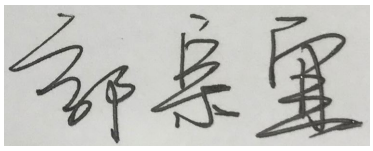
EE4212 Computer Vision CA1

A0206639Y GUO ZONGYI

February 29, 2020

I have not given or received aid from anyone else in solving the problems.



Q1 Projection Model; Homogeneous coordinates

(a)* Suppose you have two parallel lines in 3-space, one passing through the point (100,100,1000), the other through (200,200,1100). The lines are parallel to the vector (1,2,1). The lines are observed by a unit focal length camera at the origin (i.e. the camera reference frame and the world reference frame are identical). All coordinates are in camera coordinates. What is their point of intersection in the image?

Solution:
Due to the camera coordinate and the world coordinate are identical in this case, theres no need to use the equation: $P_c = R(P_w - T)$ to transform these two coordinates.

Because of parallelism of the 2 lines, therefore, $\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$ and plug in the two known points. The equation can be obtained:

$$l_1 = \begin{bmatrix} 100 \\ 100 \\ 1000 \end{bmatrix} + \lambda_1 \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$
$$l_2 = \begin{bmatrix} 200 \\ 200 \\ 1100 \end{bmatrix} + \lambda_2 \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

Now, we need to find the corresponding projection line in image coordinates. Thus, firstly, we randomly pick two points from l_1, l_2.

Set $\lambda_1 = 100 and 200 : P_{a_1} = \begin{bmatrix} 200 \\ 300 \\ 1000 \end{bmatrix}, P_{a_2} = \begin{bmatrix} 300 \\ 500 \\ 1200 \end{bmatrix}$, Then we transform to camera coordinate, using linear mapping:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix}_{im} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Because of unit focal length, f = 1 here.

Thus, $p_{a_1} = \begin{bmatrix} 200 \\ 300 \\ 1100 \end{bmatrix}$, $p_{a_2} = \begin{bmatrix} 300 \\ 500 \\ 1200 \end{bmatrix}$; the formula of this line passing through these two points is:

$l_{im_1} = \begin{bmatrix} 200 \\ 300 \\ 1100 \end{bmatrix}$ x $\begin{bmatrix} 300 \\ 500 \\ 1200 \end{bmatrix} = \begin{bmatrix} -19 \\ 9 \\ 1 \end{bmatrix}$

Similarly, we can also randomly choose 2 points from the second line. And the corresponding projection line $l_{im_2} = \begin{bmatrix} -20 \\ 9 \\ 2 \end{bmatrix}$.

Then its easy for us to find the intercept point between these two lines.

$$p_{intercept} = \begin{bmatrix} -19 \\ 9 \\ 1 \end{bmatrix} \text{x} \begin{bmatrix} -20 \\ 9 \\ 2 \end{bmatrix} = \begin{bmatrix} 9 \\ 18 \\ 9 \end{bmatrix}$$

Then convert to P_2 space, $p = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ is the intercept point in the image.

(b) Given the 3D coordinates of several corresponding points Pi and Pi in two views, you are required to find the 3D rotation R and translation T that relate the two views (Pi = RPi + T). Formulate a linear least squares algorithm (of the form Ax=b) that ignores the orthogonality constraint associated with R (that is, it is ok if the solution for R returned by your formulation is not orthogonal). State the entries of the matrix A, and the vectors x and b.

Solution:

From the question, we have $P_i^{'} = RP_i + T$, write it in matrix form:

$$\begin{bmatrix} x_i^{'} \\ y_i^{'} \\ z_i^{'} \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \\ 0 & 0 & 0 & t_1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

Then, flatten the equation into vector form:

$$\begin{aligned} x_i^{'} &= R_{11}x_i + R_{12}y_i + R_{13}z_i + t_1 + ... \\ y_i^{'} &= R_{21}x_i + R_{22}y_i + R_{33}z_i + t_2 + ... \\ z_i^{'} &= R_{31}x_i + R_{32}y_i + R_{33}z_i + t_3 + ... \end{aligned}$$

(where denotes padding zeros to the same length)

Hence, the unknown variables can be abstracted from the formulas and form a vector with length of 12.

The unknown $x = \begin{bmatrix} R_{11} \\ R_{12} \\ R_{13} \\ R_{21} \\ R_{22} \\ R_{23} \\ R_{31} \\ R_{32} \\ R_{33} \\ t_1 \\ t_2 \\ t_3 \end{bmatrix}$

Thus, rewrite it as the form of $Ax = b$, where

$$A = \begin{bmatrix} x_i & y_i & z_i & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & x_i & y_i & z_i & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_i & y_i & z_i & 0 & 0 & 1 \end{bmatrix}$$

$$b = \begin{bmatrix} x_i^{'} \\ y_i^{'} \\ z_i^{'} \end{bmatrix}$$

Note that this is only for one observe, if we consider n observes, the dimention of A and b should be $A_{3n,12}$ and $b_{3n,1}$.

(c)You are now given two sets of corresponding point clouds P_i and $P_i^{'}$, in the files pts.txt and pts_prime.txt respectively. Each row is a 3D coordinate, possibly contaminated with some noise. Write a Matlab routine to estimate the optimal values of R and T in the least squares sense via SVD, using the formulation in (b). Compute the determinant of R using the Matlab function det, and comment on the resulting value.

```
userpath('C:\Users\hp\Desktop\Seme 2\4212 CV')
P = importdata('pts.txt');
P_Prime = importdata('pts_prime.txt');
padding = zeros([200, 3]);
a = [P; padding; padding]; % Create the first 3 cols of A
b = [padding; P; padding]; % Create 4th to 6th cols of A
c = [padding; padding; P]; % Create 7th to 9th cols of A
d = [ones([200,1]) zeros([200,1]) zeros([200,1]);
    zeros([200,1]) ones([200,1]) zeros([200,1]);
    zeros([200,1]) zeros([200,1]) ones([200,1])];
A = [a b c d];
b = [P_Prime(:,1); P_Prime(:,2); P_Prime(:,3)];
[U,S,V]=svd(A);
D = [inv(S(1:12,:)) zeros([12,588])];
x = V * D * U' * b;
disp(x)
R = reshape(x(1:9,:),[3,3])';
disp(R)
T = x(10:12,:);
disp(T)
```

Display the results, R and t can be obtained:

$$R = \begin{bmatrix} 0.9782 & -0.0084 & 0.0094 \\ 0.0202 & -0.0046 & 0.9979 \\ -0.0016 & -0.9940 & -0.0007 \end{bmatrix}$$

$$t = \begin{bmatrix} -0.0365 \\ -4.0009 \\ 4.0021 \end{bmatrix}$$

The determinant of R is 0.9701, which is not equal to 1, but it's acceptable and reasonal, due to some environment noise.

(d)If the world homogeneous coordinates are (Xw, Yw, Zw, Ww), the image plane homogeneous coordinates are (u, v, w), and they are related by: $\begin{bmatrix} u \\ v \\ w \end{bmatrix} \sim M \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ W_w \end{bmatrix}$

i.find the 3x4 matrix M as a function of α , h, according to the diagram below (in the left diagram, the axis X and Xw are pointing directly out of the paper). Assume that the only intrinsic camera parameter is the focal length f (given in pixel unit). Use $P_c = R_w + t$ to relate the camera coordinates P_c and the world coordinates P_w. R is the orientation of the world with respect to the camera; t is the world origin expressed in the camera frame.

ii.* Find the 3x3 matrix for the linear transformation that maps points on the world plane $Z_w = d$ to the image plane (u, v, w).

Solution:

i.Find two points in world coordinate, which are $\begin{bmatrix} Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, which can map into the camera coordinate $\begin{bmatrix} y_c \\ z_c \end{bmatrix} = R \begin{bmatrix} Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} -\cos\alpha \\ -\sin\alpha \end{bmatrix} and \begin{bmatrix} -\sin\alpha \\ \cos\alpha \end{bmatrix}$.

Next, the translation vector is defined by the world origin expressed in the camera frame:

$$t = \begin{bmatrix} 0 \\ 0 \\ \frac{h}{\sin\alpha} \end{bmatrix}$$

Thus, the transformation matrix can be obtained:

$$M = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & -\cos\alpha & -\sin\alpha & \frac{h}{\sin\alpha} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & -f\sin\alpha & f\cos\alpha & 0 \\ 0 & -\cos\alpha & -\sin\alpha & \frac{h}{\sin\alpha} \end{bmatrix}$$

Q2 Using Singular Value Decomposition for Image Compression and PCA

1. Many thousands of outdoor cameras are currently connected to the Internet. They can be used to measure plant growth (e.g. in response to recent warming trends), survey animal populations (e.g. in national parks), monitor surf conditions, and security, etc.In this question, you are provided with a 150-frame time-lapse video of a city scene taken between 6.30-7.30pm. Each frame is of the dimension 161x261 pixels. Watch the video, and you can see that clouds are moving, and planes take off and land from a nearby airport.

(a) Using Matlab, load the first image in the video sequence provided with this question and convert it to an appropriate data type. Just type the following commands in the command window: I = im2double(imread('image001.png'));

(b) Do a singular value decomposition using the command svd: [U S V]=svd(I); This will give you the singular values and the singular vectors. The singular values in S have been sorted in descending order. Plot the singular value spectrum using the command: plot(diag(S),'b.'). Submit this plot. What do you notice in this plot?

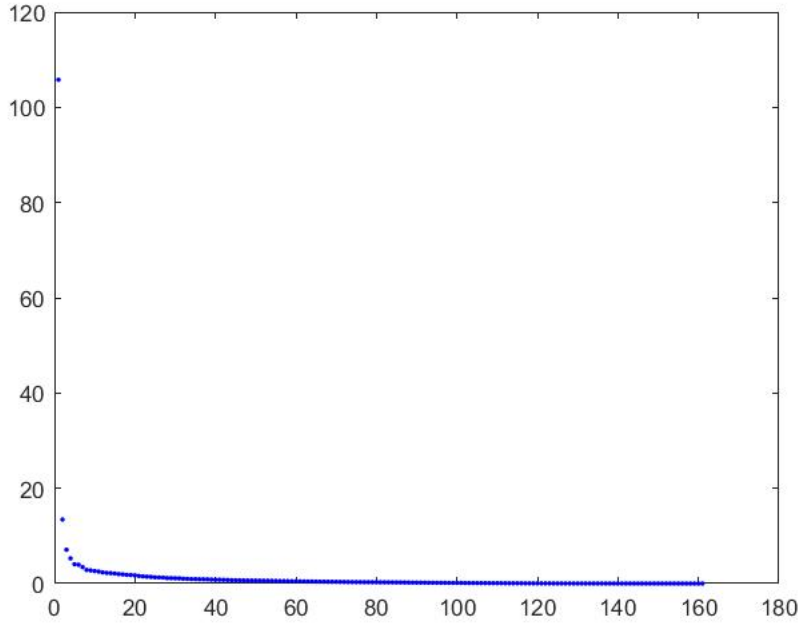


Figure 1: Singular Values

The first singular value is far more bigger than the others, which is up to 105. Although there are totally 161 singular values, however, only the first 32 values are greater than 1. The condition number, which is $C = \frac{\sigma_1}{\sigma_n}$, is extremely large, which suggests the matrix is ill-conditioned and sensitive to noise.

(c) Let K=20, Extract the first K singular values and their corresponding vectors in U and V:Uk, Vk, Sk contain the compressed version of the image.Print out a copy of this compressed image and submit it.

The main matlab code is shown below:

```
[U, S, V] = svd(I);
k = 20;
uk = U(:,1:k);
sk = S(1:k,1:k);
vk = V(:,1:k);
ik = uk * sk * vk';
imshow(ik)
```

Figure 2: Image of $K = 20$

(d)Repeat question c for K=40,60,80. Submit the compressed images for the different values of K. Compare the 4 compressed images. Briefly describe what you notice.

Change K to 40, 60, 80 relatively and show the images below.

Figure 3: Image of $K = 40$

Figure 4: Image of $K = 60$

Figure 5: Image of $K = 80$

Findings:
With increasing the number of singular values, the images become more clear and have higher resolutions. The combination of U_k, D_k, V_k can be shown in equation as follows:

$$I_k = U_1 D_1 V_1^T + U_2 D_2 V_2^T + U_3 D_3 V_3^T + ... + U_k D_k V_k^T$$

With adding more and more infomation, the image can fit the raw image to a considerable degree.

(e)Thus, in image transmission, instead of transmitting the original image you can transmit Uk, Vk, Sk, which should be much less data than the original. Is it worth transmitting when K=100 (i.e.do you save any bits in transmission when K=100)? Explain your answer.

Solution:
The usage of RAM for saving the raw image is $161 \times 261 = 42021$. And the usage for saving compressed images should be calculated as:

$$Cost = K(m + n + 1)$$

Thus, when K = 100, the cost should be $100 \times (161 + 261 + 1) = 42300$. It's even greater than saving the raw image. Therefore, it is meaningless to transmitting when K = 100.

(f)** Find an expression that bounds the per-pixel error (the difference between the original image and the compressed image for a particular pixel (i, j)) in terms of K, the singular values and the elements in U and V with the largest absolute magnitude (umax, vmax respectively).

Solution:
The raw image can be transmitted by the combination of all singular values, U and V. The compressed version of the raw image only uses the first K singular values, U_k and V_k. The expressions are shown below:

$$I_N = U_1 D_1 V_1^T + U_2 D_2 V_2^T + U_3 D_3 V_3^T + ... + U_n D_n V_n^T$$
$$I_k = U_1 D_1 V_1^T + U_2 D_2 V_2^T + U_3 D_3 V_3^T + ... + U_k D_k V_k^T$$

Make subtraction of the two equations,

$$Diff = \left|\sum_{i=k+1}^{n} U_i D_i V_i^T\right|$$

When consider a particular pixel (i, j), the above equation can be rewriten as:

$$Diff \leq \sum_{i=k+1}^{n} \left|U_i D_i V_i^T\right| = \sum_{i=k+1}^{n} D_i \left|U_i\right| \left|V_i^T\right|$$

$$Diff \leq \sum_{i=k+1}^{n} D_i U_{max} V_{max}$$

(g)** SVD is intimately related to PCA (Principal components analysis). Some of you might have learned about PCA in the EE3731C course, but it is not a pre-requisite for this question. Basically,finding the principal components of a matrix X amounts to finding an orthonormal basis that spansthe column space of X (these are the column vectors uiin the matrix U). Here we create the datamatrix X by first vectorizing each of the 150 images into a column vector (by scanning in either row-major order or column-major order), and then stacking these vectors together into a matrix of size 42021 150. In effect, we have captured the entire video sequence into the 2D matrix X.

Before we proceed further, mean subtraction (a.k.a. mean centering) to center the data at the origin is necessary for performing PCA. That is, the images are mean centered by subtracting the mean image vector from each image vector. This is to ensure that the first principal component u1 really describes the direction of maximum variance. Again, if you do not have background in PCA, it is okay; just take it as a preprocessing step (or you can do some independent learning).

Apply SVD to the resultant mean-centered matrix. Take only the first 10 principal components and reconstruct the image sequence. [NB: you can use the Matlab reshape command to convert a matrix to a vector and vice versa]. Observe the dynamics in the reconstructed video, comment on what you find, and explain why. Remember to add back this mean image vector when you are displaying your reconstructed results.

First, prove the PCA is intimatedly related to SVD. Actually, in Sci-kit learn python library, implementing PCA is just using SVD.
Eigendecomposition can be only applied to squar matrix, where SVD can be applied to all kinds of matrix.
In PCA algorithm, we need to calculate the covariance:

$$C = \frac{1}{m} X X^T = Q \Lambda Q^{-1}$$

Where Q is the eigenvectors and Λ is the eigenvalues.
Suppose A is a matrix with dimention m by n. We have:

$$A^T A = \left(U D V^T\right)^T (U D V) = V D^2 V^T$$

Due to orthogonal matrix transpose equals inverse, so assign

$$A = \frac{1}{\sqrt{m}} X^T$$

Then we just need to calculate the SVD of A. And extract the fisrt 10 eigenvectors and transform the raw dataset. The matlab code is shown below.

```
X = [];
image_path = 'time-lapse video/';
Files=dir([image_path '*.png']);
for k=1:length(Files)  % read all figures
    FileNames=Files(k).name;
    I = im2double(imread([image_path Files(k).name]));
    I = I(:);
    X = [X I];
end
mean_x = mean(X, 2);
de_mean = X - mean_x;
A = 1/sqrt(150)*de_mean';
[U,S,V] = svd(A,0);
trans_data = V(1:10,:)* X + mean_x;  % Transform step
for i=1:150     % Save transformed pictures
    Mat = trans_data(:,i);
    Name = strcat('img ', int2str(i), '.png');
    imwrite(reshape(Mat, [161, 261]) ,Name,'png');
end
```

Findings:
The reconstructed pictures expectedly become much fuzzier and unclear. Some originally dim light becomes totally unseen and other lights also become shallower. As well as lights, other details get also removed after PCA transformation. However, the whole concept of the pictures is reserved. Thus, the PCA transformation works effectively for preserving important information.

Q3: Projection Model; Homogeneous coordinates

(a)An affine camera is a simplification of the full perspective camera but is more complicated than the scaled orthographic model. It has a projection relationship given by the following equations: [x,y]T = A[X,Y,Z]T + b where A is a 23 matrix, and b is a 21 vector. If the world point (X,Y,Z) and image point (x,y) are represented by homogeneous vectors, write down the matrix representing the linear mapping between their homogeneous coordinates.
* Show that the point at infinity in space (X,Y,Z,0)T is mapped to point of infinity in the image plane. What does this result imply about the projection of parallel lines in space onto the image plane?

In homogeneous coordinate, the affine transformation can be written as:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} A & b \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Where A is a 2 by 3 matrix and b is a 2 by 1 matrix.

To find the infinit point in image plane, simply set the point in world coordinate $\begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix}$ and do the mapping. The point $\begin{bmatrix} x' \\ y' \\ 0 \end{bmatrix}$ is obtained. It denotes the infinite point in image plane. The result implies a specific point in P_2 space can denote the infinite position in real world 3D postion. Therefore, affine transformation is parallelity-invariant.

b.i.** Given an image of a scene containing a cube as shown in the following.Each vanishing point is the image of a point at infinity of the form (d, 0), where d is a Euclidean vector with 3 coordinates expressing the direction of a cube edge. Show that the coordinates of a vanishing point v can be expressed as v = K R d, where K is the intrinsic calibration matrix and R is the rotation matrix between the camera and world coordinate system. Hence express an edge direction d as a function of K, R and v. (Hint: Start from the 3x4 projection matrix equation in Ch 1)

From the world coordinate to image coordinate, we have the equation:

$$\begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = M_{int} M_{ext} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} = \begin{bmatrix} R & -RT \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} \frac{-f}{s_x} & 0 & o_x \\ 0 & \frac{-f}{s_y} & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix}$$

d denotes the infinite point in 3D space and the corresponding homogeneous expression is $\begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$. Plug into the precvious expression, the finale expression can be obtained:

$$v = M_{int} M_{ext} \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$$

$$d = (KR)^{-1} v$$

where K is M_{ext} and R is from M_{int}.

ii. ** The 3 directions of the cube edges are mutually perpendicular; therefore the dot product between any two directions is zero. Show that this condition leads to an equation in terms of the vanishing points and the unknown calibration matrix K. Note that such an equation can be written for each pair of the 3 vanishing points.

Due to perpendicular of three directions, suppose the angle between the relative 2 lines is θ, it's easy to obtain $\cos\theta = 0$. Thus, we have

$$\cos\theta = \frac{\overrightarrow{d_i}\overrightarrow{d_j}}{||d_i d_j||} = \frac{v_i^T w v_j}{||d_i d_j||}$$

where $w = ((KR)(KR)^T)^{-1} = (KK^T)^{-1}$.
The relationship should be $v_i^T w v_j = 0$, where $i \neq j$.

iii.*** Note that the derived equation above can be used to solve for K, but you are not requiredto explicitly propose a scheme for solving this equation. However, doing so will earn you bonus point.

We only have 3 valid data (vanishing point from image coordinate), but K has five degrees of freedom. Hence, we need more assumptions to solve K. From above mentioned, we know that K is just the intrinsic matrix. Originally,

$$K = \begin{bmatrix} \frac{-f}{s_x} & k & o_x \\ 0 & \frac{-f}{s_y} & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

Now suppose there's no skew(no k), and they are square pixels. Then we can finalize K:

$$K = \begin{bmatrix} \alpha & 0 & o_x \\ 0 & \alpha & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

(c) The projection of a scene point in world coordinates to pixel coordinates in an image can be represented using a camera projection matrix P as follows:

$$\begin{bmatrix} s_u \\ s_v \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

i. * Given a set of lines in a scene that are all parallel to the world X-axis, what is the vanishing point, (u, v), of these lines in the image? Can you conclude whether an infinite line in 3D space always yield an infinite line in the 2D image plane?

Suppose a line, parallel to x axis in world coordinate, is $l = (k, 0, 0, 0)^T$. Apply the projection matrix P,

$$\begin{bmatrix} s_u \\ s_v \\ s \end{bmatrix} = \begin{bmatrix} kp_{11} \\ kp_{21} \\ kp_{31} \end{bmatrix}$$

It's the expression of homogeneous coordinate and it depends on s. When s equals to 0, it denotes a infinite point in P_2 space and we can not visualize it in image plane. Otherwise, a single point $(\frac{s_u}{s}, \frac{s_v}{s})$ can denote an infinite line, parallel to x axis in world coordinate.

ii. What is the significance of the image point (represented as a homogeneous 3-vector) given bythe last column (p14, p24, p34) of P? That is,which world point gives rise to (p14, p24, p34) ?

Plug in homogeneous vector and solve the equation:

$$\begin{bmatrix} p_{14} \\ p_{24} \\ p_{34} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Hence, the corresponding world point should be $(0, 0, 0, 1)^T$.

iii. ** Consider the 1-dimensional right null-space of P, i.e., the 4-vector C such that P C = 0. In this case, the image point of C is (0, 0, 0)T which is not defined. What is the point C which possesses this property? Explain.

Plug in the condition, we have the equation:

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = 0$$

The point of C corresponding to the center of the image.

(d) The equation of a conics in inhomogeneous coordinates is given by ax2 +bxy +cy2 +dx+ey +f = 0. i. Homogenize this equation by the replacement x x1 x3 , y x2 x3 , and write down the homogeneous form. Finally, express this homogeneous form in the matrix form xT Cx = 0, where C is symmetric. Write down the elements of the symmetric matrix C.

In homogeneous coordinate, multiply x_3^2 in both sides, the conics equation becomes:

$$ax_1^2 + bx_1x_2 + cx_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0$$

To convert to the form $x^TCx = 0$, where C is the symmetric matrix,

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} a & \frac{b}{2} & \frac{d}{2} \\ \frac{b}{2} & c & \frac{e}{2} \\ \frac{d}{2} & \frac{e}{2} & f \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 0$$

As a result, the symmetric conics matrix $C = \begin{bmatrix} a & \frac{b}{2} & \frac{d}{2} \\ \frac{b}{2} & c & \frac{e}{2} \\ \frac{d}{2} & \frac{e}{2} & f \end{bmatrix}$

ii. ** If C has the special form C = lmT + mlT , clearly C is symmetric and therefore represents a conic. Show that the vector x = l × m is the null vector of C. Since a null vector exists, C is a degenerate conic. Show in this degenerate case, C contains two lines l and m, that is, show that the points on the lines l and m satisfy $x^TCx = 0$.

Solution:
In homogeneous coordinate, $x = l \times m$ means the intersection point of line l and m. It means x is a point both belong to l and m. Hence, we definitly have the equation:

$$m^T x = 0, l^T x = 0$$

In other words, x is the null vector of C.
Then, prove line l and m are both from C:
Suppose a random point from line l, which then has $l^T p = 0, p^T l = 0$.Then we just need to analyze whether p is from conic C:

$$p^T C p = (p^T l)(m^T p) + (p^T m)(l^T p) = 0 + 0 = 0$$

This shows any random point from line l is from conic C. It's the same for line m. Thus, we prove that l and m are both from C.
To conclude, the degenerate conic C has two lines, l and m, and its intersection point is the null vector of C.

(e)Given a set of two-dimensional points (x, y) as follows (not listed here, can be found in conics.txt), write a Matlab routine to find the conics best (in the least squares sense) represented by these points, in terms of the parameters a, b, c, d, e, and f, as formulated in Question 3(d).

Solution:
Any point should satisfy the equation:

$$ax_1^2 + bx_1x_2 + cx_2^2 + dx_1 + ex_2 + f = 0$$

$$(x_1^2, x_1x_2, x_2^2, x_1, x_2, 1) \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = 0$$

This becomes a least square problem to solve the linear equation $Ax = 0$. The matlab code and the plotted figure is shown below.

```
data_path = 'C:\Users\hp\Desktop\conics.txt';
[x,y] = textread(data_path,'%n%n');
xlabel('x')
ylabel('y')
scatter(x,y)
hold on
A = [x.^2, x.*y, y.^2, x, y, ones(size(x))];
    % Create the coefficient of linear function
[U, S, V] = svd(A);
c = V(:,length(V));
    % the last column is the solution
fimplicit(@(x,y) c(1)*x^2 + c(2)*x*y + c(3)*y^2 + c(4)*x + c(5)*y + c(6))
    % Create the polynomial equation
```

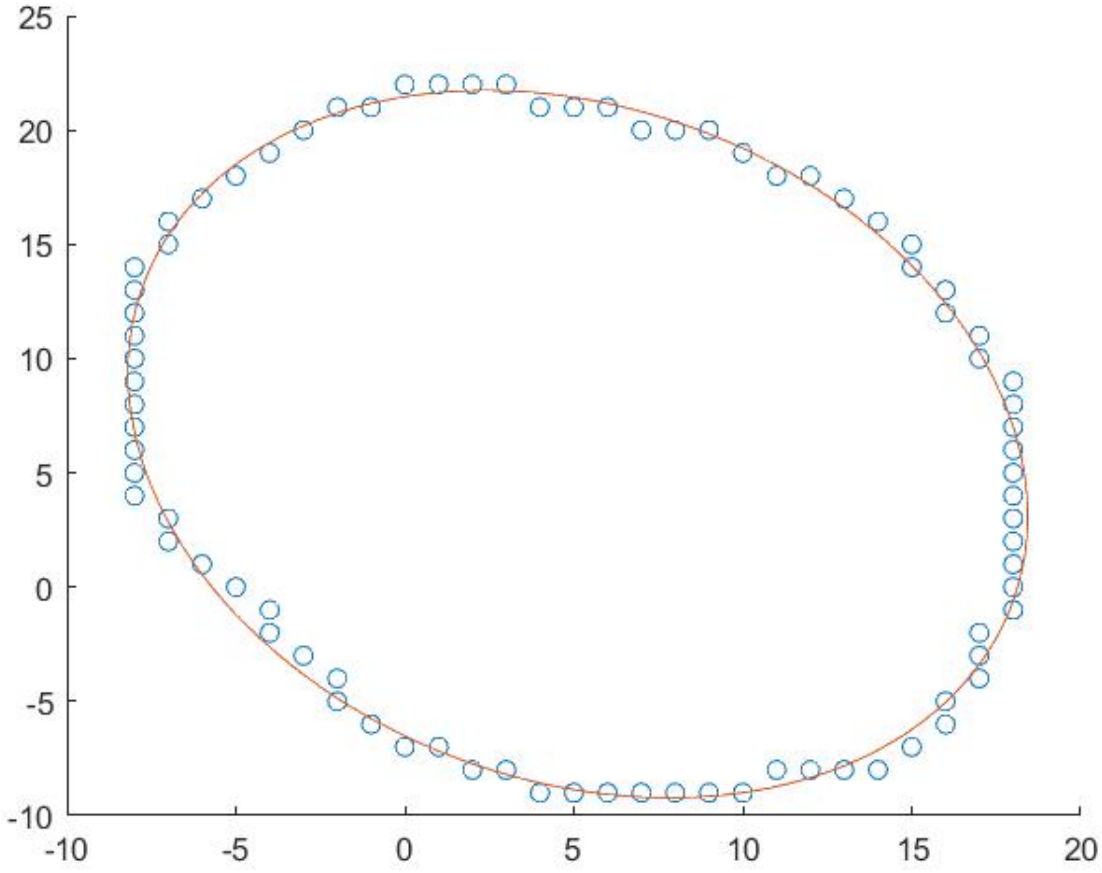


Figure 6: Best fit conics line

The best fit coefficient is:

$$\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} 0.0095 \\ 0.0033 \\ 0.0071 \\ -0.1185 \\ -0.1054 \\ -0.9872 \end{bmatrix}$$

Q4 Estimate Fundamental Matrix F using 8-Point Algorithm.

4.2.1.Establish Point Correspondences:
In this step, you need to establish point correspondences between the two images. Using Matlab, manually or automatically mark and record at least 8 pairs of corresponding points (preferably much more than 8 pairs for robustness!).

Instead of manually mark different points pairs, I used feature matching function provided in Matlab Computer Vision System toolbox based on Harris Features to automatically extract feature-matched points. The algorithm is shown below.Note that there are totally 32 point pairs found and I randomly extract 16 out of the whole.

```
image1 = imread('C:\Users\hp\Desktop\Seme 2\4212 CV\CA1\Q4_files\frc1.tif');
image2 = imread('C:\Users\hp\Desktop\Seme 2\4212 CV\CA1\Q4_files\frc2.tif');

p1 = detectHarrisFeatures(image1, 'MinQuality', 0.001);
p2 = detectHarrisFeatures(image2, 'MinQuality', 0.001);
[features1,valid_points1] = extractFeatures(I1,p1);
[features2,valid_points2] = extractFeatures(I2,p2);

indexPairs = matchFeatures(features1,features2);
matchedPoints1 = valid_points1(indexPairs(:,1),:);
matchedPoints2 = valid_points2(indexPairs(:,2),:);

rand = randsample(32, 16);
gA = matchedPoints1(rand).Location;
gB = matchedPoints2(rand).Location;

figure(100);
f1 = subplot(1,2,1);
set(f1,'position',[0.1,0.1,0.4,0.4])
imshow(image1)
a = double(gA(:,1));
b = double(gA(:,2));
text(a,b,'+','Color','yellow');

f2 = subplot(1,2,2);
imshow(image2)
set(f2,'position',[0.5,0.1,0.4,0.4])
a = double(gB(:,1));
b = double(gB(:,2));
text(a,b,'+','Color','yellow');
```

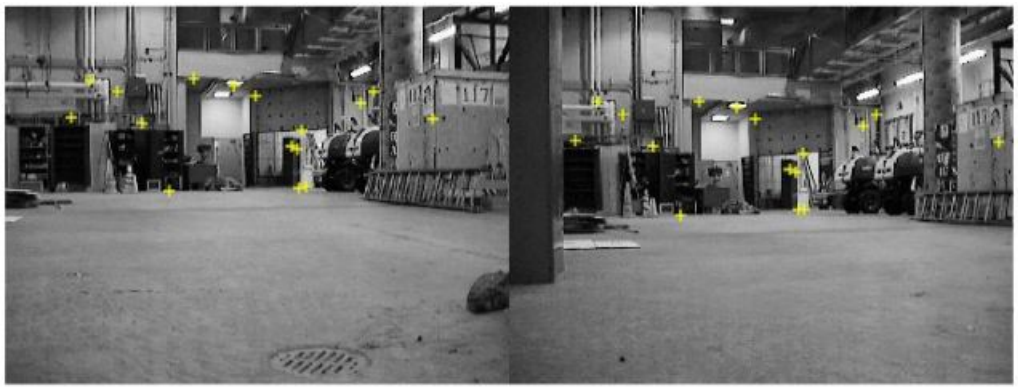


Figure 7: 16 Corresponding feature pairs

4.2.2. [Normalization of the data]
In 8-point algorithm, the stability of the results can be greatly improved by a simple normalization (translation and scaling) of the coordinates of the correspondences (originally presented in [Hartley- 1997] and also in [Hartley and Zisserman-2000]). For this normalization, you should find a similarity transformation T , consisting of a translation and scaling, that takes points xi to a new set of points xi (xi = Txi)such that the centroid of the points $\hat{x_i}$ is the coordinates origin, and their average distance (Root Mean Squares) from the origin is 2 pixels . Note: Homogeneous coordinates are used in this step.

For the image, the centroid of the image is defined by:

$$\begin{bmatrix} x_c & y_c \end{bmatrix} = \begin{bmatrix} \frac{1}{N}\sum_{i=1}^{N} x_i & \frac{1}{N}\sum_{i=1}^{N} y_i \end{bmatrix}$$

Then we calculate the demeaned data(after translation).

$$x_i = x_i - x_c = x_i + t_x y_i = y_i - y_c = y_i + t_y$$

After the translation, the points are scaled such that their mean square distance from the origin is $\sqrt{2}$, which is the rotation matrix:

$$s = \sqrt{\frac{2}{\frac{1}{N}\sum_{i=1}^{N}(x_i^2+y_i^2)}}$$

Therefore, to write the transformation and rotation equation into the matrix form:

$$T_s = \begin{bmatrix} s & 0 & st_x \\ 0 & s & st_y \\ 0 & 0 & 1 \end{bmatrix}$$

4.2.Tasks:
You are given 2 pairs of images (inria1.tif, inria2.tif, frc1.tif, frc2.tif) for this assignment. The steps for this assignment are:
a.Using one of the given pairs of images, establish point correspondences.
b.Normalize the coordinates of the correspondences.
c.Using the normalized coordinates of the correspondences, compute the fundamental matrix.
d.Perform denormalization so as to find the fundamental matrix corresponding to the original data.

In these part, we use 8-points algorithm to calculate the fundamental matrix F. In this case, I choose 16 pairs to form A matrix. Before calculation, we firstly need to do normalization, using transformation and rotation matrix. Then calculate the normalized fundamental matrix. Finally, we should perform de-norm.

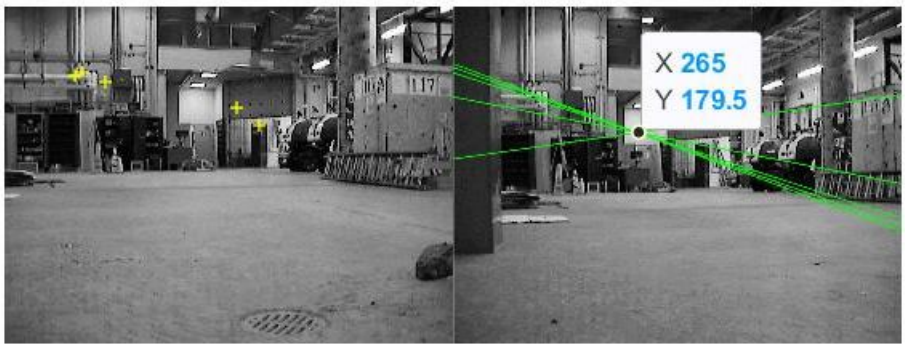
```
I1 = imread('C:\Users\hp\Desktop\Seme 2\4212 CV\CA1\Q4_files\frc1.tif');
I2 = imread('C:\Users\hp\Desktop\Seme 2\4212 CV\CA1\Q4_files\frc2.tif');
pts1 = gA;
pts2 = gB;

[n_pts1, T_s1] = normalize(pts1);
[n_pts2, T_s2] = normalize(pts2);

A = [n_pts2(:,1).*n_pts1(:,1) n_pts2(:,1).*n_pts1(:,2) n_pts2(:,1) ...
n_pts2(:,2).*n_pts1(:,1) n_pts2(:,2).*n_pts1(:,2) n_pts2(:,2) n_pts1(:,1) ...
n_pts1(:,2) ones(size(n_pts1(:,1)))];
% from svd(A), get F
[U, S, V] = svd(A);
F_temp = V(:,length(V));
F_temp = reshape(F_temp, [3,3])';
% enforce F singularity
[U_f, S_f, V_f] = svd(F_temp);
S_f(3,3) = 0;
```

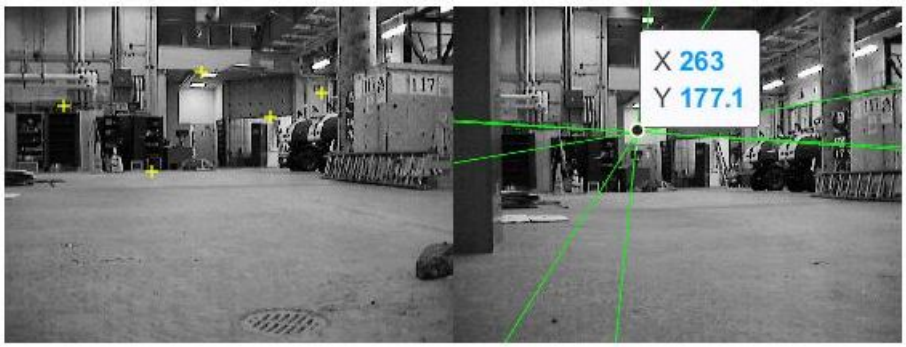
```
F_prime = U_f * S_f * V_f';

%denormalization
F = T_s2' * F_prime * T_s1;

function [ normalized_pts, T_s ] = normalize( pts )
    n = size(pts,1);
    centroid = sum(pts,1) / n;
    disp('The centroid is');
    disp(centroid);

    pts_demean = pts - centroid;
    squared_distance_sum = sum(sum(pts_demean.^2));
    scale_factor = sqrt(mean(squared_distance_sum) / 2); % scale to sqrt(2)
    normalized_pts = pts_demean / scale_factor;

    t_x = -centroid(1);
    t_y = -centroid(2);
    s = 1 / scale_factor;

    T_s = [s 0 s*t_x; 0 s s*t_y; 0 0 1];
end
```

The denormalized fundamental matrix is shown below. It's rank is 2.

$$F = \begin{bmatrix} -3.136e-10 & 7.09e-07 & -1.020e-4 \\ -8.462e-07 & 6.395e-08 & -1.633e-3 \\ 1.459e-4 & 1.7e-3 & 4.38e-2 \end{bmatrix}$$

Next, using the fundamental matrix and the points in left image, we can easily obtain the epipolar line in right image, which is $l_r = F\hat{P}_l$.
In order to shown the graphs more clearly, each time I only present 5 points and 5 corresponding epipolar lines. The images are shown below.





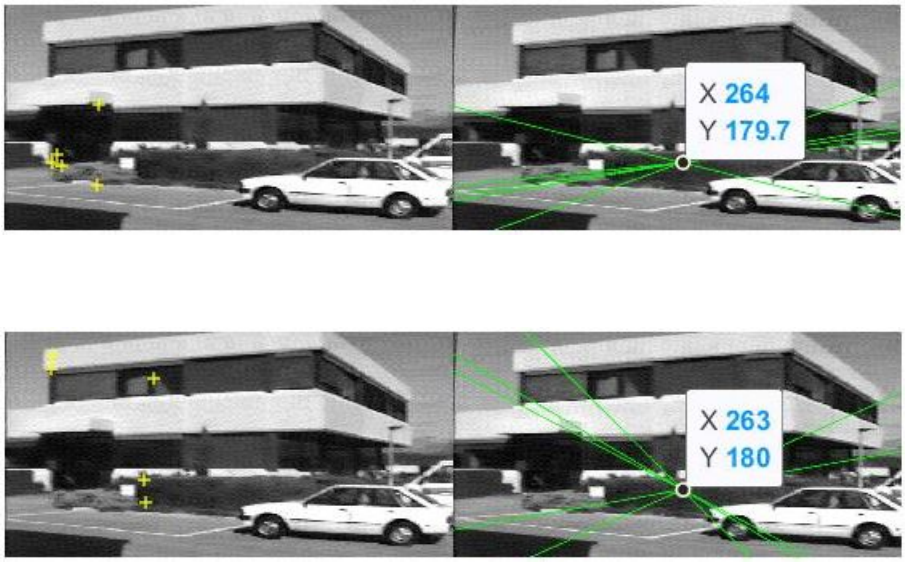


Figure 8: Randomly selected points and corresponding epipolar lines

The graphs imply that all the epiplar lines will intercept to one particular point, which is the epipole. I manualy marked the intercept point in graphs, thus it may have some error. We can see that each point from the left image can correspond to a particular epipolar line in right image. Then, it's easier for us to search the right point regarding to the left point. The matlab code is shown below.

```
ga_new = [gA(1:5,:) ones(5,1)];
r_epi_line = (F * ga_new')';

figure(100);
f1 = subplot(1,2,1);
set(f1,'position',[0.1,0.1,0.4,0.4])
imshow(I1)
text(double(ga_new(:,1)),double(ga_new(:,2)),'+','Color','yellow');

f2 = subplot(1,2,2);
imshow(I2)
set(f2,'position',[0.5,0.1,0.4,0.4])
for i=1:5
    ex_1 = r_epi_line(i,:);
    x = 0 : 1: 640;
    y = -ex_1(1)/ex_1(2) * x - ex_1(3)/ex_1(2);
    line(x,y,'Color','g')
end
hold on
```

Q5 Motion Perception

(a) In the figure below, the two squares are translating horizontally in the opposite directions as indicated by the arrows. Indicate the respective perceived motions if you are looking through the apertures 1, 2, and 3. Explain your answers.

When only look at local part of the whole, we can easily get deceived perceptions. We'll perceive:
Aperture 1: Moving upper left
Aperture 2: Moving right
Aperture 3: Moving downwards
The reason why we get false perception is that we generally perceive besed on egde moving directions. However, the edge in aperture is not the real edge from the whole image.

(b)Barber-pole with Occlusion: The figure in this question illustrates various barber-pole configurations, with occluders placed along either vertical or horizontal sides of the barber-pole. The arrows indicate the perceived direction of barber-pole motion. Explain why the perceived motion tends to be biased in the direction orthogonal to the occlusion boundary.

Without the occluders, we can no doubtedly perveive the right moving direction of the strides. However, when adding occluders in different direction, it influences our dicisions. Also, line movement can be various. The shape of the aperture thus tends to determine the perceived direction of motion for an otherwise identically moving contour. A vertically elongated aperture makes vertical motion dominant whereas a horizontally elongated aperture makes horizontal motion dominant.

In middle one, we are likely to assume that the strides are originate from the above occluder and ends at the below one. Similarly, from the right image, we are easily conclude that the strides originate from the left and terminate at the right. Thus, the problem occurs.

(c)*Referring to the diagram below, the stimulus consists of two orthogonal bars that move sinusoidally, 90 degree out of phase (Figure a and b). When presented together within an occluding aperture (Figure c), the bars perceptually cohere and appear to move in a circle as a solid cross. However, when presented alone (Figure d), they appear to move separately (the horizontal bar translates vertically and the vertical bar translates horizontally), even though the image motion is unchanged in Figures c and d. In either stimulus condition, both percepts are legitimate interpretations of the image motion. Yet a single interpretation is predominantly seen in each case. Why?

In graph d, there's no occluder, thus the endpoints are real ones. Hence, we will never get confused and make wrong decisions. In other cases, the endpoints are not real ones. We can only judge from the intersection points between the occluders and lines. Therefore, they are probably the fake edges and influence our judgement. From graph (a), the intersection points can only move up and down so that we will claim it moves up and down and so forth. As to graph (c), we will mainly focus on the edges in the middle of the picture. Thus we conclude it moves circularly.

(d)* Assume you are given enough optical flow measurements at different (x, y) locations, solve (x, y, z) in the least squares sense by writing down the least squares equation of the form $Ax = b, b \neq 0$. Explain why in this case, the equation is of the non-homogeneous form, whereas that in Question 3e is of the homogeneous form Ax=0, and explain whether it is appropriate or inappropriate to change the formulation of this Question 5d to the Ax=0 form.

The least square form can be writen as:

$$\begin{bmatrix} xy & -(x^2+1) & y \\ y^2+1 & -xy & -x \end{bmatrix} \begin{bmatrix} w_x \\ w_y \\ x_z \end{bmatrix} = \begin{bmatrix} u \\ v \end{bmatrix}$$

There are only 2 equations but 3 unkonwns. We can perform SVD and use psydo-inverse to solve the unknowns. Besides, it's not suitable to change to $Ax = 0$, because the null vector will not be unique.