REPORT

Consider two images I1 (“im1.jpg”) and I2 (“im2.jpg”) of a static scene captured from a single camera with the given intrinsic camera matrix K (“Intrinsic Matrix K.txt”).

Task 1) Find a set of ground-truth correspondences $\{(p_i, p'_i)\}_{i=1}^{n}$ using any of the existing implementations. Ensure that there are at least n = 100 true correspondences.

Task 2) Assume that the world-coordinate system is aligned with the coordinate-system of the camera location. Implement the Linear Eight-point Algorithm to find the Essential matrix E.

Task 3) Decompose the obtained Essential matrix E into the camera motion rotation matrix R and the translation vector t.

Task 4) Let Pi be the corresponding 3D point for the pixel pair (pi, p’i). Find Pi, $\forall$ i $\in$ {1, 2, . .,n} usingthe triangulation approach.

Task 5) Plot the obtained Pi, $\forall$ i $\in$ {1, 2, . . . , n} and the camera center t.

Task 1) Encrypt Find a set of ground-truth correspondences {(pi, p'i)}n i=1 using any of the existing implementations.
Ensure that there are at least n = 100 true correspondences.

We use the SIFT feature detection algorithm along with FLANN matchers to create a set of at least 100 correspondences. The set of 2D coordinates from the first and second image are stored in arrays img_points1 and img_points2 respectively.

```
[7]  img_points1

array([[ 255.85579,  729.1978 ],
       [ 599.5451 , 1055.0243 ],
       [ 605.53705, 1068.8855 ],
       ...,
       [1173.8253 ,  413.9116 ],
       [1177.1371 ,  470.41556],
       [1406.4004 ,  659.2843 ]], dtype=float32)
```

```
[8]  img_points2

array([[ 350.19003,  755.10065],
       [ 560.2085 , 1056.4618 ],
       [ 566.7841 , 1070.0416 ],
       ...,
       [1170.697  ,  443.39108],
       [1180.1462 ,  502.00748],
       [1409.2631 ,  698.8335 ]], dtype=float32)
```

We run a while loop iterating upto 100 to ensure that, in the case of total no. of correspondences being less than 100, more correspondences or 'good matches' are extracted until no of correspondences is atleast 100.

Task 2) Assume that the world-coordinate system is aligned with the coordinate-system of the camera location. Implement the algorithm taught in the class to find the Essential matrix E.

We implement the given algorithm, with the help of np.linalg . The 2D coordinates are first converted to homogenous 3D coordinates, and then from the 3D coordinates, we obtain the normalized image coordinates by performing matrix multiplication with the inverse of the camera's intrinsic matrix K. Since we have the relation $y^{\top}iExi = 0$, we take the kronecker product of the matrices, before decomposing the resultant matrix and taking the eigenvector corresponding to the smallest eigenvalue. This gives us the value of E. Now, as per linear eight-point algorithm, we need alteast 8 correspondences, and accordingly the Essential Matrix must be a matrix of rank 2 with 2 equal singular values. To ensure this, we perform Single Value Decomposition of E, update the diagonal values of the middle factor, and then multiply the factors to obtain the correct value of E.

```
E

array([[ 4.25317602e-02, -6.72013033e-01,  1.05698514e-01],
       [ 7.01592989e-01,  5.06290508e-02,  1.24334450e-01],
       [-9.64256598e-02, -1.25952884e-01,  2.14241788e-04]])
```

Task 3) Decompose the obtained Essential matrix E into the camera motion rotation matrix R and the translation vector t.

To decompose the essential matrix E into Rotation matrix R and Translational vector t, we first compute the Singular Decomposition value (SVD) of the essential matrix. Then we compute a 3x3 skew symmetric matrix by the name W. SVD returns three matrices U,S and Vt.The rotation matrix can be computed using the matrices U,W and Vt. Then we compute a scaling factor s inorder to compute the translational vector. The translation vector can be computed using this scaling factor s and matrix U.

Rotation matrix:

```
Rotation matrix:
 [[-0.99342398  0.011189   -0.11394557]
 [-0.00813751 -0.99959661 -0.02721025]
 [ 0.11420406  0.02610408 -0.9931143 ]]
```

Translational Vector:

```
Translation vector:
 [-0.97834281 -0.10313772  0.17946576]
```

Task 4) Let Pi be the corresponding 3D point for the pixel pair (pi, p'i). Find Pi, $\forall i \in \{1, 2, \ldots, n\}$ using the triangulation approach.

In this task, we calculate P1 and P2 ,the projection matrices for the two cameras. The projection matrix is a combination of the extrinsic and intrinsic properties of a camera. K(given) is the intrinsic parameters matrix, R and t (calculated in the previous tasks) represent the rotation and translation between the two cameras, respectively.

The image points p_i and p'_i are given in homogeneous coordinates, where the last element is 1. To perform triangulation, we first stack the four equations for the two cameras and two image points into a matrix A. We then use singular value decomposition (SVD) to solve the system of equations and obtain the homogeneous 3D point X.

We get the result as a matrix named 'Matrix_3D' which contains all the 3D points corresponding to each pixel pair given.

```
[90] Matrix_3D

     array([[  5.44315768,  -0.63095802, -12.91470219],
            [  1.53619322,  -1.662727  ,  -7.01845386],
            [  1.51857515,  -1.7272865 ,  -7.05186032],
            ...,
            [ -1.00503108,   1.15663128,  -8.45514176],
            [ -1.0558288 ,   0.90215845,  -8.7412803 ],
            [ -2.16172509,  -0.07678663,  -8.45588142]])
```

Task 5) Plot the obtained Pi, $\forall i \in \{1, 2, \ldots, n\}$ and the camera center t.

To plot the obtained Pi (the corresponding 3D point for the pixel pairs) and the camera center t we used the scatter function of matplotlib. To plot Pi we looped through the 3D points in Matrix_3D and plotted each point using the scatter function. These points can be seen as 'o' in the graph below. Using the same scatter function we plotted the camera center and can be seen as 'x' in the graph below.