

```

function cvexercise2
clear
clf
close all

%% Part 1
%% Step 1 Define camera 1
au1 = 100; av1 = 120; uo1 = 128; vo1 = 128;

%% Step 2 Define camera 2
au2 = 90; av2 = 110; uo2 = 128; vo2 = 128;
ax = 0.1; by = pi/4; cz = 0.2;
tx = -1000; ty = 190; tz = 230;

%% Step 3 Get the intrinsic and transformation matrices of both cameras
A1=[au1 0 uo1
    0 av1 vo1
    0 0 1];
A2=[au2 0 uo2
    0 av2 vo2
    0 0 1];

rz=[cos(cz) -sin(cz) 0
    sin(cz) cos(cz) 0
    0 0 1];
ry=[cos(by) 0 sin(by)
    0 1 0
    -sin(by) 0 cos(by)];
rx=[1 0 0
    0 cos(ax) -sin(ax)
    0 sin(ax) cos(ax)];
R=rx*ry*rz;
t=[tx;ty;tz];

K_2respto1=[R t];
K_1respto2=[R' -R'*t];

%% Step 4 Get the Fundamental matrix analytically as the product of
matrices defined in step 3
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);

```

```

%% Step 5 Define the following set of object points with respect to the
world coordinate system (or camera 1 coordinate system)
V(:, 1) = [100;-400;2000;1];
V(:, 2) = [300;-400;3000;1];
V(:, 3) = [500;-400;4000;1];
V(:, 4) = [700;-400;2000;1];
V(:, 5) = [900;-400;3000;1];
V(:, 6) = [100;-50;4000;1];
V(:, 7) = [300;-50;2000;1];
V(:, 8) = [500;-50;3000;1];
V(:, 9) = [700;-50;4000;1];
V(:, 10) = [900;-50;2000;1];
V(:, 11) = [100;50;3000;1];
V(:, 12) = [300;50;4000;1];
V(:, 13) = [500;50;2000;1];
V(:, 14) = [700;50;3000;1];
V(:, 15) = [900;50;4000;1];
V(:, 16) = [100;400;2000;1];
V(:, 17) = [300;400;3000;1];
V(:, 18) = [500;400;4000;1];
V(:, 19) = [700;400;2000;1];
V(:, 20) = [900;400;3000;1];

%% Step 6 Compute the couples of image points in both image planes by using
the matrices of step3
intr1=[A1 zeros(3, 1)];
intr2=[A2 zeros(3, 1)];
K_wrespto1=[eye(3) zeros(3, 1)];
extr1=[K_wrespto1;zeros(1, 3) 1];
K_wrespto2=K_1respto2;
extr2=[K_wrespto2;zeros(1, 3) 1];

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

%% Step 7 Open two windows in matlab, which will be used as both image
%planes, and draw the 2D points obtained in step 6
figure(15)
subplot(1, 2, 1);hold on;
x=[0;256;256;0];y=[0;0;256;256];
fill(x, y, [0.6 0.6 1]);
text(256, 256, ' image plane I1');
stem(V_I1(1, :), V_I1(2, :), 'LineStyle', 'none', 'Marker', 'o', 'Color', 'm');

```

```

title(' Camera 1 ');
subplot(1,2,2);hold on;
fill(x,y,[1 0.6 1]);
text(256,256,' image plane I2');
stem(V_I2(1,:),V_I2(2:,:),'LineStyle','none','Marker','o','Color','r');
title(' Camera 2 ');

figure(1)
draw_object_points(V_I1,V_I2);

%% Step 8. Compute the fundamental matrix by using the 8-point method and
least-squares by means of the 2D points obtained in step 6

F1_8pm=eightpointsmethod1(V_I1,V_I2);
F1_normal=F1/F1(3,3);

F2_8pm=eightpointsmethod2(V_I1,V_I2);
F2_normal=F2/F2(3,3);

%% Step 9. Compare the step 8 matrix with the one obtained in step 4

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

%% Step 10. Draw in the windows of step 7 all the epipolar geometry
figure(1)
subplot(1,2,2);
ylabel(' epipolar geometry using LS 8 points method without noise ');
%boundary of x-axis
x1_max=600;
x1_min=-600;
x2_max=600;
x2_min=-600;

draw_epipolar_geometry(F1_normal,F2_normal,V_I1,V_I2,x1_min,x1_max,x2_min,x
2_max);
draw_focal_point(t,intr1,extr1,intr2,extr2);

%% Step 11 Add some Gaussian noise to the 2D points producing discrepancies
between the range [-1,+1] pixels for the 95% of points.
mu=0;
sigma=0.5102;

```

```

V_I1_noise=V_I1+[normrnd(mu, sigma, 2, 20);zeros(1,20)];% add Gaussian noise
to the point
V_I2_noise=V_I2+[normrnd(mu, sigma, 2, 20);zeros(1,20)];

%% Step 12. Again repeat step 8 up to 10 with the noisy 2D points.

%Compute the fundamental matrix by using the 8-point method and
%least-squares by means of the 2D points with noise obtained in step 11

F1_8pm_noise=eightpointsmethod1(V_I1_noise,V_I2_noise);
F2_8pm_noise=eightpointsmethod2(V_I1_noise,V_I2_noise);

%Draw all the epipolar geometry
figure(2);
subplot(1,2,2);
ylabel('epipolar geometry using LS 8 points method with noise (no unique
eipiplar)');

%boundary of x-axis
x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise,V_I2_noise);
draw_epipolar_geometry(F1_8pm_noise,F2_8pm_noise,V_I1_noise,V_I2_noise,x1_m
in,x1_max,x2_min,x2_max);
draw_focal_point(t,intr1,extr1,intr2,extr2);

%compute the closest rank-2 matrix
[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';

figure(3)
subplot(1,2,2);
ylabel('epipolar geometry using LS 8 points method with noise (unique
eipiplar)');
x1_max=700;
x1_min=-700;

```

```

x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise, V_I2_noise);
draw_epipolar_geometry(F1_8pm_noise_svd, F2_8pm_noise_svd, V_I1_noise, V_I2_noise,
x1_min, x1_max, x2_min, x2_max);
draw_focal_point(t, intr1, extr1, intr2, extr2);

%% Step 13. Increase the Gaussian noise of step 11 (now in the range [-2,+2]
for the 95% of points) and repeat step 8-12.
mu=0;
sigma=0.5102;
sigma2=2*sigma;

V_I1_noise2=V_I1+[normrnd(mu, sigma2, 2, 20);zeros(1,20)];% add Gaussian noise
to the point
V_I2_noise2=V_I2+[normrnd(mu, sigma2, 2, 20);zeros(1,20)];

%Compute the fundamental matrix by using the 8-point method and
%least-squares by means of the 2D points with noise obtained above

F1_8pm_noise2=eightpointsmethod1(V_I1_noise2, V_I2_noise2);
F2_8pm_noise2=eightpointsmethod2(V_I1_noise2, V_I2_noise2);

%Draw all the epipolar geometry
figure(4)
subplot(1,2,2);
ylabel('epipolar geometry using LS 8 points method with noise2 (no unique
epipolar)');

%boundary of x-axis
x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise2, V_I2_noise2);
draw_epipolar_geometry(F1_8pm_noise2, F2_8pm_noise2, V_I1_noise2, V_I2_noise2,
x1_min, x1_max, x2_min, x2_max);
draw_focal_point(t, intr1, extr1, intr2, extr2);

%compute the closest rank-2 matrix

```

```

[Q1, S1, V1] = svd(F1_8pm_noise2);
S1(3,3)=0;
F1_8pm_noise2_svd=Q1*S1*V1';
[Q2, S2, V2] = svd(F2_8pm_noise2);
S2(3,3)=0;
F2_8pm_noise2_svd=Q2*S2*V2';

figure(5)
subplot(1,2,2);
ylabel('epipolar geometry using LS 8 points method with noise2 (unique eipiplar)');

x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise2, V_I2_noise2);
draw_epipolar_geometry(F1_8pm_noise2_svd, F2_8pm_noise2_svd, V_I1_noise2, V_I2_noise2, x1_min, x1_max, x2_min, x2_max);
draw_focal_point(t, intr1, extr1, intr2, extr2);

%% Part 2.
%% Step 14. Compute the fundamental matrix by using the 8-point method and SVD from the points 2D obtained in step 6 and without noise.
F1_svd=svdmethod1(V_I1, V_I2);
F2_svd=svdmethod2(V_I1, V_I2);
different3=F1_svd-F1_normal;
different4=F2_svd-F2_normal;

%% Step 15. Repeat step 10 up to 13 (with the matrix of step 14 instead of step 8) for points with Gaussian noise

%first Gaussian noise in the rang [-1, 1] for the 95% of points
F1_svd_noise=svdmethod1(V_I1_noise, V_I2_noise);
F2_svd_noise=svdmethod2(V_I1_noise, V_I2_noise);
%Draw all the epipolar geometry
figure(6)
subplot(1,2,2);
ylabel('epipolar geometry using SVD 8 points method with noise (no unique eipiplar)');

```

```

%boundary of x-axis
x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise,V_I2_noise);
draw_epipolar_geometry(F1_svd_noise,F2_svd_noise,V_I1_noise,V_I2_noise,x1_m
in,x1_max,x2_min,x2_max);
draw_focal_point(t,intr1,extr1,intr2,extr2);

%compute the closest rank-2 matrix
[Q1,S1,V1] = svd(F1_svd_noise);
S1(3,3)=0;
F1_svd_noise_svd=Q1*S1*V1';
[Q2,S2,V2] = svd(F2_svd_noise);
S2(3,3)=0;
F2_svd_noise_svd=Q2*S2*V2';

figure(7)
subplot(1,2,2);
ylabel('epipolar geometry using SVD 8 points method with noise (unique
eipiplar)');

x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise,V_I2_noise);
draw_epipolar_geometry(F1_svd_noise_svd,F2_svd_noise_svd,V_I1_noise,V_I2_no
ise,x1_min,x1_max,x2_min,x2_max);
draw_focal_point(t,intr1,extr1,intr2,extr2);

%second Gaussian noise in the rang [-2, 2] for the 95% of points
F1_svd_noise2=svdmethod1(V_I1_noise2,V_I2_noise2);
F2_svd_noise2=svdmethod2(V_I1_noise2,V_I2_noise2);

%Draw all the epipolar geometry
figure(8)
subplot(1,2,2);
ylabel('epipolar geometry using SVD 8 points method with noise2 (no unique

```

```

eipiplar)');

%boundary of x-axis
x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise2,V_I2_noise2);
draw_epipolar_geometry(F1_svd_noise2,F2_svd_noise2,V_I1_noise2,V_I2_noise2,
x1_min,x1_max,x2_min,x2_max);
draw_focal_point(t,intr1,extr1,intr2,extr2);

%compute the closest rank-2 matrix
[Q1,S1,V1] = svd(F1_svd_noise2);
S1(3,3)=0;
F1_svd_noise2_svd=Q1*S1*V1';
[Q2,S2,V2] = svd(F2_svd_noise2);
S2(3,3)=0;
F2_svd_noise2_svd=Q2*S2*V2';

figure(9)
subplot(1,2,2);
ylabel('epipolar geometry using SVD 8 points method with noise2 (unique
eipiplar)');

x1_max=700;
x1_min=-700;
x2_max=700;
x2_min=-700;

draw_object_points(V_I1_noise2,V_I2_noise2);
draw_epipolar_geometry(F1_svd_noise2_svd,F2_svd_noise2_svd,V_I1_noise2,V_I2
_noise2,x1_min,x1_max,x2_min,x2_max);
draw_focal_point(t,intr1,extr1,intr2,extr2);

%% function
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
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;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function draw_object_points(V_I1,V_I2)
```

```
subplot(1,2,1);hold on;
```

```
x=[0;256;256;0];y=[0;0;256;256];
```

```
fill(x,y,[0.6 0.6 1]);
```

```
text(256,256,' image plane I1');
```

```
stem(V_I1(1,:),V_I1(2:3,:),'LineStyle','none','Marker','.', 'Color','m');
```

```
title(' Camera 1');
```

```
subplot(1,2,2);hold on;
```

```
fill(x,y,[1 0.6 1]);
```

```
text(256,256,' image plane I2');
```

```
stem(V_I2(1,:),V_I2(2:3,:),'LineStyle','none','Marker','.', 'Color','r');
```

```
title(' Camera 2');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
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];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
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
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
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;

```

