VISUAL PERCEPTION Lab Report 3
Reconstruction from Two Views

Raabid Hussain

April 24, 2016

1 Abstract

The objective of this lab work was to reconstruct 3D points from 2 view-points using MATLAB. Known the position and orientation of two cameras, 3D points in world frame were projected onto the camera image planes. These points were used to get the fundamental matrix matrix of the stereo images using two methods, namely 8 point linear mean square and 9 point single value decomposition. Epipolar geometry was constructed to observe the relation between the stereo images. The lab work concluded with a comparison between the two techniques.

2 Introduction - Triangulation & Epipolar Geometry

Triangulation is the terms referred to the process of determining the 3D coordinates of a point in world coordinate system, given its projection on two images. In order to accomplish this, the camera projection parameters that transform from 3D space to 2D image plane need to be determined first. These parameters can be represented by the fundamental matrix. A line between the two corresponding points is called an epipolar line. All of the corresponding lines cross through one point called the epipole. Unfortunately in real case scenarios, the coordinates in image plane can not be measured with perfect accuracy due to noise. As a consequence the epipolar lines do not cross through exactly 1 point. In this lab work, different techniques were used to determine the fundamental matrix and the epipoles.

3 Methodology

3.1 Step 1 & 2

I step 1, we were provided with the intrinsic parameters of camera 1 and we just had to copy them in MATLAB and set the world coordinate system to the coordinate system of camera 1. This meant that the transformation matrix from camera 1 to world system was kept to be identity.

In step 2, we provided with camera parameters of camera 2 and just like in step 1, we just ad to copy them into MATLAB.

3.2 Step 3

In step 3, we had to find the intrinsic and transformation matrices of both the cameras. The standard intrinsic, rotation and translation matrix formulations were used. For camera 2 the rotation was eular XYZ so when creating the rotation matrix, the rotation matrices for each angle were multiplied in XYZ order. Figure 1 shows the results. For camera 1, as described in step 1, the rotation matrix was identity and translation matrix was zero, so it is not displayed.

```
A1 =

   100     0   128
     0   120   128
     0     0     1

A2 =

    90     0   128
     0   110   128
     0     0     1

R =

    0.6930   -0.1405    0.7071
    0.2669    0.9611   -0.0706
   -0.6697    0.2376    0.7036

T =

       -1000
         190
         230
```

Figure 1: The intrinsic matrix of both cameras folloed by rotation and translation matrices of camera 2.

3.3 Step 4

In step 4, we had to compute the fundamental matrix of the camera geometry. For computing the matrix the formula shown in figure 2 was used. For translation matrix, since matrix multiplication is being used in this step, the dimensions needed to match so we used the anti-symmetric form of the translation vector. The output using the equation was not normalized so at the end the last element of the fundamental matrix was divided by each element. The result is shown in the same figure.

$$F = \mathbf{A}'^{-t} R^{t} [t]_{x} \mathbf{A}^{-1}$$

```
F =

    0.0000    0.0001   -0.0066
    0.0000   -0.0000    0.0130
   -0.0096   -0.0173    1.0000
```

Figure 2: The fundamental matrix for the current camera positions.

3.4 Step 5

In step 5, we were provided with a set of 20 points in world system and similar to step 1 and 2, we just had to copy them in MATLAB. For the next steps, these points were used to compute the epipolar geometry. Also, in the last step, we had to use more than 20 points so a simple for loop was written to obtain the chosen number of points randomly. The range of these points was chosen as such so they are in sync with the 20 points provided. The loop is shown in figure 3.

```
for i=1:num
    V(:,i)=[round(rand*1000);round(rand*800)-400;round(randn*2000)+2000;1];
end
```

Figure 3: The program written to generate multiple random 3D points.

3.5 Step 6 & 7

In step 6, we had to find the corresponding points on the image planes. In this stage, the intrinsic matrix is now of 3x4 dimension, so we had to recompute that. Also the rotation and translation calculated above are for the reverse case so new transformation matrices were also computed as can be shown in 4. After computing the intrinsic and extrinsic matrices, they were simply multiplied with the 3D points to get the projections.

```
I1=[au1 0 uo1 0;0 av1 vo1 0;0 0 1 0];
I2=[au2 0 uo2 0;0 av2 vo2 0;0 0 1 0];
E2=[R' -R'*T];
E2=[E2;0 0 0 1];
E1=[R1 T1];
E1=[E1;0 0 0 1];

c1=I1*E1*V;
c2=I2*E2*V;
for i=1:num
    c1(1,i)=c1(1,i)/c1(3,i);
    c1(2,i)=c1(2,i)/c1(3,i);
    c1(3,i)=c1(3,i)/c1(3,i);
    c2(1,i)=c2(1,i)/c2(3,i);
    c2(2,i)=c2(2,i)/c2(3,i);
    c2(3,i)=c2(3,i)/c2(3,i);
end
```

Figure 4: The program written to find projections of 3D points onto the two image planes.

In step 7 we had to display the two image planes and the points mapped. The result is as shown in 5

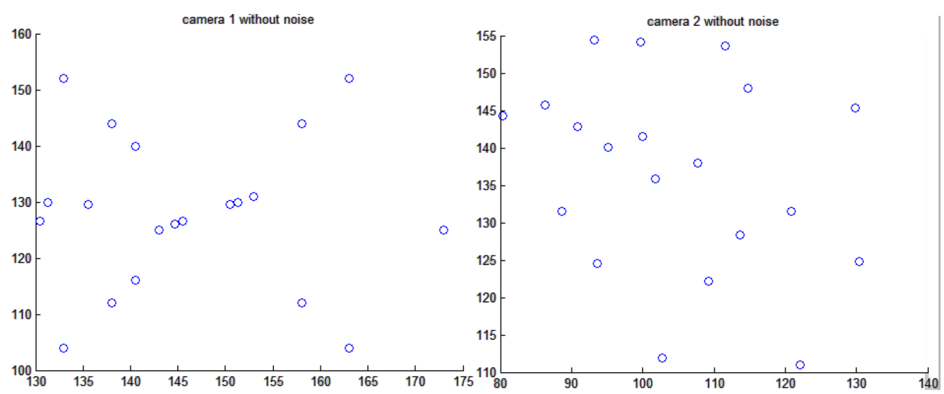

Figure 5: The camera image planes along with their respective projected points.

3.6 Step 8 & 9

In step 8, we had to compute the fundamental matrix using the 8 point linear mean square (LMS) method. Since the fundamental matrix has 9 element so the last element was fixed to 1. For this step, equations from the lecture notes were consulted. The only change was that the equations in the notes were from "mFm'=0" but in this exercise the equations were the opposite (m'Fm=0). This was catered by simply replacing m with m' in the formulation and vice versa. The code for this is shown in figure 6.

```
for i=1:num
    U(i,:)=[c1(1,i)*c2(1,i) c1(2,i)*c2(1,i) c2(1,i) c1(1,i)*c2(2,i) c1(2,i)*c2(2,i) c2(2,i) c1(1,i) c1(2,i)];
end
A=-(U'*U);
b=U'*ones(num,1);
f=A\b;
f=[f; 1];
f8LMS=reshape(f,3,3)';
```

Figure 6: The program written for the LMS method.

In step 9 we had to compare the result of this method with the original fundamental matrix which turned out to be exactly same. (For the sake of page limit, it is not repeated here)

3.7 Step 10

In step 10, we had to visualize the entire epipolar geometry for each camera separately. The gradient and offset of each individual epipolar line was found through the equation [x,y,1] [u1;u2;u3]=0. Using these two extracted parameters and taking 2 arbitrary x-axis points (-500, 500), the epipolar line was drawn for each points. These lines were seen to intersect at one point called the epipolar point. Since all the lines crossed this epiploar point, so in order to obtain th ecoordinates of this point, intersection of any two was found using 'polyxpoly' function of MATLAB. The code for one camera is as shown in figure 7 and the result visualized in figure 8.

```
%epipolar lines
C1=f8LMS*c1;
for i=1:num
    m2(i)=-C1(1,i)/C1(2,i);
    d2(i)=-C1(3,i)/C1(2,i);
end
%epipolar point
[ep1x ep1y]=polyxpoly([-500 500],[y11(1) y12(1)],[-500 500],[y11(num) y12(num)])
```

Figure 7: The program written for determining the epipolar lines and point.

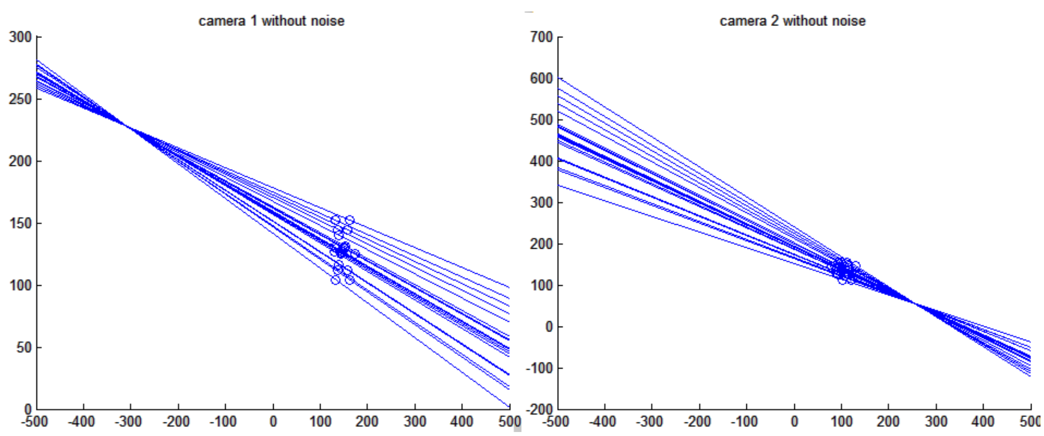

Figure 8: Visualization of the epipolar geometry.

3.8 Step 11, 12 & 13

In step 11 and 12, we worked on some real case scenarios by adding gaussian noise to the 2D points. After adding noise to the points, the above LMS technique was repeated to find the noisy fundamental matrix. When we displayed the noisy matrix in MATLAB window it turned out too be same as when displaying MATLAB only shows 4 decimal places. However, when we used this matrix to draw the epipolar geometry, significant differences were seen. The epipolar geometry was redrawn. It was found that the lines do not pass through 1 single point. So, in order to approximate the epipolar point, so modification needed to done. For this the rank of the fundamental matrix reduced to 2 by first applying SVD and then forcing the smallest eigenvalue to 0. Although this introduced more noise into the system but the resulting lines passed through one point which was found using the same 'polyxpoly' function for any two lines. The result is shown in figure 10. In step 13, noise was increased. The result indicated that the new epipolar point is further away from the area where originally epipolar lines should meet.

```
%using SVD to force rank of F to 2
[Us,S,V] = svd(f8LMS);
[~,i]=min(diag(S));
S(i,i)=0;
f8LMS=Us*S*V'
```

Figure 9: The program written for determining the noisy epipolar lines and point.

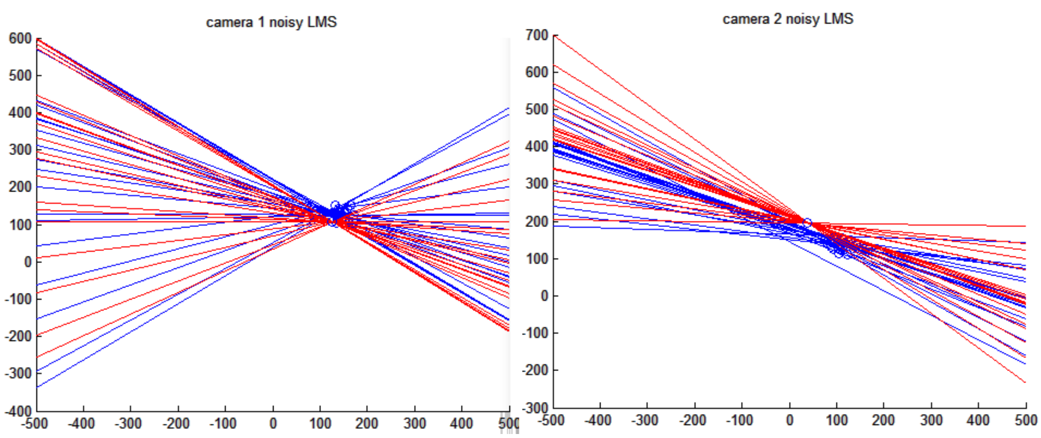

Figure 10: Visualization of the noisy epipolar geometry. Blue line represent the result obtained and the red line represent the approximation of these lines.

3.9 Step 14 & 15

In step 14 and 15, we had to compute the epipolar geometry using the 8 point SVD (Eig) analysis. Same noiseless and noisy points were used in order to do a fair comparison later. Equations from the lecture notes were used. Entire formulation was same as for LMS method. The only difference how to compute the fundamental matrix (figure 11) The result had to be normalized so that the last element is 1. The fundamental matrix was also similar with a very slight change in epipolar geometry.

```
%eigen decompositon
[U,D,V]=svd(U2);
f9Eig=reshape(V(:,9),3,3)';
f9Eig=f9Eig/f9Eig(3,3);
```

Figure 11: The program written for determining the noisy epipolar lines and point.

3.10 Step 16

At the end, it was time to compare the two techniques implemented during this lab. The distance from the noisy points to their corresponding epipolar line (before forcing rank to 2) was found. So different number of points and noise were used to draw some conclusions. As expected as the noise increased, the error increased however when the number of points increased the error decreased. Eig method performed slightly better at times however, the results were pretty much similar for both the cases. The results are shown in tabulated form. In one case the result was a little unexpected but that was probably due to the nature of the points generated.

Points	.........10.........		.........20.........		.........50.........		.........100.........	
Noise	LMS	Eig	LMS	Eig	LMS	Eig	LMS	Eig
0.05 (mean)	0.3515	0.3572	0.0509	0.0509	0.0578	0.0578	0.0649	0.0649
0.1 (mean)	0.3093	0.3093	0.1512	0.1512	0.1407	0.1407	0.1505	0.1505
0.5 (mean)	0.5546	0.5546	1.7882	1.7881	0.8850	0.8848	0.8282	0.8281
0.05 (sd)	0.7102	0.7357	0.0376	0.0376	0.0980	0.0980	0.0682	0.0682
0.1 (sd)	0.3038	0.3037	0.1216	0.1215	0.1188	0.1188	0.1483	0.1483
0.5 (sd)	0.4655	0.4655	1.5217	1.5217	0.9661	0.9661	1.1549	1.1546

4 Step 17

We were also offered to do an optional part. In this step, we had to draw the epipolar geometry of both the camera in one window. In order to accomplish this, two sample 3D points were taken at random and their projections on the image plane found using steps 1-7. Here the catch was that since we were drawing in 3D space and not in the image plane directly, we modifications were required for the obtained points. Using the focal length, pixel adjustment parameters and image buffer parameters, their corresponding points in 3D space were found. Then the pi-plane was drawn from the origins of the two cameras and the 3D points. Each point had a different pi-plane and their respective projections on the image planes, were found to lie on these lines. Since the triangle formed by joining these three points is actually on the pi plane so an additional plane was not drawn and the pi planes were represented by this triangulation. Next epipolar geometry was found using step 10 and they were drawn after the modifications mentioned above. A sample result for two points is shown in figure 12.

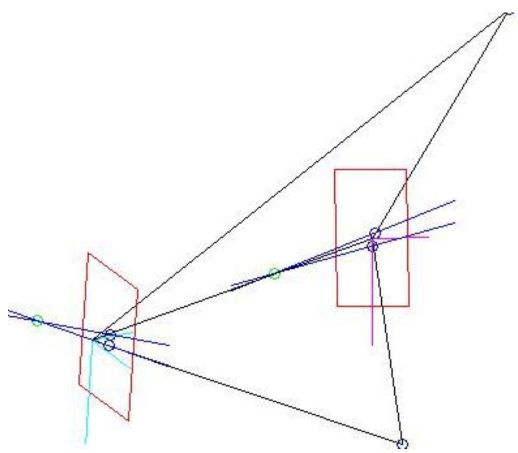

Figure 12: Epipolar geometry for 2 sample 3D points. The black lines indicate the triangulation and pi-planes, The red lines represent the image planes, the green dots represent the epipoles, the blue dots represent the camera and 3D points, the blue lines represent the epipolar lines and the magenta and cyan lines represent the coordinate systems of the two image planes.

5 Conclusion

In this lab work, epipolar geometry was computed using two famous 8 point techniques namely LMS and SVD. Real case scenarios were also tested and the results of the two techniques compared. The exercises were completed during the lab times whereas the report was written during the coming week at home.