Homework Assignment 4

Abhay Gupta (Andrew Id: abhayg)

November 11, 2018

1 Theory

1.1

Consider the fundamental matrix F_{3*3} given by $F = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$

The property of the fundamental matrix is that $x_1 T F x_1 = 0$. Here $x_1^T = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$, since the principal point is at the origin.

$$\begin{aligned} x_1^T F x_1 &= 0 \\ \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} &= 0 \\ \begin{bmatrix} f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} &= 0 \\ f_{33} &= 0 \end{aligned}$$

1.2

In the calibrated case, assume the translation vector is given by $t = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$ and since the translation is parallel to the x-axis, we have t_2 and $t_3 = 0$. Since we have pure translation, we also have that the rotation matrix R is an identity, that is, $R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Now, the properties of the essential matrix E are given by $x_2^T E x_1 = 0$ and $l_2 = E x_1$, where $E = \bar{T} R$, where $\bar{T}$ is the cross-product matrix of the translation vector, given by $\bar{T} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$. Since t_2 and t_3 are zero (parallel translation to the x-axis), we get, $\bar{T} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -t_1 \\ 0 & t_1 & 0 \end{bmatrix}$. Now,

$$\begin{aligned} E &= \bar{T} R \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -t_1 \\ 0 & t_1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -t_1 \\ 0 & t_1 & 0 \end{bmatrix} \end{aligned}$$

Now using the essential matrix properties, we get,

$$
\begin{aligned}
l_2 &= Ex_1 \\
&= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -t_1 \\ 0 & t_1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0 \\ -t_1 \\ t_1 y_1 \end{bmatrix}
\end{aligned}
$$

Now consider the epipolar line C_2, which is given by, $x_2^T l_2 = 0$, from which we get,

$$
\begin{aligned}
\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -t_1 \\ t_1 y_1 \end{bmatrix} &= 0 \\
-t_1 y_2 + t_1 y_1 &= 0
\end{aligned}
$$

The line C_2 is parallel to the x-axis and vice-versa for the C_1 epipolar line. This also holds for the uncalibrated case. Thus, the epipolar lines in the two cameras are also parallel to the x-axis for pure translation along the x-axis.

1.3

Let X denote the point in the real world and x_i represent the different points in the image. From the properties of the camera extrinsics, we get,

$$
\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = K(R_i \mid t_i) \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}
$$

Consider two points x_1 and x_2 at the two time frames,

$$
\begin{aligned}
\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} &= K(R_1 \mid t_1) \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\
&= K(R_1 \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t_1) \\
\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} &= K(R_2 \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t_2) \\
\implies \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} &= R_2^{-1}(K^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} - t_2)
\end{aligned}
$$

Substituting the last equation in the point equation for the first frame, we get,

$$
\begin{aligned}
\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} &= K\left(R_1 \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t_1\right) \\
&= K\left(R_1 \left(R_2^{-1}(K^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} - t_2)\right) + t_1\right) \\
&= KR_1R_2^{-1}K^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} - KR_1R_2^{-1}t_2 + KR_1t_1
\end{aligned}
$$

Thus, t_{rel} and R_{rel} are given by,

$$
\begin{aligned}
t_{rel} &= -KR_1R_2^{-1}t_2 + KR_1t_1 \\
R_{rel} &= KR_1R_2^{-1}K^{-1}
\end{aligned}
$$

and the fundamental matrix F and essential matrix E are given by,

$$
\begin{aligned}
E &= \bar{t_{rel}}R_{rel} \\
F &= K^{-1}EK \\
&= K^{-1}\bar{t_{rel}}R_{rel}K
\end{aligned}
$$

Here, $\bar{t_{rel}}$ is the cross-product matrix of the translation vector t_{rel}.

1.4

Assume that the fundamental matrix is given by F and the real world coordinates of the image is A and the reflection is given by A'. Also, the corresponding points in the image are given by a and a'. From the properties of camera instrinsics, we get $\lambda_1 a = KA$ and $\lambda_2 a' = KA'$, where K is the instrinsic matrix of the camera.

$A = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$. Since there is only a reflection between A and A', we get that the rotation matrix is identity, that is, $R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and there exists only a translation, which can be described as $T = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$.

Now, using the above definitions, we get,

$$
\begin{aligned}
A' &= RA + T \\
\lambda_2 K^{-1}a' &= \lambda_1 RK^{-1}a + T \\
\lambda_2 K^{-1}\bar{T}a' &= \lambda_1 RK^{-1}\bar{T}a + T\bar{T}
\end{aligned}
$$

where $\bar{T}$ means taking the cross-product with T and $T\bar{T} = 0$ since the angle between them is zero ($\sin 0 = 0$). This implies,

$$
\lambda_2 K^{-1}\bar{T}a' = \lambda_1 RK^{-1}\bar{T}a
$$

Now taking the dot product with A' on both sides, we get,

$$
\begin{aligned}
(\lambda_2 K^{-1}a')^T(\lambda_2 K^{-1}\bar{T}a') &= (\lambda_2 K^{-1}a')^T(\lambda_1 RK^{-1}\bar{T}a) \\
(\lambda_2)^2 K^{-T}a'^T K^{-1}\bar{T}a' &= \lambda_2\lambda_1 K^{-T}a'^T RK^{-1}\bar{T}a
\end{aligned}
$$

Now, $K^{-T}a'^T K^{-1}\bar{T}a' = 0$, since the volume is zero, which implies,

$$
\begin{aligned}
\lambda_2\lambda_1 K^{-T}a'^T RK^{-1}\bar{T}a &= 0 \\
K^{-T}a'^T RK^{-1}\bar{T}a &= 0 \\
a'^T\left(K^{-T}RK^{-1}\bar{T}\right)a &= 0 \\
a'^T Fa &= 0
\end{aligned}
$$

where $F = K^{-T}RK^{-1}\bar{T}$. Now since rotation is an identity matrix and the instrinsics matrix K will not affect the skew-symmetricity of the F matrix. Now consider $\bar{T}$, which is given by, $\bar{T} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$.

Now taking $-\bar{T}^T$, we get, $-\bar{T}^T = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} = \bar{T}$

This implies that $\bar{T}$ is a skew-symmetric matrix and hence, F is a skew-symmetric matrix, which implies the fundamental matrix F is a skew-symmetric matrix.

Practice

2 Fundamental Matrix Estimation

2.1 Eight Point Algorithm

The fundamental matrix obtained is $F = \begin{bmatrix} 4.47299494e-09 & 1.21213372e-07 & -1.19108124e-03 \\ 6.86335630e-08 & 3.26770775e-09 & -2.65963702e-05 \\ 1.14422069e-03 & 8.94807235e-06 & 4.12106109e-03 \end{bmatrix}$ and the points with epipolar lines are shown in Figure 1. Code is implemented in `eightpoint(pts1,pts2,M)` and can be tested by running `python submission.py 1`. Results are saved in `q2_1.npz`.

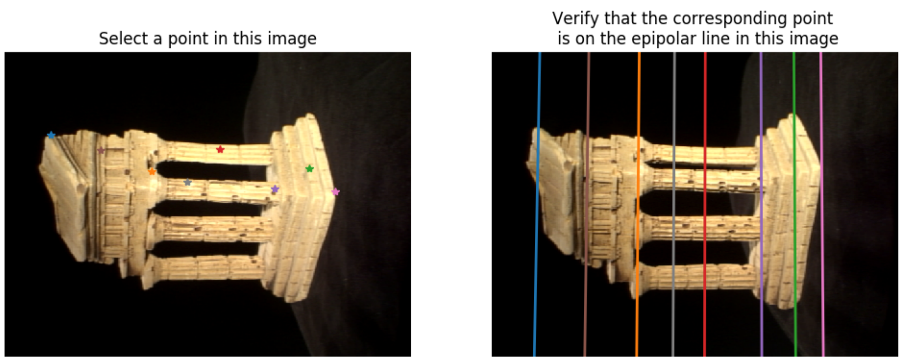


Figure 1: Eight Point Algorithm

2.2 Seven Point Algorithm

The fundamental matrix obtained along with points and epipolar lines are shown below. Code is implemented in `sevenpoint(pts1, pts2,M)` and can be tested by running `python submission.py 2`. Results are saved in `q2_2.npz`.

$F_1 = \begin{bmatrix} -1.14160558e-09 & -1.18621569e-09 & 1.34445217e-04 \\ 2.55567652e-07 & -5.59628774e-07 & -7.79399039e-02 \\ -1.27169075e-03 & 1.45975917e-03 & -2.44545384e+01 \end{bmatrix}$ and the corresponding image is Figure 2

$F_2 = \begin{bmatrix} -4.08669446e-08 & 3.05551144e-07 & 8.36363607e-02 \\ -5.53854907e-09 & 1.86614157e-08 & 1.39751433e-03 \\ 3.69395646e-05 & -1.38352678e-04 & -2.13058184e+01 \end{bmatrix}$ and the corresponding image is Figure 3.

$F_3 = \begin{bmatrix} 1.22182144e-03 & -1.39745242e-03 & 2.38091034e+01 \\ -5.73578682e-05 & 1.59988545e-04 & 1.99716020e+01 \\ 5.19772625e-03 & -7.03329341e-03 & 6.75013270e+01 \end{bmatrix}$ and the corresponding image is Figure 4

We can see the F_1 is the best fundamental matrix as it gives the most relevant epipolar lines.

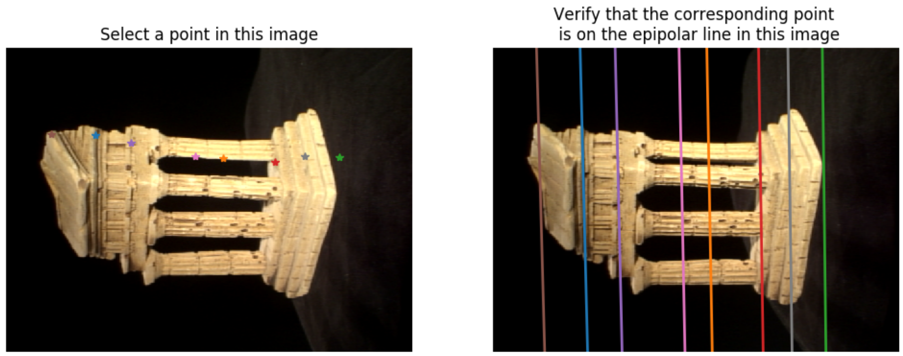


Figure 2: 7-Point: F_1

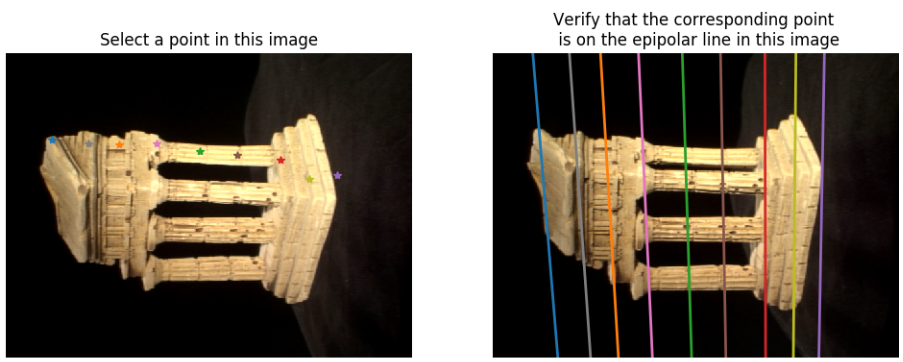


Figure 3: 7-Point: F_2

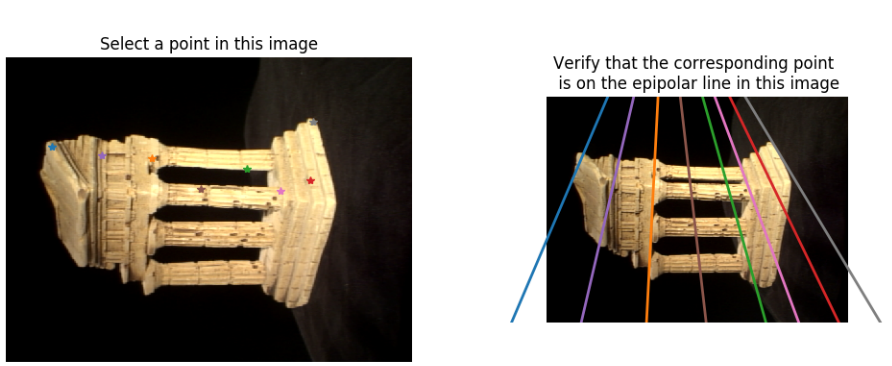


Figure 4: 7-Point: F_3

3 Matrix Reconstruction

3.1 Essential Matrix

The essential matrix E given the fundamental matrix from 8-point algorithm is given by

$$E = \begin{bmatrix} 1.03398474e-02 & 2.81212401e-01 & -1.76336755e+00 \\ 1.59228381e-01 & 7.60843534e-03 & -7.69110425e-03 \\ 1.76749013e+00 & 7.08017571e-02 & 3.74300526e-04 \end{bmatrix}$$

Code is implemented in `essentialMatrix(F,K1,K2)` and can be tested by running `python submission.py 3`.

3.2 Triangulate

Let $p1_{i1}$ and $p1_{i2}$ represent the x and y coordinate of the i^{th} point from set of points 1. Similarly for set of points 2, we have $p2_{i1}$ and $p2_{i2}$ as the x and y coordinates respectively. Let $C1_{i:}$ represent the i^{th} row of C_1 and let $C2_{i:}$ represent the i^{th} row of C_2. Here either of $C1_{i:}$ or $C2_{i:}$ is a vector of length 4. Code is implemented in `triangulate(C1,pts1,C2,pts2)`.

$$A_i = \begin{bmatrix} p1_{i1}C1_{3:} - C1_{1:} \\ p1_{i2}C1_{3:} - C1_{2:} \\ p2_{i1}C2_{3:} - C2_{1:} \\ p2_{i2}C2_{3:} - C2_{2:} \end{bmatrix}$$

3.3 Find best M2

The best M_2 is given by $M_2 = \begin{bmatrix} 0.99938017 & 0.034815 & 0.00521366 & 0.00438072 \\ -0.03503057 & 0.96886256 & 0.24510855 & -1.0 \\ 0.00348214 & -0.24513926 & 0.96948162 & 0.09006165 \end{bmatrix}$

Code is implemented in `findM2.py` and can be tested by running `python findM2.py`. Results are saved in `q3_3.npz`.

4 3D Visualization

4.1 Epipolar Correspondence

The result generated is shown in Figure 5. Code is implemented in `epipolarCorrespondence(im1, im2, F, x1, y1)` and can be tested by running `python submission.py 4`. Results are saved in `q4_1.npz`.

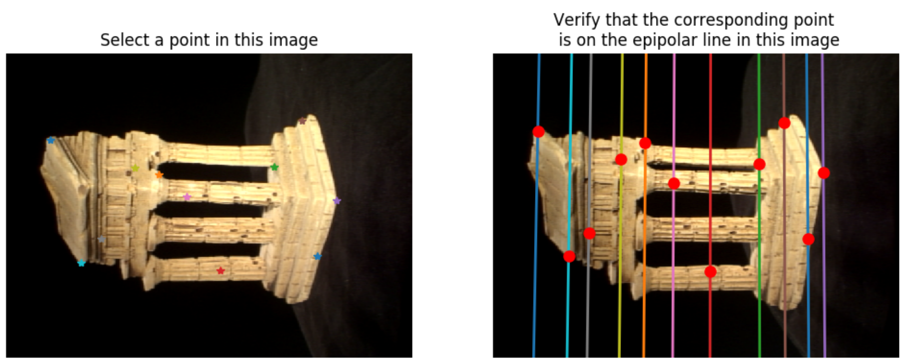
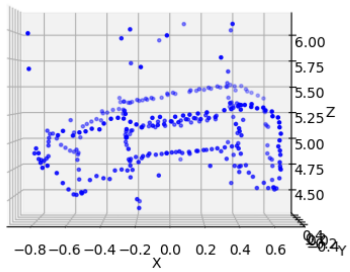


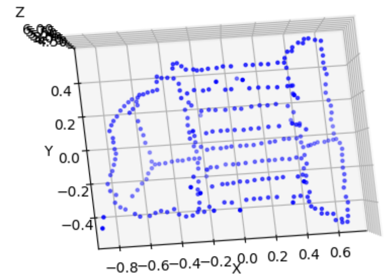
Figure 5: Epipolar Correspondence

4.2 Visualization

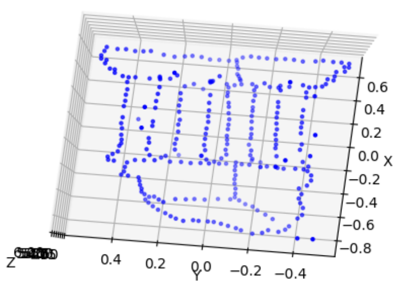
The visualization is shown in Figures 6a–6e. Code is implemented in `visualize.py` and can be tested by running `python visualize.py`. Results are saved in `q4_2.npz`



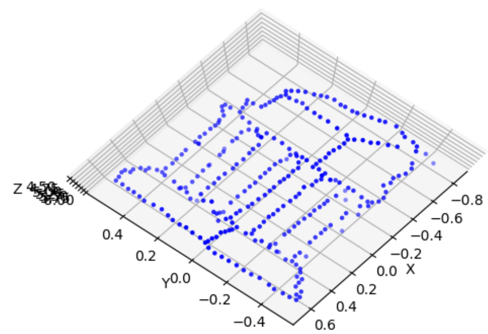
(a) View 1



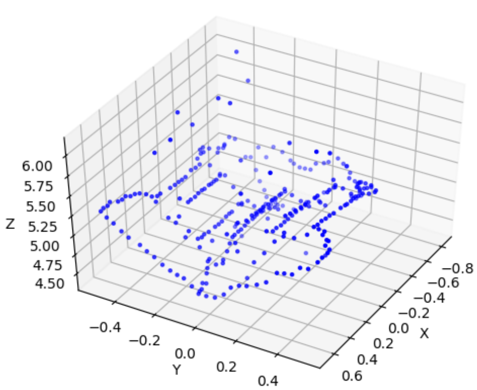
(b) View 2



(c) View 3



(d) View 4



(e) View 5

Figure 6: Visualizing with Epipolar Correspondence

5 Bundle Adjustment

5.1 Ransac

For ransac, I selected 7 random points and obtained the possible fundamental matrices using the `sevenpoint(pts1, pts2, M)` code. Then for each of the matrices from sevenpoint, I compute the epipolar line $E = \begin{bmatrix} a & b & c \end{bmatrix}^T$ for each point in image 1 p_{i1} and then compute the correspondence in image 2 as $p_{2i}E^T$, where p_{2i} and p_{i1} is the i^{th} point in image 2. Since this equation has to be zero for correspondence, the actual error metric can be defined as the value obtained from $p_{2i}E^T$ and this has to be small. I take a threshold of 0.02 and run it for approximately 50 iterations - giving the algorithm sufficient computation to find good set of inliers. Then using these set of inliers, I get the refined fundamental matrix using the `eightpoint(pts1,pts2,M)` code.

The result of eightpoint without refining is shown below in Figure 7 and the result with refining is shown below in Figure 8. Code is implemented in `ransacF(pts1, pts2, M)` and can be tested by running `python submission.py 5`.

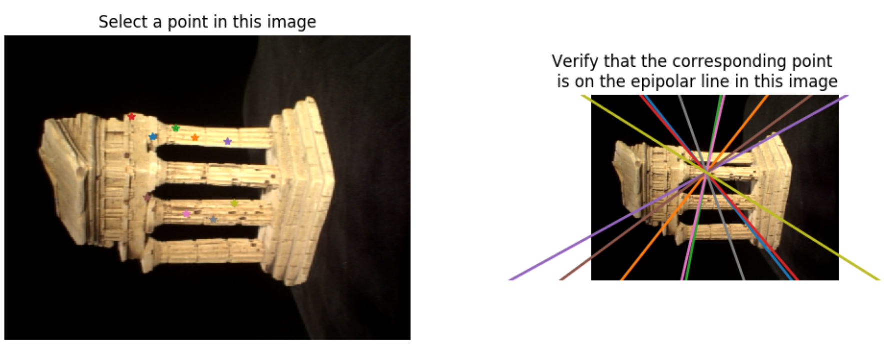


Figure 7: Without ransac on noisy points

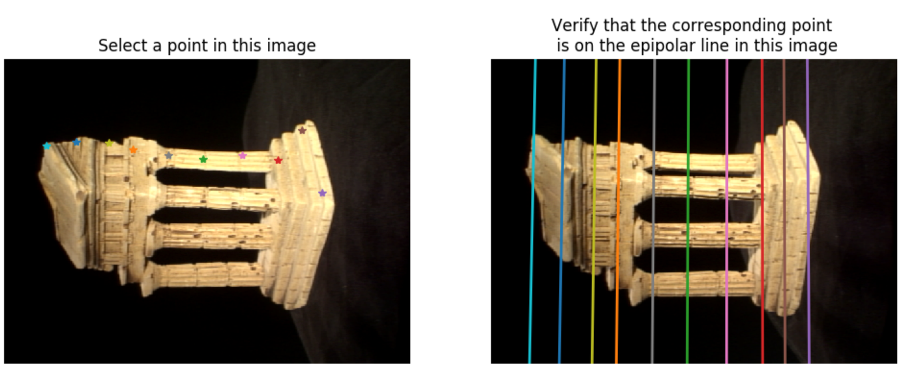


Figure 8: With ransac on noisy points

5.2 Rodrigues Vector and Matrix Computation

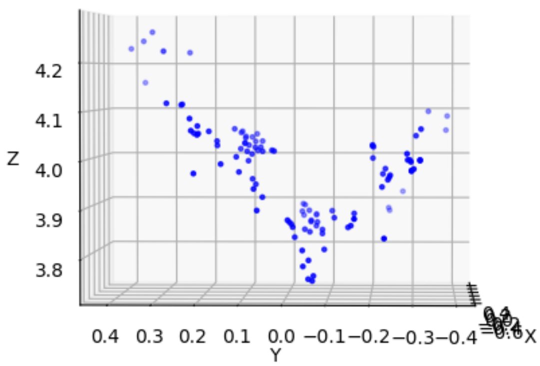
The Rodrigues vector for $R = I$ computed using `invRodrigues(R)` is given by $r = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$

The rodrigues matrix for $r = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ computed using `rodrigues(r)` is given by $R = \begin{bmatrix} 0.22629564 & -0.18300792 & 0.95671228 \\ 0.95671228 & 0.22629564 & -0.18300792 \\ -0.18300792 & 0.95671228 & 0.22629564 \end{bmatrix}$.

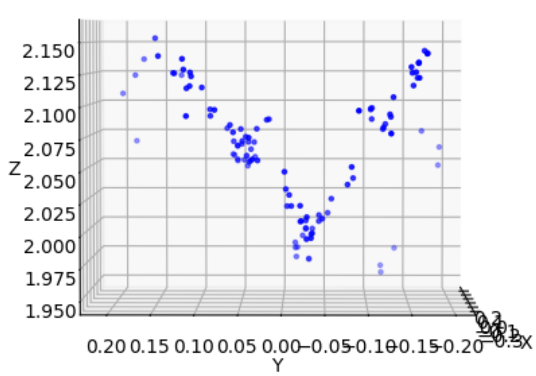
Code is implemented in `invRodrigues(R)` and `rodrigues(r)` and can be tested by running `python submission.py 6`.

5.3 Residuals and BA

The initial 3D points are given in Figure 9a and the final 3D points are given in Figure 9b. The initial re-projection error is 71.4897 and the final re-projection error is 4.8708. Code is implemented in `bundleAdjustment(K1, M1, p1, K2, M2_init, p2, P_init)` and can be tested by running `python submission.py 7`.



(a) Without BA



(b) With BA

Figure 9: Bundle Adjustment

`Note`: Including `helper.py` and `checkA4Format.py` in zip submission as they have also been modified.