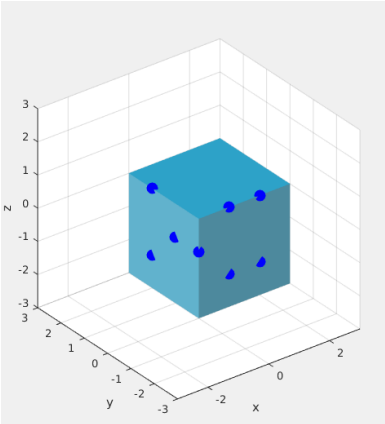
EE417 Computer Vision & Image Processing
Hakan Buğra Erentuğ 23637
10/12/19 Post Lab Report #8

In this assignment, we need to do pose recovery of a calibrated camera. Thanks to TA's the 3D world and basic cube object is already created. In the assignment there are 19 different point on various planes of the cube. To obtain better result we need to choice the most 8 point. It is strongly suggested to select point from different faces of the cube. Below, you can find the which 8 point is selected in our experiment.

There are various sub-tasks in our assignment. The first thing we need to do is calculated p1 and p2 camera pose matrices by the formula pi= inverse of K multiply Ui. Luckily, the u values were already calculated in the upper part of the codes which is I am not responsible. Then, we need to calculate a matrix which a = [x1x2 x1y2 x1z2 y1x2 y1y2 y1z2 z1x2 z1y2 z1z2] T for each point and X matrix consists of each a value from 1 to 8 for each point.



Then we go forward to next task, estimation of E and cure the results. The first thing we need to do is multiplying the X and transpose of X and take singular value decomposition of that. As you already know, every square matrix can decompose to three disticnt matrix. Sigma diagonal matrix, U and V square matrix which has determinant=1. Then we can Estimate the E essential matrix by V matrix's ninth element. The next thing we need to do is curing the estimation. If we take

1 0 0
0 1 0 as new sigma and calculate U and V by SVD of new estimation and found U*S'* V'
0 0 0

we will found true E value with a coefficient. In our example If we multiply our estimation with -3 we get very similar solution. Please check output in the appendices to verify.

The next task we need to do find the epipoles and epipolar lines. Which has very hard theory behind that. We found the epiploes by finding null space of E estimation and found epipolar lines bny taking E_estimation and pi multiplication. And after done that, c1=l1' * e1, c2=l2' * e2.

The last thing is find the rotation matrix, and found the transform head matrix 3x3 and taking the transform by (3,2) -(3,1) (2,1) points of 3x3.

EE417 Computer Vision &
Hakan Buğra Erentuğ 2363
10/12/19 Post Lab Report #

Appendices:

```
E_est =

   -0.0141   -0.9384   -0.0443
   -0.3188    0.0570   -2.9823
    0.0327    2.8487    0.0417


l1 =

    0.0842
    3.0601
    0.5027


l2 =

    0.0965
   -2.8869
   -0.3826

c1
  -6.9389e-17
c2
  -4.1633e-17

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

Estimated E =
   -0.0141   -0.9384   -0.0443
   -0.3188    0.0570   -2.9823
    0.0327    2.8487    0.0417

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

Estimated R1 & R2 :
R1_est =
    0.9111    0.0058   -0.4122
    0.0021    0.9998    0.0189
    0.4122   -0.0181    0.9109

-------------
R2_est =
    0.9774   -0.0246    0.2099
   -0.0215   -0.9996   -0.0168
    0.2103    0.0119   -0.9776

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

Estimated T1 & T2 :
T1_est =
    2.8491
   -0.0292
    0.9391

-------------
T2_est =
   -2.8491
    0.0292
   -0.9391

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

EE417 Computer Vision & Image Processing
Hakan Buğra Erentuğ 23637
10/12/19 Post Lab Report #8

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

p1 = pinv(K)*(u1);
p2 = pinv(K)*(u2);

a = zeros(9, 8);

for i=1:8
    a(:,i) = [p1(1, i) * p2(1,i) p1(1, i) * p2(2,i) p1(1, i) * p2(3,i) p1(2, i) * p2(1,i) p1(2, i) *
p2(2,i) p1(2, i) * p2(3,i) p1(3, i) * p2(1,i) p1(3, i) * p2(2,i) p1(3, i) * p2(3,i)]';
end

X = a';
%% Estimate E, cure it and check for Essential Matrix Characterization

[U,S,V] = svd(X'*X);
Es = V(:,9);
E_est = [Es(1) Es(4) Es(7);
    Es(2) Es(5) Es(8);
    Es(3) Es(6) Es(9)];

[U_est,S_est,V_est] = svd(E_est);

S_est(1,1) = 1; S_est(2,2) = 1; S_est(3,3) = 0;


disp(det(U_est));
disp(det(V_est));

E_est = U_est*S_est'*V_est';
E_est=E_est*-3
%% 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;

disp("values")
disp(c1);
disp(c2);
%% Recover the rotation and the translation
az = pi/2;
Rz = [cos(az) -sin(az) 0;
      sin(az) cos(az)  0;
      0          0     1];

T1 = U_est*Rz*S_est*U_est';
R1 = U_est*Rz'*V_est';

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

az = -pi/2;
Rz = [cos(az) -sin(az) 0;
      sin(az) cos(az)  0;
      0          0     1];


T2 = U_est*Rz*S_est*U_est';
R2 = U_est*Rz'*V_est';
T2=[T2(3,2); -T2(3,1); T2(2,1)];
```