EE417 - Assignment 8 - Post-Lab Report

Baris Sevilmis
Lab Date: 11/12/2018
Due Date: 17/12/2018

Overview

Eight lab assignment of EE417 focuses on pose recovery of a camera from different viewpoints. Therefore, essential matrix(E) must be estimated and 8-point algorithm will be used for estimation. Knowledge of epipolar geometry is essential in this process. Required knowledge of epipolar geometry will be provided in next section. Furthermore, pose recovery will be done on a cubic 3D image that has 3 different plane, each with 9 points, though some of the points are common for multiple planes.

Pose Recovery of a Calibrated Camera Through Essential Matrix Using 8-Point Algorithm

From Stereo Vision, 3D Structure and Motion Recovery, we have already learned that different viewpoints on the same image plane are estimated through rigid body motion with the following formula:

$$\lambda_2 x_2 = R\lambda_1 x_1 + T$$

For sake of clarity, an euclidean transformation is applied between camera poses for which $x_1 \& x_2$ are measurements of world coordinates and $\lambda_1, \lambda_2, R \& T$ are unknowns to be estimated. As a consequence, an optimization problem stands out, in which Rotation(R), Translation(T) and Depth(λ_1, λ_2) are to be estimated in order to minimize reprojection error. Following formula is required for estimation given a naive approach to the problem:

$$\sum_{j=1}^{n} \left\| x_1^j - \pi(R_1, T_1, X) \right\|^2 + \left\| x_2^j - \pi(R_2, T_2, X) \right\|^2$$

However, solving this optimization problem by a naive approach would be unreasonably difficult given the unknown parameters and the view structures. For this reason, Algebraic Elimination of Depth by Longuet-Higgins alters this optimization problem into a much more reasonable, efficient and reliable problem. In this technique, previous optimization formula changes into following form by taking cross product of both sides with T, in other words matrix multiplication with skew symmetric matrix $\hat{T}$:

$$x_2^T \hat{T} R x_1 = 0$$

$$\hat{T} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

$$T = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Proof of this equation is as following by the skew symmetric matrix properties:

$$x_2^T \hat{T} x_2 = -x_2^T \hat{T}^T x_2 = -x_2^T \hat{T} x_2 \equiv 0$$

$\hat{T}R$ will be specified as the essential matrix(E):

$$x_2^T E x_1 = 0$$

This matrix has 5 degrees of freedom instead of 6 or 8. Since Rotation matrix has 3 degrees of freedom and Translation Matrix ahs 2 degrees of freedom, essential matrix should have 6 degrees of freedom instead of 8. However, Translation is up to scale in addition. For this reason, E has 5 degrees of freedom instead of 6.

Before continuing with the pose recovery, there are few more properties that should be mentioned, namely epipolar geometry properties. These following properties will be used in pose recovery problem:

$$l_1 \sim E^T x_2$$
$$l_2 \sim E x_1$$
$$l_i^T x_i = 0$$
$$l_i^T e_i = 0$$
$$E e_1 = 0$$
$$e_2 E^T = 0$$

As for the pose recovery problem, we will be working on a cubic image. Data is provided in Figure 1, in which first graph corresponds to the world coordinates and other two images correspond to image points from different views.

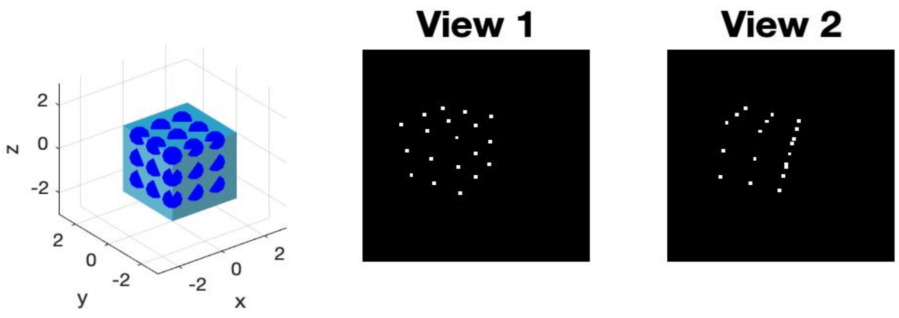


Figure 1: 3D world points and image points from two different views

Intrinsic parameters are already known, to be more specific camera is already calibrated. Therefore, as to begin pixel coordinates are multiplied with inverse of the K matrix. As we will be using 8 point approach, in order to avoid degeneration since $rank(x^T x) >= 8$ for reliable solutions. $x^T x$ is the minimum eigenvector of:

$$min_{E^s} \left\| xE^S \right\|^2 = min_E \sum_{j=1}^{n} (x_2^{jT} E x_1^j)^2$$

E_s is column stacked version of essential matrix E.

As for the algorithm, 8 points are selected as mentioned previously. These points are placed in vector a and a_i are placed in X in the following way:

$$a^T = \begin{bmatrix} x_1x_2 & x_1y_2 & x_1z_2 & y_1x_2 & y_1y_2 & y_1z_2 & z_1x_2 & z_1y_2 & z_1z_2 \end{bmatrix}$$

$$X = \begin{bmatrix} a_1^T; & a_2^T; & & a_8^T \end{bmatrix}$$

Listing 1: MatLab Code for X matrix and transforming pixel coordinates

```
%% Transform pixel coordinates and construct X matrix using
   Equations 1 and 2
p1_new = pinv(K)*u1;
p2_new = pinv(K)*u2;
lst = [4 5 7 8 12 13 14 15 16 17 18 19];
X = [];
for ii = 1:length(lst)
    a = [p1_new(1,lst(ii))*p2_new(1,lst(ii)); p1_new(1,lst(ii))*
       p2_new(2,lst(ii));...
         p1_new(1,lst(ii))*p2_new(3,lst(ii)); p1_new(2,lst(ii))*
            p2_new(1,lst(ii));...
         p1_new(2,lst(ii))*p2_new(2,lst(ii)); p1_new(2,lst(ii))*
            p2_new(3,lst(ii));...
         p1_new(3,lst(ii))*p2_new(1,lst(ii)); p1_new(3,lst(ii))*
            p2_new(2,lst(ii));...
         p1_new(3,lst(ii))*p2_new(3,lst(ii))];
    X = [X; a'];
end
```

Furthermore, $XE_s = 0$ indicates that we are required to use SVD or eigenvector technique on X such that we obtain E_s. E_s is turned in to 3x3 E format. Moreover, SVD is technique is used on E in order to cure the matrice. Diagonal matrix is manually changed in its diagonals as $\sigma_1 = \sigma_2 = 1, \sigma_3 = 0$. As last setp of curing the essential matrix E, resulting U and V matrices from SVD(E) taken into consideration, in which it is controlled whether $U, V \in SO(3)$, namely if $det(U) = det(V) = 1$. Listing 2 provides MatLab Code for this step. $det(U) = det(V) = 1$ is checked in addition with display method.

Listing 2: MatLab Code for Curing Essential Matrix

```
%% Estimate E, cure it and check for Essential Matrix
    Characterization
[[V, D] = eig(X'*X);
minn = D(1,1);
mineig_vec = V(:,1);
for kk = 1:size(V,1)
    if minn > D(kk,kk)
        minn = D(kk,kk);
        mineig_vec = V(:,kk);
    end
end
E_s = mineig_vec;
[U_e, S_e, V_e] = svd(E);
E = [E_s(1) E_s(4) E_s(7); E_s(2) E_s(5) E_s(8); E_s(3) E_s(6)
    E_s(9)];
S_e(1,1) = 1;
S_e(2,2) = 1;
S_e(3,3) = 0;
det_u = det(U_e);
det_v = det(V_e);

E = U_e * S_e * V_e';
```

Next step requires finding the epipoles and epipolar lines. Formulas were provided above. Listing 3 depicts the code piece for finding epipoles and epipolar lines.

Listing 3: MatLab Code for Finding Epipoles and Epipolar Lines

```
%% Find epipoles and epipolar lines
e1 = null(E);
e2 = null(E');
l1 = E'*p2_new(:,5);
l2 = E*p1_new(:,5);
%% Verify epipoles and epipolar lines
c1 = l1'*e1;
c2 = l2'*e2;
disp(l1'*e1 and l2'*e2);
disp(c1);
disp(c2);
if c1 == 0
    disp('Verification of l1 and e1: Done');
end
if c2 == 0
    disp('Verification of l2 and e2: Done');
end
```

Lastly, Rotation and Translation of the camera are recovered. In other words, camera poses are recovered up to a scale, since translation could be estimated up to scale. Following formulas are used for the pose recovery:

$$(\hat{T}_1, R_1) = (UR_z(\frac{\pi}{2})\Sigma U^T, UR_z^T(\frac{\pi}{2})V^T))$$

$$(\hat{T}_2, R_2) = (UR_z(-\frac{\pi}{2})\Sigma U^T, UR_z^T(-\frac{\pi}{2})V^T))$$

These $R\&\hat{T}$ are complementary of each other such that one is in front of the camera and one is right in the back in camera. Since it would be physically impossible for camera to take an image of its back, only positive point will be considered. Listing 4 ensures MatLab Code for this last phase.

Listing 4: MatLab Code for Finding Rotation and Translation

```
%% Recover the rotation and the translation
az = 90 * DEG_TO_RAD;
Rz = [cos(az) -sin(az)  0;
      sin(az)  cos(az)  0;
      0        0        1];
az2 = -90 * DEG_TO_RAD;
Rz2 = [cos(az2) -sin(az2)  0;
      sin(az2)  cos(az2)  0;
      0        0        1];
T1_hat = U_e * Rz * S_e * U_e';
T2_hat = U_e * Rz2 * S_e * U_e';
T1 = [T1_hat(3,2); T1_hat(1,3); T1_hat(2,1)];
T2 = [T2_hat(3,2); T2_hat(1,3); T2_hat(2,1)];

R1 = U_e * Rz' * V_e';
R2 = U_e * Rz2' * V_e';
```

We have tried 5 cases in our lab assignment in terms of point selection in the following manner:

- 8 points from different planes
- 8 points from same plane (For all 3 planes)
- All points
- More than 8 points ensuring optimal pose recovery

We have shown that there are two translation and rotation matrix estimation as one of them is physically impossible. Therefore, non-negative $R\&T$ are picked always. In our four cases, only the non-negative estimation matrices will be provided.

Case 1: 3-Planes, 8-Points

8 points were selected from all the planes. World point matrix is same as in Listing 5. Points that were selected are as follows:

$$points = \begin{bmatrix} 4 & 5 & 7 & 8 & 12 & 13 & 14 & 16 \end{bmatrix}$$

First four point is from the left plane, fifth, sixth and seventh are from the right plane and last point is from the right plane. These 8 points were selected in terms of achieving similar results as in provided lab reports.

Listing 5: MatLab Code for World Coordinates

```
%% World Coordinates
P_W=[0  2       0       1;
     0  1       0       1;
     0  0       0       1;
     0  2      -1       1;
     0  1      -1       1;
     0  0      -1       1;
     0  2      -2       1;
     0  1      -2       1;
     0  0      -2       1;
     1  0       0       1;
     2  0       0       1;
     1  0      -1       1;
     2  0      -1       1;
     1  0      -2       1;
     2  0      -2       1;
     1  1       0       1;
     2  1       0       1;
     1  2       0       1;
     2  2       0       1];

P_W = P_W';
```

Visualization of these 8-points are depicted in Figure 2. On the other hand, results are screen shots of terminal windows in terms of showing all the verification steps. Figure 3 provides the estimation results. Dee to noise there is no exact result, however similarity of estimated matrices with the true $R\&T$ matrices is definite. Moreover, these estimation results are also similar with the lab report results. Discussing these verifications and matrices step by step will prove useful. First commands indicate that the determinants of U and V matrices (results of $svd(E)$) are $\in SO(3)$. Then, by using $l_i^T e_i = 0$ formula from epipolar geometry, epipole and epipolar line intersection is proved, since results are ~ 0. Lastly, estimations are compared with each other, as it is obvious that signs of the estimations are same. Since translation is up to scale, E matrix and T matrix are up to scale. However, multiplying both of the matrices element wise by 3 will give a very close value to their true forms. Therefore, R is the only matrix that has the same scale with its true form. Estimated R is almost same with the true R. It is not as good as the estimation in the lab report that is provided for the lab assignment, however main point is clear. Estimation using 8 point algorithm works fine given the better estimable points in both views.

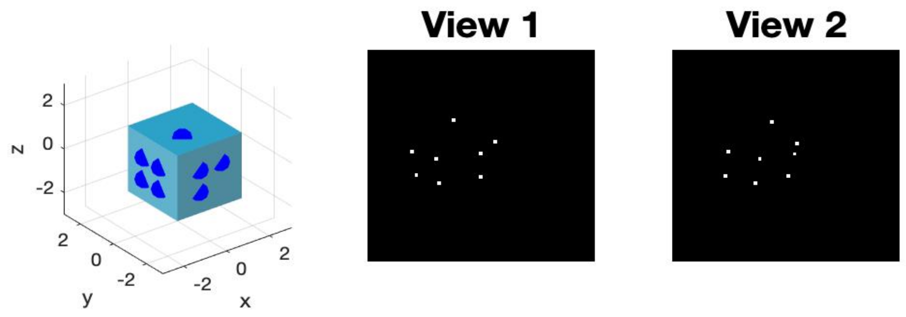
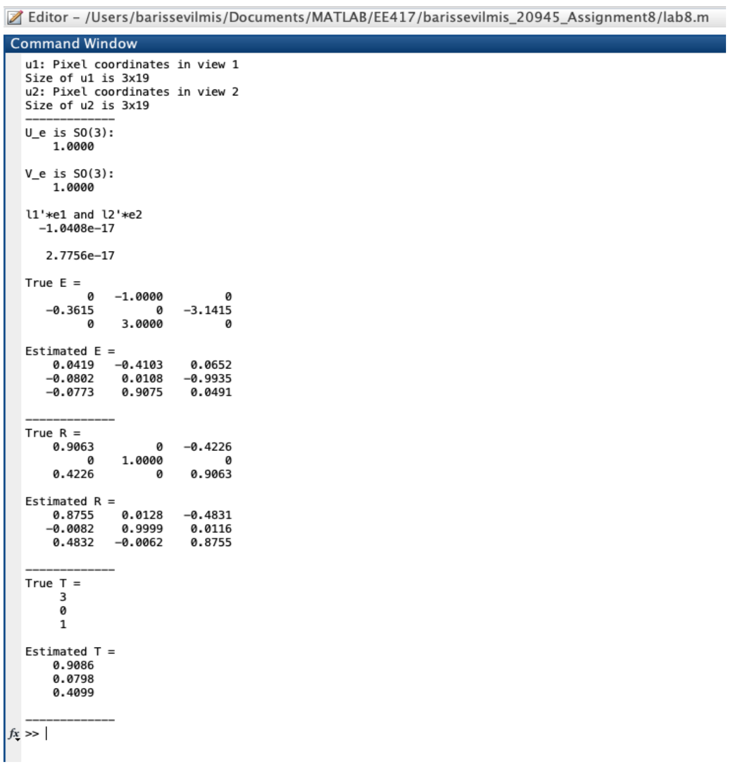


Figure 2: Visualization of Points-Case1


Editor – /Users/barissevilmis/Documents/MATLAB/EE417/barissevilmis_20945_Assignment8/lab8.m
Command Window

```
u1: Pixel coordinates in view 1
Size of u1 is 3x19
u2: Pixel coordinates in view 2
Size of u2 is 3x19
-------------
U_e is SO(3):
    1.0000

V_e is SO(3):
    1.0000

l1'*e1 and l2'*e2
  -1.0408e-17

   2.7756e-17

True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
    0.0419   -0.4103    0.0652
   -0.0802    0.0108   -0.9935
   -0.0773    0.9075    0.0491

-------------
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R =
    0.8755    0.0128   -0.4831
   -0.0082    0.9999    0.0116
    0.4832   -0.0062    0.8755

-------------
True T =
     3
     0
     1

Estimated T =
    0.9086
    0.0798
    0.4099

-------------
fx >> |
```

Figure 3: Results of Case 1

Case 2: 1-Plane, 8-Points

In this case, we have selected all the planes one by one and took 8 points from each of them. In other words, there are no more point from different planes. Selected points are depicted in Figure 4, 6 and 8 according to their planes. Figure 4 depicts the points on the left plane, Figure 6 depicts points on the right plane and Figure 8 refers to points on the upper plane. On the other hadn, Figure 5, 7 and 9 provide results for this case in the same order of the Figure 4, 6 and 8. All of them are verified in terms of their $U\&V$ matrices and epipole-epipolar line intersections. Figure 3 has the partially correct estimations, though they have mistakes as well. However, Figure 4 and Figure 5 has much more inaccurate results, especially Figure 4 is the worst. Points from the same plane are the main cause, however points in left plane are much more clearly visible in both views in comparison to the other two planes. Upper plane is also visible in both views, even if it is not as clear as left plane. However, points in right plane are barely distinguishable from each other. As a consequence of both of these two reasons, points from one plane does not ensure decent pose recovery matrix estimations by using 8-points algorithm. Again, decrease in estimation accuracy can be seen in order of Figure 5, Figure 7 and Figure 9.

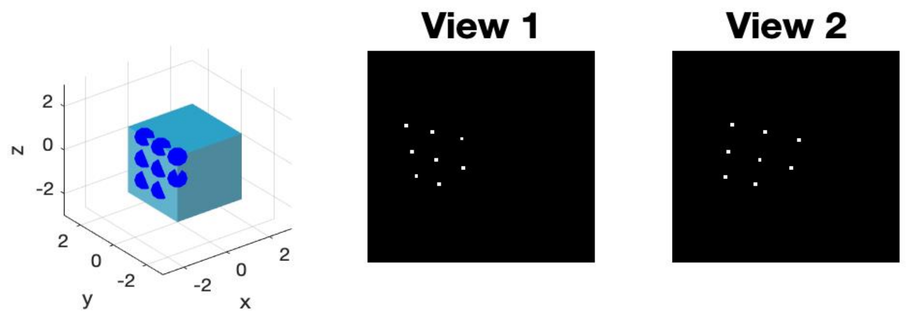
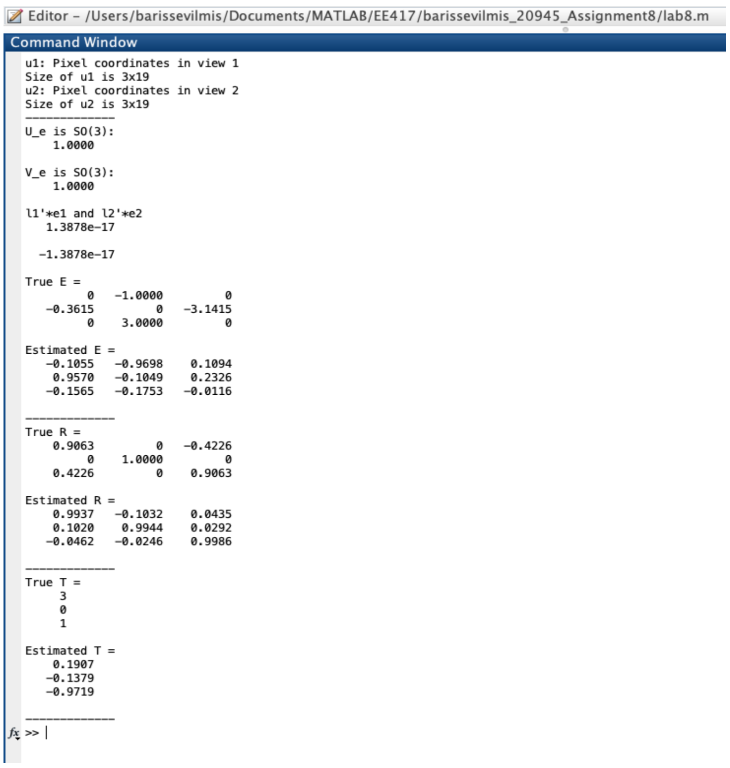


Figure 4: Visualization of Points-Left Plane-Case 2


Editor – /Users/barissevilmis/Documents/MATLAB/EE417/barissevilmis_20945_Assignment8/lab8.m
Command Window

```
u1: Pixel coordinates in view 1
Size of u1 is 3x19
u2: Pixel coordinates in view 2
Size of u2 is 3x19
-------------
U_e is SO(3):
    1.0000

V_e is SO(3):
    1.0000

l1'*e1 and l2'*e2
   1.3878e-17

  -1.3878e-17

True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
   -0.1055   -0.9698    0.1094
    0.9570   -0.1049    0.2326
   -0.1565   -0.1753   -0.0116

-------------
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R =
    0.9937   -0.1032    0.0435
    0.1020    0.9944    0.0292
   -0.0462   -0.0246    0.9986

-------------
True T =
     3
     0
     1

Estimated T =
    0.1907
   -0.1379
   -0.9719

-------------
fx >> |
```

Figure 5: Results of Case 2- Left Plane

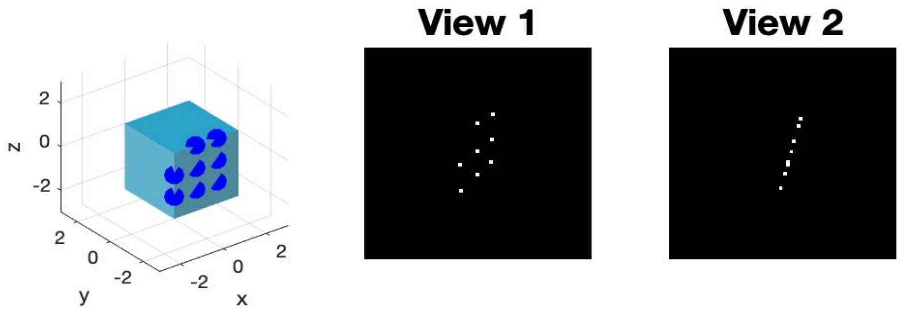
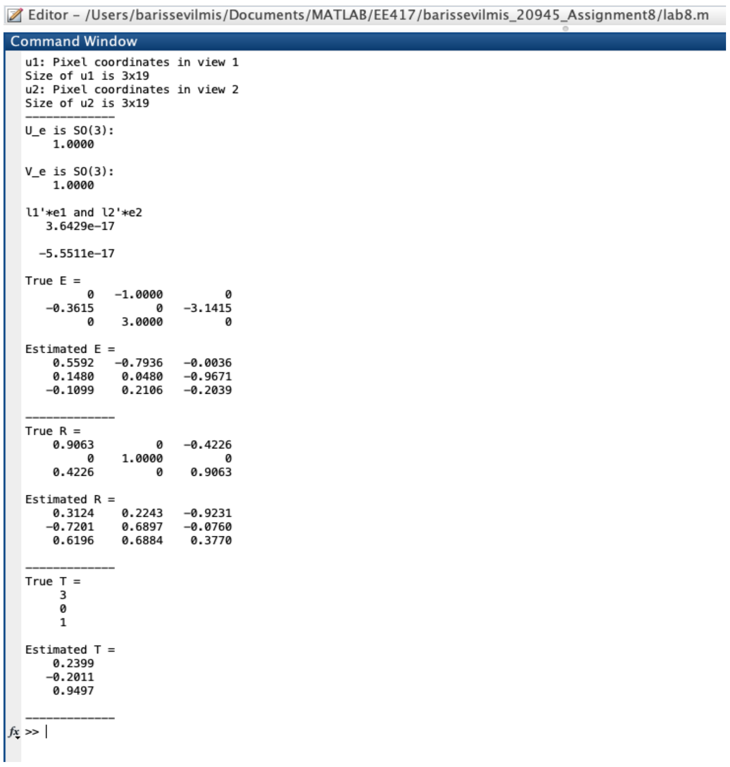


Figure 6: Visualization of Points-Right Plane-Case 2



```
Editor - /Users/barissevilmis/Documents/MATLAB/EE417/barissevilmis_20945_Assignment8/lab8.m
Command Window
u1: Pixel coordinates in view 1
Size of u1 is 3x19
u2: Pixel coordinates in view 2
Size of u2 is 3x19
-------------
U_e is SO(3):
    1.0000

V_e is SO(3):
    1.0000

l1'*e1 and l2'*e2
   3.6429e-17

  -5.5511e-17

True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
    0.5592   -0.7936   -0.0036
    0.1480    0.0480   -0.9671
   -0.1099    0.2106   -0.2039

-------------
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R =
    0.3124    0.2243   -0.9231
   -0.7201    0.6897   -0.0760
    0.6196    0.6884    0.3770

-------------
True T =
     3
     0
     1

Estimated T =
    0.2399
   -0.2011
    0.9497

-------------
fx >> |
```

Figure 7: Results of Case 2- Right Plane

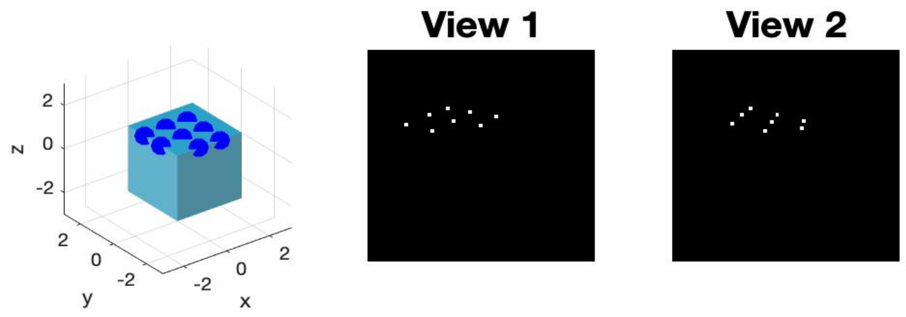
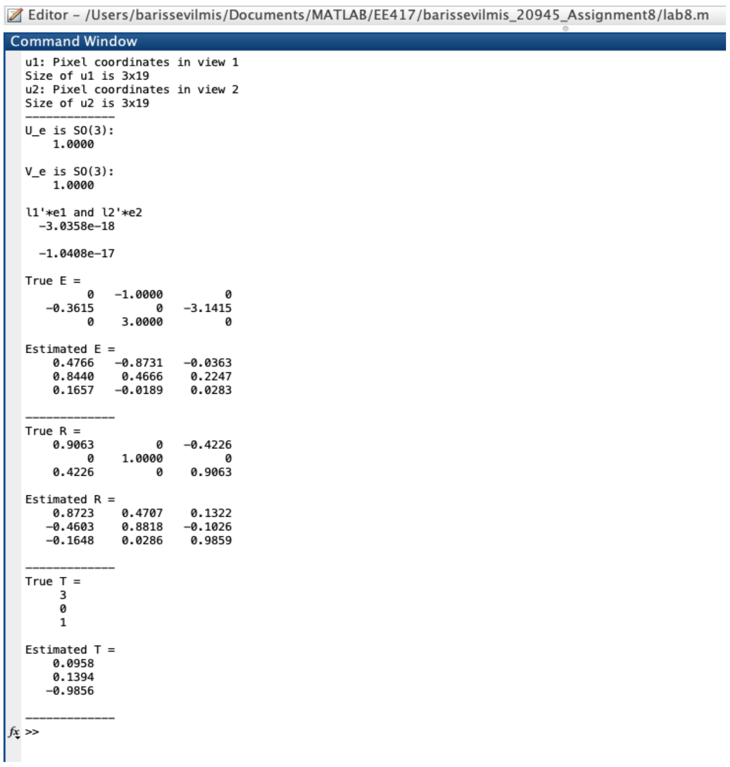


Figure 8: Visualization of Points-Upper Plane-Case 2



Editor – /Users/barissevilmis/Documents/MATLAB/EE417/barissevilmis_20945_Assignment8/lab8.m
Command Window

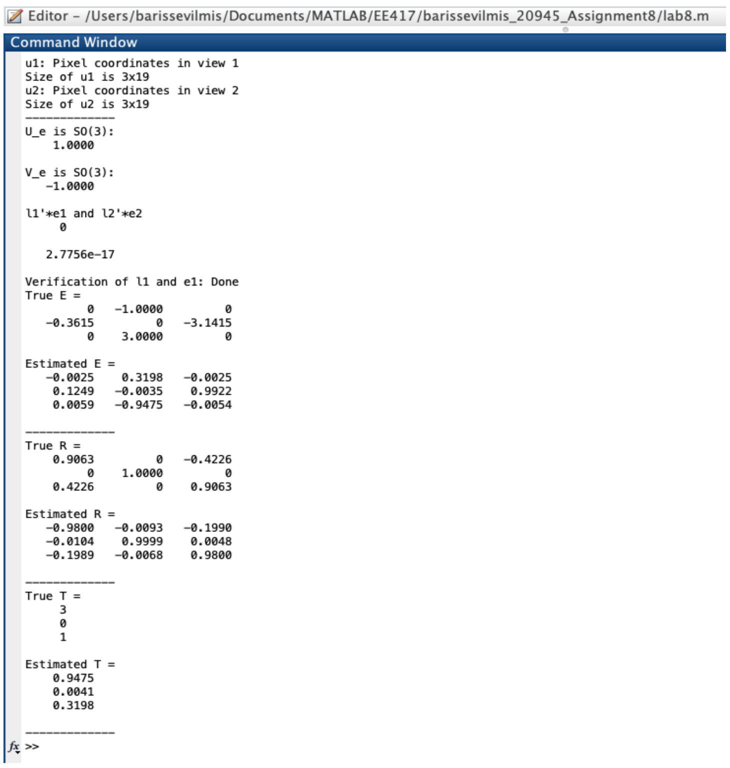
```
u1: Pixel coordinates in view 1
Size of u1 is 3x19
u2: Pixel coordinates in view 2
Size of u2 is 3x19
-------------
U_e is SO(3):
    1.0000

V_e is SO(3):
    1.0000

l1'*e1 and l2'*e2
  -3.0358e-18

  -1.0408e-17

True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
    0.4766   -0.8731   -0.0363
    0.8440    0.4666    0.2247
    0.1657   -0.0189    0.0283

-------------
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R =
    0.8723    0.4707    0.1322
   -0.4603    0.8818   -0.1026
   -0.1648    0.0286    0.9859

-------------
True T =
     3
     0
     1

Estimated T =
    0.0958
    0.1394
   -0.9856

-------------
>>
```

Figure 9: Results of Case 2- Upper Plane

Case 3: All points

In this case, an overall assumption is tested, which is caused by the Camera Calibration assumption. This assumption implies that picking more points will result in better recovery. However, picking every possible point is not a well thought approach, even in 8 points picking random 8 points did not always give the best result. For the sake of visualization, Figure 1 demonstrates this case. In our case, picking every possible point results in Figure 10, as Matrix Characterization Theorem is not satisfied. Therefore, essential matrix E is not cured and rest of the estimated results(R&T) are wrong. For instance, signs of T matrix seems to be correct, though the signs of R matrix are incorrect. This is because of the non-cured essential matrix. As mentioned earlier, R&T matrices provided in Figure 6, are the physically correct estimations and the other estimation results, that are not provided, are negative version of our estimations.

We have seen that picking all points causes an non-curable essential matrix. However, picking more than 8 points can also cause better estimation results. Case 4 proves this situation.



Editor – /Users/barissevilmis/Documents/MATLAB/EE417/barissevilmis_20945_Assignment8/lab8.m

Command Window

```
u1: Pixel coordinates in view 1
Size of u1 is 3x19
u2: Pixel coordinates in view 2
Size of u2 is 3x19
-------------
U_e is SO(3):
    1.0000

V_e is SO(3):
   -1.0000

l1'*e1 and l2'*e2
     0

   2.7756e-17

Verification of l1 and e1: Done
True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
   -0.0025    0.3198   -0.0025
    0.1249   -0.0035    0.9922
    0.0059   -0.9475   -0.0054

-------------
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R =
   -0.9800   -0.0093   -0.1990
   -0.0104    0.9999    0.0048
   -0.1989   -0.0068    0.9800

-------------
True T =
     3
     0
     1

Estimated T =
    0.9475
    0.0041
    0.3198

-------------
>>
```

Figure 10: Results of Case 3- All Points

Case 4: More than 8 points ensuring optimal pose recovery

As shown in Figure 10, picking all the points may not always prove useful in terms of pose recovery. However, picking more than 8 points carefully ensures better estimations. Following 12 points were used instead of the 18 points:

$$points = \begin{bmatrix} 4 & 5 & 7 & 8 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 \end{bmatrix}$$

Notice that these points are are 4 points on each plane that are not on intersecting lines between planes. Figure 11 clearly shows these points from both views and on the graph.

Figure 12 depicts the results for this 14 points as shown below. Verification steps are same as before, as $U \& V \in SO(3)$ and epipoles are on epipolar lines. For the essential matrix E, due to scaling values are not same as true E, yet similarity between them is obvious despite the scale difference. R matrices are very similar, and as mentioned earlier R is not up to scale, therefore can be directly compared. Estimated R values are almost the same with true R. Lastly, estimated T matrix can be compared with true T, and it will be clear that T is also up to scale. However similarity between estimated matrices and true matrices are depicted clearly in Figure 7.

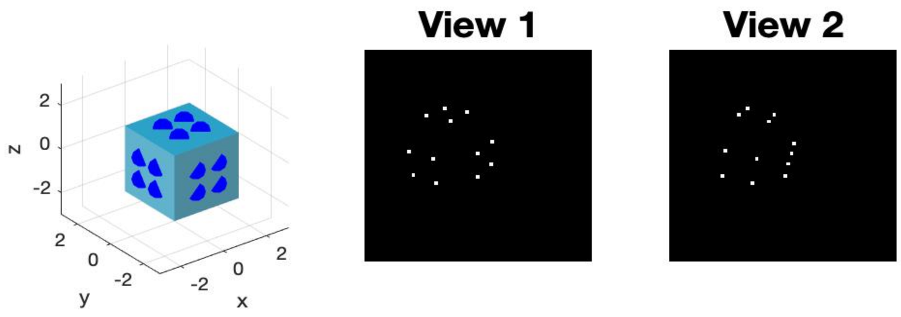
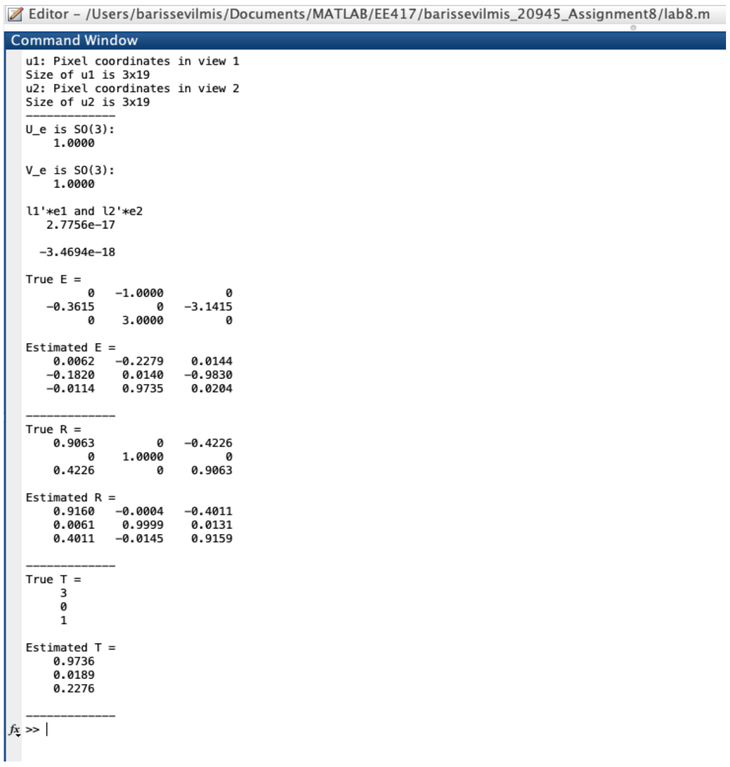


Figure 11: Visualization of Points-Upper Plane-Case 4


Editor – /Users/barissevilmis/Documents/MATLAB/EE417/barissevilmis_20945_Assignment8/lab8.m
Command Window

```
u1: Pixel coordinates in view 1
Size of u1 is 3x19
u2: Pixel coordinates in view 2
Size of u2 is 3x19
-------------
U_e is SO(3):
    1.0000

V_e is SO(3):
    1.0000

l1'*e1 and l2'*e2
   2.7756e-17

  -3.4694e-18

True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
    0.0062   -0.2279    0.0144
   -0.1820    0.0140   -0.9830
   -0.0114    0.9735    0.0204

-------------
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R =
    0.9160   -0.0004   -0.4011
    0.0061    0.9999    0.0131
    0.4011   -0.0145    0.9159

-------------
True T =
     3
     0
     1

Estimated T =
    0.9736
    0.0189
    0.2276

-------------
>> |
```

Figure 12: Results of Case 3- 12 points