Computer Vision Lab#8

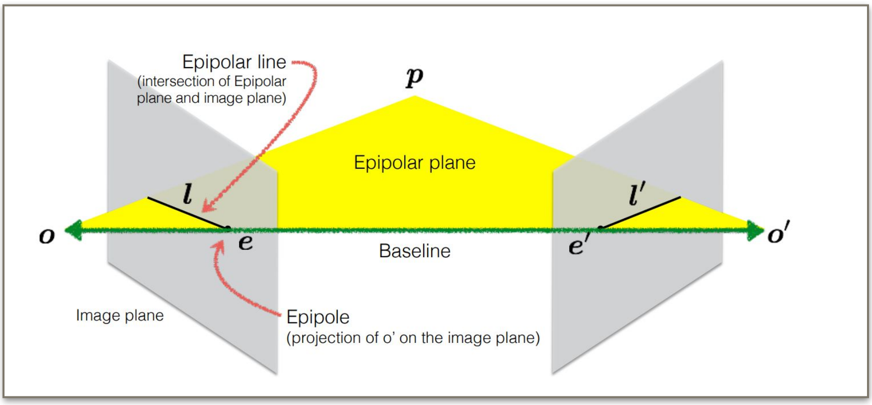
Goktug Korkulu

INTRODUCTION

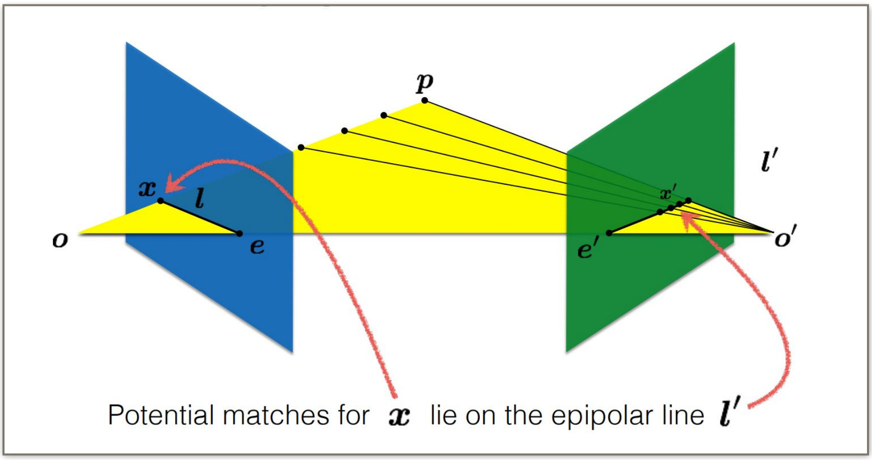
Hello and welcome to the week 8 lab report. I hope it will be an enjoyable read for you.

First, let's remember what we did last week. Last week, we had 2 different images of almost the same scene, captured with a very small angle change, and we described them as the 'left image' and the 'right image'. Using these two different images, we tried to recover the depth information of each object in the scene. While doing this, the most important and time-consuming part was to obtain the corresponding pixel coordinates in the right picture for each pixel coordinates in the left picture. Fortunately, our job of searching for the corresponding pixel for each single pixel on the left image was only on one single line, not all points on the right image. Otherwise, considering that even now this process takes 3-4 minutes in total, it would probably cost us hours to search through the whole image. So, how did we drop this search process on only 1 line instead of all the coordinates on the picture? We will learn the answer to this question in our laboratory work this week.

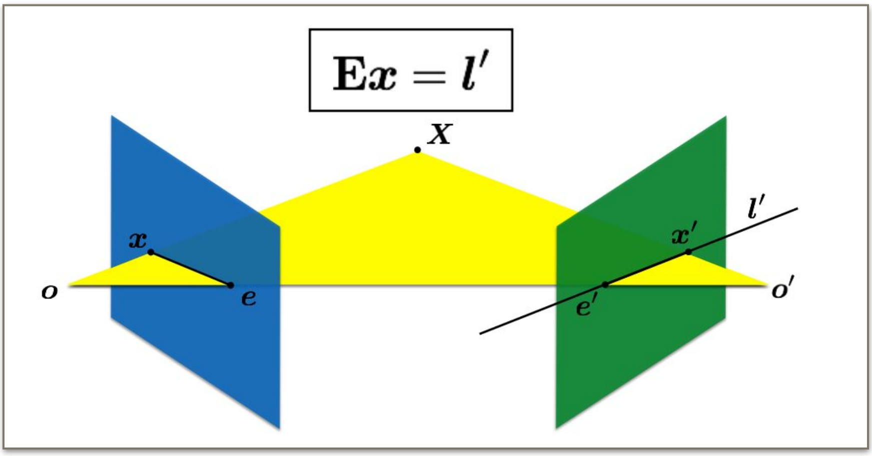
Let's wrap up everything we talked so far and give more scientific look. The *corresponding point search line* we are talking about is referred to as 'Epipolar Line' in computer vision literature. If we need to visualize it geometrically, the lines named l and l' in the diagram below is our Epipolar Lines.


Epipolar line
(intersection of Epipolar plane and image plane)
p
Epipolar plane
l
l′
o
e
Baseline
e′
o′
Image plane
Epipole
(projection of o′ on the image plane)

To sum up, we want to avoid search over entire image and Epipolar Constraint reduces search to a single line. The question is now, how do we compute the Epipolar Line? Answer : Essential Matrix. Essential matrix is a 3x3 matrix that encodes epipolar geometry.
Epipolar Constraint asserts that, potential match for x coordinate on the left image lies on the epipolar line of the other image plane. Visually,



Therefore, by combining all the things mentioned so far; Given a point in one image, multiplying by the ***essential matrix*** will tell us the epipolar line on the second view.



Note that, since x is on the epipolar line l, tran(x)*l = 0. Similarly, tran(x')*l' = 0. Thus,

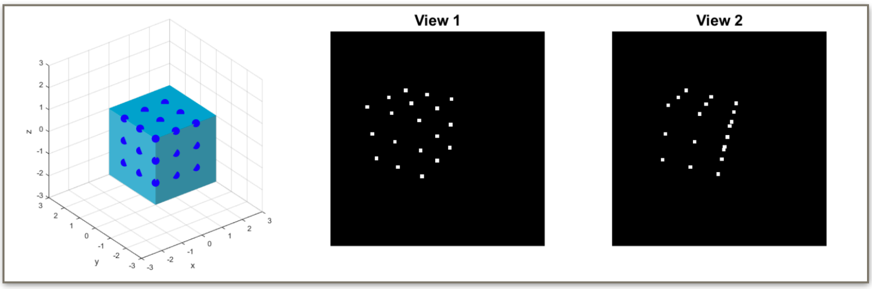
So if $\boldsymbol{x}'^{\top}\boldsymbol{l}' = 0$ and $\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'$ then

$$\boxed{\boldsymbol{x}'^{\top}\mathbf{E}\boldsymbol{x} = 0}$$

Notice that, by this equation and sufficient amount of known x - x' pairs, we can estimate the Essential matrix and apply it onto the entire coordinate values in order to get the corresponding coordinates on the other image plane.

METHOD

In this lab we will recover the pose of a camera from two images taken from different viewpoints by estimating the Essential Matrix (E) using 8-point algorithm. To do that we will make use of epipolar geometry. First, let's run "lab8.m" file to generate 3D world points and obtain their corresponding image points from two different views.



3D world points and their corresponding image points from two different views

STEP 1:
Note that the matrix K which includes intrinsic camera parameters is already known. Thus, I will start my implementation by transforming points in pixel coordinates in both views by multiplying them with the inverse of K matrix. Initially, I will select 8 point pairs properly to avoid degenerate configuration for estimating a reliable Essential Matrix. For each point pair, I will obtain the vector **'a'** appropriately and stack 8 of them in the rows of the **'X'** matrix as follows:

$$a = [x_1x_2 \quad x_1y_2 \quad x_1z_2 \quad y_1x_2 \quad y_1y_2 \quad y_1z_2 \quad z_1x_2 \quad z_1y_2 \quad z_1z_2]^T$$
$$X = [a_1^T;\ a_2^T;\ \dots\ a_8^T]$$

```
%% Transform pixel coordinates and construct X matrix using Equations 1 and 2

p1 = K^(-1)*(u1);
p2 = K^(-1)*(u2);

p1 = p1';
p2 = p2';

a = [p1(:,1).*p2(:,1) p1(:,1).*p2(:,2) p1(:,1).*p2(:,3) p1(:,2).*p2(:,1)
p1(:,2).*p2(:,2) p1(:,2).*p2(:,3) p1(:,3).*p2(:,1) p1(:,3).*p2(:,2)
p1(:,3).*p2(:,3)]';

X = [a(:,1)' ; a(:,2)' ; a(:,3)' ; a(:,4)' ; a(:,5)' ; a(:,6)' ; a(:,7)' ;
a(:,8)'];
```

STEP 2:

Since $XE^S = 0$, where E^S is the stacked version of the Essential Matrix, the eigenvector associated with the smallest eigenvalue of $X^T X$ gives the solution in stacked form. Now, I will obtain the Essential Matrix E and cure it to satisfy Essential Matrix Characterization theorem, and obtain the normalized essential matrix.

```
%% Estimate E, cure it and check for Essential Matrix Characterization
[U,S,V] = svd(X'*X);
E_stack = V(:,9);

E_estimation = [E_stack(1) E_stack(4) E_stack(7); E_stack(2) E_stack(5)
E_stack(8); E_stack(3) E_stack(6) E_stack(9)];

[U1, S1, V1] = svd(E_estimation);
S_essential = [1 0 0; 0 1 0; 0 0 0];

E_est = U1 * S_essential * V1' ;
```

STEP 3:

Next, find the epipoles (e_1,e_2) and the coefficients of epipolar lines for a selected point pair. Then, verify that the image point and the epipole for one view are on the epipolar line.

```
%% Find epipoles and epipolar lines

e1 = null(E_est);
e2 = null(E_est');

l1 = E_est' * p2(1,:)';
l2 = E_est  * p1(1,:)';

%% Verify epipoles and epipolar lines

c1 = l1' * e1;
c2 = l2' * e2;
```

STEP 4:

Finally, recover the rotation and the translation of the camera. Possible poses are recovered as follows:

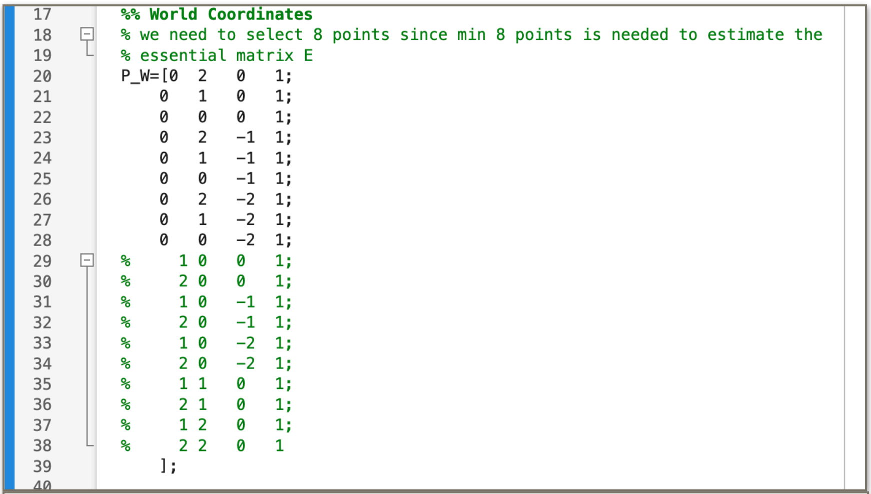
$$(\hat{T}_1, R_1) = (UR_z(+\frac{\pi}{2})\Sigma U^T, UR_z^T(+\frac{\pi}{2})V^T) \qquad (\hat{T}_2, R_2) = (UR_z(-\frac{\pi}{2})\Sigma U^T, UR_z^T(-\frac{\pi}{2})V^T)$$

```
%% Recover the rotation and the translation

%1)
az_PLUS = 90 * DEG_TO_RAD;
Rz_PLUS = [cos(az_PLUS) -sin(az_PLUS) 0;
            sin(az_PLUS) cos(az_PLUS)  0;
            0           0      1];
T1 = U1*Rz_PLUS*S_essential*U1';
R1 = U1*Rz_PLUS'*V1';

T1  = [T1(3,2); T1(1,3); T1(2,1)];

%2)
az_MINUS = -90 * DEG_TO_RAD;
Rz_MINUS = [cos(az_MINUS) -sin(az_MINUS) 0;
            sin(az_MINUS) cos(az_MINUS)  0;
            0           0      1];
T2 = U1*Rz_MINUS*S_essential*U1';
R2 = U1*Rz_MINUS'*V1';

T2  = [T2(3,2); T2(1,3); T2(2,1)];
```

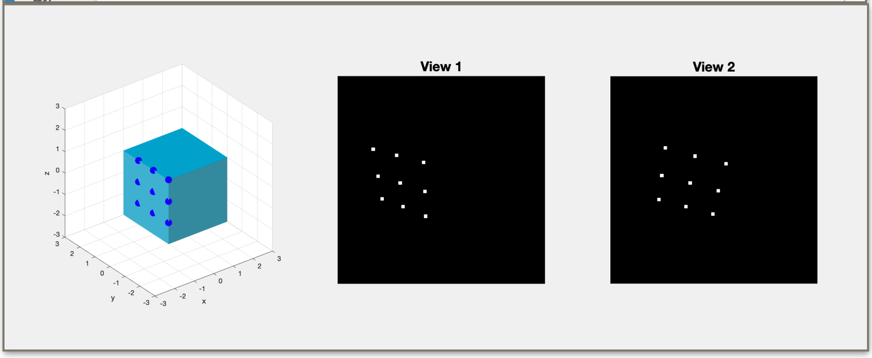
POST LAB DISCUSSION & CONCLUSION

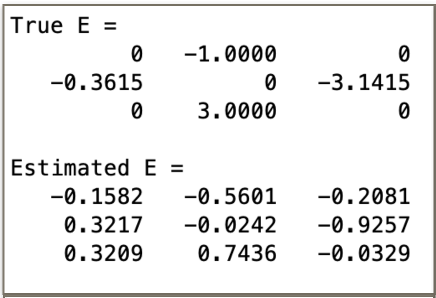
Select 8 points on the same plane and recover the pose of the camera by estimating the essential matrix. Repeat the same procedure for each plane.

STEP1: LEFT PLANE

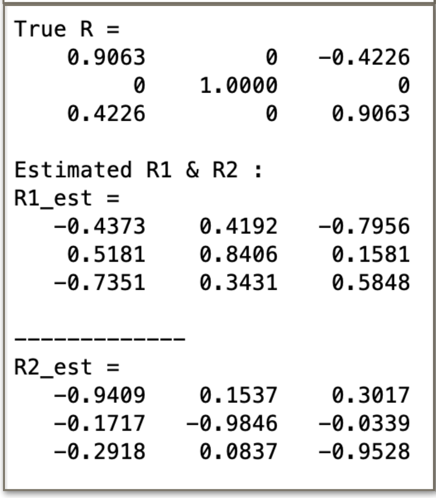


```
%% World Coordinates
% we need to select 8 points since min 8 points is needed to estimate the
% essential matrix E
P_W=[0  2   0   1;
    0   1   0   1;
    0   0   0   1;
    0   2   -1  1;
    0   1   -1  1;
    0   0   -1  1;
    0   2   -2  1;
    0   1   -2  1;
    0   0   -2  1;
%      1 0   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
    ];
```

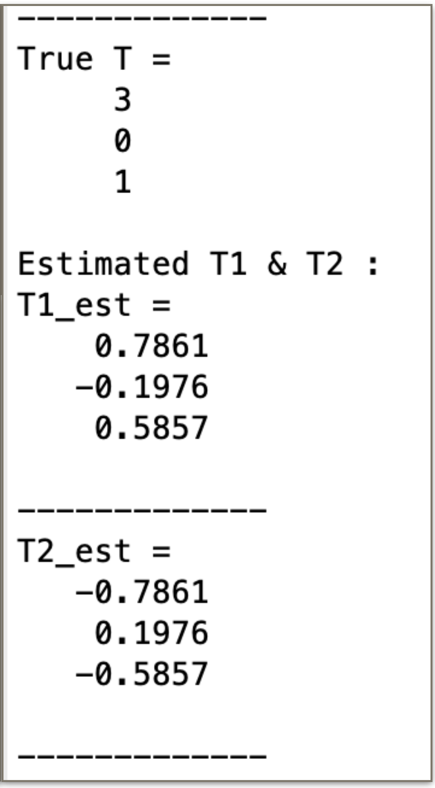




```
True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
   -0.1582   -0.5601   -0.2081
    0.3217   -0.0242   -0.9257
    0.3209    0.7436   -0.0329
```

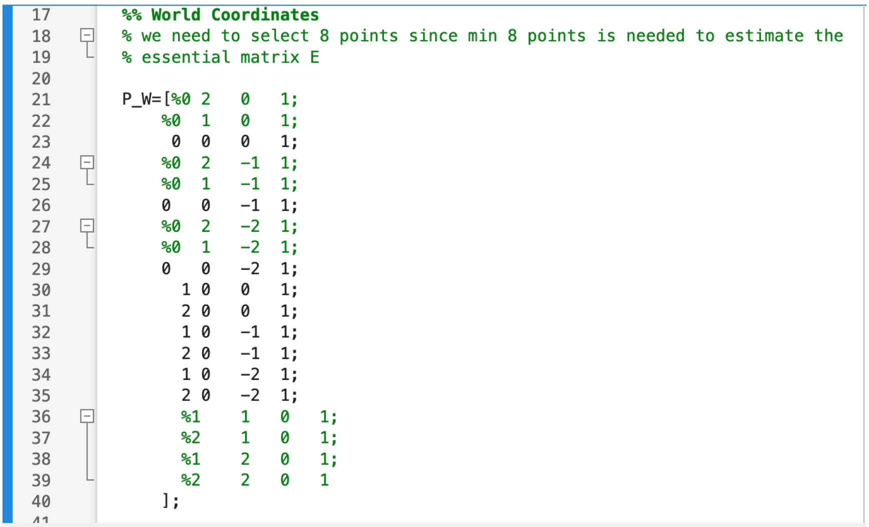


```
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R1 & R2 :
R1_est =
   -0.4373    0.4192   -0.7956
    0.5181    0.8406    0.1581
   -0.7351    0.3431    0.5848

-------------
R2_est =
   -0.9409    0.1537    0.3017
   -0.1717   -0.9846   -0.0339
   -0.2918    0.0837   -0.9528
```

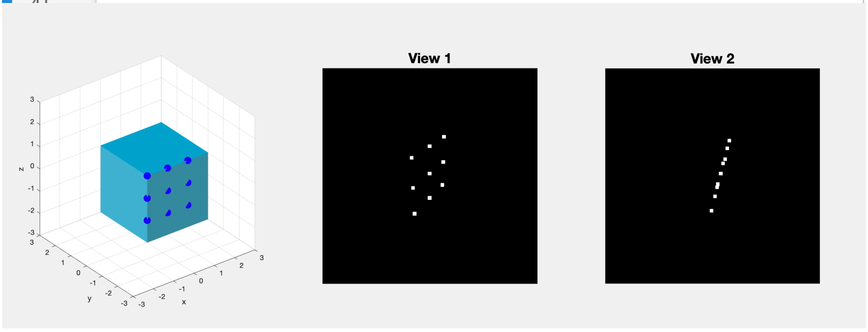


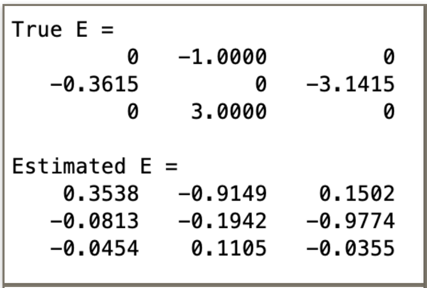
```
-------------
True T =
     3
     0
     1

Estimated T1 & T2 :
T1_est =
    0.7861
   -0.1976
    0.5857


-------------
T2_est =
   -0.7861
    0.1976
   -0.5857


-------------
```

STEP2: RIGHT PLANE

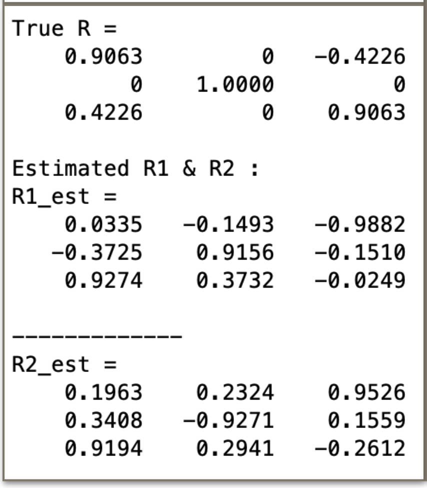


```
%% World Coordinates
% we need to select 8 points since min 8 points is needed to estimate the
% essential matrix E

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
    ];
```

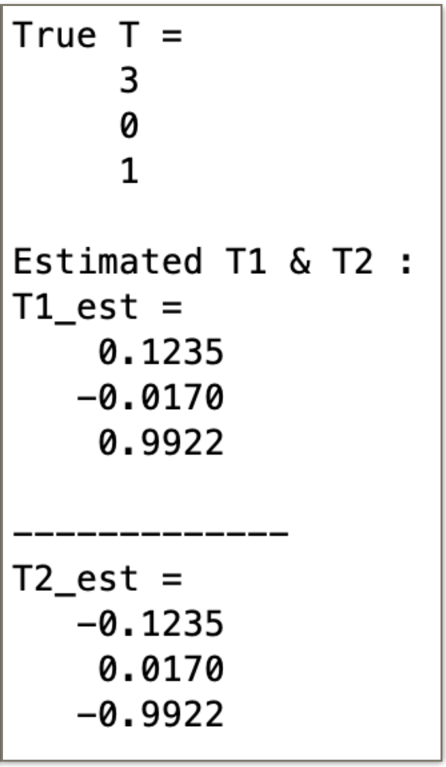




```
True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
    0.3538   -0.9149    0.1502
   -0.0813   -0.1942   -0.9774
   -0.0454    0.1105   -0.0355
```

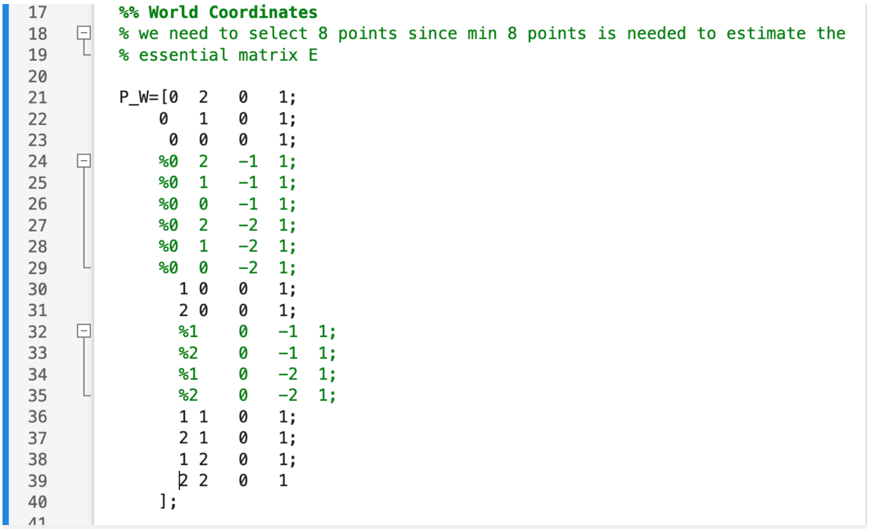


```
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R1 & R2 :
R1_est =
    0.0335   -0.1493   -0.9882
   -0.3725    0.9156   -0.1510
    0.9274    0.3732   -0.0249

-------------
R2_est =
    0.1963    0.2324    0.9526
    0.3408   -0.9271    0.1559
    0.9194    0.2941   -0.2612
```

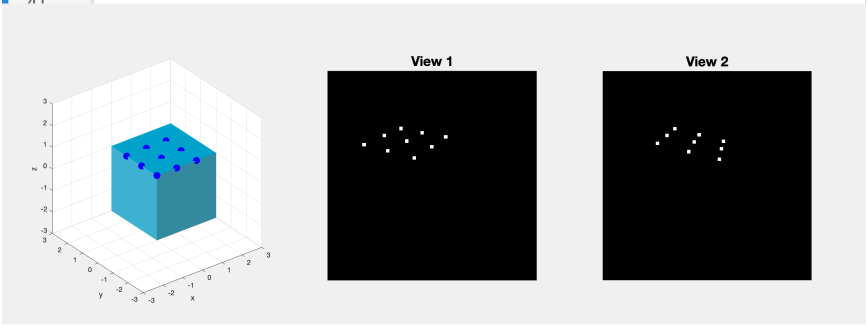


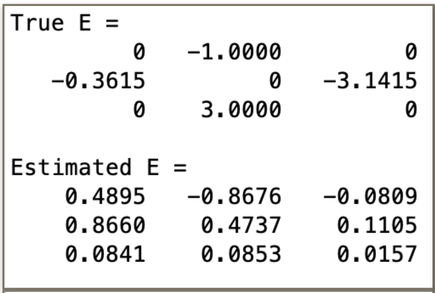
```
True T =
     3
     0
     1

Estimated T1 & T2 :
T1_est =
    0.1235
   -0.0170
    0.9922


-------------
T2_est =
   -0.1235
    0.0170
   -0.9922
```

STEP3: TOP PLANE

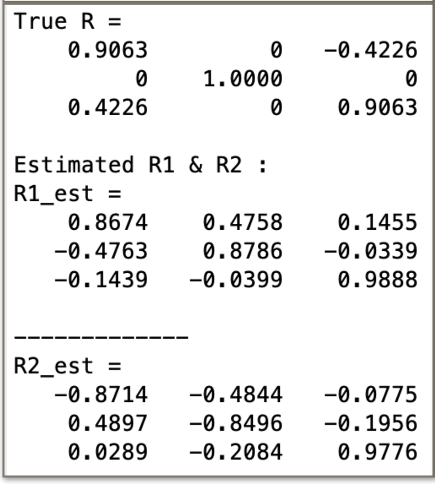


```
%% World Coordinates
% we need to select 8 points since min 8 points is needed to estimate the
% essential matrix E

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
    ];
```

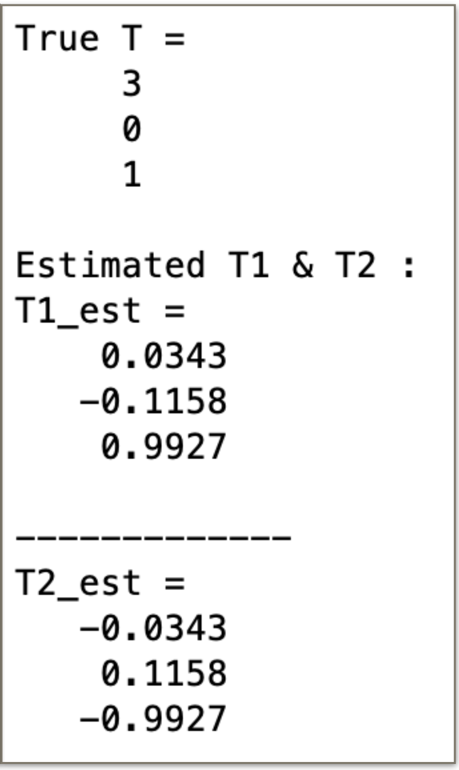




```
True E =
         0   -1.0000         0
   -0.3615         0   -3.1415
         0    3.0000         0

Estimated E =
    0.4895   -0.8676   -0.0809
    0.8660    0.4737    0.1105
    0.0841    0.0853    0.0157
```

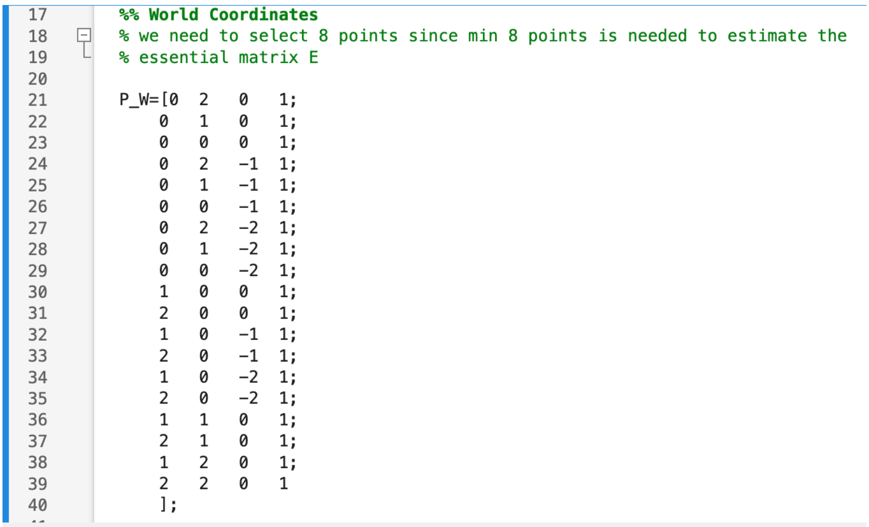


```
True R =
    0.9063         0   -0.4226
         0    1.0000         0
    0.4226         0    0.9063

Estimated R1 & R2 :
R1_est =
    0.8674    0.4758    0.1455
   -0.4763    0.8786   -0.0339
   -0.1439   -0.0399    0.9888

-------------
R2_est =
   -0.8714   -0.4844   -0.0775
    0.4897   -0.8496   -0.1956
    0.0289   -0.2084    0.9776
```

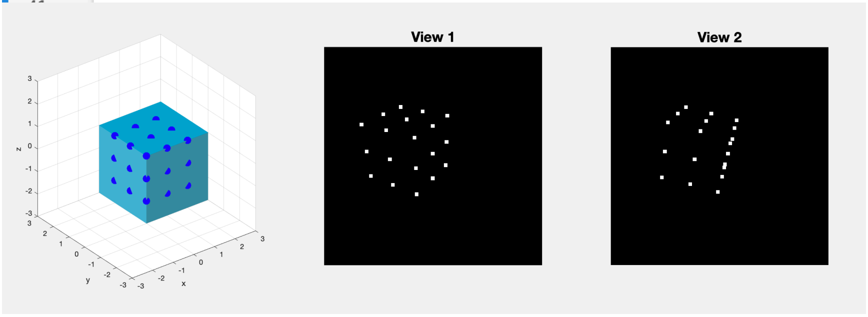


```
True T =
     3
     0
     1

Estimated T1 & T2 :
T1_est =
    0.0343
   -0.1158
    0.9927

-------------
T2_est =
   -0.0343
    0.1158
   -0.9927
```

STEP4: Now select all the points and recover the pose of the camera by estimating the essential matrix.



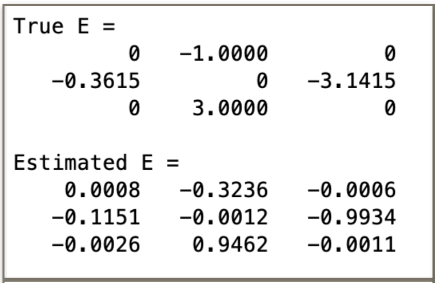
```
%% World Coordinates
% we need to select 8 points since min 8 points is needed to estimate the
% essential matrix E

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
    ];
```



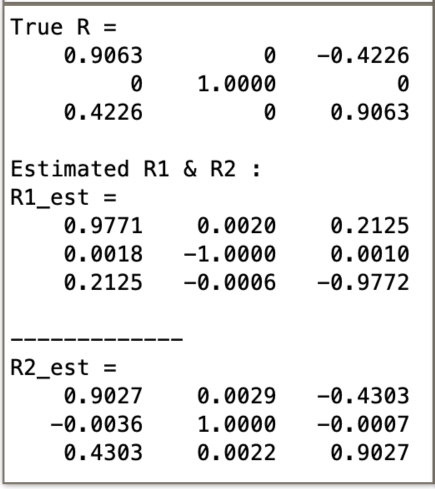
Also note that, I have changed my code in the part of creating X matrix as;

```
X = [a(:,1)'; a(:,2)'; a(:,3)'; a(:,4)'; a(:,5)';
a(:,6)'; a(:,7)'; a(:,8)'; a(:,9)';
a(:,10)';a(:,11)'; a(:,12)'; a(:,13)'; a(:,14)';
a(:,15)'; a(:,16)'; a(:,17)'; a(:,18)'; a(:,19)’];
```

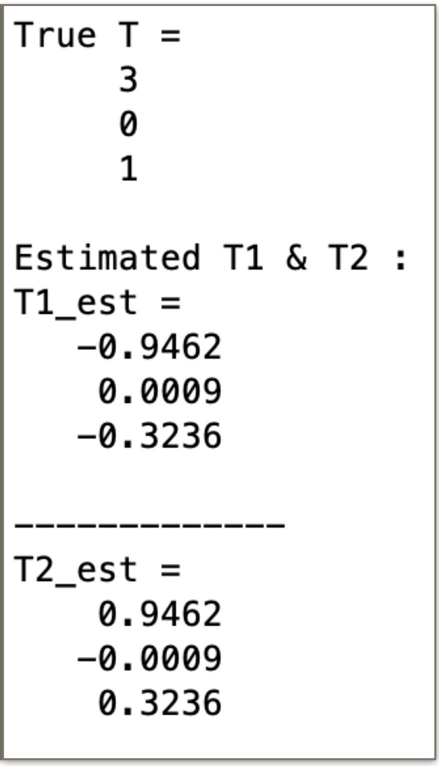
in order to select all he points to recover the pose of the camera.



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

Estimated E =
    0.0008   -0.3236   -0.0006
   -0.1151   -0.0012   -0.9934
   -0.0026    0.9462   -0.0011
```



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

Estimated R1 & R2 :
R1_est =
    0.9771    0.0020    0.2125
    0.0018   -1.0000    0.0010
    0.2125   -0.0006   -0.9772

-------------
R2_est =
    0.9027    0.0029   -0.4303
   -0.0036    1.0000   -0.0007
    0.4303    0.0022    0.9027
```



```
True T =
     3
     0
     1

Estimated T1 & T2 :
T1_est =
   -0.9462
    0.0009
   -0.3236

-------------
T2_est =
    0.9462
   -0.0009
    0.3236
```

As seen above, I have performed four different pose recovery method which is indicated by four steps above. First, I have picked the points form only left-plane by commenting out the other coordinate points and tried to recover the pose of the camera. As seen there, the estimated values are too far away from the real values. This situation holds also for step2 and step3 where I have chosen point only from right plane and top plane respectively. When you see the estimations and the real values of those single-plane estimations, you will realize that those methods are non-sense and never should be chosen. However, on the other hand, when I chose all of those points in step4, I have estimated the values of R, T, and E very close to real values. Please keep in mind that, rotation estimate is very close to the true rotation, while transition estimate is recovered up to a scale. So for all these reasons, I would choose the last recovered pose (the one with all the points have been used).

Thank you,

Goktug Korkulu

27026