Visual Perception Lab 2

<< Calibrated SfM>>

Master in Computer Vision

UNIVERSITE DE BOURGOGNE

Centre Universitaire Condorcet - UB, Le Creusot

20 May 2017

Submitted to: Prof. David Fofi

Submitted by: Mohit Kumar Ahuja

Functions Made/Used:

1. **VP_Project** - Complete Code file.
2. **calculate_2D_point** - Will calculate 2D Points form projection Matrix.
3. **plot3D** - Used for plotting.

Task:

1. **Start from the very same simulation done in #1. Consider that your system is calibrated. Compute the Essential Matrix E, knowing R and T (ground-truth of your simulation).**

Solution: I used an internal Point Cloud of matlab ('teapot.ply') for 3D scene. And as shown in figure there are two camera's shown, Red one is the first camera and the orange one is the second camera. In the starting of the code, We assigned {x,y,z} values to both the camera's. As shown below,

```
Camera_1_Coordinate = [0 0 0];
Camera_2_Coordinate = [2 0 0];
```

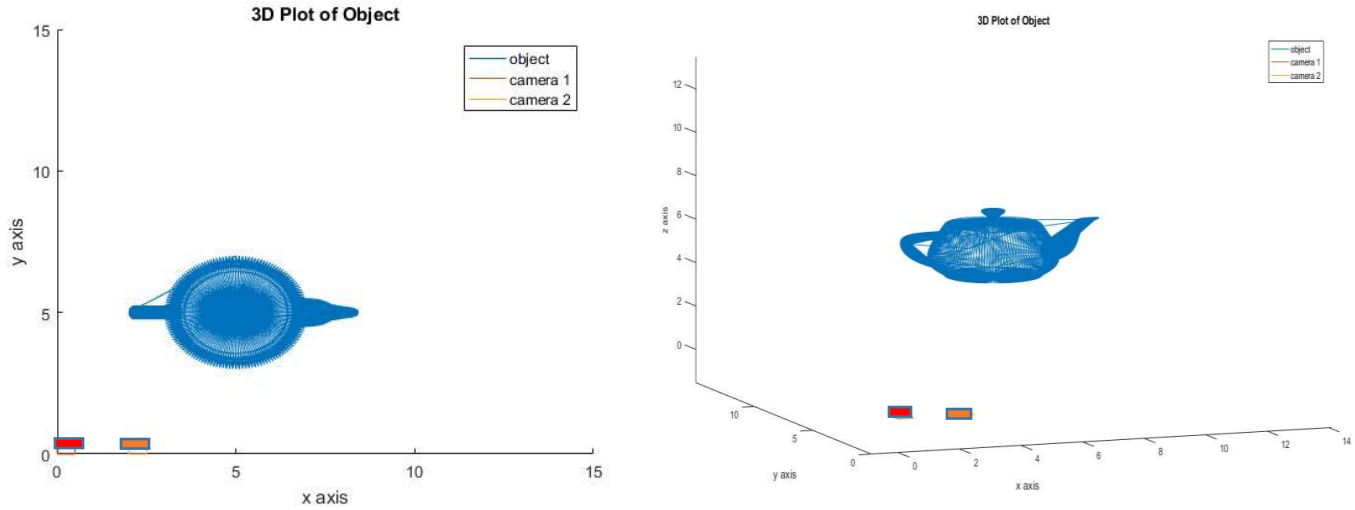

Figure 1: 3D Scene and stereovision system

```
RT_Matrix = [Rotation_Matrix Translation_Matrix];
Proj_Geo_1 = Intrinsic_Parameters*e1*RT_Matrix;
RT_Matrix2 = [Rotation_Matrix2 Translation_Matrix2];
Proj_Geo_2 = Intrinsic_Parameters2*e2*RT_Matrix2;
```

P1 = Proj_Geo_1*a;
P2 = Proj_Geo_2*a;

In code, RT Matrices represents the external parameter matrices for both camera 1 and camera 2. The Intrinsic_Parameters matrices contains the intrinsic parameters of both camera's which is then further used to calculate the projective matrices of both camera's. Then, P1 and P2 are calculated in "calculate_2D_point" function. In which we calculate the 2D {x,y} coordinates of 3D object.

As the Internal and External parameters are known to us, we can calculate the Essential matrix using the formula;

$$E = R[t]x$$

The [t]x is calculated by cross product which results in;

$$[\mathbf{a}]_\times \stackrel{\text{def}}{=} \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}.$$

*Replace **a** with **t**.

By this, we will get the Essential Matrix(Ground Truth) as,

$$\textbf{Essential\_Matrix} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{2} \\ \mathbf{-2} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

2. **Based on "An Efficient Solution to the Five-Point Relative Pose Problem" Paper, extract R and T from E.**

Solution: As the Essential Matrix is known to us, we can calculate the Translation and Rotation parameters by using that essential matrix. So, by decomposing the Essential Matrix, we can get some values of U, V and S.

[U,S,V] = svd(Essential_Matrix)

$$D = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

So, The translational Matrix is given by,

$$t \sim tu \equiv [\, U13\ U23\ U33]^T$$

And the Rotational Matrix is given by,

$$R_a \equiv UDV^T\ ,\ R_b = UD^TV^T$$

So, by using this formula we can calculate both Translation and Rotation Matrix from Essential Matrix. The Result is shown below;

$$t_u = \begin{matrix} 1 \\ 0 \\ 0 \end{matrix}$$

$$R_a = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

$$R_b = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3. **Use the so-called cheirality constraint to choose between the four putative solutions.**

Solution: Any combination of R and t according to the above prescription satisfies the epipolar constraint. To resolve the inherent ambiguities, it is assumed that the first camera matrix is [I | 0] and that is of unit length. There are then the following four possible solutions for the second camera matrix:

PA = [Ra | tu], PB = [Ra | −tu], PC = [Rb | tu], PD = [Rb | −tu]

One of the four choices corresponds to the true configuration. Another one corresponds to the twisted pair which is obtained by rotating one of the views 180 degrees around the baseline. The remaining two correspond to reflections of the true configuration and the twisted pair. For example, PA gives one configuration. PC corresponds to its twisted pair, which is obtained by applying the transformation. PB and PD correspond to the reflections.

In order to determine which choice corresponds to the true configuration, the cheirality constraint 1 is imposed. One point is sufficient to resolve the ambiguity. The point is triangulated using the view pair ([I | 0], PA) to yield the space point Q and cheirality is tested.

a) If c1 = Q3Q4 < 0, the code will display in the command window that "the point is behind the first camera" and will plot the 3D view of the Points.

b) If c2 = (PAQ)3Q4 < 0, the code will display in the command window that "the point is behind the second camera" and will plot the 3D view of the Points.

c) If c1 > 0 and c2 > 0, the code will display in the command window that "P* and Q correspond to the true configuration" and will plot the 3D view of the Points.

So, we used this technique to solve our problem of selecting true configuration. In my case, PC had a values of C1 and C2 greater than Zero.

The resulting Images are shown below;

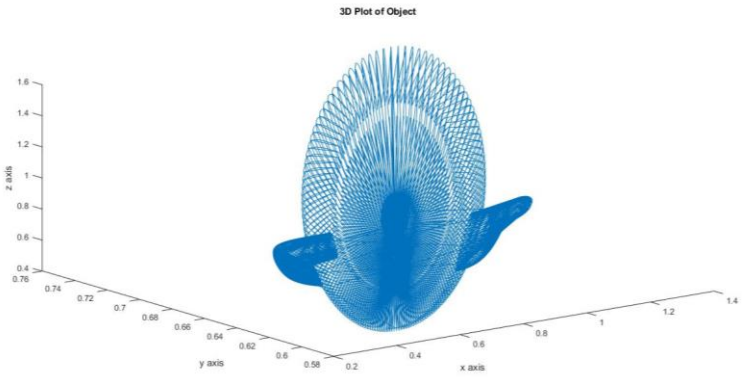

Figure 2: Showing PA as second Camera Matrix

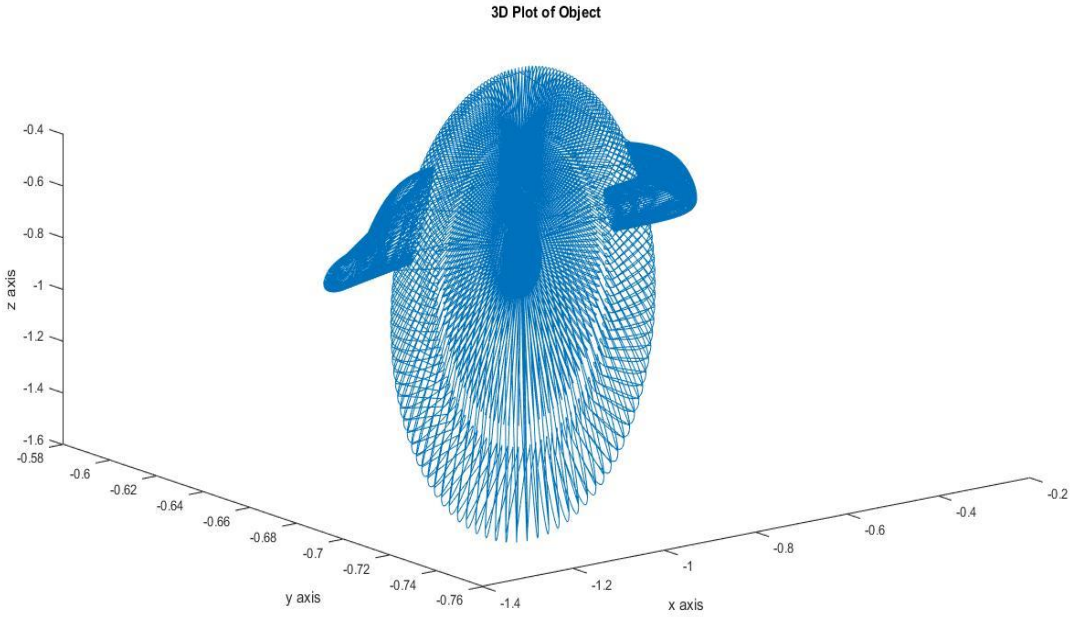

Figure 3: Showing PB as second Camera Matrix

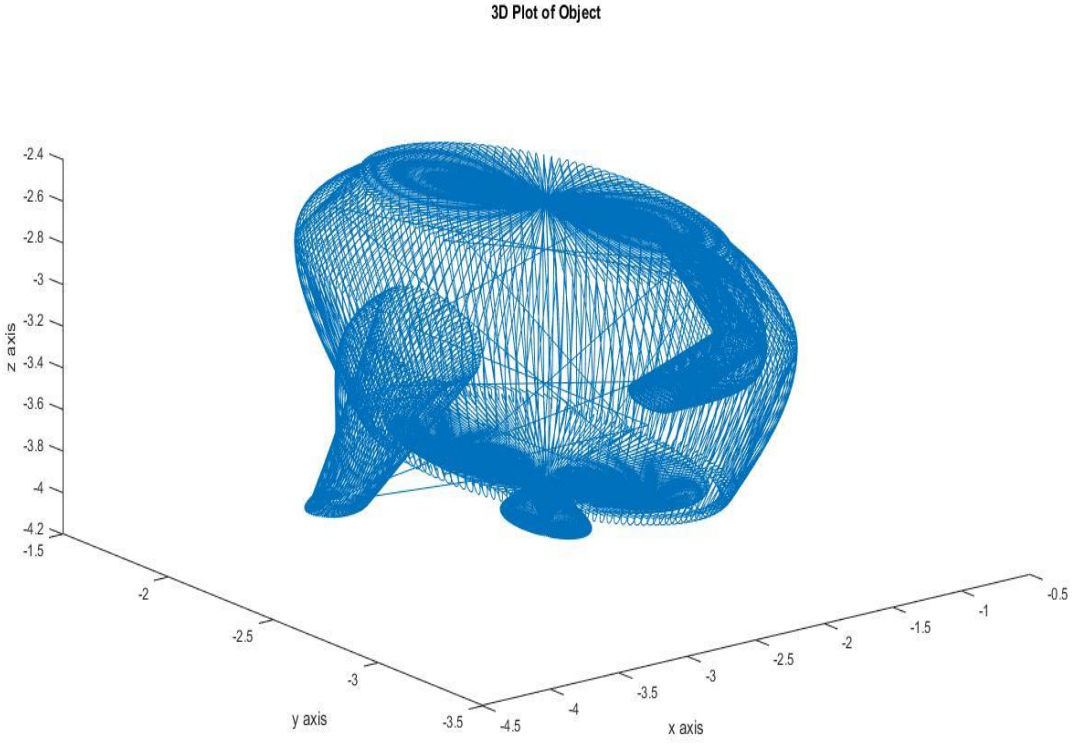

Figure 4: Showing PD as second Camera Matrix

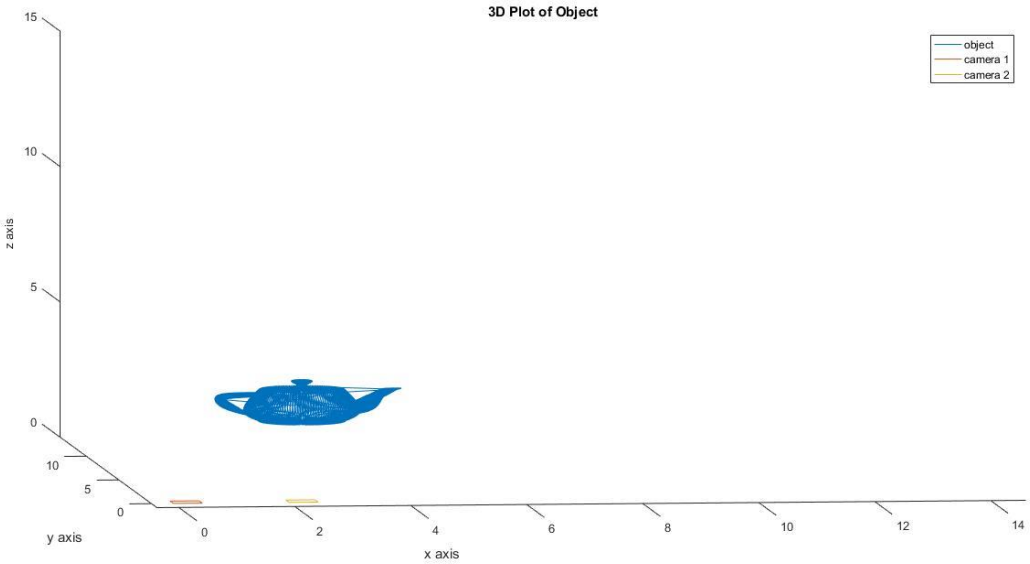

Figure 5: Showing Output when we took PC as second camera projection matrix.

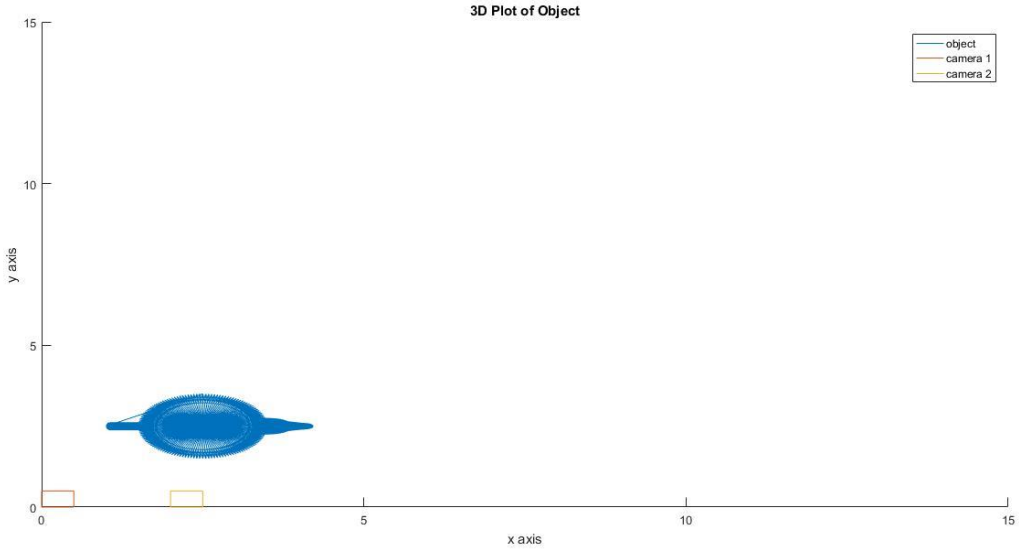

Figure 6: Showing Output when we took PC as second camera projection matrix.

4. Compute and display the 3D reconstruction of a scene. Compare with the ground-truth.

Solution: Now for 3D scene, it will automatically link the rebuild 3D shape with the Original Image if the configuration is matched. Otherwise it will keep showing error messages and the rebuilt 3D structure will open in a different window.

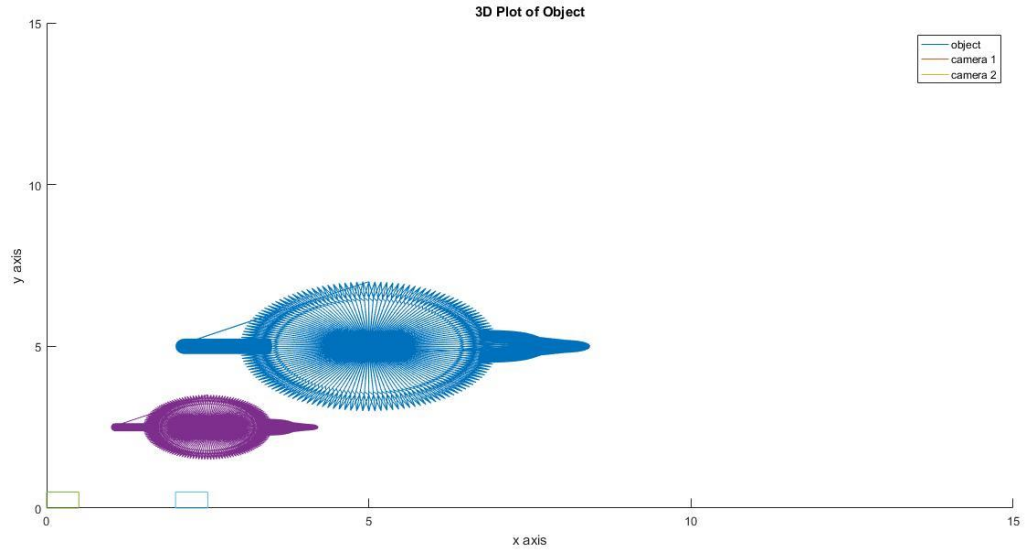

Figure 7: Showing Output when we took PC as second camera projection matrix And merged with Original Structure.

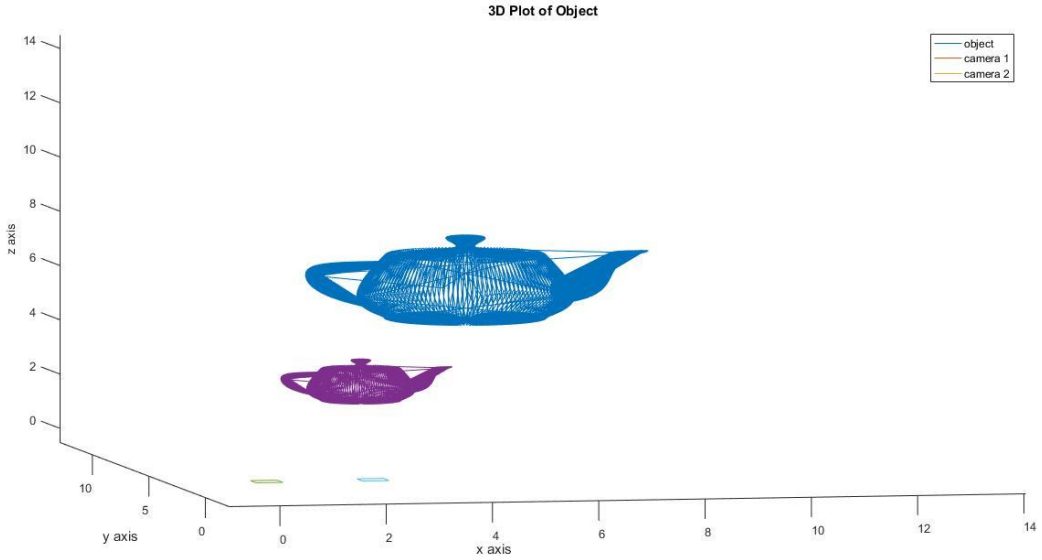

Figure 8: Showing Output when we took PC as second camera projection matrix And merged with Original Structure.

I calculated 3D points from 2D points using a prebuild function “triangulate”. Syntax of which is;

```
Three_D_Points =
triangulate(Image_1(1:2,:)',Image_2(1:2,:)',Proj_3,Proj_4);
```

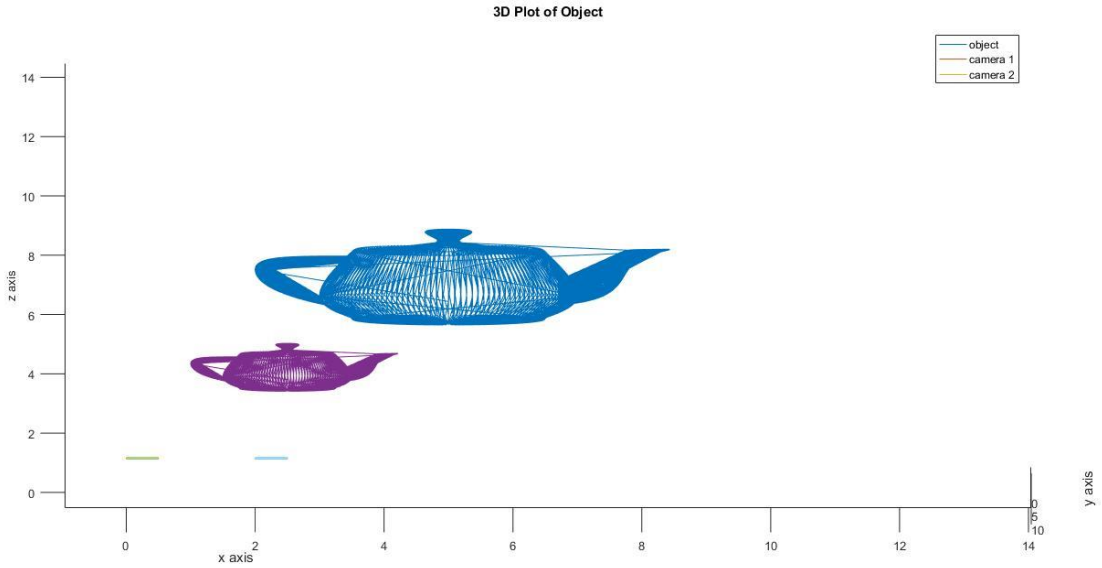

Figure 9: Showing Output when we took PC as second camera projection matrix And merged with Original Structure.

The Shape is recovered because we are considering the Camera parameters to be shifted, so in wrong configuration, the points will shift to –ve side or they will rotate but the shape will remain the same (As shown in class).