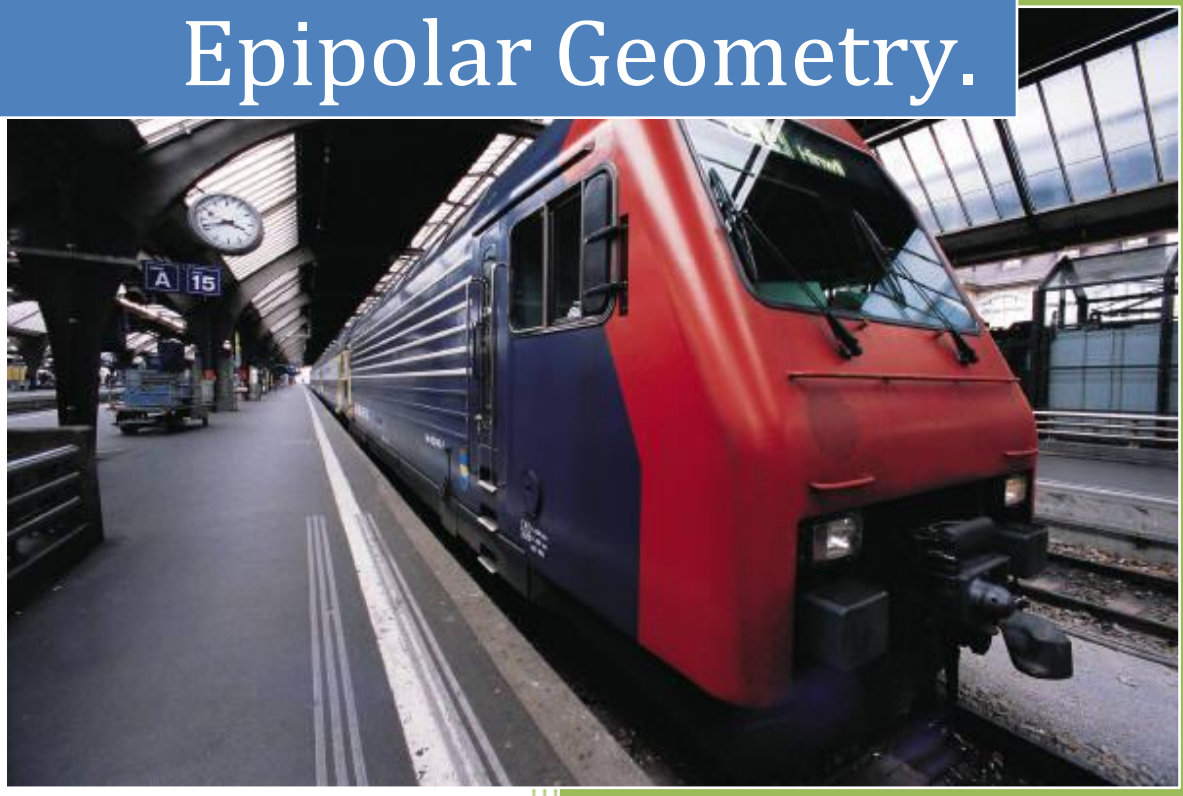
2018

Computer Vision. Practical 3 Epipolar Geometry.



Surajudeen Abdulrasaq and

Ezukwoke Ifeanyi Kenneth.

Machine Learning and Data Mining

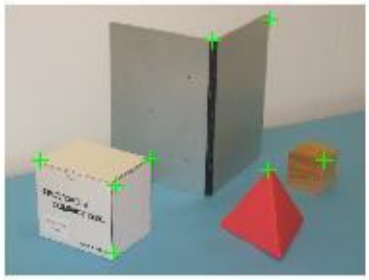
3/15/2018

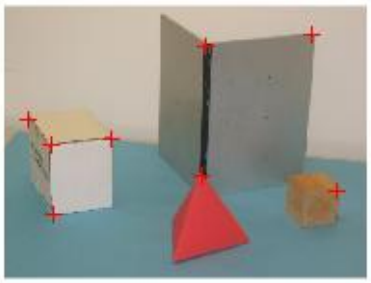
Introduction:

The epipolar geometry referred to the intrinsic projective geometry involving two views. This depends only on the cameras' internal parameters and relative pose, and completely independent of world view. Two images of the same view are related by the epipolar geometry. To determine this epipolar geometry we need to estimate the 3x3 singular matrixes known as the Fundamental matrix.

Step1:

Here we find the correspondence points by manually matching them, which is necessary in other to estimate the Fundamental matrix required for step 2.





Manually obtain 8 corresponding points from the two images stereo1 (left) and stereo2 (right).

(Results)

Calculation of the fundamental matrix algorithm by 8 points

ans = 368.4144 346.0319 190.4465 533.2660 190.4465 189.4362 187.0241 491.6230

67.6658 300.0745 327.7727 330.7128 30.0187 235.3936 450.9813 444.1364

Step 2:

Inside a MATLAB function called ***matF.m,*** Calculate the fundamental matrix with 8 point algorithm using SVD. You can use the reference:

```
function F=MatF(left_image_points,right_image_points)
% p1,p2 : vectors of coordinates ...x...

% initialisation
A = [];
N=8;


% Write a martix of homogeneous linear system Af=0
load left_image_points;
load right_image_points;
% Write a martix of homogeneous linear system Af=0
% ---> TODO <---
[a b] = size(left_image_points);
for i=1:a

    x1 = left_image_points(i,1);
    y1 = left_image_points(i,2);
    x2 = right_image_points(i,1);
    y2 = right_image_points(i,2);
    A(i,:) = [x1*x2 y1*x2 x2 x1*y2 y1*y2 y2 x1 y1 1];

end

% Solving the linear system for the estimate of F by SVD
% Ist step: Linear solution
[U,S,V] = svd(A)
Fest = reshape(V(:,end),3,3)

% 2nd step: Constraint enforcement
Fest=Fest';
[U,S,V] = svd(Fest);
F = U*diag([S(1,1) S(2,2) 0])*V';
```

(Results of the matrix)

0.3275 -0.1012 -0.0844

-0.1012 -0.0282 0.0802

-0.0844 -0.0802 0.9168

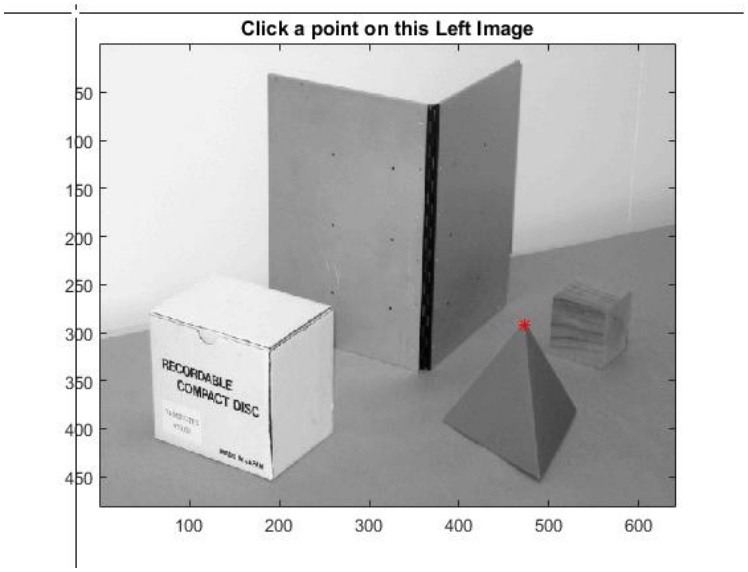
Step 3 and 4:

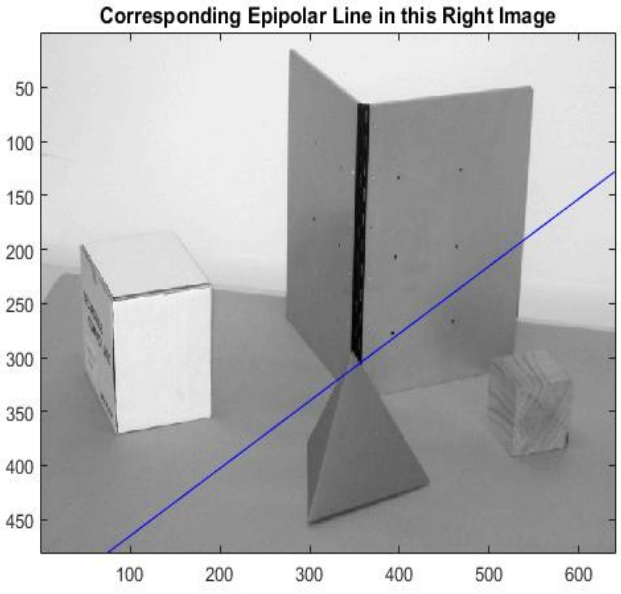
Calculate the Epipoles of two cameras and implements a function which will allow you to see the epipolar line in the right image corresponding to any point in the left image. That means, if you click one point in the left image you should see the epipolar line in the right image.

```
clear all;
close all;

N = 8;
%I1 = imread('stereo1.jpg');
%I2 = imread('stereo2.jpg');

X1 = ones(2,N); % Empty vector to store the 8 points
X2 = ones(2,N);
F = zeros(3,3);  % Empty matrix to calculate the fundamental matrix

disp('Calculation of the fundamental matrix algorithm by 8 points');
% Step 1 : Manually obtain 8 corresponding points from the two images
% stereo1 (left) and stereo2 (right)
% In order to save your time for this task, I encourage you to use the
% function: clickPoints
% ---> TODO <---

%load('FM1.mat');
%load('X4.mat');

 % Step 2 : Calculate the fundamental matrix with 8 point algorithm
% Reference: Chapter 11: Multiview Geometry and lecture slides (56-58)
% Completely writing the function is a part of your exercise session.
% May be you can exploit the function MatF in you exercise subdirectory
% which is 70% complete.
% ---> TODO <---
%load('X3','X4')
load('C:\mlab\sesion3\betterPointsX1X2.mat')
FM1 = MatF1(X1,X2);

% Step 3 : Calculate the Epipoles of two camera
% Reference: Chapter 9, page : Multiview Geometry
% Book chapter download link: http://www.robots.ox.ac.uk/~vgg/hzbook/hzbook2/HZepipolar.pdf
% Use of MATLAB function 'NULL' may be helpful for you.
% ---> TODO <---
close all;
left_image = double(rgb2gray(imread('stereo1.jpg')));
right_image = double(rgb2gray(imread('stereo2.jpg')));

[m n] = size(left_image);

figure(1),imagesc(left_image); colormap(gray); title('Click a point on this Left Image');
figure(2),imagesc(right_image); colormap(gray); title('Corresponding Epipolar Line in this Right Image');

list =['r' 'b' 'g' 'y' 'm' 'k' 'w' 'c'];
for i=1:100

   % Clicking a point on the left image:
   figure(1);
   [left_x, left_y] = ginput(1);
```

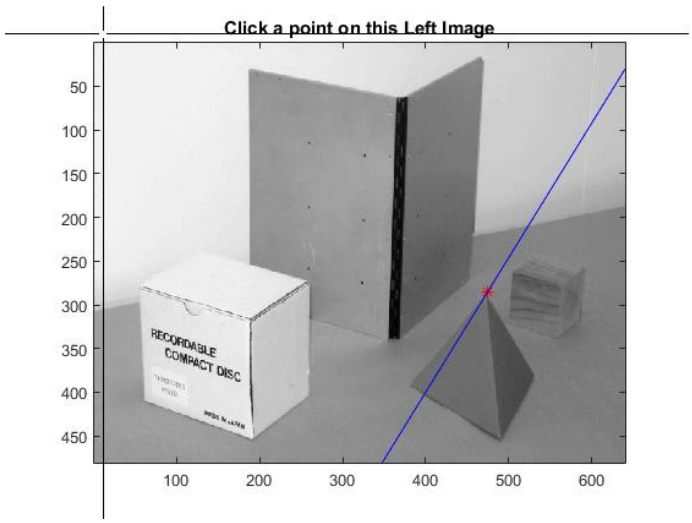
```
hold on;
plot(left_x,left_y,'r*');

% Finding the epipolar line on the right image:
left_P = [left_x; left_y; 1];

right_P = FM1*left_P;

right_epipolar_x=0:n;
% Using the eqn of line: ax+by+c=0; y = (-c-ax)/b
right_epipolar_y=(-right_P(3)-right_P(1)*right_epipolar_x)/right_P(2);
figure(2);
hold on;
plot(right_epipolar_x,right_epipolar_y,list(mod(i,8)+1));

% Now finding the other epipolar line
% We know that left epipole is the 3rd column of V.
% We get V from svd of F. F=UDV'
left_epipole = FV(:,3);
left_epipole = left_epipole/left_epipole(3);

left_epipolar_x = 1:2*m;
left_epipolar_y = left_y + (left_epipolar_x-left_x)*(left_epipole(2)-left_y)/(left_epipole(1)-left_x);
figure(1);
hold on;
plot(left_epipolar_x,left_epipolar_y,list(mod(i,8)+1));

end
```

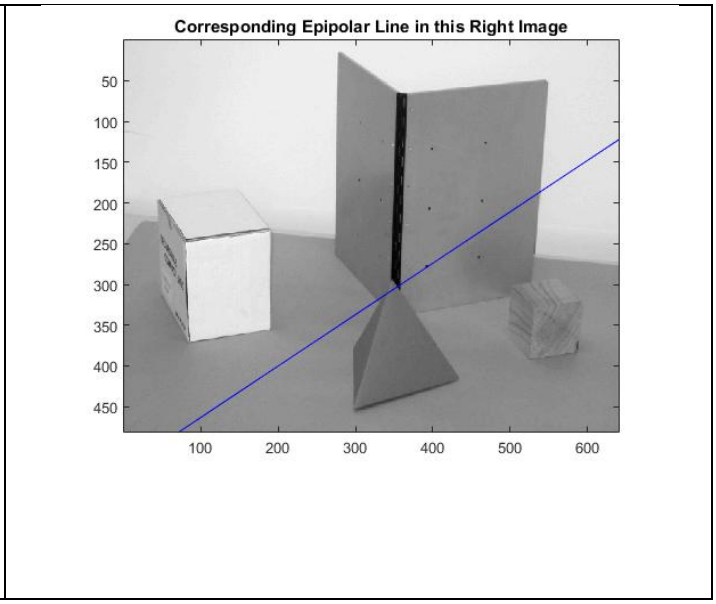
Observation1: the results below is not too good, we were unable to Match exactly the tip of the cone, so we will try re-matches the images with the 8-point algorithm and re-estimates the Fundamental matrix.



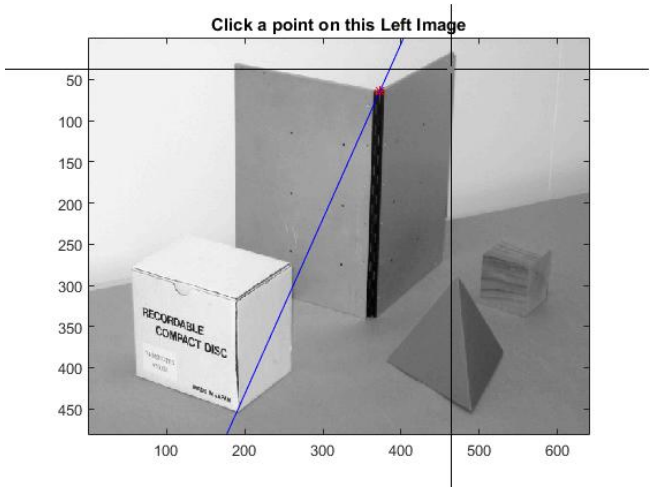


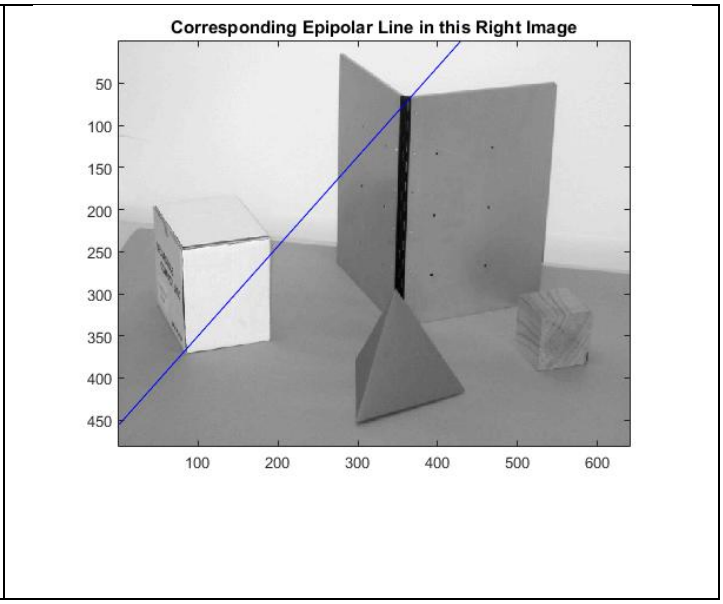
Observation2: After several try and increasing the no of iterations, this is the closest we can get, so I guess the background at the tip of the cone in the correspondence image is a big factor in estimating epipolar line, if we look closely at the background of the correspondence image in the right there's a black background line of cup-board which I think is responsible for my result.



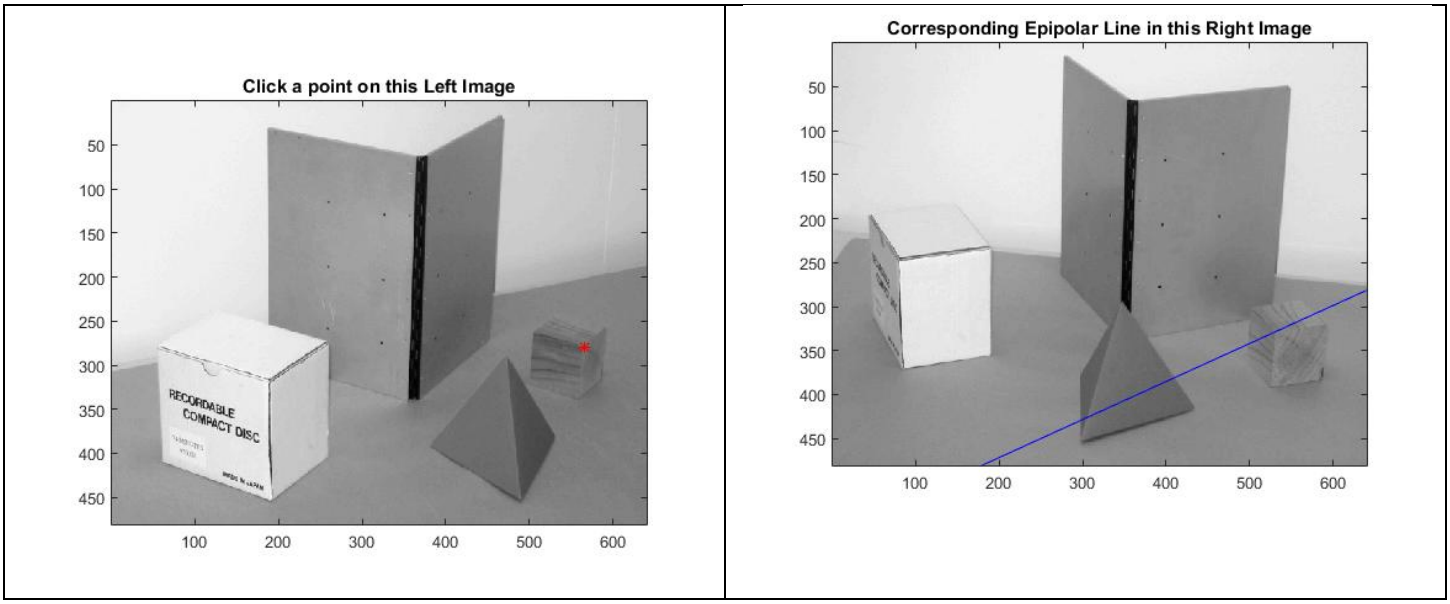


Observation3: In order to test my assertion above we decided to try other point and this time I got a better results.



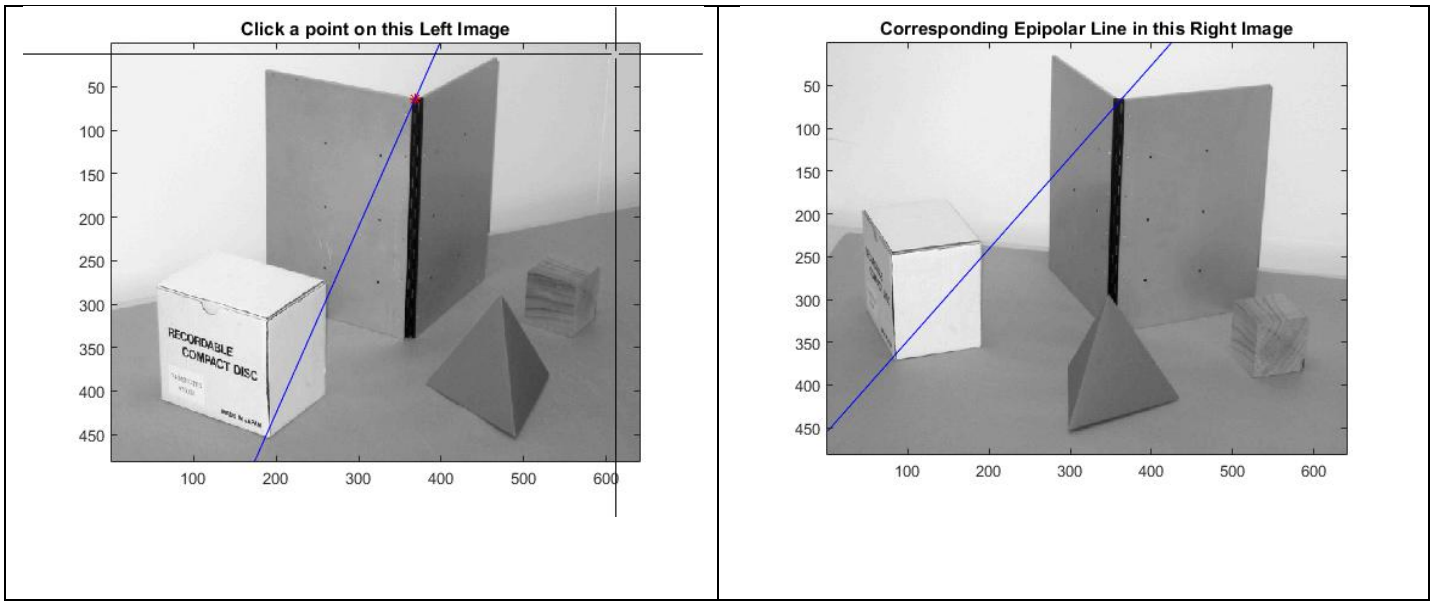


Observation4: Again trying with other point: Not too bad results

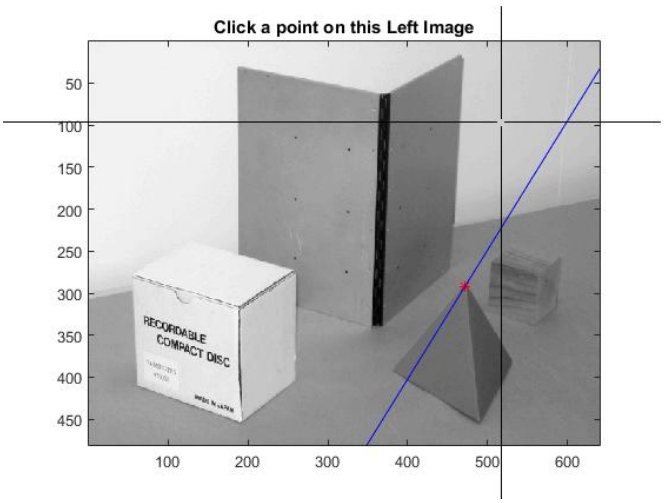


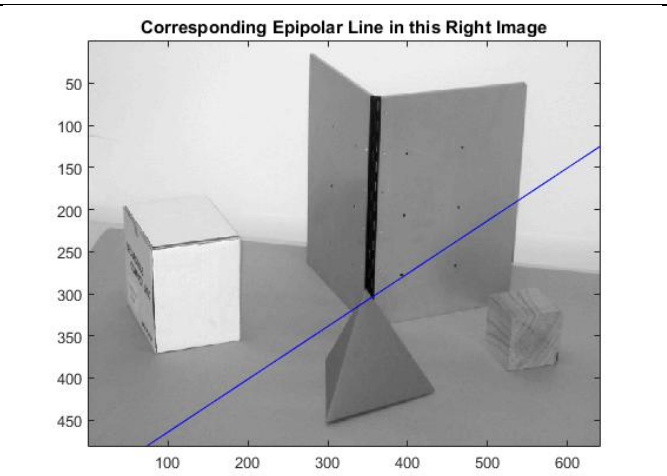
Use the points saved in the file *betterPointsX1X2.mat*: Using the file 'betterPointsX1X2', we got an improved result; we decided to test this using the same points in the previous experiments to get a better evaluation.

Observation5: A better point matching compare to the observation 3, although the results are close but this gives us almost 100% accuracy.



Now back to the cone tip: Now we decided to try the Cone tip using the ***"betterPointsX1X2.mat"*** to know if we will get different results but our results is not different.





Conclusion: I think using just the 8-points algorithm might not be enough to estimate accurately the fundamental matrix, maybe more points will be needed in order to arrive at better results. In addition to this, wide range of points should be chosen at different corner all over the image in order to get uniformly distributed outcome.