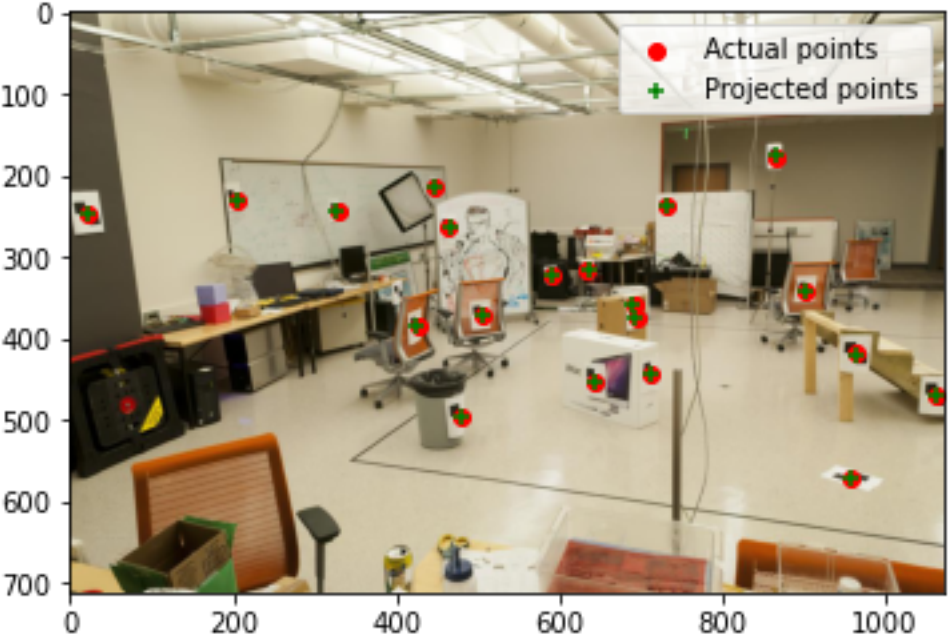
The first three rows of the transformation matrix are just the extrinsic parameters of our camera, why is that?

Since we are simply transforming from world coordinates to camera coordinates both in homogenous coordinates with no scaling or calibration, the first 3 rows are just extrinsic parameters of the camera.

Say if we know the coordinates of 3D points in camera 1 coordinate system, and we want to transform them into camera 2 coordinate system, what should we do? You may assume that the poses of both cameras are known.

In order to transform 3D coordinates of camera tc1 to camera tc2, we first subtract off tc2 (which is a point in tc1's coordinate system). Since we also know the poses, we can perform the appropriate rotation and translation of axes to align with the axes of the camera 2. ie. Z axis of camera 2 points towards camera 1.

Part 4: DLT


Actual points
Projected points
0
100
200
300
400
500
600
700
0
200
400
600
800
1000

Part 4: DLT

Why do we need the cross product trick?

The cross product trick suddenly allows us to convert the eq

x = PX into a homogenous linear system of equations instead. which allows us to solve for the 11 DOF in P directly using eigen decomposition. This is possible because our vectors are in the same direction!

Ie. x **X** PX = 0

We pick up the eigenvector with the smallest eigenvalue. Will this eigenvalue be exactly zero? Why/why not?

We try to solve the homogenous system Ap = 0 by getting the eigenvector with the smallest eigenvalue as the 0 eigenvalue is not possible. This is due to

Measurement error: The measurements are not entirely accurate in terms of cms (or whatever distance metric) to pixels in the image.

EC: RANSAC Iterations Questions

How many RANSAC iterations would we need to find the fundamental matrix with 99.9% certainty from your Mount Rushmore and Notre Dame SIFTNet results assuming that they had a 90% point correspondence accuracy?

One might imagine that if we had more than 9 point correspondences, it would be better to use more of them to solve for the fundamental matrix. Investigate this by finding the number of RANSAC iterations you would need to run with 18 points.

If our dataset had a lower point correspondence accuracy, say 70%, what is the minimum number of iterations needed to find the fundamental matrix with 99.9% certainty?

EC: RANSAC

<insert the number of inliers from your best model here>

<insert the image and total residual here>

Tests

```
(proj4) 2012s-MacBook-Pro:proj4_release_v0 a2012$ pytest unit_tests/
============================================================ test session starts ============================================================
platform darwin -- Python 3.6.12, pytest-6.1.1, py-1.9.0, pluggy-0.13.1
rootdir: /Users/a2012/Desktop/cv/proj4_release_v0
collected 11 items

unit_tests/part1_unit_test.py .....                                                                                                   [ 45%]
unit_tests/part3_unit_test.py ...                                                                                                     [ 72%]
unit_tests/test_dlt.py ..                                                                                                             [ 90%]
unit_tests/test_ransac.py F                                                                                                           [100%]

================================================================= FAILURES ==================================================================
_____________________________________________ test_calculate_num_ransac_iterations _____________________________________________
```

Conclusions

Learnt:

1. Projection from 3-d to 2-d (multiple applications in VR, 3D reconstruction, game programming, I just found this during this project!)
2. Simple to implement for cubes. Optimization algorithms least_squares and SVD

The challenges in this assignment were surprisingly in the understanding as the coding part was fairly straightforward.