HW 2: Assignment Report

Titash Mandal (tm2761@nyu.edu, N13592626)

4. a Epipolar geometry from F-matrix

1 Scope of the Experiment

This objective of this assignment is to take a stereo pair of images, and explore the epiploar geometry .

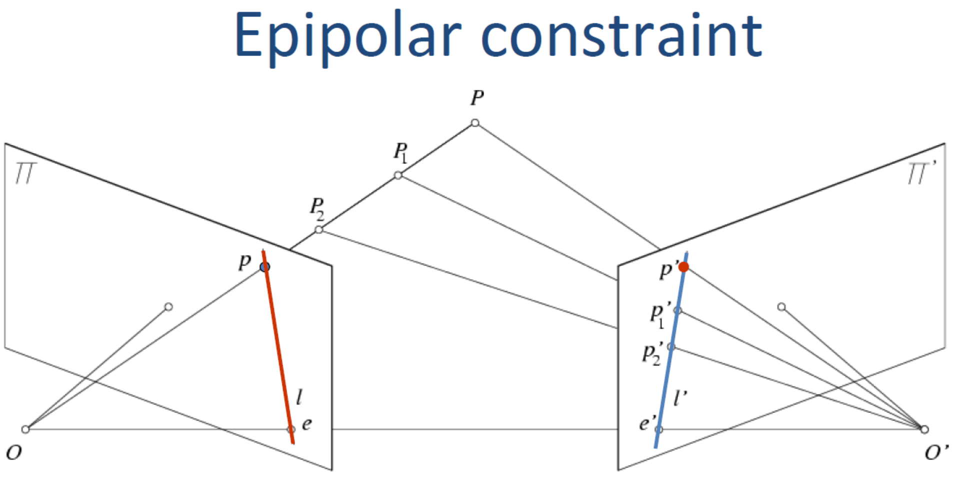


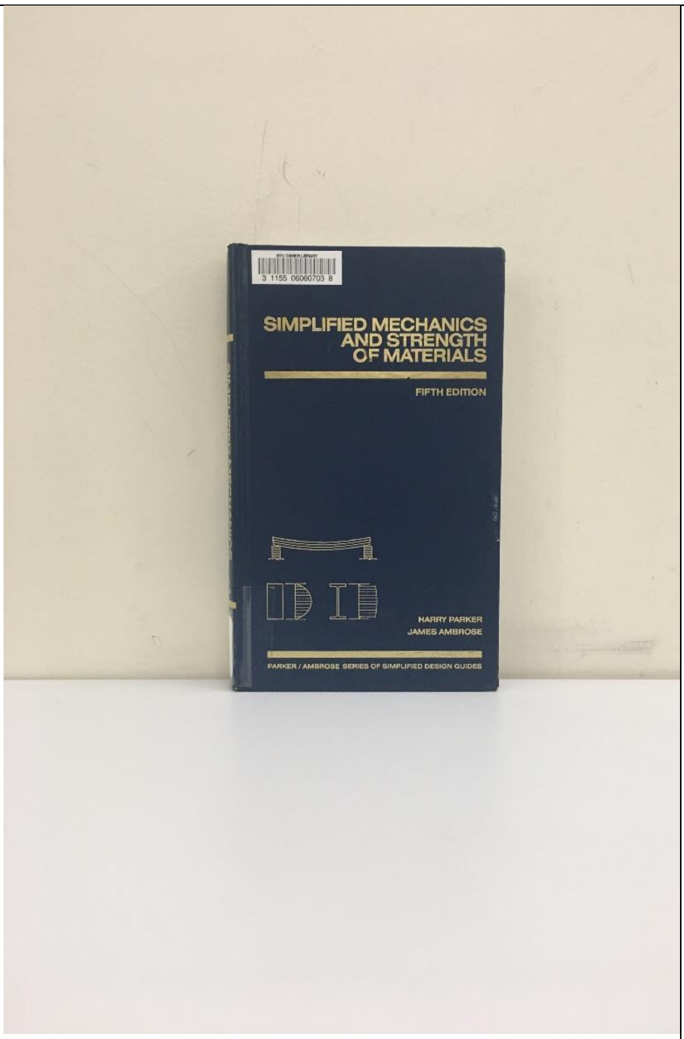
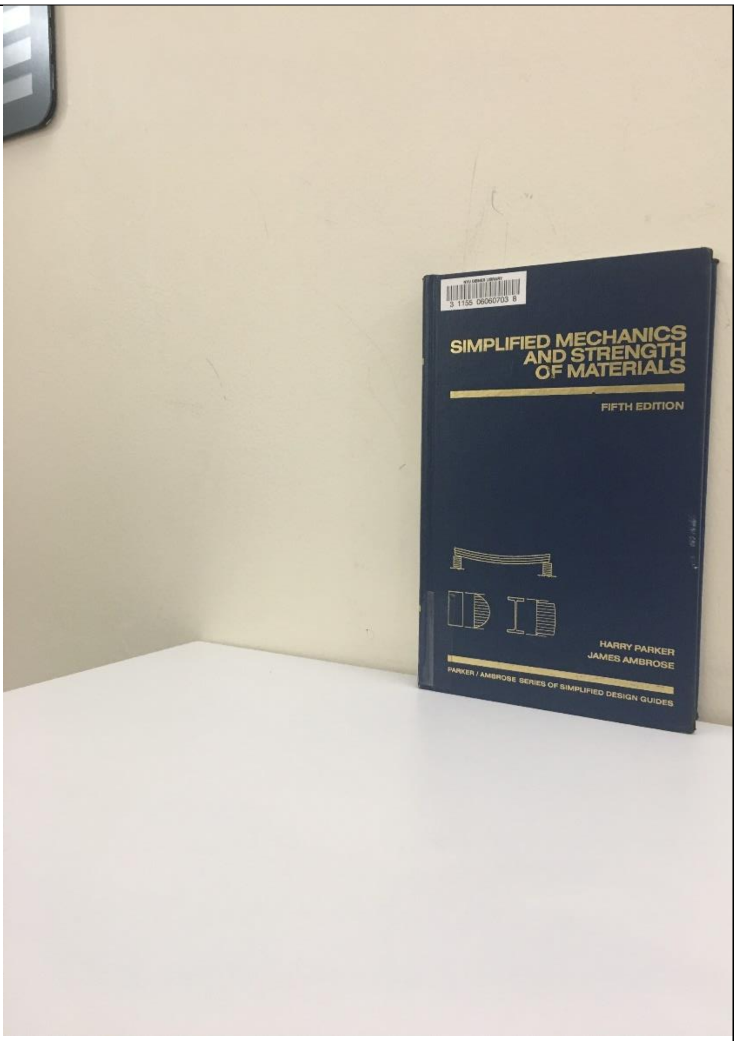
Figure 1.

The Figure 1. Shows us the epipolar geometric constraint. Let us consider the images p and p' of a point P observed by a camera with optical centers O and O'. These five points all belong to the epipolar plane defined by the two intersecting rays OP and O'P. The point p' lies on the epipolar line l' where this plane and the retina Π' of the second camera intersect. The line l' is the epipolar line associated with the point p, and it passes through the point e' where the baseline joining the optical centers O and O' intersects Π'. Likewise, the point p lies on the epipolar line l associated with the point p', and this line passes through the intersection e of the baseline with the plane Π.

2 Experimental Procedure

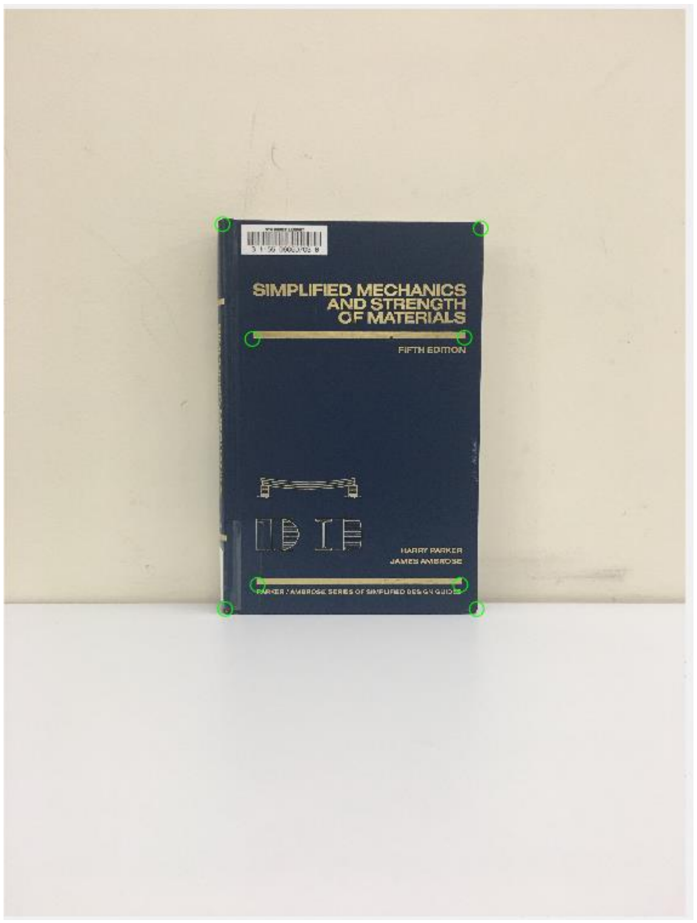
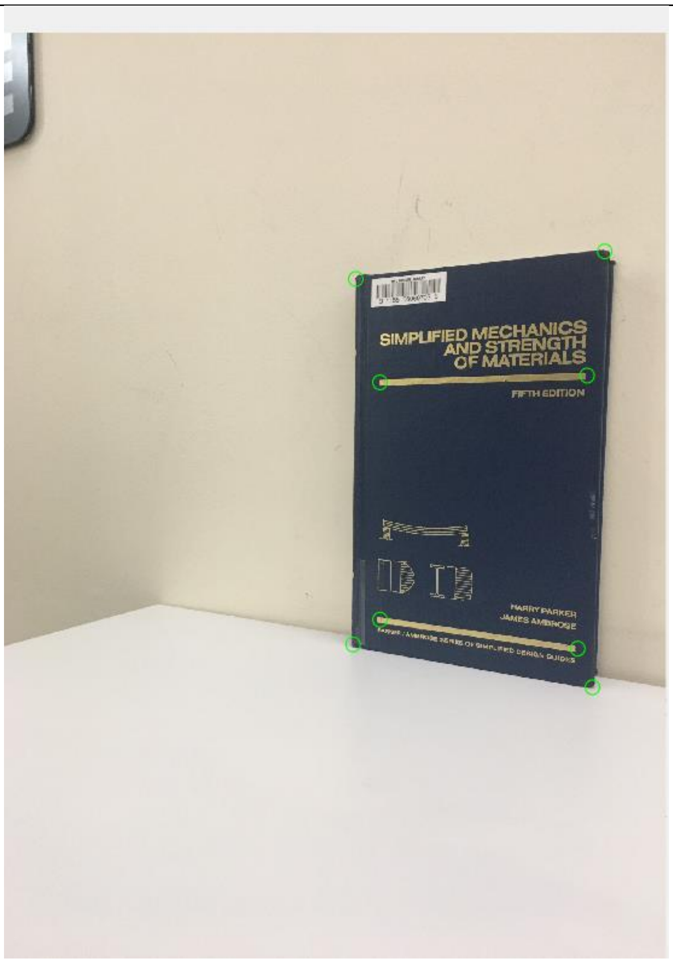
2.1 Data Capture

Taking a camera with a focal length 4.15mm kept at 542mm from the origin of the world co-ordinate, a stereo-pair of images were taken of the same object. The first image was taken and this camera's position (with optical center O) was set as the frame of reference. The camera was then translated and rotated to take the second image of the same object. The translation was along the x direction and it was 166 mm from the camera's first optical center. Thus, the baseline distance between the camera's optical center was 166mm.

 SIMPLIFIED MECHANICS AND STRENGTH OF MATERIALS FIFTH EDITION HARRY PARKER JAMES AMBROSE PARKER / AMBROSE SERIES OF SIMPLIFIED DESIGN GUIDES **Figure 2.1.1-** This is the image of the object with the camera's optical center O chosen as the frame of reference.	 SIMPLIFIED MECHANICS AND STRENGTH OF MATERIALS FIFTH EDITION HARRY PARKER JAMES AMBROSE PARKER / AMBROSE SERIES OF SIMPLIFIED DESIGN GUIDES **Figure 2.1.2-** This is the image of the object taken after the camera undergoes a rotation and translation with camera's optical center now being O' shifted 166mm from the center O.

2.2 Calculating the pixel co-ordinates from the stereo pair of images.

After the two images were taken, eight points were selected in the first image. Using the MATLAB function ginput() the pixel co-ordinates (u,v) of those eight points in the first image with camera's optical center O, was calculated. The same procedure was followed for the second shifted and rotated image. The points selected in the second image were corresponding to the points selected in the first image.

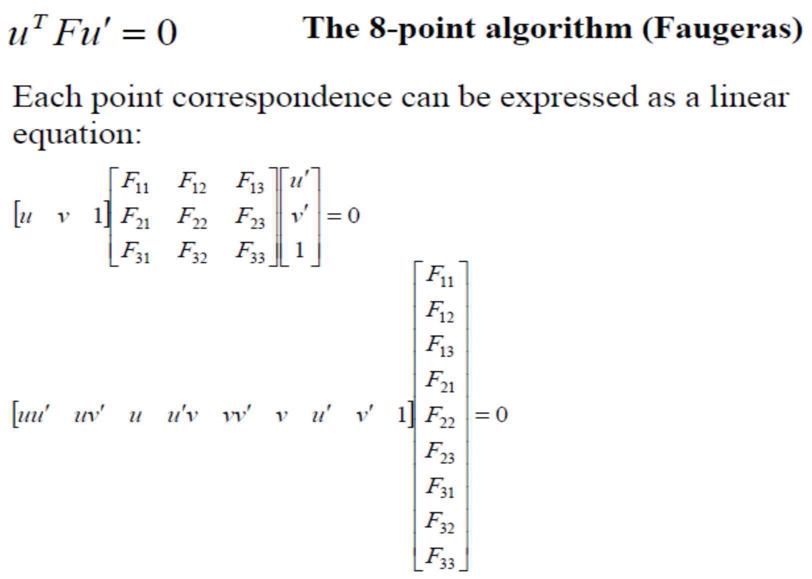
 **Figure 2.2.1** The eight points chosen in the first image are shown in the image marked as green.	 **Figure 2.2.2** The eight points chosen in the second image correspond to the points chosen in the first image and are shown marked as green.

2.3 Fundamental Matrix from the stereo pair of images.

We now have eight sets of points in the first image and corresponding eight sets of points in the second image. We know from Epipolar geometry that,

$$u^T F u' = 0 \qquad F = K_1^{-T} \mathcal{E} K_2^{-1}$$

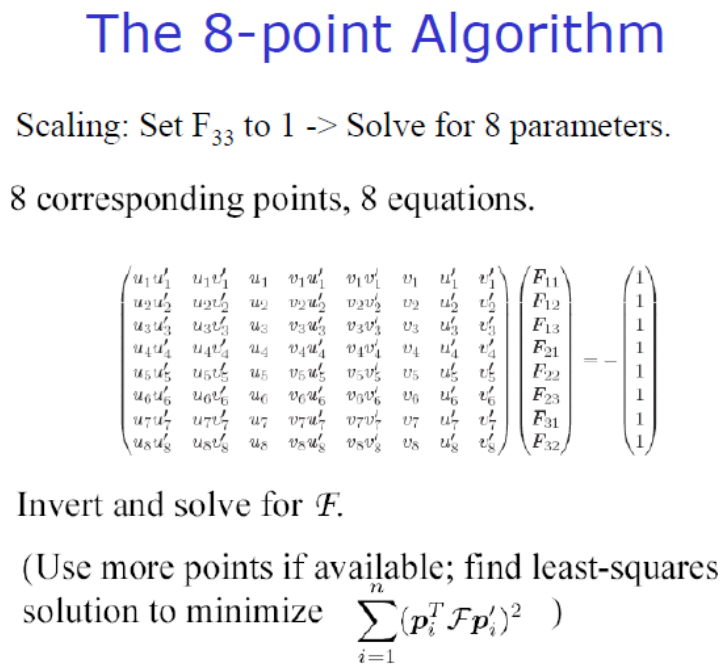
Where u represents the pixel co-ordinates in the first image, and u' represents the pixel coordinates in the second image. **F** is called as the **Fundamental Matrix**. The epipolar geometry is the intrinsic projective geometry between the two stereo views. It is independent of scene structure, and only depends on the cameras' internal parameters and relative pose. The **fundamental matrix F encapsulates this intrinsic geometry**. It is a 3 × 3 matrix of rank 2. It can be computed from correspondences of imaged scene points that we have already calculated, without requiring knowledge of the cameras' internal parameters or relative pose.


$u^T F u' = 0$ **The 8-point algorithm (Faugeras)**

Each point correspondence can be expressed as a linear equation:

$$[u \quad v \quad 1]\begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix}\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = 0$$

$$[uu' \quad uv' \quad u \quad u'v \quad vv' \quad v \quad u' \quad v' \quad 1]\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$


The 8-point Algorithm

Scaling: Set F_{33} to 1 -> Solve for 8 parameters.

8 corresponding points, 8 equations.

$$\begin{pmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' \\ u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' \\ u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' \\ u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' \\ u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' \\ u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' \end{pmatrix}\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{pmatrix} = -\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Invert and solve for $\mathcal{F}$.

(Use more points if available; find least-squares solution to minimize $\sum_{i=1}^{n}(\boldsymbol{p}_i^T \mathcal{F} \boldsymbol{p}_i')^2$)

The Fundamental Matrix was calculated using the 8-point Algorithm. Having all the 8 points in the first image corresponding to the second image, we can calculate an 8*8 matrix and use the above equation to estimate the fundamental matrix F. The following equation is how F was calculated.

```
[U,S,V]=svd(EightPointsMatrix);..........................................(1)
F = reshape(V(:,9),3,3)';
```

The fundamental matrix can also be calculated using an inbuilt function of MATLAB as follows.

[Fundamental,inliers]= estimateFundamentalMatrix(ShiftedPixels,OriginalPixels,'Method','Norm8Point').(2)

The fundamental matrix calculated using the two methods are:

Using (1),

F_formula=

$$\begin{pmatrix} 6.028261148043974E-8 & -3.776535778239122E-7 & 0.0012914115390374 \\ 3.5732526457037527E-7 & -5.018250673484659E-9 & -0.0010010103453802 \\ -0.00148472571965862 8 & 8.245724850484517E-4 & 0.9999972229481467 \end{pmatrix}$$

Using (2),

F_Matlab_Method= $\begin{pmatrix} 5.845920014452083E\text{-}8 & -3.5802422958760593E\text{-}7 & 0.0012268784586190195 \\ 3.4392460727119534E\text{-}7 & 3.2984128862172615E\text{-}9 & -9.762324850456811E\text{-}4 \\ -0.0014240846451785989 & 7.695865416182974E\text{-}4 & 0.9999974607293916 \end{pmatrix}$

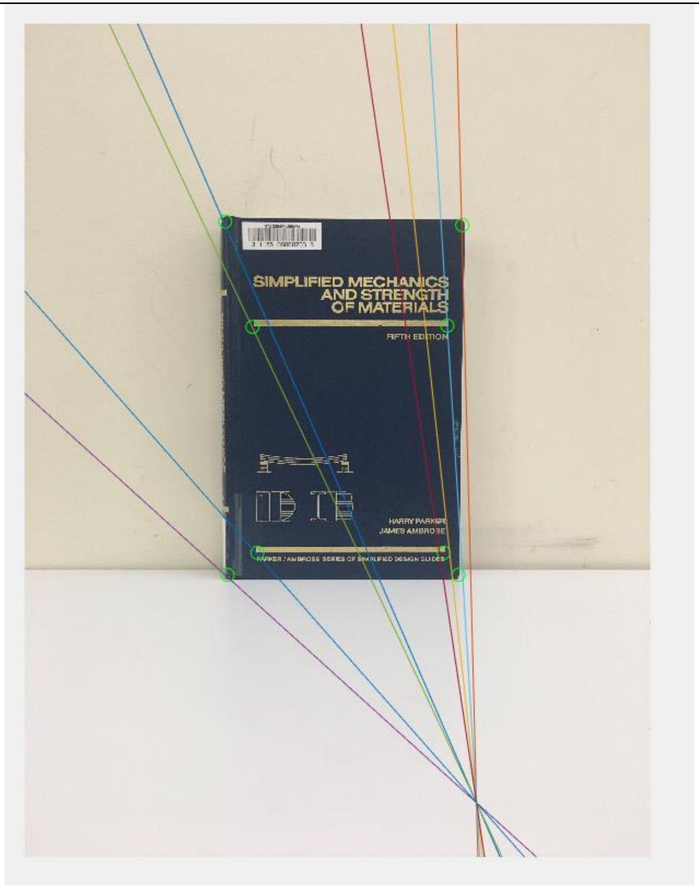
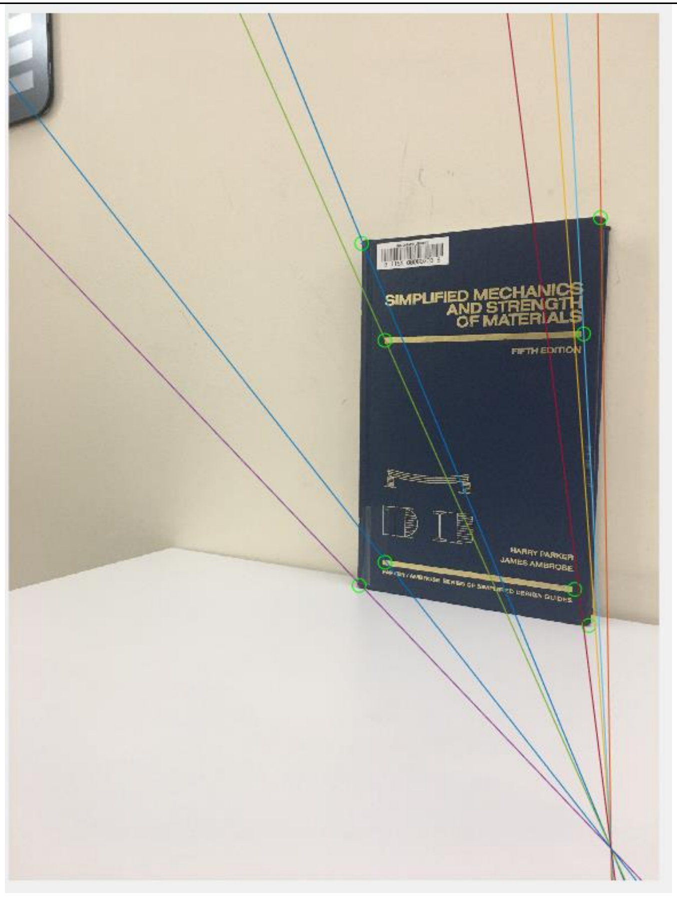
2.4 Determining the epipolar line from the Fundamental Matrix equation.

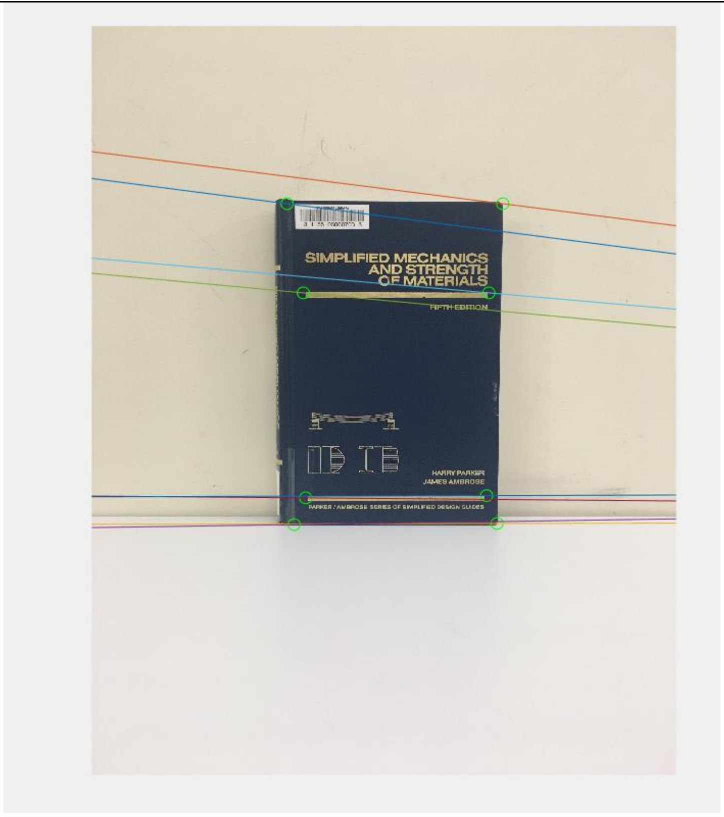
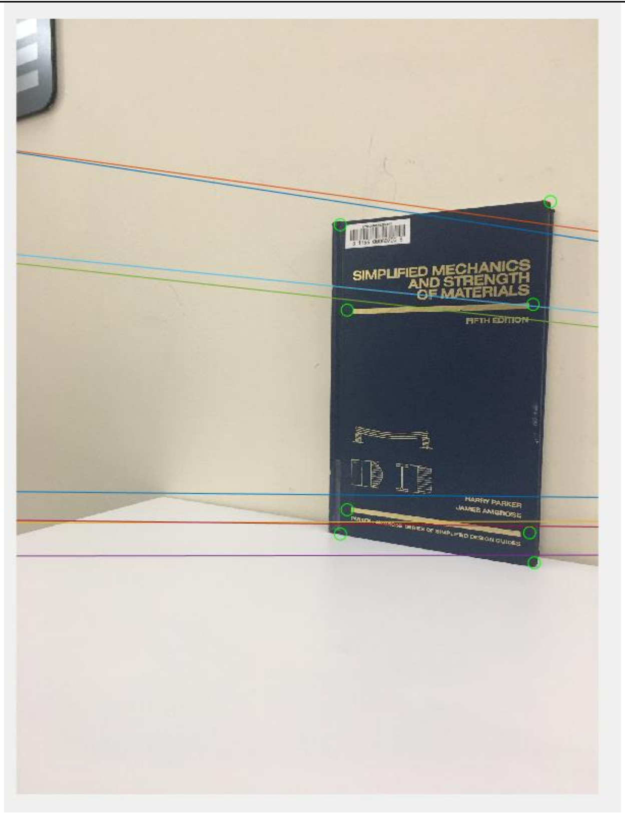
Choosing a point in the first image, the epipolar line in the rotated and translated image was calculated.
From the equation of epipolar geometry we know that,

$$p^T \mathcal{F} p' = 0,$$

Here l=Fp' can be interpreted as the coordinate vector of the epipolar line l associated with the point p' in the first image. By symmetry, it is also clear that the coordinate vector representing the epipolar line associated with p in the second image is l'=Fp. Using these equations, the epipolar lines were drawn passing through the set of selected points in the images.

There was a difference observed while plotting the epipolar lines using the two different **F matrices found out in two different methods.** The results are given as follows.

 **Figure 2.5.1-** The epipolar lines are drawn through the selected six points using the fundamental matrix **F_Matlab_Method** .	 **Figure 2.5.2-** The epipolar lines are drawn through the selected six points using the fundamental matrix **F_Matlab_Method** .

 **Figure 2.5.2** The epipolar lines drawn using the fundamental matrix , **F_formula**.	 **Figure 2.5.2** The epipolar lines drawn using the fundamental matrix , **F_formula**.

We get a more accurate result using the F matrix calculated in equation (2).

2.5 Calculating the epipole of the two images.

The epipole is the projection of the camera's optical center onto the opposite image planes. The epipole is calculated in the following way.

```
%Epipole of the first camera's optical center into the second image plane.
[U,V] =eigs(F*F');
e1 = U(:,1)./V(3,1);
%Epipole of the first camera's optical center into the second image plane.
[U,V] =eig(F'*F);
e2 = U(:,1)./V(3,1);
```

We now compare the result of the epipole coordinates obtained using the previous method with coordinates obtained using the inbuilt MATLAB function. The MATLAB inbuilt function for calculating the epipole is,

```
imageSize[4032 3024];
[isIn,epipole]=inEpipoleInImage[flMedS,imageSize]
```

The co-ordinates of the epipole of the two images obtained from MATLAB function	**The co-ordinates of the epipole of the two images obtained from calculation.**

XRight= 2801.2557697314865
YRight=3884.1997702767794
	XRight=1424.0831588243435
	YRight=2958.5153226498323

XLeft= 3211.7677697329805
YLeft= 4067.11371224367794
	XLeft= 2791.7677697329805
	YLeft= 3992.71737122436779

To verify which method gives more accurate result, the actual epipole is obtained by the ginput function. The actual co-ordinate of the epipole in the right image(without rotation and translation) is:

X=2192.437007874016 Y=3775.870078740157.

The actual co-ordinate of the epipole in the left image (with rotation and translation) is:

X=3172.897217879222 Y=4001.234567993840.

Thus, the epipoles estimated by the calculation has more error from the actual epipole than the epipole calculated using the MATLAB function.

The MATLAB code for the assignment is as follows.

```
%Reading the images from the directory.
FrameofReference=imread('C:\NYU_SEM_2\Computer Vision\Assignment 2\Original.jpg');
FrameofRef=imrotate(FrameofReference,-90);
imshow(FrameofRef);
%Calculating the eight set of points from the first image
[X,Y]=ginput(8);

 %Following the same steps for the second image
RotatedTranslatedImage=imread('C:\NYU_SEM_2\Computer Vision\Assignment 2\Rotated.jpg');
RotatedTranslatedImage=imrotate(RotatedTranslatedImage,-90);
imshow(RotatedTranslatedImage);
[XP,YP]=ginput(8);

%Creating an 8*2 Matrix using the (u,v) coordinates from the first and second image.
Original_Pixels=[X,Y];
Shifted_Pixels=[XP,YP];

%calling the function to calculate the 8 point matrix with corresponding points
EightPointsMatrix=EightPointsMatrix(X,Y,XP,YP);
[U,S,V]=svd(EightPointsMatrix);

%Calculating the Fundamental Matrix
Fundamental_Matrix = reshape(V(:,9),3,3)';

TempF=[Fundamental_Matrix(1,:),Fundamental_Matrix(2,:),Fundamental_Matrix(3,1:2)];
%To check to see if the equation holds and we get a 8*1 matrix of only 1’s
EightPointsMatrix*TempF'

%estimating the epipolar lines corresponding to the points
[F_formula,inliers] = estimateFundamentalMatrix(Original_Pixels,Shifted_Pixels,'Method','Norm8Point');
imshow(FrameofRef);
hold on;
plot(Original_Pixels(inliers,1),Original_Pixels(inliers,2),'go')
```

```
%plotting epipolar lines on the Rotated and Translated image
imshow(RotatedTranslatedImage);
hold on;
plot(Shifted_Pixels(inliers,1),Shifted_Pixels(inliers,2),'go')
epiLines = epipolarLine(F_formula,Original_Pixels(inliers,:));
points = lineToBorderPoints(epiLines,size(RotatedTranslatedImage));
line(points(:,[1,3])',points(:,[2,4])');
truesize;

%plotting epipolar lines on the Original image
[F_formula,inliers] = estimateFundamentalMatrix(Shifted_Pixels,Original_Pixels,'Method','Norm8Point');
imshow(RotatedTranslatedImage);
hold on;
plot(Shifted_Pixels(inliers,1),Shifted_Pixels(inliers,2),'go')

%plotting epipolar lines on the Rotated and Translated image
imshow(FrameofRef);
hold on;
plot(Original_Pixels(inliers,1),Original_Pixels(inliers,2),'go')
epiLines = epipolarLine(F_formula,Shifted_Pixels(inliers,:));
points = lineToBorderPoints(epiLines,size(FrameofRef));
line(points(:,[1,3])',points(:,[2,4])');
truesize;

%Epipole of the first camera's optical center into the second image plane.
[U,V] =eigs(Fundamental_Matrix*Fundamental_Matrix');
e1 = U(:,1)./V(3,1);
%Epipole of the first camera's optical center into the second image plane.
[U,V] =eig(Fundamental_Matrix'*Fundamental_Matrix);
e2 = U(:,1)./V(3,1);
```

CalculatingEightPoint.m

```
%The function to calculate the 8 point Matrix using the corresponding points of the two images.
function matrix=EightPointsMatrix(X,Y,XP,YP)
matrix=[]
for t=1:8
   result=IndividualRowEquation(X(t), Y(t), XP(t), YP(t));
   matrix=[matrix;result];
end
end
```

OnePointEquation.m

%To calculate the following Matrix

$$[uu' \quad uv' \quad u \quad u'v \quad vv' \quad v \quad u' \quad v' \quad 1]$$

```
function calculates= IndividualRowEquation(u,v,up,vp)
calculates=[u*up, u*vp, u, up*v, v*vp, v, up, vp,1];
end
```

4. b 3D Object geometry via triangulation

1. Scope of the Experiment

The scope of this experiment is to take stereo pair of images of an object with simple geometry and then select certain points in the image to help in its reconstruction by displaying a set of lines joining these key points.

2. Experimental Procedure

Taking a camera with a focal length 4.15mm a stereo-pair of images were taken of the same object. The first image was taken and this camera's position (with optical center O) was set as the frame of reference. The camera was then translated along the x direction and it was 228.6mm from the camera's first optical center. Thus, the baseline distance between the camera's optical center was 228.6mm.

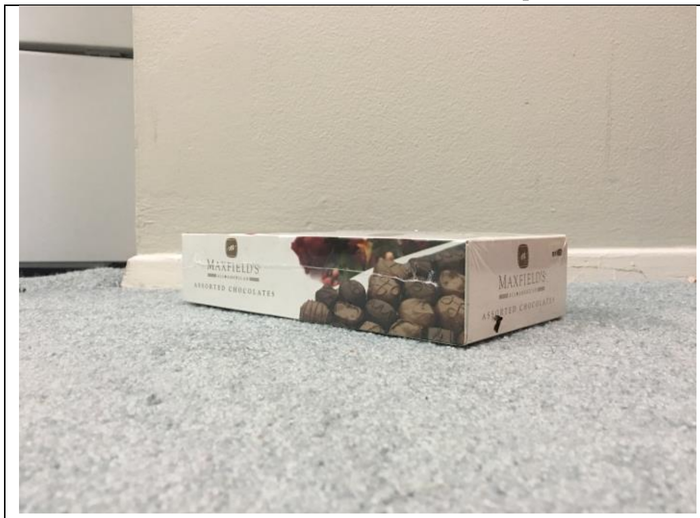


Figure 2.1: Original Non-Translated Image

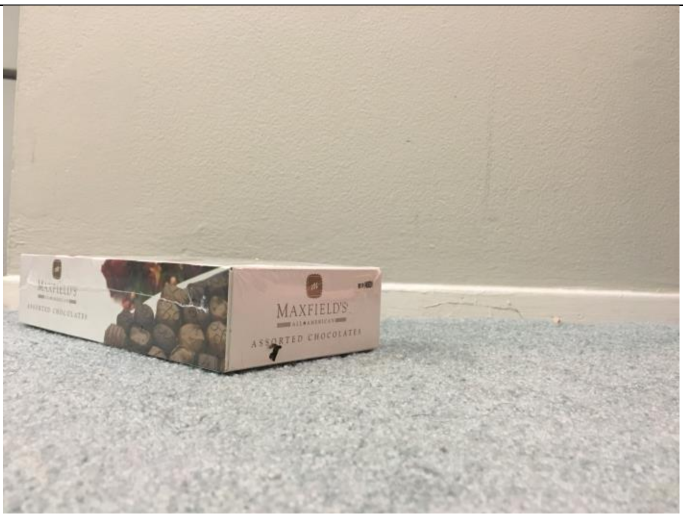


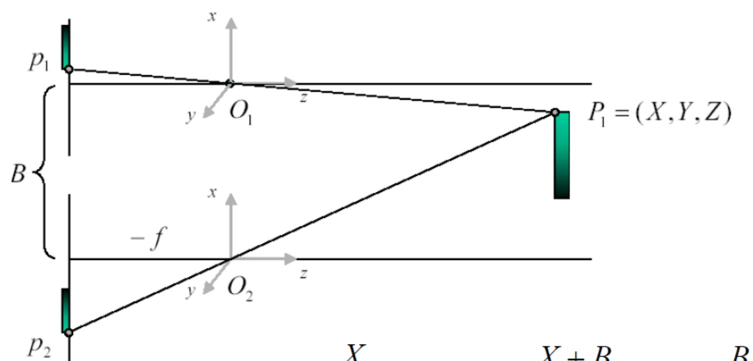
Figure 2.2: Translated Image.

2.1 Data Capture and Procedure for Experiment

The two images were taken with a translation of 228.6mm along the X axes. After the two images were taken, six points were selected in the first image. Using the MATLAB function ginput() the pixel co-ordinates (u,v) of those points were calculated. The same procedure was followed for the second image.
After this, the horizontal and vertical disparity in pixel units and mm-units were calculated using the formula, disparity_in_X=(X_Shifted-X_Original).

The depth is calculated using the following formula:

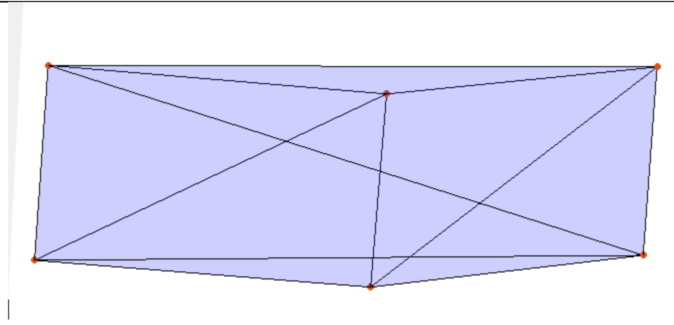
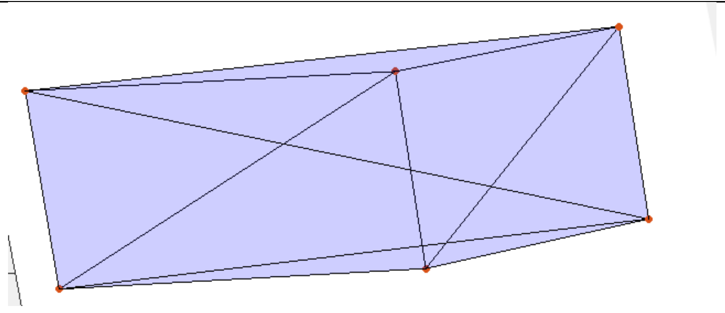
Z=-fB/disparity_in_X



$$x_1 = -f\frac{X}{Z} \qquad x_2 = -f\frac{X+B}{Z} = x_1 - f\frac{B}{Z}$$

$$Z = \frac{fB}{x_1 - x_2}$$

Using MATLAB functions, the points selected on the image were plotted on the 3D plane and then joining the points with straight lines, the 3D structure of the object was reconstructed.

 **Figure 2.2**- 3D Reconstructed Image of the Original Image	 **Figure 2.3**- 3D Reconstructed Image of the Translated Image

The MATLAB code for the assignment is as follows.

```
%Read the first image
Original_Image=imread('C:\NYU_SEM_2\Computer Vision\Assignment 2\Straight.jpg');
imshow(Original_Image);
%select the 6 definite boundary points in the image
[X_Original,Y_Original]=ginput(6)
%Read the second image
Shifted_Image=imread('C:\NYU_SEM_2\Computer Vision\Assignment 2\Shift.jpg');
imshow(Shifted_Image)
%select the 6 definite boundary points in the image
[X_Shifted,Y_Shifted]=ginput(6)
format long g
% disparity horizontal and vertical in pixel units
disparity_in_X=(X_Shifted-X_Original)
disparity_in_Y=(Y_Shifted-Y_Original)
%depth Z= fB/d
Depth_Z=(3.6*228.6)./disparity_in_X;
%calculating the world co-ordinates from the depth and the pixel
%co-ordinates
[World_Xp_Original]=(X_Original.*Depth_Z)./4.15
[World_Yp_Original]=(Y_Original.*Depth_Z)./4.15
```

```
[World_Xp_Shifted]=(X_Shifted.*Depth_Z)./4.15
[World_Yp_Shifted]=(Y_Shifted.*Depth_Z)./4.15

% disparity horizontal and vertical in mm units
disparity_in_X_World=(World_Xp_Shifted-World_Xp_Original)
disparity_in_Y_World=(World_Yp_Shifted-World_Yp_Original)

%plotting the World co-ordinates of the image in the original position
plot3(World_Xp_Original,World_Yp_Original,Depth_Z,'k-')
grid on
%make a matrix containing the X,Y,Z world co-ordinates to help in plotting
%lines using them
Matrix_of_All_Coordinates=[World_Xp_Original,World_Yp_Original,Depth_Z];
plot3(Matrix_of_All_Coordinates(:,1),Matrix_of_All_Coordinates(:,2),Matrix_of_All_Coordinates(:,3),'.')
hold on;
k = boundary(Matrix_of_All_Coordinates,0);
j = boundary(Matrix_of_All_Coordinates,1);
plot3(Matrix_of_All_Coordinates(:,1),Matrix_of_All_Coordinates(:,2),Matrix_of_All_Coordinates(:,3),'.','MarkerSize',10)
trisurf(k,Matrix_of_All_Coordinates(:,1),Matrix_of_All_Coordinates(:,2),Matrix_of_All_Coordinates(:,3),'Facecolor','blue'
```

CONCLUSION:

Thus, through this assignment, we understood the concept of stereoscopy and which allows us to reconstruct depth using two or more images. We illustrated how we acquired a stereo pair of images: a set of images with rotation and translation and a set with horizontal translation. We presented and illustrated the computation of the fundamental Matrix and the epipolar lines that match associated point pairs on both images and we also plotted the location of the epipoles. This experiment allowed us to understand and implement the techniques to create a matching between points in a stereo pair of images. We also performed 3D Reconstruction to determine World co-ordinates from the set of pixel co-ordinates and use them to reconstruct the original 3D Image.