

Visual Perception

Compute the Fundamental Matrix from two simulated cameras

1. Introduce

In this course work, we focus on the epipolar geometry by computing the fundamental matrix from two simulated cameras. 20 object points in the 3D world have been used; we compared the results from different kinds of projection situations, with noise (distortion) and without noise. 8 points method is used to get the fundamental matrix. During the mathematical calculation when using 8 points method, two different ways, Least square and SVD, are used to solve the multiple linear regression problem.

All the detail analysis will be discussed in the following report. The stereo vision system we discussed is shown in figure 1.

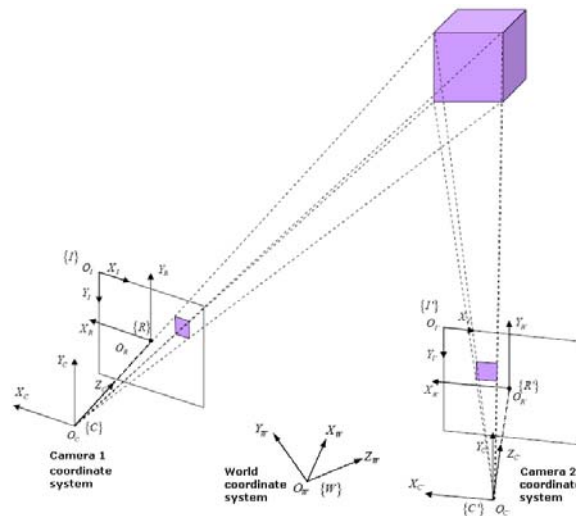


Fig. 1 Stereo vision system

2. Procedure

Part 1

Step 1 to Step 3

In these 3 steps, we use given parameters simulate two cameras in the world space with different intrinsic and extrinsic matrices. The world coordinate system is located at the focal point of camera 1 (camera on the left).

Step 1. Define camera 1 with the following parameters and set the world coordinate system to the coordinate system of camera 1 (Rotation=Identity and translation=0):

au1 = 100; av1 = 120; uo1 = 128; vo1 = 128;

Image size: 256 x 256

Step 2. Define camera 2 with respect to camera 1 with the following parameters:

au2 = 90; av2 = 110; uo2 = 128; vo2 = 128;

ax = 0.1 rad; by = pi/4 rad; cz = 0.2 rad; XYZ EULER

tx = -1000 mm; ty = 190 mm; tz = 230 mm;

Image size: 256 x 256

Step 3. Get the intrinsic transformation matrices of both cameras, and the rotation and translation between both cameras. Be aware that the dimensions of the intrinsic matrices are 3x3, the rotation matrix is 3x3 and the translation vector is 1x3.

From the MATLAB calculation, the results of the calibration of the two cameras are as follow:

Camera 1 Intrinsic matrix =

100	0	128
0	120	128
0	0	1

Camera 1 Extrinsic matrix =

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Camera 2 Intrinsic matrix =

90	0	128
0	110	128
0	0	1

Camera 2 Rotation matrix =

0.6930	-0.1405	0.7071
0.2669	0.9611	-0.0706
-0.6697	0.2376	0.7036

Camera 2 Translation Matrix =

-1000
190
230

Step 4

From the calibration of the first 3 steps and some epipolar geometry, shown in figure 2, we can get the fundamental matrix

$$F = A'^{-t} R^t [t]_x A^{-1} \quad F' = A^{-t} [t]_x R A'^{-1}$$

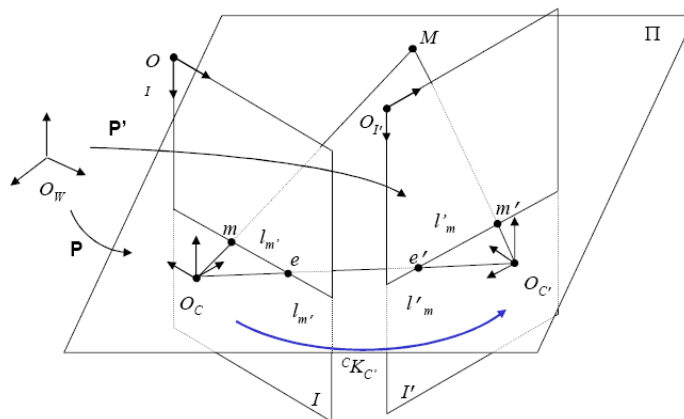


Fig. 2 Epipolar geometry

In the formula, A' and A are 3x3 matrixes and:

$$[t]_x = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}$$

The MATLAB actualization is as follow:

```
t_antisym=[0 -tz ty
           tz 0 -tx
           -ty tx 0];
F1=inv(A2')*R'*t_antisym*inv(A1);
F2=inv(A1')*t_antisym*R*inv(A2);
```

The result is:

The fundamental matrix F =

0.0210	0.0473	-4.3028
0.0160	-0.0156	8.4389
-6.2288	-11.2758	650.1769

The fundamental matrix F' =

-0.0210	-0.0160	6.2288
-0.0473	0.0156	11.2758
4.3028	-8.4389	-650.1769

Step 5 to Step 7

During this 3 steps, 20 given 3D world space points will be projected to the both image plane of the two cameras by using the calibration of these two camera obtain in previous steps.

```
intr1=[A1 zeros(3,1)];
intr2=[A2 zeros(3,1)];
K_wresptol=[eye(3) zeros(3,1)];
extr1=[K_wresptol;zeros(1,3) 1];
K_wrespto2=K_lrespto2;
extr2=[K_wrespto2;zeros(1,3) 1];

V_I1=point_projection(intr1,extr1,V);
V_I2=point_projection(intr2,extr2,V);
```

```
function V_I1=point_projection(intr1,extr1,V)
V_I1=intr1*extr1*V;
V_I1(1,:)=V_I1(1,:)./V_I1(3,:);
V_I1(2,:)=V_I1(2,:)./V_I1(3,:);
V_I1(3,:)=1;
```

The result show in figure 3:

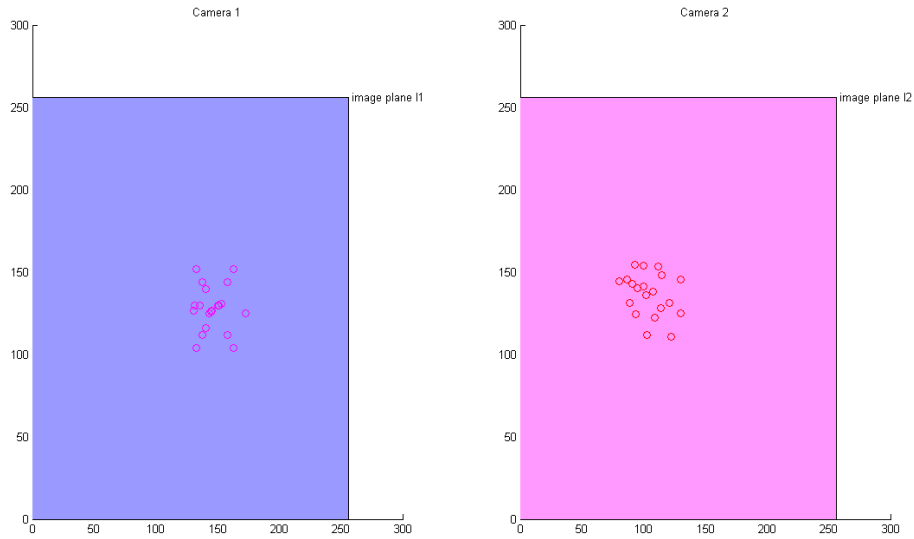


Fig. 3 Points projection in two camera image planes

Step 8

In this step, we will compute the fundamental matrix using least-squares 8-point method by mean of the 2D points obtained in previous steps.

The epipolar geometry is defined as:

$$m^T \mathbf{F}' m' = 0 \quad [x_i \quad y_i \quad 1] \mathbf{F}' \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = 0$$

Operating, we obtain:

$$U_n f = 0$$

$$U_n = (u_1, u_2, \dots, u_n)$$

$$u_i = (x'_i x_i, y'_i x_i, x_i, x'_i y_i, y'_i y_i, y_i, x'_i, y'_i, 1)$$

$$f = (F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33})^t$$

The first solution for: $U_n f = 0$ is $f = 0$ which is not wanted.

\mathbf{F} is defined up to a scale factor, so we can fix one of the components to 1.

Let's fix $\mathbf{F}_{33}=1$.

$$U'_n f' = -1_n$$

$$U'_n = (u'_1, u'_2, \dots, u'_n)$$

$$u'_i = (x'_i x_i, y'_i x_i, x_i, x'_i y_i, y'_i y_i, y_i, x'_i, y'_i)$$

$$f' = (F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32})^t$$

Then,

$$U_n'^{-1} U'_n f' = -U_n'^{-1} 1_n$$

$$f' = -U_n'^{-1} 1_n \quad \Longrightarrow \quad f' = -(U_n'^t U'_n)^{-1} U_n'^t 1_n$$

The actualization using MABLAB is as follow:

```
function F1_8pm=eightpointsmethod1(V_I1,V_I2)
    U1=[V_I1(1,:).*V_I2(1,:)
        V_I1(2,:).*V_I2(1,:)
        V_I2(1,:)
        V_I1(1,:).*V_I2(2,:)
        V_I1(2,:).*V_I2(2,:)
        V_I2(2,:)
        V_I1(1,:)
        V_I1(2,:)]';
    F1_8pm=-inv(U1'*U1)*U1'*ones(20,1);
    F1_8pm=[F1_8pm(1) F1_8pm(2) F1_8pm(3)
            F1_8pm(4) F1_8pm(5) F1_8pm(6)
            F1_8pm(7) F1_8pm(8) 1];
```

```
function F2_8pm=eightpointsmethod2(V_I1,V_I2)
    U2=[V_I2(1,:).*V_I1(1,:)
        V_I2(2,:).*V_I1(1,:)
        V_I1(1,:)
        V_I2(1,:).*V_I1(2,:)
        V_I2(2,:).*V_I1(2,:)
        V_I1(2,:)
        V_I2(1,:)
        V_I2(2,:)]';
    F2_8pm=-inv(U2'*U2)*U2'*ones(20,1);
    F2_8pm=[F2_8pm(1) F2_8pm(2) F2_8pm(3)
            F2_8pm(4) F2_8pm(5) F2_8pm(6)
            F2_8pm(7) F2_8pm(8) 1];
```

The first part is for camera 1 and the second part is for camera 2.

The result of fundamental matrix gotten from least-squares 8 point method is as follows:

Least-square 8 point method Fundamental matrix F =

0.000032234708306	0.000072675805435	-0.006617839383324
0.000024596847303	-0.000023922569127	0.012979428506412
-0.009580203229746	-0.017342632844176	1.000000000000000

Least-square 8 point method Fundamental matrix F' =

0.000032234708306	0.000024596847304	-0.009580203229657
0.000072675805432	-0.000023922569127	-0.017342632843523
-0.006617839383090	0.012979428506156	1.000000000000000

Step 9

In this step, we will compare the fundamental matrix gotten from least-square 8 points method with the real fundamental matrix.

```
F1_normal=F1/F1(3,3);
F2_normal=F2/F2(3,3);
```

Before compare, we need normalize the real fundamental matrix by setting value of F_{33} and F'_{33} equal to 1.

MATLAB code is as follow:

```
different1=F1_8pm-F1_normal;
different2=F2_8pm-F2_normal;
```

The different is as follows:

Different1 = 1.0e-011 *

-0.000112627599304	-0.001126785568063	0.080470352603612
0.000151821846653	0.000465342960563	-0.261489961628225
0.027829474835706	0.250218595843066	0

Different2 = 1.0e-011 *

-0.000150788805270	0.000167089446120	0.036710218198621
-0.001487602630055	0.000518076521353	0.315586792809519
0.103917829202826	-0.287067765392113	0

From the order of magnitude of the different, 10^{-11} , compare with the order of magnitude in the fundamental matrix, the different is so small that we can say virtually the those two matrix are the "same".

Step 10

In this step, the aim is to plot all the epipolar geometry properly.

[Hint] An epipolar line is defined by $lm'=Fm=[u1,u2,u3]r$. Besides any point $[x,y,1]$ that lies on its corresponding epipolar line satisfies $[x,y,1][u1,u2,u3]r=0$. From this equation you can extract $y=mx+d$, so that you obtain m and d from the components of the epipolar line. Hence, fixing x in both boundaries of the image plane, you can obtain the y component and draw the epipolar line by using the Plot function.

[Hint] You can compute the epipoles by projecting the focal point of each camera to the image plane of the other. Moreover, the epipoles can be computed by intersecting two or more epipolar lines as all the epipolar lines cross at the epipole in the absence of noise.

Follow the hints; complete the MATLAB code as follows:

Function

```
draw_epipolar_geometry(F1,F2,V_I1,V_I2,x1_min,x1_max,x2_min,x2_max)
    l2=F1*V_I1;
    l1=F2*V_I2;

    x1=x1_min:x1_max;
    x2=x2_min:x2_max;

    m1=-l1(1,:)./l1(2,:);
    d1=-l1(3,:)./l1(2,:);
    m2=-l2(1,:)./l2(2,:);
    d2=-l2(3,:)./l2(2,:);

    for i=1:20
        y1(i,:)=m1(i)*x1+d1(i);
        y2(i,:)=m2(i)*x2+d2(i);
        subplot(1,2,1);hold on;
        plot(x1,y1(i,:), 'Color', 'b');
        xlim([x1_min x1_max]);
        ylim([x1_min x1_max]);
        axis square;

        subplot(1,2,2);hold on;
        plot(x2,y2(i,:), 'Color', 'm');
        axis square;
        xlim([x2_min x2_max]);
        ylim([x2_min x2_max]);
    end
```

Then draw of projection of the focal point of each camera on the others' image plane:

```
function draw_focal_point(t,intr1,extr1,intr2,extr2)
    oc2_resptow=[t;1];
    oc2_I1=intr1*extr1*oc2_resptow;
    oc2_I1=oc2_I1/oc2_I1(3);
    oc1_resptow=[0;0;0;1];
    oc1_I2=intr2*extr2*oc1_resptow;
    oc1_I2=oc1_I2/oc1_I2(3);
    subplot(1,2,1);hold on;
    stem(oc2_I1(1),oc2_I1(2), 'LineStyle', 'none', 'Color', 'r');
    subplot(1,2,2);hold on;
    stem(oc1_I2(1),oc1_I2(2), 'LineStyle', 'none', 'Color', 'b');
```

The result show in figure 4:

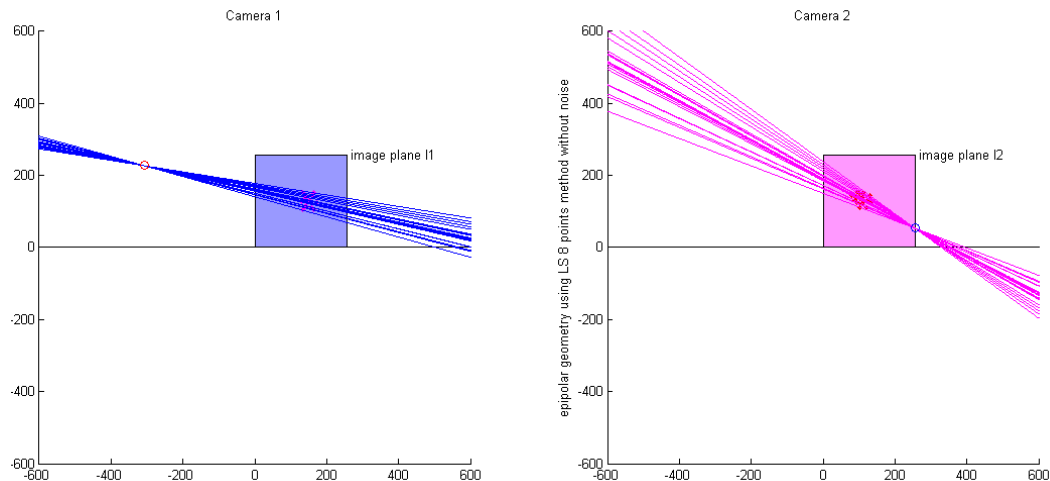


Fig. 4 Epipolar geometry using LS 8 points method without noise

In figure 4, the left part is Camera 1; the light blue square is the image plane of camera 1, 20 magenta points in the light blue square is the 20 projection points of object point in the world space, we can see that each of 20 points is lad on the one of the 20 blue lines which are the epipolar lines. All the epipolar lines are intersect at one point. The small red circle is the projection of the focal point of camera 2. We can see that this projection of the focal point is the same point of intersection of all the epipolar lines.

the right part is Camera 2; the light magenta square is the image plane of camera 2, 20 red points in the light magenta square is the 20 projection points of object point in the world space, we can see that each of 20 points is lad on the one of the 20 magenta lines which are the epipolar lines. All the epipolar lines are intersect at one point. The small blue circle is the projection of the focal point of camera 1. We can see that this projection of the focal point is the same point of intersection of all the epipolar lines.

Step 11 and Step 12

In this step, some Gaussian noise is added to the 2D points producing discrepancies between the range $[-1,+1]$ pixels for the 95% of points. Then repeat the above step to plot the epipolar geometry.

MATLAB code to add Gaussian noise:

```
mu=0;
sigma=0.5102;
V_I1_noise=V_I1+[normrnd(mu,sigma,2,20);zeros(1,20)];
V_I2_noise=V_I2+[normrnd(mu,sigma,2,20);zeros(1,20)];
```

The result is shown in figure 5, in the figure we can see that the epipolar lines are not intersect at a unique point and the projection of the focal point also is far away.

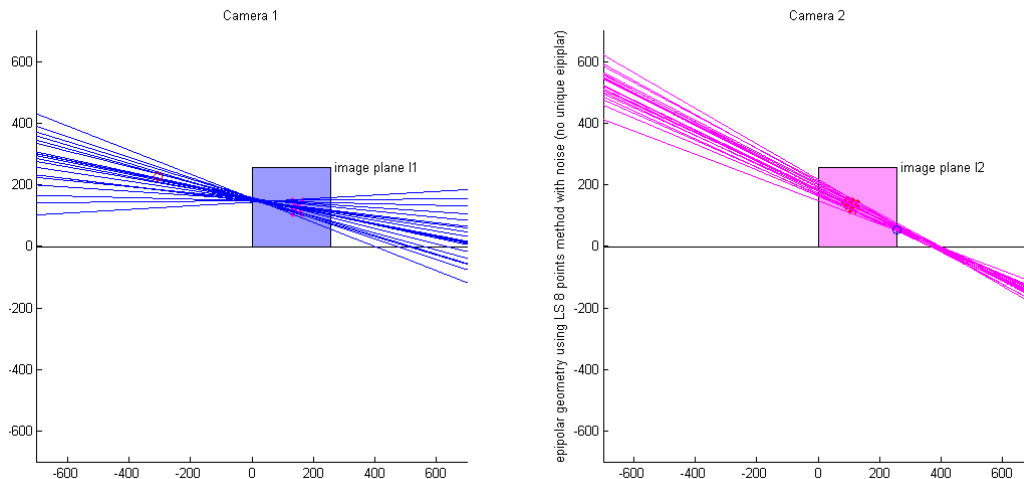


Fig. 5 Epipolar geometry using LS 8 points method with noise (no unique epipole)

The epipole is not unique, because the Fundamental matrix obtained is rank-3. We can compute the closest rank-2 matrix from any rank-3 matrix by SVD (Use the SVD function). That is, $F=UDV^T$. The diagonal matrix D is 3×3 and if we set the smallest eigenvalue to zero and calculate F again, we end up with a fundamental matrix of rank 2. Then, you can compute the epipolar lines from the rank-2 matrix and verify if all cross in the epipole.

The MATLAB code to do this is:

```
[Q1,S1,V1] = svd(F1_8pm_noise);
S1(3,3)=0;
F1_8pm_noise_svd=Q1*S1*V1';
[Q2,S2,V2] = svd(F2_8pm_noise);
S2(3,3)=0;
F2_8pm_noise_svd=Q2*S2*V2';
```

The result is shown in figure 6; we can compare it with figure 5. In fig. 6 all the epipolar lines are intersect at the unique point but not the projection of the focal point and the 2D points become away from its epipolar line not like in figure 5 where they are around the epipolar lines.

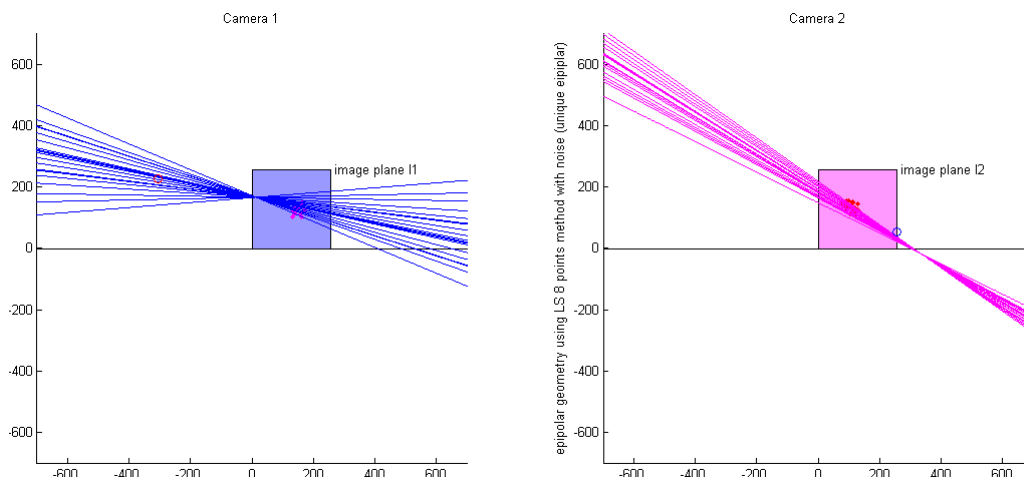


Fig. 6 Epipolar geometry using LS 8 points method with noise (unique epipole)

Step 13

In this step, Increase the Gaussian noise of step 11 (now in the range $[-2,+2]$ for the 95% of points) and repeat step 8-12.

The MATLAB code to do this is:

```
mu=0;
sigma=0.5102;
sigma2=2*sigma;
V_I1_noise2=V_I1+[normrnd(mu,sigma2,2,20);zeros(1,20)];
V_I2_noise2=V_I2+[normrnd(mu,sigma2,2,20);zeros(1,20)];
```

Figure 7 illustrates the situation without modify the fundamental matrix:

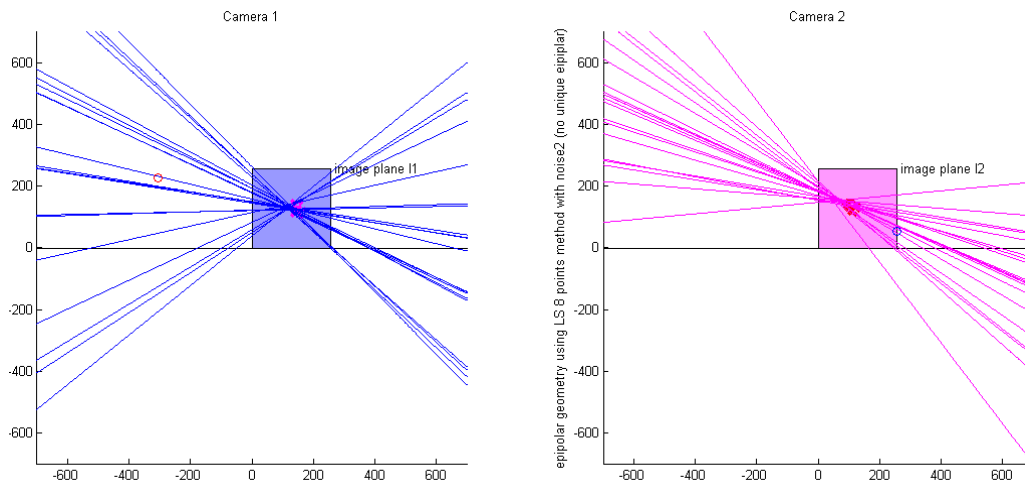


Fig. 7 Epipolar geometry using LS 8 points method with noise2 (no unique epipole)

Figure 8 illustrates the situation after modify the fundamental matrix to rank-2:

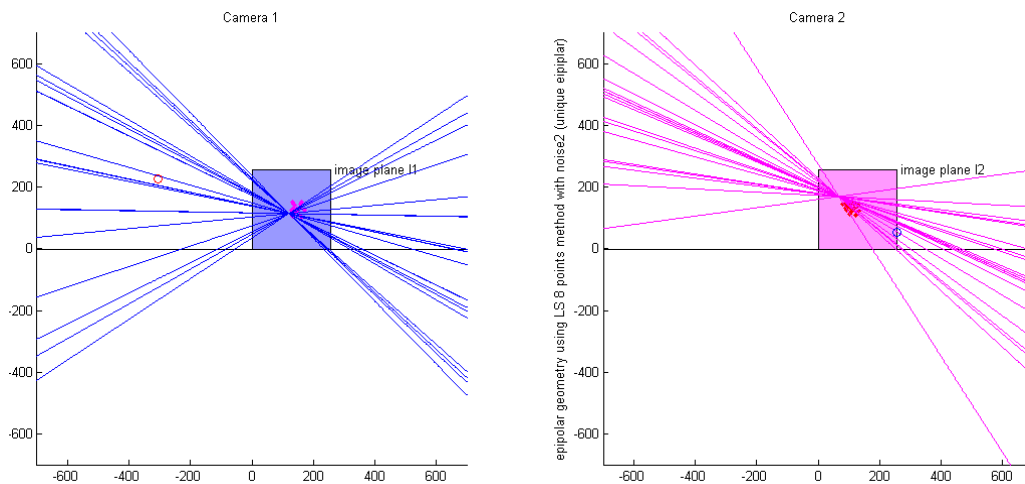


Fig. 8 Epipolar geometry using LS 8 points method with noise2 (unique epipole)

Part 2

Step 14

In the steps above, the fundamental matrices are obtained by using the least-square 8 points method. In this step we will use SVD 8 points to get the fundamental matrices.

The same as above:

$$U_n f = 0 \quad f = (F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33})^t$$

This equation can be solved by SVD so that f lies in the null space of $U_n = [UDV^T]$. Hence f corresponds to a multiple of the column of V that belongs to the unique singular value of D equal to 0.

MATLAB code is:

```
function F1_svd=svdmethod1(V_I1,V_I2)
U1_svd=[V_I1(1,:).*V_I2(1,:)
        V_I1(2,:).*V_I2(1,:)
        V_I2(1,:)
        V_I1(1,:).*V_I2(2,:)
        V_I1(2,:).*V_I2(2,:)
        V_I2(2,:)
        V_I1(1,:)
        V_I1(2,:)
        ones(1,20)]';
[Q3,S3,V3] = svd(U1_svd);
F1_svd=V3(:,9);
F1_svd=F1_svd/F1_svd(9);
F1_svd=[F1_svd(1) F1_svd(2) F1_svd(3)
        F1_svd(4) F1_svd(5) F1_svd(6)
        F1_svd(7) F1_svd(8) F1_svd(9)];
```

```
function F2_svd=svdmethod2(V_I1,V_I2)
U2_svd=[V_I2(1,:).*V_I1(1,:)
        V_I2(2,:).*V_I1(1,:)
        V_I1(1,:)
        V_I2(1,:).*V_I1(2,:)
        V_I2(2,:).*V_I1(2,:)
        V_I1(2,:)
        V_I2(1,:)
        V_I2(2,:)
        ones(1,20)]';
[Q4,S4,V4] = svd(U2_svd);
F2_svd=V4(:,9);
F2_svd=F2_svd/F2_svd(9);
F2_svd=[F2_svd(1) F2_svd(2) F2_svd(3)
        F2_svd(4) F2_svd(5) F2_svd(6)
        F2_svd(7) F2_svd(8) F2_svd(9)];
```

The fundamental matrices using SVD is as follows:

Fundamental matrices using SVD F =

0.000032234708307	0.000072675805446	-0.006617839384129
0.000024596847302	-0.000023922569132	0.012979428509026
-0.009580203230024	-0.017342632846678	1.000000000000000

Fundamental matrices using SVD F' =

0.000032234708307	0.000024596847302	-0.009580203230024
0.000072675805446	-0.000023922569132	-0.017342632846678
-0.006617839384129	0.012979428509026	1.000000000000000

The difference between the fundamental matrices using SVD and the real fundamental matrices is as follows:

```
different3=F1_svd-F1_normal;
different4=F2_svd-F2_normal;
```

Different3 = 1.0e-015 *

-0.000941900637347	-0.003577867169202	0.180411241501588
-0.000735224598217	0.003388131789017	-0.634908792207511
0.220309881449054	0.423272528138341	0

Different4 = 1.0e-015 *

-0.000420128341838	-0.000450621527939	0.206432093641240
-0.001843143693225	0.000210064170919	0.503069808033274
0.215973072759112	-0.603683769639929	0

From the order of magnitude of the difference, 10^{-15} , compare with the order of magnitude in the fundamental matrix, the different is so small that we can say virtually the those two matrices obtain by SVD are the “same” as the real fundamental matrices.

Step 15

In this step, we will repeat the step 10 to step 13 (with the fundamental matrix obtained by SVD of step 14 instead of step 8) for some Gaussian noise: first in rang of [-1, 1] and then in rang of [-2, 2] for the 95% of points.

The result shows in the following four figures, from figure 9 to figure 12.

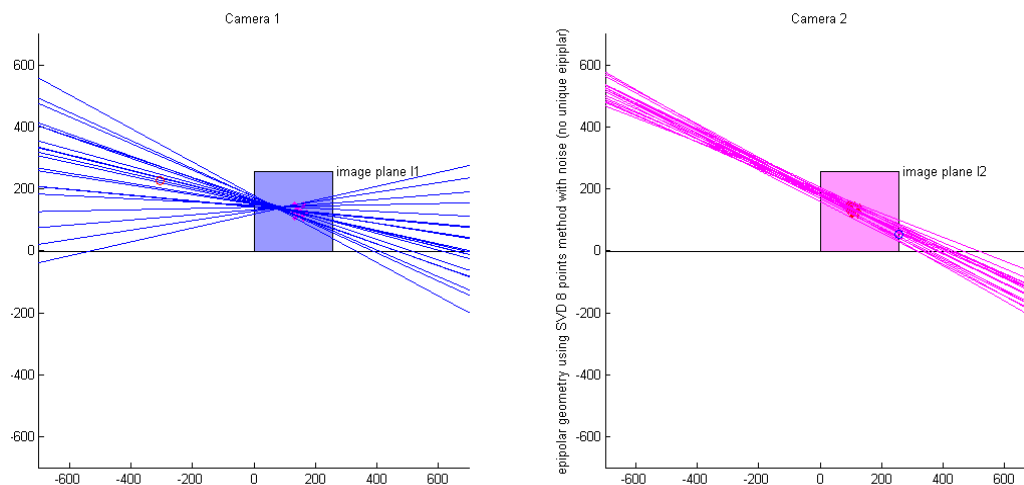


Fig. 9 Epipolar geometry using SVD 8 points method with noise [-1,1] (no unique epipole)

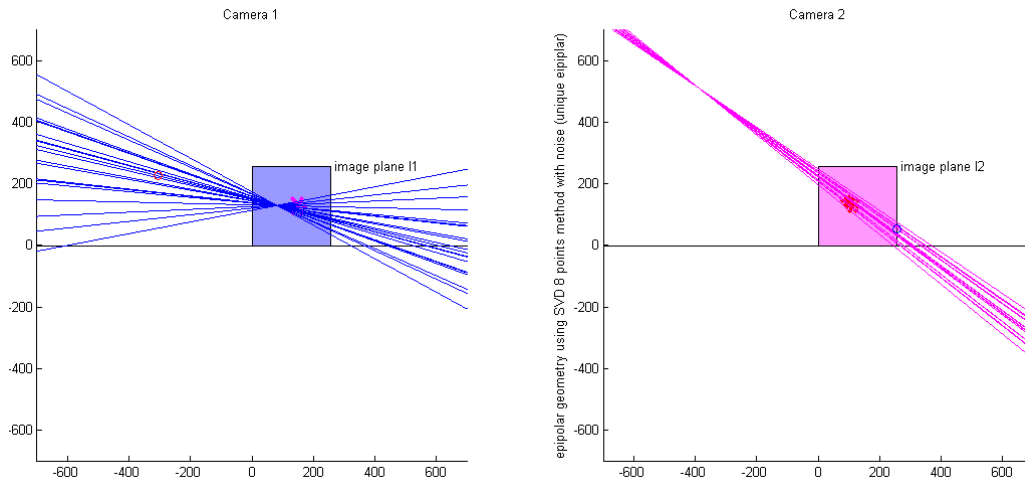


Fig. 10 Epipolar geometry using SVD 8 points method with noise [-1,1] (unique epipole)

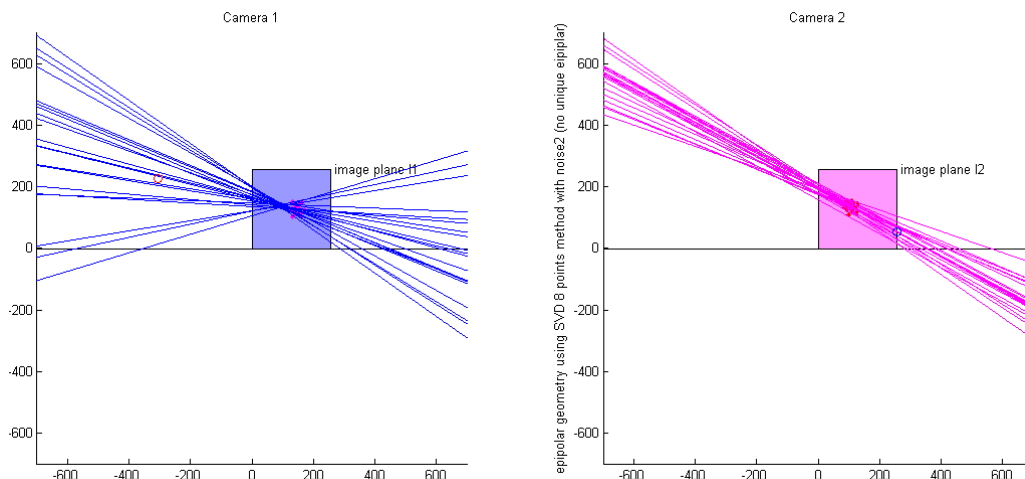


Fig. 11 Epipolar geometry using SVD 8 points method with noise [-2,2] (no unique epipole)

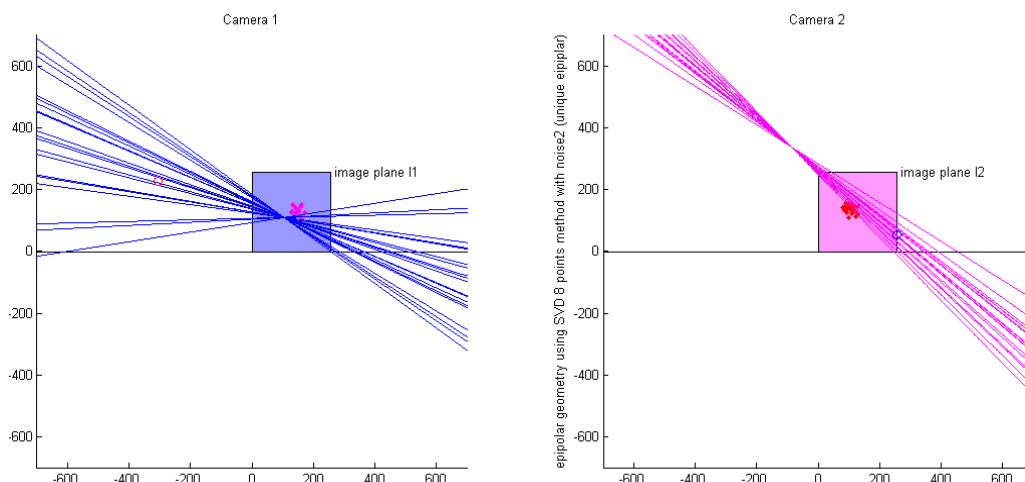


Fig. 12 Epipolar geometry using SVD 8 points method with noise [-2,2] (no unique epipole)

In this step, we will compare the method of least-squares and SVD.

The difference between least-squares and SVD is illustrated in figure 13:

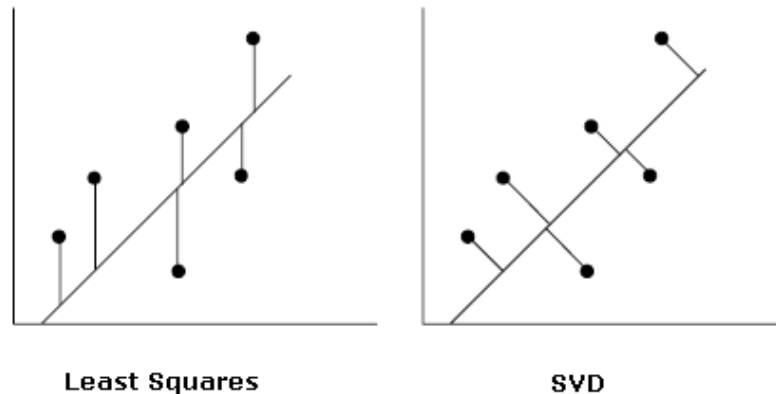


Fig. 13 comparison between least-squares and SVD

Both least-square and SVD are methods to solve the multiple linear regression problem. The difference is: least-square is finding the line that has the minimum value of the sum of y-axis direction distance (where the distance is vertical) of all the point to the line. The SVD is finding the line that has the minimum value of the sum of distance (perpendicularity with the line) from all the point to the line, as shown in figure 13. So the average distance between points to line finding by SVD is smaller than the average distance between points to line finding by least-square.

Then we can compare the value of fundamental matrices.

In the first case, there is Gaussian noise in rang of $[-1, 1]$ for 95% points:

$F_{SVD} - F_{normal} =$

-0.00008042608759	-0.000070637432529	0.004184087026946
0.000008984183626	0.000026258575143	-0.015718856363885
0.001895206925778	0.014544154936074	0

$F_{LS} - F_{normal} =$

-0.00008043165677	-0.000070637501791	0.004184065479026
0.000008983195737	0.000026259854988	-0.015719048698520
0.001895449631960	0.014544152231376	0

$F'_{SVD} - F'_{normal} =$

-0.00008042608759	0.000008984183626	0.001895206925777
-0.000070637432529	0.000026258575143	0.014544154936074
0.004184087026946	-0.015718856363885	0

$F'_{LS} - F'_{normal} =$

-0.00008043165677	0.000008983195737	0.001895449631945
-0.000070637501791	0.000026259854988	0.014544152231335
0.004184065479011	-0.015719048698532	0

Comparing difference between $F(F')$ using SVD and real $F(F')$, difference between $F(F')$ using least-square and $F(F')$, there is not very obviously different as in the theory that SVD should more accurate than least-square.

In the second case, there is Gaussian noise in rang of $[-2, 2]$ for 95% points:

$F_{SVD} - F_{normal} =$

-0.000015398159311	-0.000068898383424	0.004038547238031
-0.000000615413325	0.000039117035767	-0.017396552728360
0.004462894171758	0.013942166223257	0

$F_{LS} - F_{normal} =$

-0.000015398194524	-0.000068898065212	0.004038513032294
-0.000000615644740	0.000039117077945	-0.017396520377489
0.004462928596335	0.013942121419825	0

$F'_{SVD} - F'_{normal} =$

-0.000015398159311	-0.000000615413325	0.004462894171758
-0.000068898383424	0.000039117035767	0.013942166223257
0.004038547238031	-0.017396552728360	0

$F'_{LS} - F'_{normal} =$

-0.000015398194524	-0.000000615644740	0.004462928596315
-0.000068898065213	0.000039117077945	0.013942121419825
0.004038513032293	-0.017396520377542	0

In this case, the difference still very limited, as the previous one, so in this experiment we didn't find the obvious advantage of SVD.

3. conclusion

In this course work, we focus on the computation of the fundamental matrix from two simulated cameras and epipolar geometry. The time cost on this course work is much more than expected. It is difficult but interesting. It is good for understanding the whole system.