CS512 Assignment 5 Report

Ronit Rudra
A20379221

November 26, 2017

Abstract

This report describes the implementation of a Epipolar Line Estimation in Stereo Image Pairs by estimating the Fundamental Matrix through a number of corresponding points using the Normalized 8-point Algorithm. The desired outcomes of the framework, the problems faced, the implemented solutions would be discussed in the upcoming sections.

1 Problem Statement

The objective is to:

1. Estimate Epipolar Lines corresponding to an image when a particular point is known in the other image.

2. Estimate the position of the left and right epipole.

In order to do achieve the aforementioned objectives, the Fundamental Matrix is to be estimated. This is achieved through only the corresponding image points in both the images. The Normalized 8-point algorithm is employed to determine this matrix.

The features required for the program are:

- Would be run from the command prompt/terminal

- Would be passed a filename as argument. The file should be in the data folder.

- Read the left and right images using the passed filename
- Display the two images and allow user to select at least 8 corresponding points on each image.
- Calculate the Fundamental Matrix from the given set of corresponding points.
- Allow the user to select a point on any image and draw the Epipolar Line in the other image.
- Display the numerical coordinates of the left and right epipole (can be outside the bounds of the image)

2 Proposed Solution

The solution revolves around the estimation of the Fundamental Matrix F from the obtained image points. The Fundamental Matrix is a $3x3$ matrix relating the points in one image to the other.

The solution follows the algorithm as defined below:

Algorithm 1 Fundamental Matrix Estimation

```
1: $p_l \leftarrow$ Set of Matched Points in Left Image as 1x3 vectors
2: $p_r \leftarrow$ Set of Matched Points in Right Image as 1x3 vectors
3: $P_l \leftarrow$ Normalized $p_l$ using $M_l$
4: $P_r \leftarrow$ Normalized $p_r$ using $M_r$
5: $A \leftarrow$ An Empty $Nx9$ Matrix
6: $N \leftarrow$ Number of Matched Points
7: for all $i \in P_l r \in P_r$ do
8:     $A_i \leftarrow i^T \cdot r$ is the $i^{th}$ row of $A$
9: end for
10: $U, D, V^T \leftarrow$ SVD Decomposition of $A$
11: $F \leftarrow$ Right Null Space of $V$
12: $F \leftarrow$ New $F$ with Rank 2 enforced
13: $F \leftarrow M_l^T F M_r$ with normalization undone
```

The program was implemented in *Python 3.6.1* with *OpenCV 3*.

The next section details the implementation of the program.

3 Implementation Details

Assume that there are two cameras $Left$ and *Right*, the subscripts l and r would be used to reference them. The left camera produces image I_l and the right camera produces image I_r. The user selects N corresponding points in each image and $N \geq 8$. The estimation of $F$is through the **Normalized 8-point Algorithm** and subject to the following constraint:

$$(p_i)_T F p'_i = 0 \tag{1}$$

The above equation is known as the *Epipolar Constraint* and means that for a homogeneous point p'_i in the left image and a corresponding homogeneous point p_i in the second image, the matrix product is always zero.

The matrix F is estimated by rearranging the equation as a system of linear equations represented as $\tilde{P}\hat{F} = 0$ where $\tilde{P}$ is an $Nx9$ matrix and $\hat{F}$ is the vectorized version of F i.e a $9x1$ vector.

Let $(x'_i, y'_i, 1)$ be the coordinates of the left image and $(x_i, y_i, 1)$ be of the second image and $1 \leq i \leq N$. Then the equation becomes:

$$\begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ . & & & & & & & & \\ . & & & & & & & & \\ . & & & & & & & & \\ . & & & & & & & & \end{bmatrix} \cdot \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0 \tag{2}$$

The Matrix F can be estimated by *SVD* decomposition the $\tilde{P}$ matrix into U, D *and* $V_T$and selecting the right null space of V. It is to be noted that F is a Rank 2 matrix but the F obtained through V is not. To enforce the Rank 2 constraint, we again take the *SVD* of F, set the last element of D to zero and then recompute F, which would be a rank 2 matrix.

This is the *8 Point Algorithm* but is prone to errors as the first few terms of each row in $\tilde{P}$ are very large. It is necessary to normalize the points before estimating F. Hence, each point can be normalized (zero mean and unit standard deviation) through a simple matrix operation of translating the point by the negative of mean in the x and y directions and scaling it down by the standard deviation.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sigma} & 0 & 0 \\ 0 & \frac{1}{\sigma} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & & -\mu_x \\ 0 & 1 & -\mu_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3}$$

Let us call the transformation matrix M_l for the left image and M_r for the right image. After normalization is performed, we obtain the F matrix as defined in the algorithm. But now the matrix satisfies the Epipolar Constraint with regards to the normalized points. Therefore, we need to undo the normalization by:

$$F = M_l^T F M_r \tag{4}$$

Thus, the *Fundamental Matrix* is obtained.

The Epipoles can be calculated easily be *SVD* decomposition of F and selecting the left and right null spaces. The left epipole is the right null space (last column of V) while the left epipole is the left null space (last column of U)

The Epipolar Lines corresponding to the point in left image p' or point in right image p can be calculated as:

$$l = Fp' \tag{5}$$

$$l' = F^T p \tag{6}$$

4 Results and Discussion

The following images show how the program performed for different images when running the *main.py* script with command line parameters *tsukuba* and *corridor* respectively:

Figure 1: User Selected Corresponding points

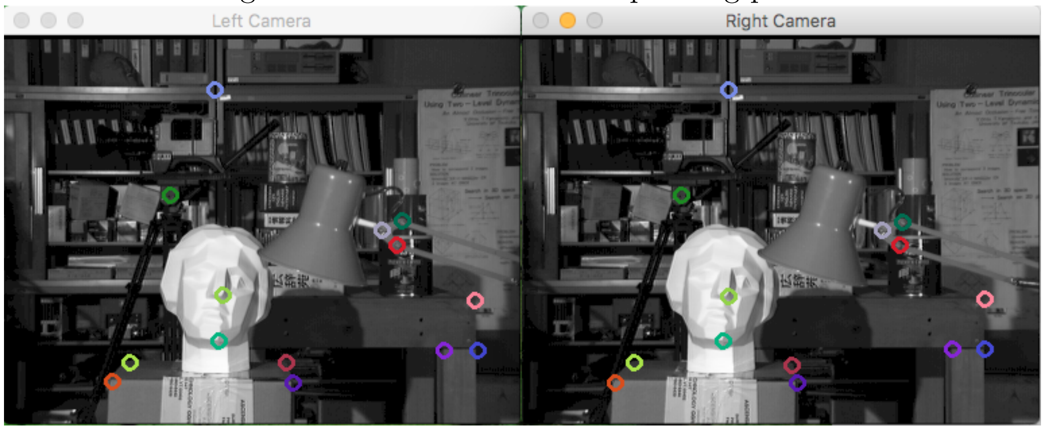


Figure 2: Drawn Epipolar Lines

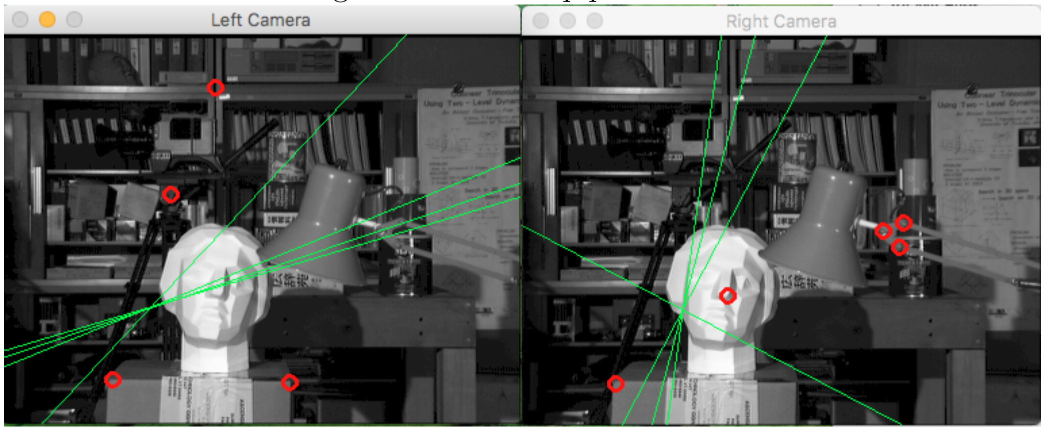


Figure 3: User Selected Corresponding points

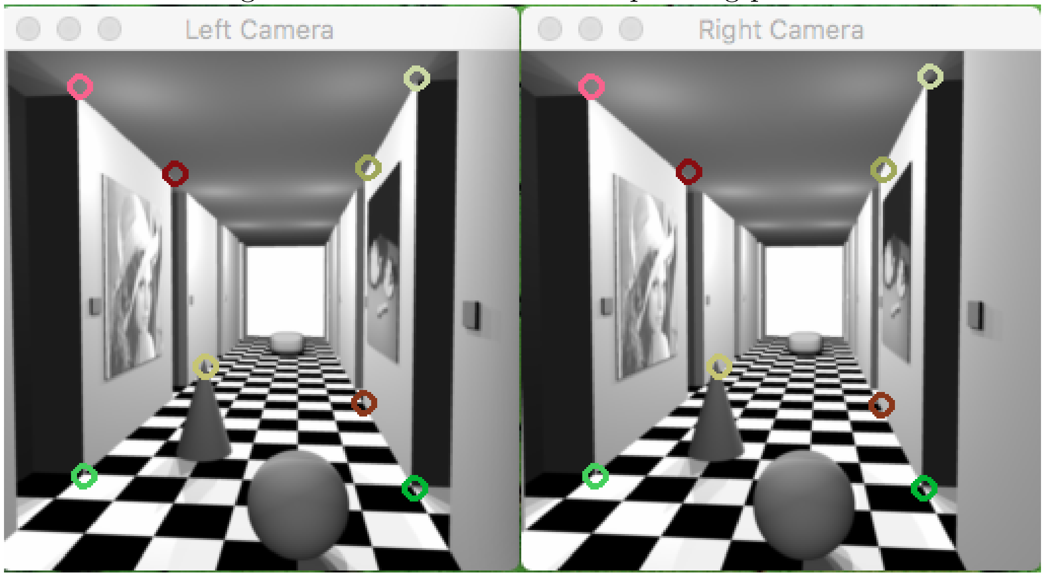
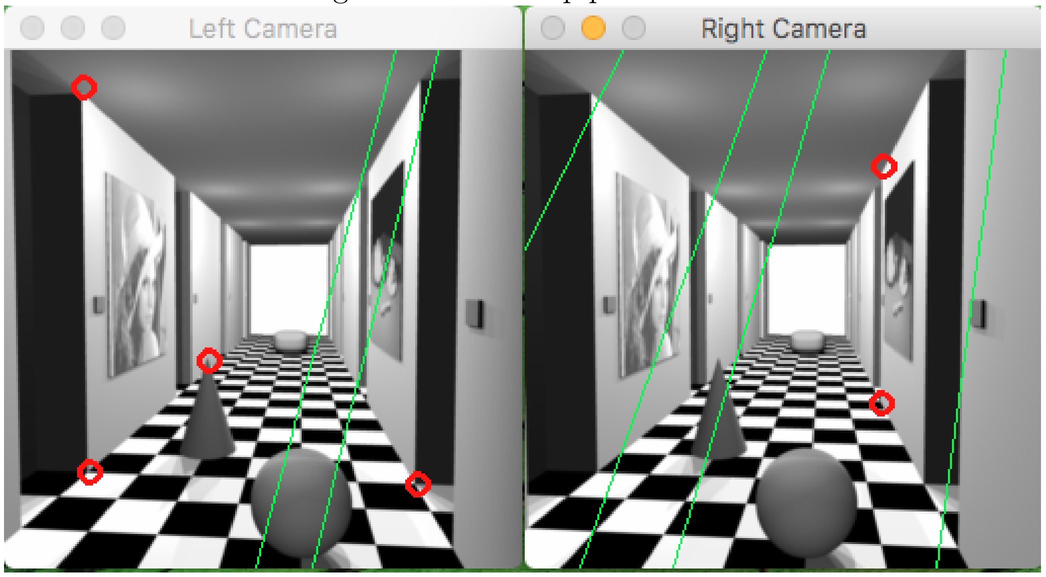
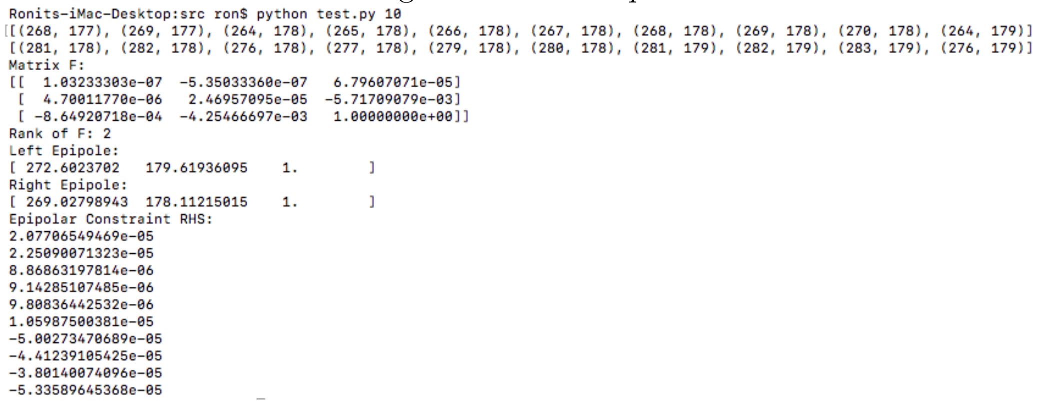


Figure 4: Drawn Epipolar Lines



The *test.py* is run with 10 as the number of precomputed matching points (the matched points are stored in *data* folder).

Figure 5: Test Output



```
Ronits-iMac-Desktop:src ron$ python test.py 10
[[(268, 177), (269, 177), (264, 178), (265, 178), (266, 178), (267, 178), (268, 178), (269, 178), (270, 178), (264, 179)]
[(281, 178), (282, 178), (276, 178), (277, 178), (279, 178), (280, 178), (281, 179), (282, 179), (283, 179), (276, 179)]
Matrix F:
[[  1.03233303e-07  -5.35033360e-07   6.79607071e-05]
 [  4.70011770e-06   2.46957095e-05  -5.71709079e-03]
 [ -8.64920718e-04  -4.25466697e-03   1.00000000e+00]]
Rank of F: 2
Left Epipole:
[ 272.6023702   179.61936095    1.        ]
Right Epipole:
[ 269.02798943  178.11215015    1.        ]
Epipolar Constraint RHS:
2.07706549469e-05
2.25090071323e-05
8.86863197814e-06
9.14285107485e-06
9.80836442532e-06
1.05987500381e-05
-5.00273470689e-05
-4.41239105425e-05
-3.80140074096e-05
-5.33589645368e-05
```

The Program performed satisfactorily for the test images albeit with some error in the location of epipolar lines. This was due to the errors while choosing the points manually. This can be rectified by using a robust estimation method such as RANSAC for choosing the best matching points while removing outliers.

5 References

1. `https://www.robots.ox.ac.uk/~vgg/hzbook/hzbook1/HZepipolar.pdf`

2. `https://docs.opencv.org/trunk/da/de9/tutorial_py_epipolar_geometry.html`

3. `ftp://ftp.cs.utoronto.ca/pub/jepson/teaching/vision/2503/epiPolarGeom.pdf`

4. `http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/Lecture22%20-%20Epipolar%20Geometry.pdf`