3D Reconstruction

16-720 Computer Vision - Homework 4

Q. 1.1

Camera C1 and C2 capture point P. The fundamental matrix F defines the following relationship between the corresponding pixels in both the images as ${p_1}^T * F * p_2 = 0$.

$$\text{where, } F = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}, p_1 = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}, p_2 = \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

When the image coordinates are normalized, such that the coordinate center lies on the principle axis, the coordinates p1 and p2 that represent P are,

$$p_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, p_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

The relationship ${p_1}^T * F * p_2 = 0$ is now,

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

$$\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & f_{13} \\ 0 & 0 & f_{23} \\ 0 & 0 & f_{33} \end{bmatrix} = 0$$

$$\mathrm{f}_{33} = 0$$

Q. 1.2

For two cameras C1 and C2 that view an object such that the second camera differs only by pure translation, we can write the Camera Matrices as,

$$R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, T_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, T_2 = \begin{bmatrix} t \\ 0 \\ 0 \end{bmatrix}$$

The Essential Matrix E that relates the point in one matrix to a line in the other matrix is given by,

$$E = [tx]R = \begin{bmatrix} 0 & -tz & ty \\ tz & 0 & -tx \\ -ty & tx & 0 \end{bmatrix} \begin{bmatrix} r11 \\ r22 \\ r33 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -tx \\ 0 & tx & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

For a point p1(x1, y1) in image1, the epipolar line l2 in image2, the equation for l2 is given by,

$$l_2 = Ep_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -tx \\ 0 & tx & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

$$\mathrm{l}_1 = \begin{bmatrix} 0 \\ -t_x \\ y_1 t_x \end{bmatrix}$$

The line in image1 for a point p2(x2, y2) in image2 is given by,

$$l_2 = E^T p_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & tx \\ 0 & -tx & 0 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

$$\mathrm{l}_2 = \begin{bmatrix} 0 \\ t_x \\ -y_2 t_x \end{bmatrix}$$

The equation for a line is given by, $ax + by + c = 0$. The equations for the epipolar lines l1 and l2 have a = 0, which represents a line that is parallel to the x axis with a distance given by the c coefficients, $y_1 t_x and - y_2 t_x$ The lines l1 and l2 are lines that are parallel to x axis.

Q. 1.3

Let a point P in the real world be captured by the robot at time i. The image point p1 is given by,

$$p_1 = R_i P + t_i$$

The point p can be represented as,

$$P = R_i^{-1}(p_1 - t_i)$$

The same point world point P captured at time i+1 will be as p2,

$$p_2 = R_{i+1} P + t_{i+1}$$
$$\mathrm{p}_2 = R_2(R_i^{-1}(p_1 - t_i)) + t_{i+1}$$
$$\mathrm{p}_2 = R_2 R_i^{-1} p_1 - R_2 R_i^{-1} t_i + t_{i+1}$$

From this equation, we can derive the expression for the rotation and translation between the two image frames as follows,

$$R_{rel} = R_2 R_i^{-1}$$
$$\mathrm{t}_{rel} = -R_2 R_i^{-1} t_i + t_{i+1}$$

The Essential Matrix E is given by

$$E = [t_{rel}] R_{rel}$$
$$\mathrm{F} = \mathrm{K}^{-T} [t_{rel}] R_{rel} K^{-1}$$

Q. 1.4

The Camera C captures the real world points X and X' as image points x and x', where X' is the reflection of X in a mirror. X and X' are separated by pure translation. The fundamental matrix F for a pure translation case is given by the formula derived in Q. 1.3.

$$F = K^{-T}[t_{rel}]R_{rel}K^{-1}$$

$$\mathrm{F} = \mathrm{K}^{-T} \begin{bmatrix} 0 & -tz & ty \\ tz & 0 & -tx \\ -ty & tx & 0 \end{bmatrix} K^{-1}$$

Taking the transpose,

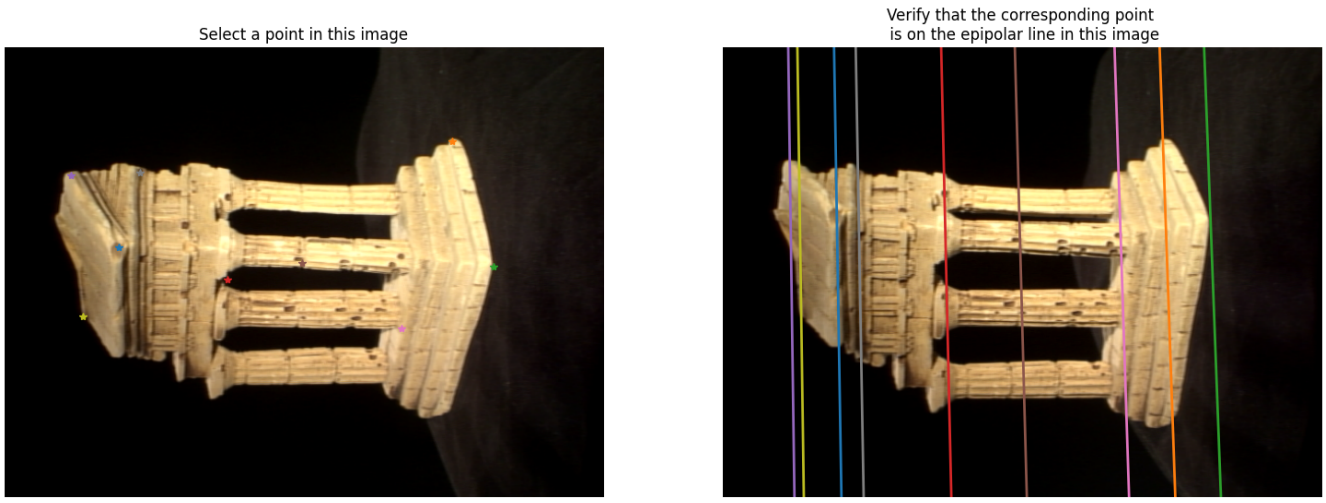
$$F^T = K^{-T} \begin{bmatrix} 0 & tz & -ty \\ -tz & 0 & tx \\ ty & -tx & 0 \end{bmatrix} K^{-1}$$

$$\mathrm{F}^T = -F$$

This proves that the fundamental Matrix is a skew-symmetric matrix.

Q. 2.1 The Eight Point Algorithm

The epipolar lines for a set of points visualized using the helper function, displayEpipolarF.



The F matrix obtained is as follows.

$$F = \begin{bmatrix} 9.7883e-10 & -1.3213e-07 & 1.1258e-03 \\ -5.7384e-08 & 2.968e-09 & -1.1751e-05 \\ -1.0826e-03 & 3.0484e-05 & -4.4703e-03 \end{bmatrix}$$

```
F:  [[ 9.78833282e-10 -1.32135929e-07  1.12585666e-03]
 [-5.73843315e-08  2.96800276e-09 -1.17611996e-05]
 [-1.08269003e-03  3.04846703e-05 -4.47032655e-03]]
```

Q. 3.2 Triangulate

For given set of 3D coordinates , we write the homogeneous coordinates,

The 2D homogeneous coordinates $w_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}$ is given by,

$$u_i = \frac{m_1^T P}{m_3^T P}$$
$$\mathrm{v}_i = \frac{m_2^T P}{m_3^T P}$$

where M is the 3x4 Camera projection matrix,

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{14} \end{bmatrix}$$

Expanding the expressions for u_i and v_i, we get,

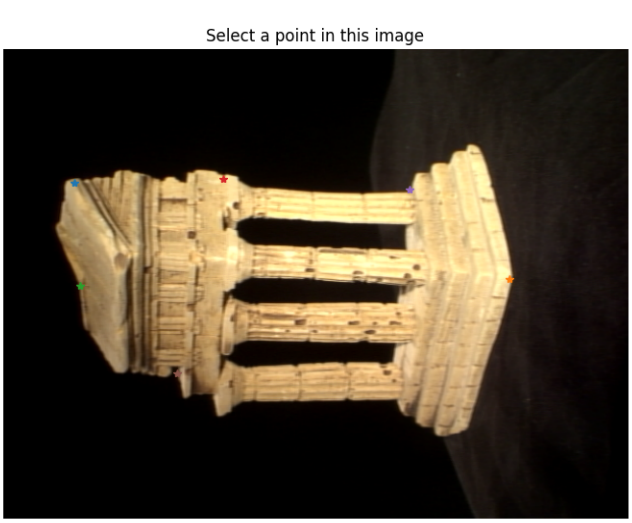
$$(m_{33}u_i - m_{11})x_i + (m_{32}u_i - m_{12})y_i + (m_{33}u_i - m_{13})z_i + (m_{34}u_i - m_{14}) = 0$$
$$(\mathrm{m}_{33}v_i - m_{21})x_i + (m_{32}v_i - m_{22})y_i + (m_{33}v_i - m_{23})z_i + (m_{34}v_i - m_{24}) = 0$$

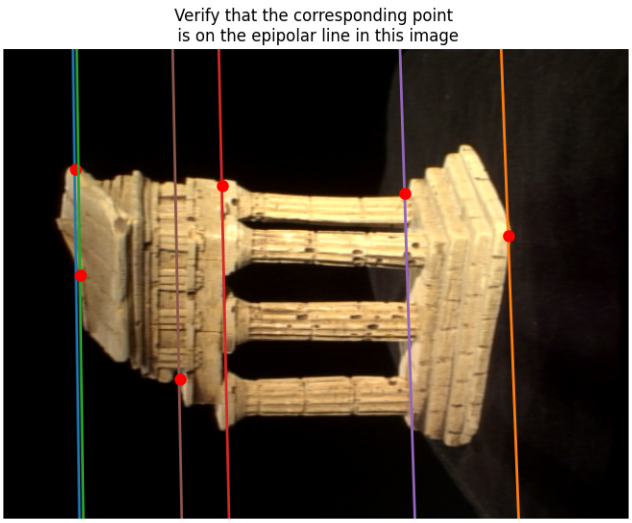
The expression for A in $A_i w_i = 0$ is obtained by taking corresponding set of image points $(u_1, v_1), (u_2, v_2)$ for the same point (x, y, z) in 3D,

$$(m_{33}u_1 - m_{11})x + (m_{32}u_1 - m_{12})y + (m_{33}u_1 - m_{13})z + (m_{34}u_1 - m_{14}) = 0$$
$$(\mathrm{m}_{33}v_1 - m_{21})x + (m_{32}v_1 - m_{22})y + (m_{33}v_1 - m_{23})z + (m_{34}v_1 - m_{24}) = 0$$
$$(\mathrm{m}_{33}u_2 - m_{11})x + (m_{32}u_2 - m_{12})y + (m_{33}u_2 - m_{13})z + (m_{34}u_2 - m_{14}) = 0$$
$$(\mathrm{m}_{33}v_2 - m_{21})x + (m_{32}v_2 - m_{22})y + (m_{33}v_2 - m_{23})z + (m_{34}v_2 - m_{24}) = 0$$

Q. 4.1 3D Visualization

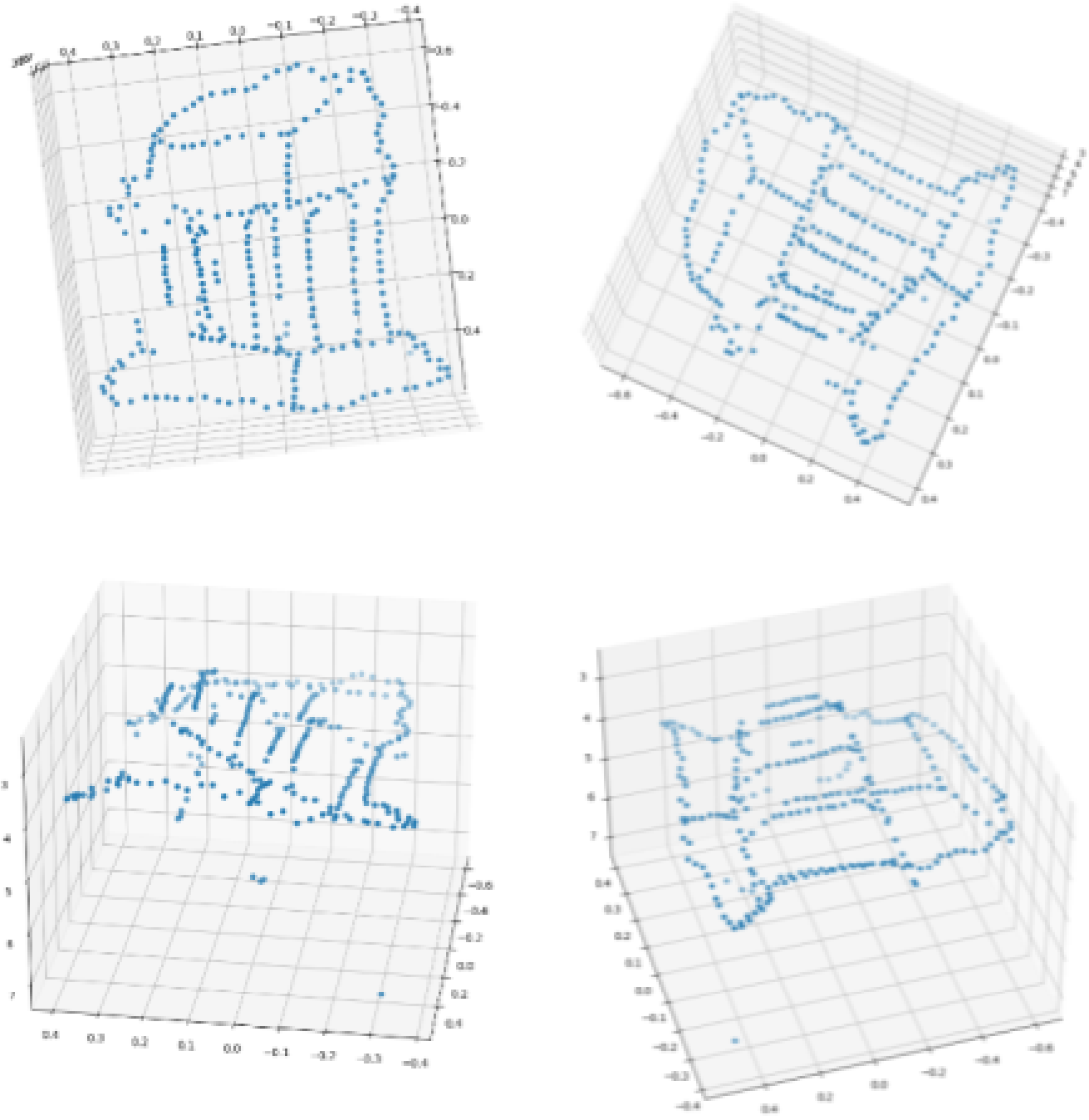
The corresponding points in image 2 for a set of points in image 1 visualized using the helper function, displayMatchGUI.





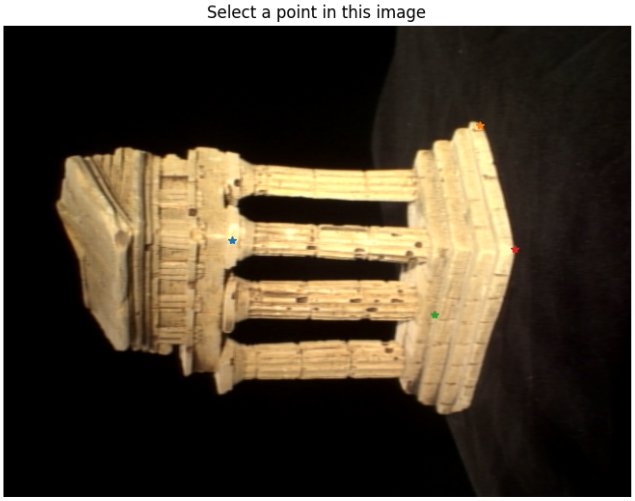
Q. 4.2

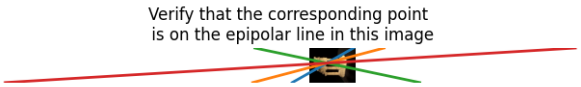
The 3D Visualization of the temple is show below.



Q. 5.1 RANSAC

Using the noisy correspondence data, the eight point algorithm is first run. The image below shows the correspondences as plotted using displayEpipolarF





Using the RANSAC method, we can calculate a Fundamental Matrix that is accurate to calculate the epipolar lines. The RANSAC algorithm was built by taking 8 random points from the noisy correspondence data and calculating F using the eightpoint algorithm. The predictions for the points is done using the calculated F and the function epipolarCorrespondence().
The error between the predicted points and actual points is calculated using the following equation

$$error = (p_{2pred})^T F p_1$$

We run RANSAC for 500 iterations with a tolerance of 0.001 to find the inlier using the calculated error. After the iterations, the Fundamental Matrix F is calculated with all the points that were the maximum inliers for some iteration of the RANSAC algorithm.

Q. 5.2 Rodrigues Vector

Rodrigues Formula Implementation

$$(axis, angle) = (\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}, \theta)$$

Inverse Rodrigues - Rotation Matrix R to Axis-angle r

$$\theta = arccos(\frac{Tr(R)-1}{2})$$
$$\omega = \frac{1}{2sin\theta}\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$
$$\mathrm{r} = \theta * \omega$$

Rodrigues - Axis-angle r to Rotation Matrix R

$$K = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix}$$
$$\mathrm{R} = \mathrm{I} + (\sin\theta)K + (1 - cos\theta)K^2$$
where θ *is the magnitude of* r

```
Q5.2:Extra Credit  Rodrigues formula.
    Input:  r, a 3x1 vector
    Output: R, a rotation matrix
'''
def rodrigues(r):
    # Replace pass by your implementation
    theta = np.sqrt(r[0]**2 + r[1]**2 + r[2]**2)
    r0 = r / theta
    K = [[0, -r0[2], r0[1]], [r0[2], 0, -r0[0]], [-r0[1], r0[0], 0]]
    R = np.eye(3) + (math.sin(theta)*K) + ((1 - math.cos(theta))*(K**2))

    return R


'''
Q5.2:Extra Credit  Inverse Rodrigues formula.
    Input:  R, a rotation matrix
    Output: r, a 3x1 vector
'''
def invRodrigues(R):
    # Replace pass by your implementation
    theta = math.acos((np.trace(R) - 1)/2)
    omega = (1 / 2*math.sin(theta))* [[R[2][1] - R[1][2]][R[0][2] - R[2][0]][R[1][0] - R[0][1]]]
    r = theta * omega

    return r
```