NYU CS-GY 6643, Computer Vision
Assignment 2 (Practical Problems)

Name: Amitesh Kumar Sah
NYU ID: N19714360

Problem 4: 3D from stereo image pairs

4a: Epipolar geometry from F-matrix

Matlab Code

```
clc,close all
% Reading the stereo pair of scene
left=imread('Left.jpg');
right=imread('Right.jpg');

% Load the left points and right points
load left_points.mat
load right_points.mat

% Plotting the point selected
figure(1),imshow(left),title('Points on the left image'),title('Left Image and 12 points'),
hold on, plot(left_points(:,1),left_points(:,2),'r*')

figure(2),imshow(right),title('Points on the right image'),title('Right Image and 12 corresponding points'),
hold on, plot(right_points(:,1),right_points(:,2),'r*')


%% Calculate the F-matrix
% Normalize the points( Hartley preconditioning algorithm)
% The coordinates of corresponding points can have a wide range leading to numerical instabilities.
% It is better to first normalize them so they have average 0 and stddev 1 (mean distance from the center is sqrt(2)
% and denormalize F at the end

l=left_points';
r=right_points';
% Normalising left points
centl=mean(l, 2); %x_bar and y_bar
tl=bsxfun(@minus,l, centl);

% compute the scale to make mean distance from centroid sqrt(2)
meanl=mean(sqrt(sum(tl.^2)));
if meanl>0 % protect against division by 0
   sl=sqrt(2)/meanl;
else
   sl=1;
end
T=diag(ones(1,3)*sl);
T(1:end-1,end)=-sl*centl;
T(end)=1;
if size(l,1)>2
   left_normPoints=T*l;
else
   left_normPoints=tl*sl;
end
% Normalising the right points
centr=mean(r, 2); %x_bar and y_bar
tr=bsxfun(@minus,r,centr);
```

```
% compute the scale to make mean distance from centroid sqrt(2)
meanr=mean(sqrt(sum(tr.^2)));
if meanr>0 % protect against division by 0
   sr=sqrt(2)/meanr;
else
   scaler=1;
end
T_bar=diag(ones(1,3)*sr);
T_bar(1:end-1,end)=-sr*centr;
T_bar(end)=1;
if size(r,1)>2
   right_normPoints=T_bar*r ;
else
   right_normPoints=tr*sr;
end
%% Expressing the linear equation
u=left_normPoints(1,:)';
v=left_normPoints(2,:)';
ud=right_normPoints(1,:)';
vd=right_normPoints(2,:)';

% Defining the image points in one matrix
P=[u.*ud,v.*ud,ud,u.*vd,v.*vd,vd,u,v,ones(size(u,1),1)];
% % Perform SVD of P
[U,S,V]=svd(P);
[min_val,min_index]=min(diag(S(1:9,1:9)));
% % m is given by right singular vector of min. singular value
m=V(1:9,min_index);
% Projection matrix reshaping
F=[m(1:3,1)';m(4:6,1)';m(7:9,1)'];
% F1=F;
% To enforce rank 2 constraint:
% Find the SVD of F: F = Uf.Df.VfT
% Set smallest s.v. of F to 0 to create D?f
% Recompute F: F = Uf.D?f.VfT
[Uf,Sf,Vf]=svd(F);
Sf(end)=0;
F=Uf*Sf*Vf';
% Denormalise F and transform back to original scale
F=T_bar'*F*T;
% % Normalize the fundamental matrix.
F=F/norm(F);
if F(end)<0
  F=-F;
end
%%

% Choose a point in the left image and calculate the epipolar line in the right image
% figure,imshow(left);
% cpselect(left,right)

 load el_left;
 load el_right;
 er_right=el_right;
epi_lineR=zeros(size(el_left,1),3);
el_left(:,3)=ones(size(el_left,1),1);
for i=1:size(el_left,1)
epi_lineR(i,:)=(F*el_left(i,:)')';
epi_lineR(i,:)=epi_lineR(i,:)./epi_lineR(i,3);
end
```

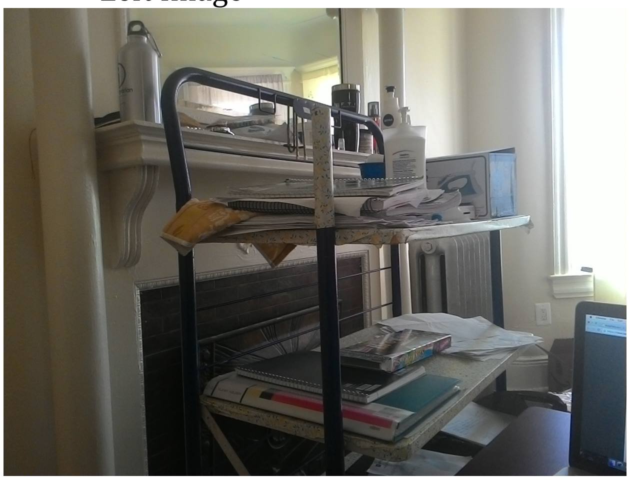
```
% Plotting the epipole in right image
al=epi_lineR(:,1);
bl=epi_lineR(:,2);
cl=epi_lineR(:,3);
xl=1:1:size(right,2);
yl=zeros(size(el_left,1),size(right,2));
figure(3),imshow(left),title('Points on the left image')
hold on, plot(el_left(:,1),el_left(:,2),'r*');hold off
figure(4),imshow(right);hold on, title('Corresponding Epipolar lines on right image')
plot(el_right(:,1),el_right(:,2),'r*');hold on
for i=1:size(al)
    yl(i,:)=-(al(i)/bl(i)).*xl-(cl(i)/bl(i));
    figure(4),
     plot(xl(1,:),yl(i,:))
   hold on
end

%%
% Choose a point in the right image and calculate the epipolar line in the left image
epi_lineL=zeros(size(er_right,1),3);
er_right(:,3)=ones(size(er_right,1),1);
for i=1:size(er_right,1)
epi_lineL(i,:)=er_right(i,:)*F;
epi_lineL(i,:)=epi_lineL(i,:)./epi_lineL(i,3);
end

% Plotting the epipole in right image
ar=epi_lineL(:,1);
br=epi_lineL(:,2);
cr=epi_lineL(:,3);
xr=1:1:size(left,2);
yr=zeros(size(er_right,1),size(left,2));
figure(5),imshow(right),hold on,title('Points on the right image')
plot(er_right(:,1),er_right(:,2),'r*');
figure(6),imshow(left);hold on, title('Corresponding Epipolar lines on left image')
plot(el_left(:,1),el_left(:,2),'r*');
for i=1:size(ar)
    yr(i,:)=-(ar(i)/br(i)).*xr-(cr(i)/br(i));
    figure(6)
 hold on
     plot(xr(1,:),yr(i,:))
end
% % % Calculate the position of the epipole of the left camera
[~, ~, v]=svd(F);
ep=v(:, 3)';
epipole= ep(1:3)/ep(3)
%%
% Plotting the epipole along with the epipolar line in left image

figure,
xr1=1:1:(epipole(1,1)+1000);
yr1=zeros(size(er_right,1),(floor(epipole(1,1)+1000)));

figure(7),imagesc(left);hold on,title('Epipolar lines and epipoles on left image')
 plot(epipole(1,1),epipole(1,2),'r*');
 hold on
% plot(er_left(:,1),er_left(:,2),'r*');
for i=1:size(ar)
    yr1(i,:)=-(ar(i)/br(i)).*xr1-(cr(i)/br(i));
    figure(7)
```

```
 hold on
    plot(xr1(1,:),yr1(i,:))
end
xlim([1,10000])
ylim([1,4000])
```

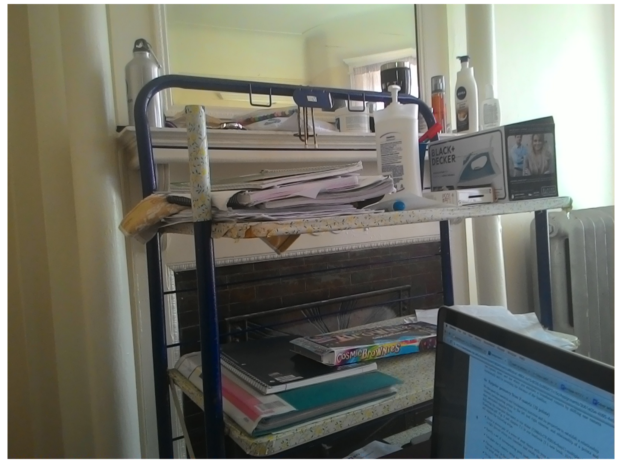
A) Shoot a stereo-pair of a scene of your choice with your camera, where you take a left and a right picture where you translate and rotate the camera.

INPUT image

Left Image



Right Image



B) Specify a set of corresponding pixel landmark pairs in the left and right camera views.

12 Points Selected in Left and Right Image to calculate F matrix

Left Image and 12 points

Right Image and 12 corresponding points

C) Calculate the F-matrix using instructions as discussed in the slides.?

The F-matrix calculated is
F =

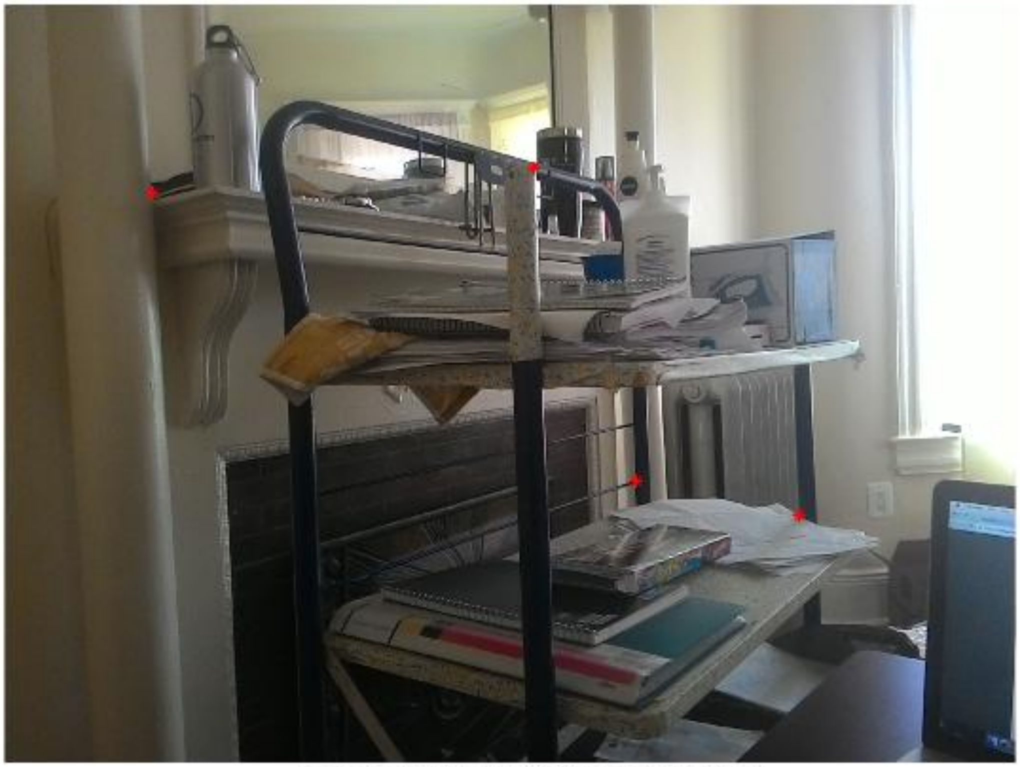
$$\begin{matrix} 0.0000 & 0.0000 & -0.0012 \\ 0.0000 & 0.0000 & -0.0080 \\ -0.0003 & 0.0061 & 0.9999 \end{matrix}$$

D) Choose a point in the left image and calculate the epipolar line in the right image. Display the point and the line as overlays in your images.

The point selected in left image is
[X Y]
1256.5 , 808.25
998.5 , 755.75
235.0 , 299.75
838.0 , 259.25

The corresponding calculated epipolar line in right image is
[a b c]
7.64900263510643e-06 -0.00119696356544386 1
-1.01272815834046e-05-0.00129680346901401 1
-0.000273888032278490 -0.00277820687094460 1
-0.000322836453387364 -0.00305312403492266 1

Points on the left image



Corresponding Epipolar lines on right image

E) • Choose point in the right image and calculate the epipolar line in the left image. Display the point and the line as overlays in your images.

The four points in right image is

[X	Y]
1403.5 ,	841.5
1147.5 ,	759.5
289.5 ,	331.5
517.5 ,	271.5

The corresponding calculated epipolar line in left image is

[a b c]
-7.90479608932467e-05-0.00111762801551848 1
-7.59573084031497e-05-0.00122664301685011 1
-1.65554073839198e-05-0.00332189582817765 1
-8.78994599545833e-07-0.00387484190219904 1

Points on the right image

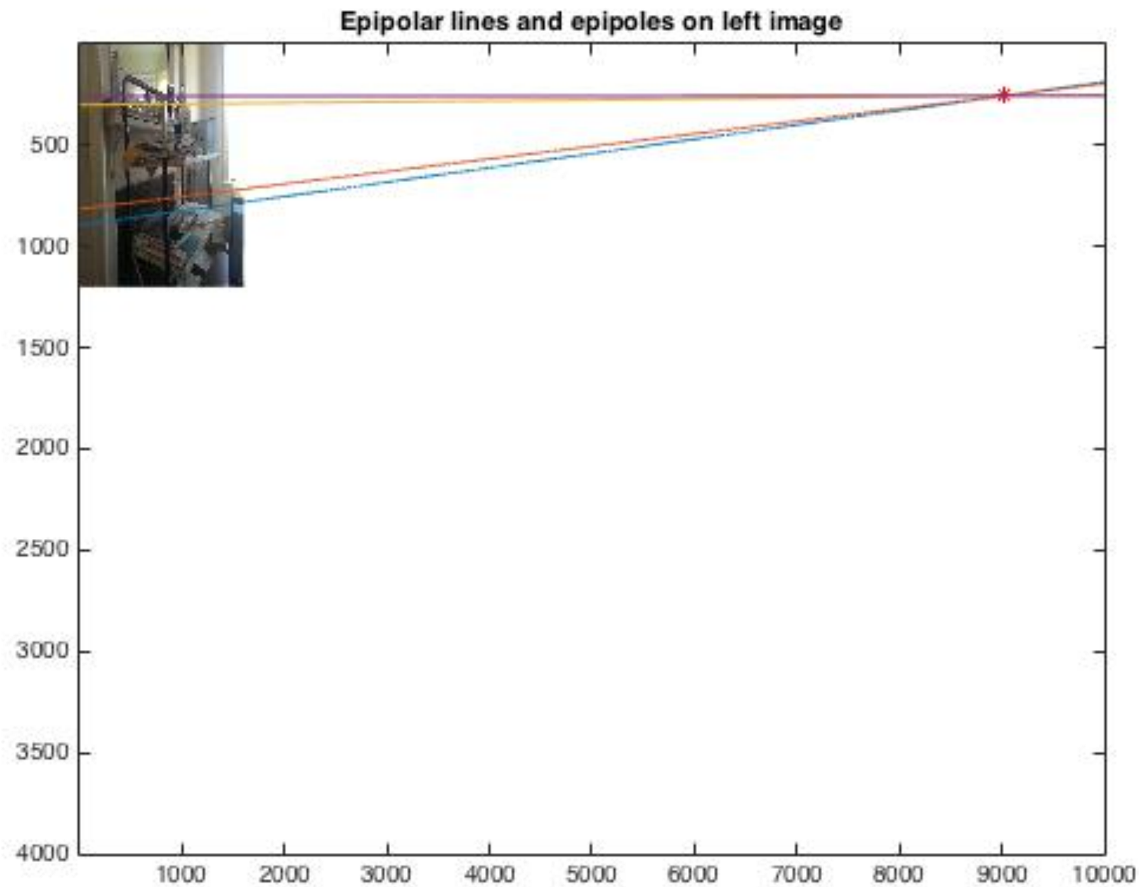
Corresponding Epipolar lines on left image

F) • Calculate the position of the epipole of the left camera (see last two slides for instructions). Discuss if the calculated position seems reasonable.

epipole =

   1.0e+03 *

    9.0307   0.2560   0.0010



Through formula and theoretical calculation, the position of epipole of the left camera is at (9030,256). We can also see that all the four epiplolar lines meet at a point and that point is same as the point obtained through formula e.F=0. Hence, I am quite satisfied with the result obtained.

4b: 3D Object geometry via triangulation

Matlab Code

```
clc;clear all;close all;

% Read stereo pair of images
left_image=imread('left.jpg');
right_image=imread('right.jpg');
figure,subplot(1,2,1),imshow(left_image);title('Left image');
 subplot(1,2,2),imshow(right_image);title('Right image');

% cpselect(left_image,right_image)
load left_points;
load right_points;
```

```
% Intrinsic Parameters
 alpha=2424.3;
 beta=2459.6;
 teta=89.771;
 x0=1775.7;
 y0=1065.1;

% Landmark coordinate with respect to image center
 for i=1:size(left_points,1)
 left_center(i,1)=left_points(i,1)-x0;
 left_center(i,2)=left_points(i,2)-y0;
 right_center(i,1)=right_points(i,1)-x0;
 right_center(i,2)=right_points(i,2)-y0;
 end

%  Calculation of horizontal and vertical disparity in pixel width
 dis_x=left_center(:,1)-right_center(:,1);
 dis_y=left_center(:,2)-right_center(:,2);
 figure,
 plot(dis_x,dis_y,'r*'),title('Disparity of corners');
 xlabel('Horizontal disparity in pixel unit');
 ylabel('Vertical Disparity in pixel unit');

%  Horizontal Disparity in mm
f=3.1;      % in mm
pdx=alpha/f;
dis_xmm=dis_x/pdx;

% vertical disparity in mm
pdy=beta/f;
dis_ymm=dis_y/beta;

% x and y coordinates of left image landmark points in mm
xmm=left_points(:,1)./pdx;
ymm=left_points(:,2)./pdy;


% Using triangulation
B=705-148;
Z=(B*f)./dis_xmm;

% Using Perspective projection
X=(Z.*xmm)/f;
Y=(Z.*ymm)/f;

% Making a semi-cuboid  by arranging the coordinates
 X1=[X(1),X(2),X(4),X(3),X(6),X(7),X(5),X(2),X(1),X(3),X(4),X(7)];
 Y1=[Y(1),Y(2),Y(4),Y(3),Y(6),Y(7),Y(5),Y(2),Y(1),Y(3),Y(4),Y(7)];
 Z1=[Z(1),Z(2),Z(4),Z(3),Z(6),Z(7),Z(5),Z(2),Z(1),Z(3),Z(4),Z(7)];

 %Plotting the cuboid and the corner point
figure,plot3(X1,Y1,Z1), title('3D Reconstruction Image of cuboid'),
xlabel('x-axis in mm');
ylabel('y-axis in mm');
zlabel('z-axis in mm');
hold on
scatter3(X,Y,Z);
```

A) • Define a small set of corresponding key locations by either manual definition of landmarks or a correlation-based image processing method (with selection of only the major key points).

Here, I have taken 7 sets of corresponding key location manually by using a function cpselect().
The corner points of a cuboid in left image is:
left_points =

1.0e+03 *

1.9749	1.5252
1.6276	1.5621
2.3579	1.5781
1.9814	1.6291
1.6224	1.9856
2.3631	2.0164
1.9904	2.1406

The corresponding corner points of a cuboid in the right image is:
Right points =

1.0e+03 *

1.0679	1.5316
0.5991	1.5754
1.2879	1.5719
0.7666	1.6289
0.5929	1.9914
1.2924	2.0006
0.7681	2.1296

B) Calculate horizontal and vertical disparity in pixel units and mm-units, and from those the 3D point coordinates (X,Y,Z).

Horizontal Disparity in pixel unit
dis_x =
1.0e+03 *
0.9070
1.0285
1.0700
1.2148
1.0295
1.0708
1.2223

Horizontal Disparity in mm
dis_xmm =
1.1598
1.3152
1.3682
1.5533
1.3164
1.3692
1.5629

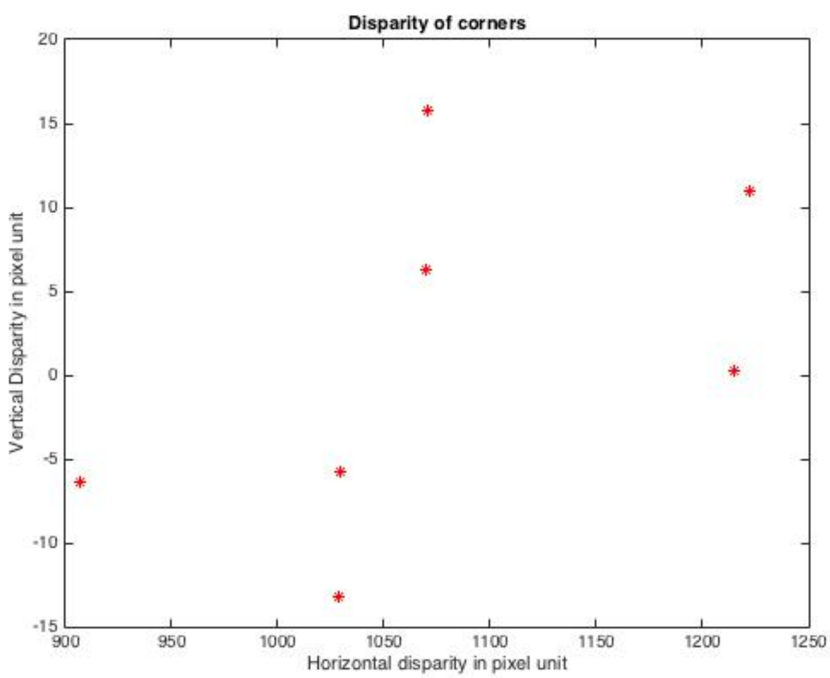
Vertical disparity in pixel unit
dis_y =
-6.3750
-13.2500
6.2500
0.2500
-5.7500
15.7500
11.0000

Vertical disparity in mm unit
ymm =
1.9224
1.9689
1.9890
2.0533
2.5026
2.5414
2.6980

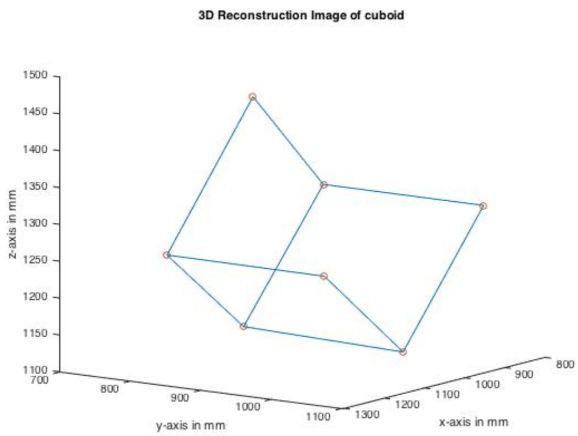
3D Point XYZ in cm

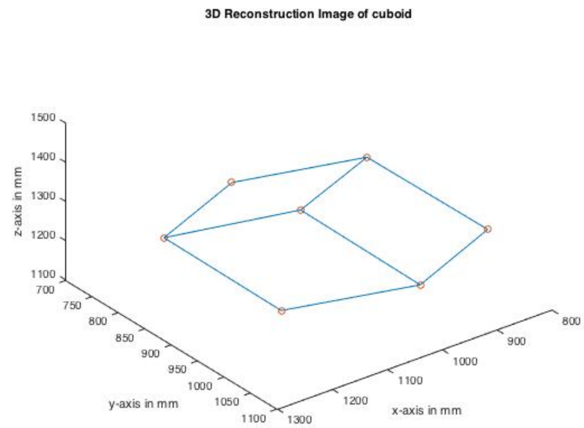
121.2795	92.3232	148.8793
88.1465	83.3851	131.2917
122.7417	80.9720	126.1995
90.8521	73.6283	111.1616
87.7769	105.8883	131.1642
122.9288	103.3857	126.1111
90.7048	96.1518	110.4795

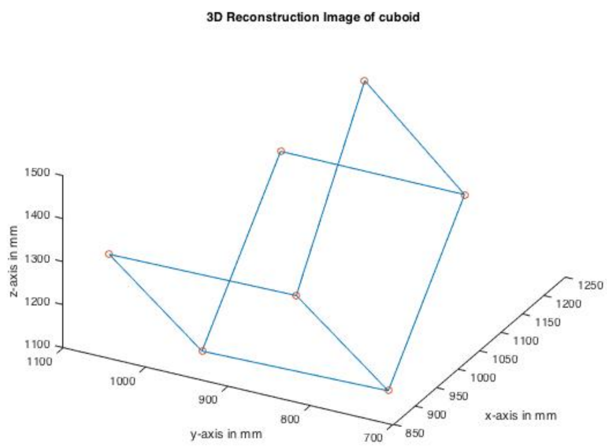
Sl. No	Left image points (x,y) in pixels * 1.0e+03	Right image points (x,y) in pixels * 1.0e+03	Horizontal Disparity		Vertical Disparity		3D point coordinate (X,Y,Z) in cm
			In pixel Unit *1.0e+03	In mm unit	In pixel unit	In mm unit	
1.	1.9749 1.5252	1.0679 1.5316	0.9070	1.1598	-6.3750	1.9224	(121.2795 , 92.3232 , 148.8793)
2.	1.6276 1.5621	0.5991 1.5754	1.0285	1.3152	-13.250	1.9689	(88.1465 , 83.3851 , 131.2917)
3.	2.3579 1.5781	1.2879 1.5719	1.0700	1.3682	6.2500	1.9890	(122.7417 , 80.9720 , 126.1995)
4.	1.9814 1.6291	0.7666 1.6289	1.2148	1.5533	0.2500	2.0533	(90.8521 , 73.6283 , 111.1616)
5.	1.6224 1.9856	0.5929 1.9914	1.0295	1.3164	-5.7500	2.5026	(87.7769 , 105.8883, 131.1642)
6.	2.3631 2.0164	1.2924 2.0006	1.0708	1.3692	15.7500	2.5414	(122.9288 , 103.3857 , 126.1111)
7.	1.9904 2.1406	0.7681 2.1296	1.2223	1.5629	11.0000	2.6980	(90.7048 , 96.1518 , 110.4795)

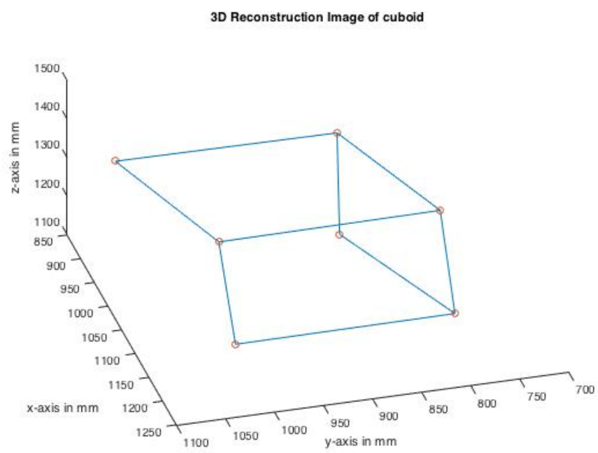


C) • Use Matlab or your software to display the 3D points and edge lines for the reconstructed object (only those visible in your images). Choose display viewpoints different from the camera views to verify the quality of 3D reconstruction









Here , I have shown the reconstructed object in different display view of 7 points . I have also plotted the seven points in the same 3D image .

DISCUSSION

I took the picture of the box of dimension 262mm(length)* 262mm(breadth)*218mm(height). From the reconstruction , I can see that the length of the box is 305.6174mm(length), 295.065mm (breadth),225.071mm(height). There is a small difference of 4 mm in length which is tolerable due to the error in selecting the exact coordiantes or due to the light deviation from horizontal while moving the camera. It can be also due to the intrinsic camera calibration error.

Disparity between the two image provides the depth map of the image using the triangulation formula. Further using projection matrix, one can come to know the approximate dimension of the object in the world and the location of the object in the world We have to first calibrate our camera and get the intrinsic parameters . We also need the information of camera i.e. focal length to calculate the distance in mm.