Homework 4:

I have implemented the Non-Coplanar camera calibration method to estimate the intrinsic and extrinsic camera parameters given the correspondence between image and world points.

Approach:

1. Estimate the projection matrix M given the correspondence between image and world points. We use Singular Value Decomposition for this:
   $A = U D V^T$

   And, we get the solution as the column of V corresponding to 0 eigen value. So it is the last column of V as it is sorted by value of the eigen value.
2. Then we solve for the camera parameters by breaking down the estimated projection matrix into K*, R* and T*. We use the fact that the rows are orthonormal vectors in the rotation matrix. We take the dot product of the equations we got, which in turn eliminated some unknowns as their dot product is 0. And similarly, we use the cross product too, to change the vectors in the equations.

   In the code, I have directly used the final equations derived and given to find the parameters.

I have created the csv file mapping image points and world points as given on the course website.

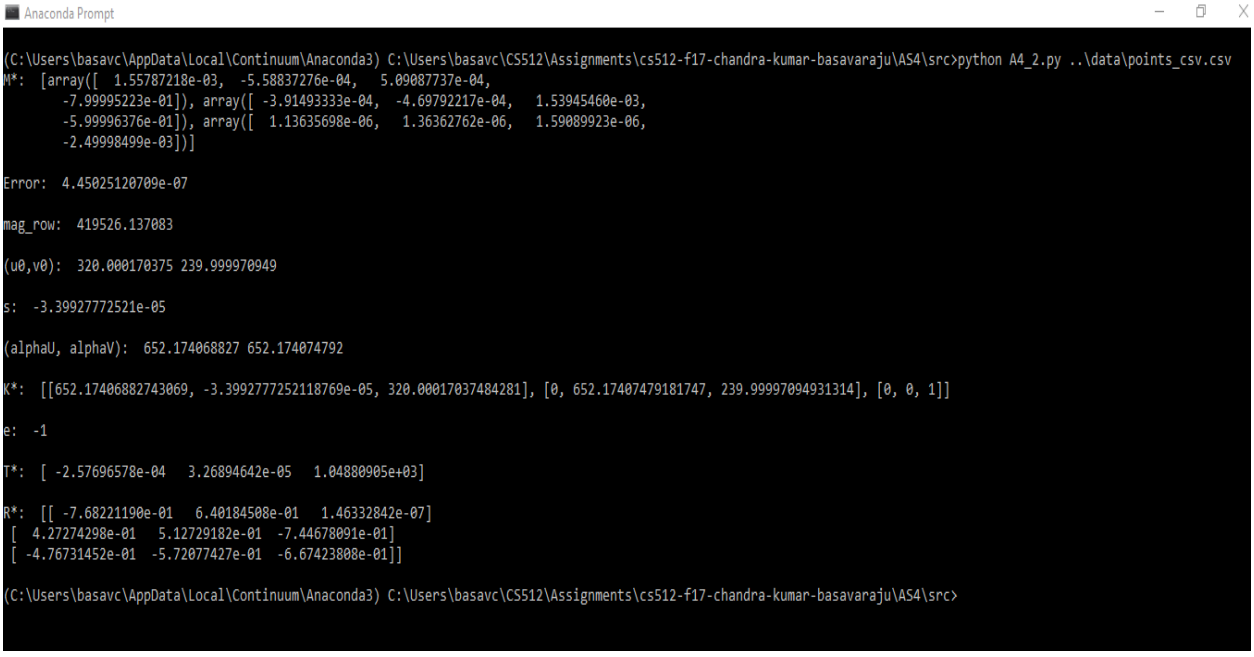
1. points_csv.csv for the calibration data
2. Noisy1.csv for the first noise data
3. Noisy2.csv for the second noise data

To execute the code, use below command from the command line:

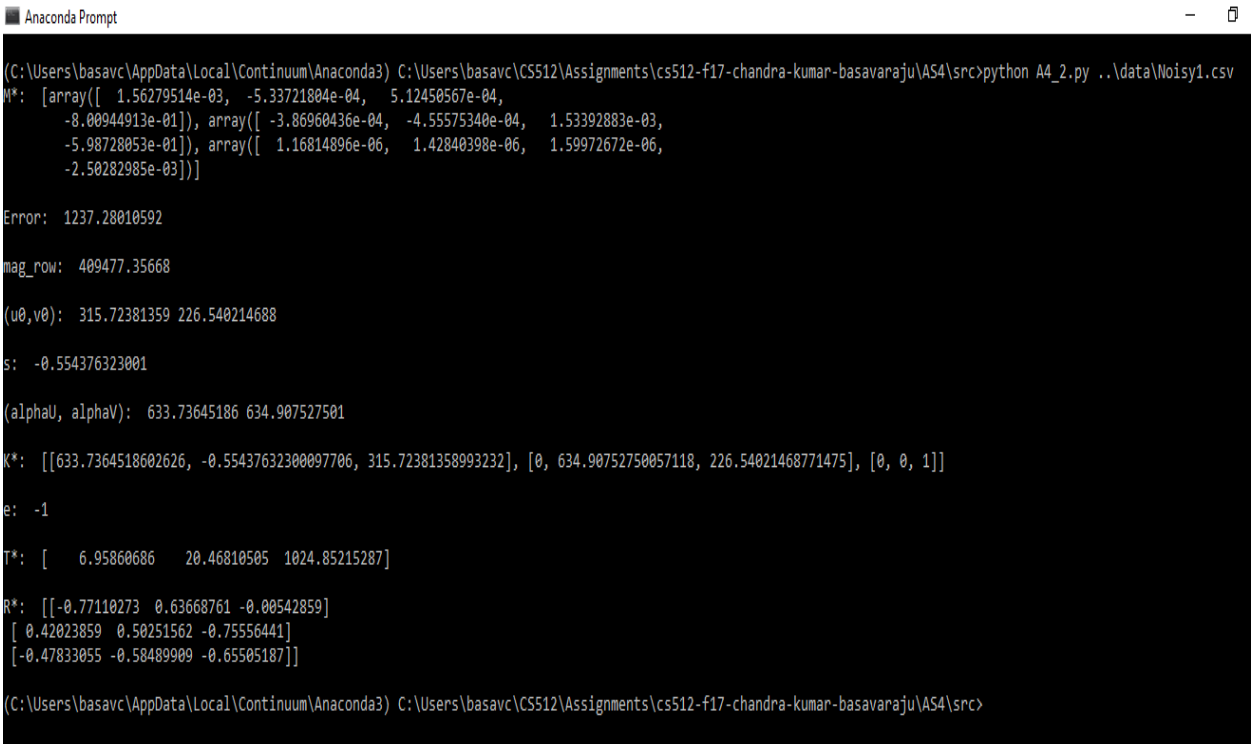
python A4_2.py [path for the csv file holding the data]

if file name is not specified while executing, user id prompted to enter the csv data file path

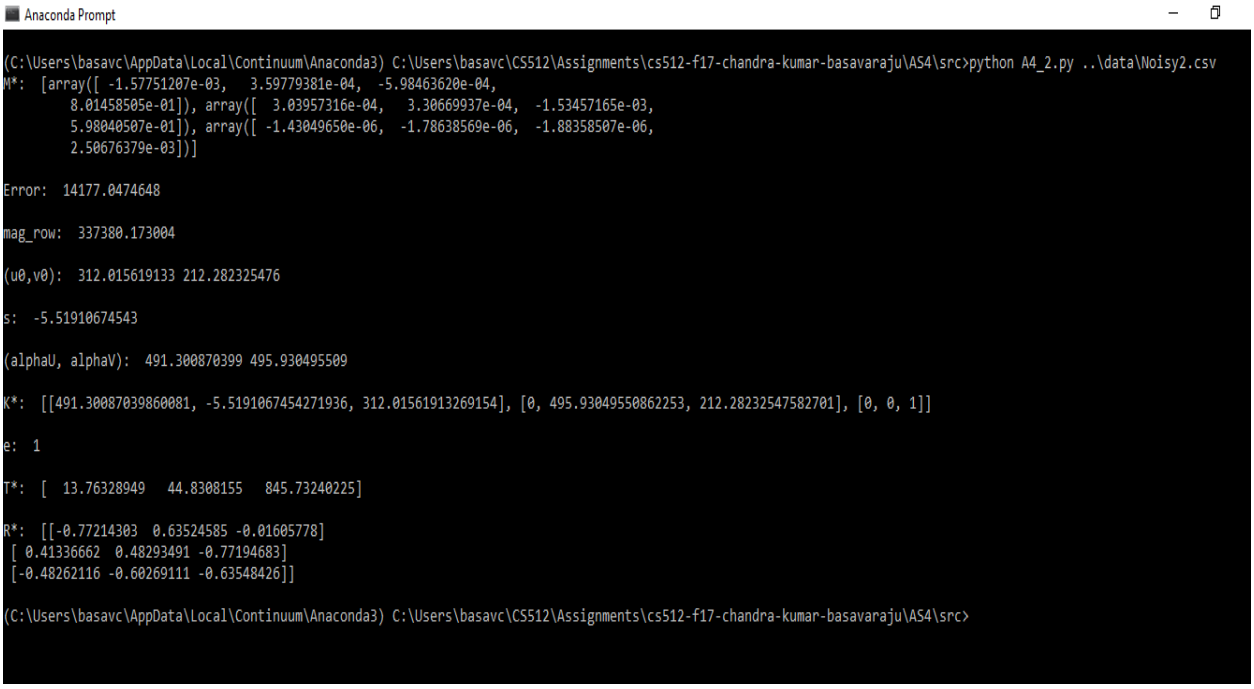
Below is my output of camera calibration i.e the intrinsic and extrinsic camera parameters for the calibration data:



Below is my output of camera calibration i.e the intrinsic and extrinsic camera parameters for the first noise data:



Below is my output of camera calibration i.e the intrinsic and extrinsic camera parameters for the second noise data:



To overcome the problem with outliers/noise, we do robust estimation to find the best date to build the model.

RANSAC (Random Sample Consensus) is one of the robust estimation technique.

The basic idea of RANSAC is to choose a fixed number(n) of points from the data to fit the model, this number is chosen to be close to the minimum number(d) of points required to fit the model.

We fit the model using the chosen points.

Then, using the fit model we find the inliers from the whole data.

And, if there are more inliers than d, we recomputed the model using all these inliers.

We do this process k (number of trials, user specified) times and choose the model which has the most inliers.

The algorithm has a way to re estimate the k value.

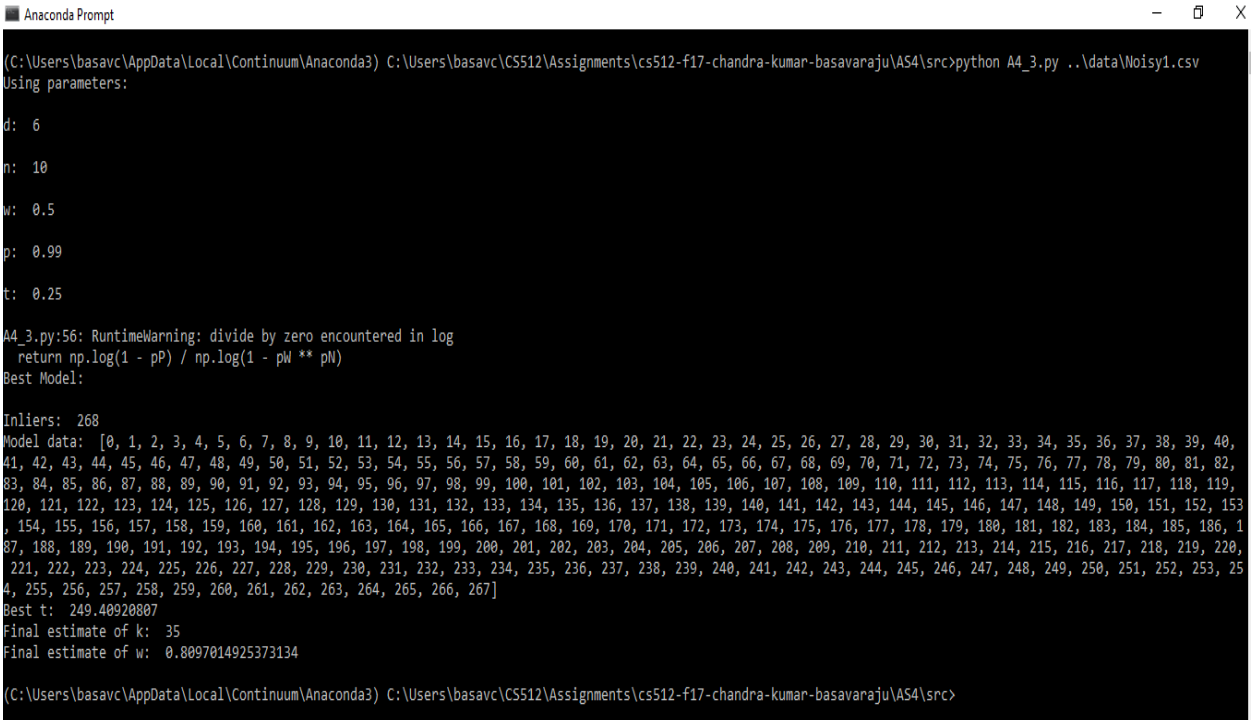
The idea behind choosing just over the required points is that there is less chance to pick outliers and we could be lucky to pick the sample with no outliers.

Below are the sample ouputs from my implementation for the data provided to us:

Without noise:

```
(C:\Users\basavc\AppData\Local\Continuum\Anaconda3) C:\Users\basavc\CS512\Assignments\cs512-f17-chandra-kumar-basavaraju\AS4\src>python A4_3.py ..\data\points_csv.csv
Using parameters:

d:  6

n:  10

w:  0.5

p:  0.99

t:  0.25

Best Model:

Inliers:  250
Model data:  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82,
83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119,
120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153
, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 1
87, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 209, 210, 211, 212, 213, 214, 215, 216, 217, 220, 221, 222, 223,
 224, 225, 226, 227, 231, 232, 233, 234, 235, 236, 237, 241, 242, 243, 244, 245, 246, 247, 252, 253, 254, 255, 256, 257, 263, 264, 265, 266, 267]
Best t:  250.515452754
Final estimate of k:  53
Final estimate of w:  0.7798507462686567

(C:\Users\basavc\AppData\Local\Continuum\Anaconda3) C:\Users\basavc\CS512\Assignments\cs512-f17-chandra-kumar-basavaraju\AS4\src>
```

first noise data:



Second noise data: