LAB 6

Fundamental Matrix Estimation

Matteo Dicenzi
Marco Demutti

1. Introduction

Stereo vision represents the problem of inferring 3D information from two or more images taken from different viewpoints.
In this regard, this experiment aims at estimating *the fundamental matrix F*, which is used to describe the relationship between corresponding points in a stereo system.
In particular, given two corresponding points **x** and **x′** in a stereo image pair (expressed in homogeneous coordinates), **Fx** describes the **epipolar line**, that is the line on which **x′** must lie. More specifically, F will be estimated by implementing the **8 points algorithm** to two sets of images: "*Mire*" and "*Rubik*". Each one of these sets has two images that show the same object from two different points of view. In addition, to compute the fundamental matrix F, we are going to use the provided sets of corresponding points contained in the files "*Mire1.points*", "*Mire2.points*", "*Rubik1.points*", "*Rubik2.points*". We will repeat this procedure twice: one simply applying the algorithm using the corresponding points as they are, and then after normalizing such points (i.e. applying a transformation that allows to translate and scale them).

2. Procedure

The following procedure is repeated for both *Mire* and *Rubik* set of images, therefore we will illustrate all the steps considering only one set of images.
After loading in Matlab the images along with the corresponding point, we wrote function **EightPointsAlgorithm**, which implements a first version of the *8 points algorithm*, in order to perform the estimation of matrix **F**. It takes as input two sets of points represented in homogeneous coordinates and stored as column vectors of two matrices P_1 and P_2. As a first step, it computes the Matrix A with the following formula:

$$\begin{bmatrix} x_1{}'x_1 & x_1{}'y_1 & x_1{}' & y_1{}'x_1 & y_1{}'y_1 & y_1{}' & x_1 & y_1 & 1 \\ x'_2x_2 & x'_2y_2 & x'_2 & y'_2x_2 & y'_2y_2 & y'_2 & x_2 & y_2 & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix}$$

Figure 2.1 - construction of matrix A

Then, matrix F is found by solving the homogeneous system **Af = 0**, where f is the column vector $[f_{11}, f_{12}, \dots, f_{33}]$ (f_{ij} represents the [i, j] element of matrix F). In particular, the goal is to choose a solution h that minimizes the cost function $||Ax||$, subject to $||h|| = 1$, and it is achieved by decomposing the matrix A with the **SVD algorithm (**obtaining $A = U^*D^*V^T$), and choosing the last column of V as the solution. Finally, the function returns matrix F, forced to have rank 2 by applying once again SVD algorithm and setting $D(3,3) = 0$.
Then, we wrote function **EightPointsAlgorithmN,** which implements a second, enhanced version of the *8 points algorithm*, estimating the *Fundamental Matrix* starting from a set of *normalized points*, obtained by normalizing the original points with the provided function *normalise2dpts* (i.e. performing translation and scaling). Besides the normalization step, which also implies having to de-normalize the final matrix F (as $T_2^T * F * T_1$), the steps of the algorithm are just the same as the previous case.
The two functions **EightPointsAlgorithm** and **EightPointsAlgorithmN (**i.e. starting from the normalized set of points) were then used to compute the two fundamental matrices F_1 and F_2, respectively.
At this point, we checked whether the epipolar constraint $x_2^T * F * x_1 = 0$ (for each corresponding pair (x_1, x_2) in P_1 and P_2) holds for the fundamental matrices F_1 and F_2, and used the provided function *visualizeEpipolarLines* to display the two sets of points and the corresponding epipolar lines onto the two stereo images.
After that, we computed the epipoles of both F_1 and F_2 by writing function ***computeLeftAndRightEpipoles.*** It takes as input the Fundamental matrix F and returns the epipoles in cartesian coordinates of both the left and right images. More specifically, the computation of the epipoles involves two steps: the first one gives us the left and right epipoles expressed in homogeneous coordinates, by decomposing F as $F = U * W * V^T$ (computing once again the SVD decomposition of F), and selecting the last column of U and V (since the left and right epipoles are respectively the right and left null space of F); the second step consists in converting both epipoles from homogenous coordinates to cartesian coordinates, which are simply the first two coordinates, both divided by the third one (obtaining $[\frac{X_1}{X_3}, \frac{X_2}{X_3}]$ from $[X_1, X_2, X_3]$).
Finally, function ***visualizeEpipolesAndEpLines*** was written in order to visualize the epipolar lines and the right and left epipoles all together, and check whether left and right epipoles were, as expected, the points of intersection of the corresponding epipolar lines.

3. Results

Figures 3.1-3.2 show the two stereo images of *Mire* set, along with the corresponding points and the epipolar lines (as output of *visualizeEpipolarLines* function): in figure 3.1 we can see the results of the first, not-normalized version of the algorithm, while figure 3.2 displays the results of the second, normalized version.

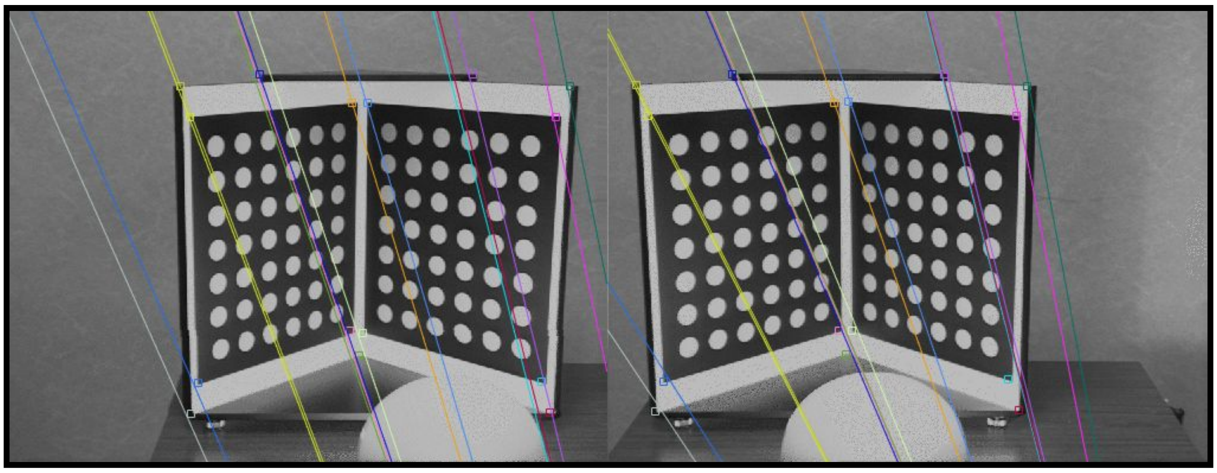


Figure 3.1 - Mire *stereo images, not normalized points (first algorithm)*

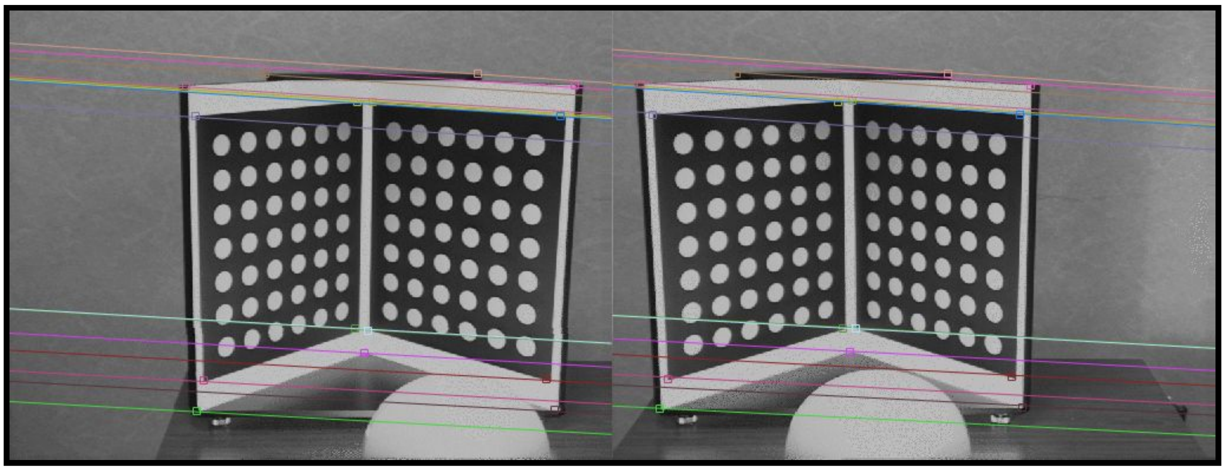
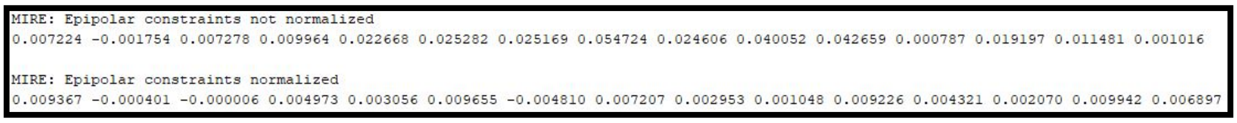


Figure 3.2 - Mire *stereo images, normalized points (second algorithm)*

We can notice that the estimation of matrix F in the second version of the algorithm is more precise: the resulting epipolar lines (*Figure 3.2*) are more reliable. This is due to the fact that matrix F, estimated using this algorithm, is not invariant to point transformation and normalizing the points allows to overcome this issue.
However, we can check that in both cases, the epipolar constraint holds with a quite satisfactory precision (*Figure 3.3*), still slightly better in the second case.



```
MIRE: Epipolar constraints not normalized
0.007224 -0.001754 0.007278 0.009964 0.022668 0.025282 0.025169 0.054724 0.024606 0.040052 0.042659 0.000787 0.019197 0.011481 0.001016

MIRE: Epipolar constraints normalized
0.009367 -0.000401 -0.000006 0.004973 0.003056 0.009655 -0.004810 0.007207 0.002953 0.001048 0.009226 0.004321 0.002070 0.009942 0.006897
```

Figure 3.3 - Mire *epipolar constraint*

Also, in both cases, the left and right epipoles correctly represent the points of intersection of the epipolar lines in the left and right image, respectively: this can be displayed using ***visualizeEpipolesAndEpLines*** function (*Figures 3.4-3.5*).
Of course, being the epipolar lines almost parallel, the corresponding epipoles are distant from the origin, both on x-axis and on y-axis, especially in the second version of the algorithm (*Figure 3.5*).

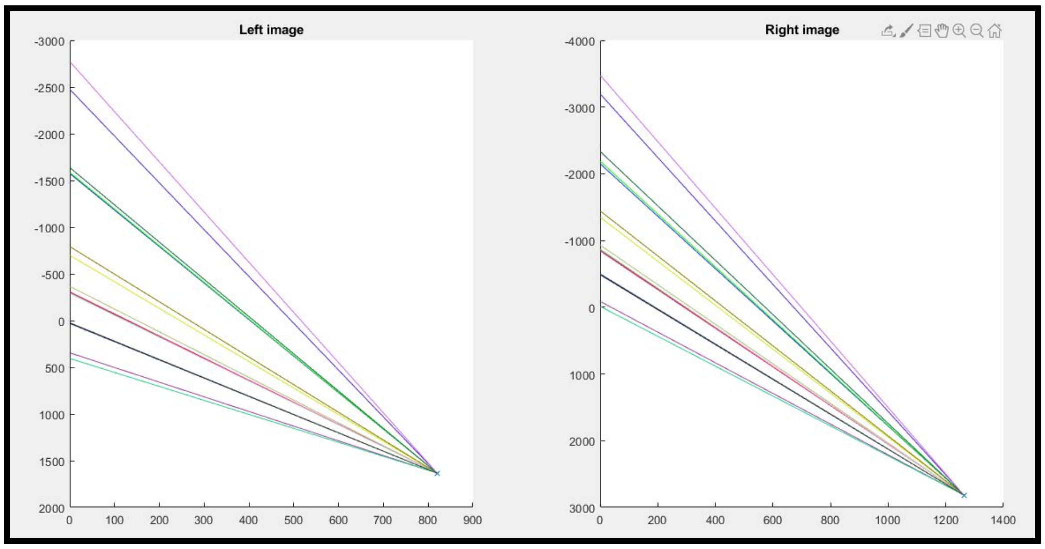


Figure 3.4 - Mire *epipoles and epipolar lines in left and right image (first version)*

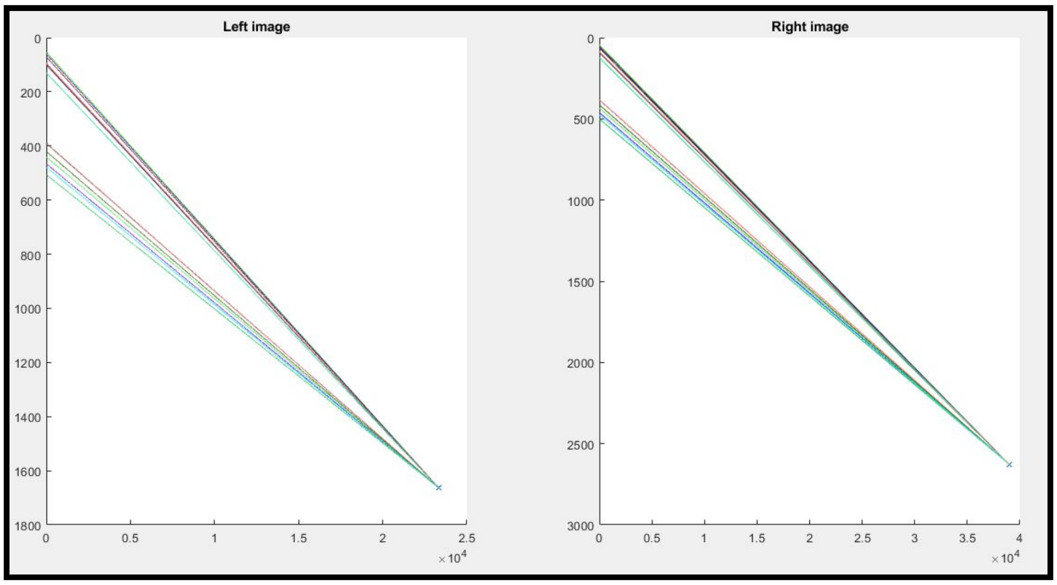


Figure 3.5 - Mire *epipoles and epipolar lines in left and right image (second version)*

Figures 3.6-3.7 show the results of the same procedure applied to *Rubik* stereo images (as output of *visualizeEpipolarLines* function): once again, in *figure 3.6* we can see the results of the first, not-normalized version of the algorithm, while *figure 3.7* displays the results of the second, normalized version.

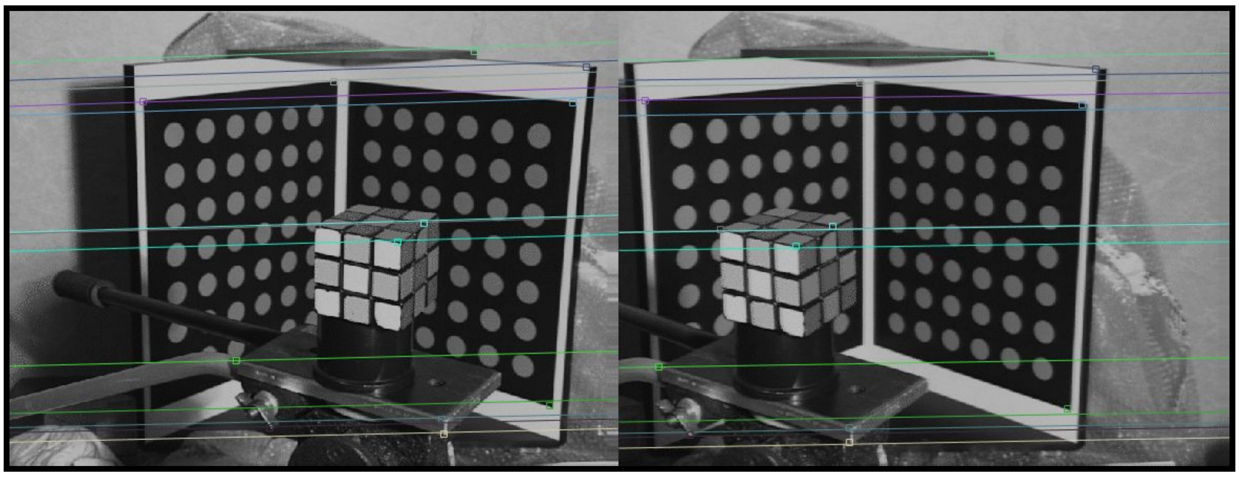


Figure 3.6 - Rubik *stereo images, not normalized points (first algorithm)*

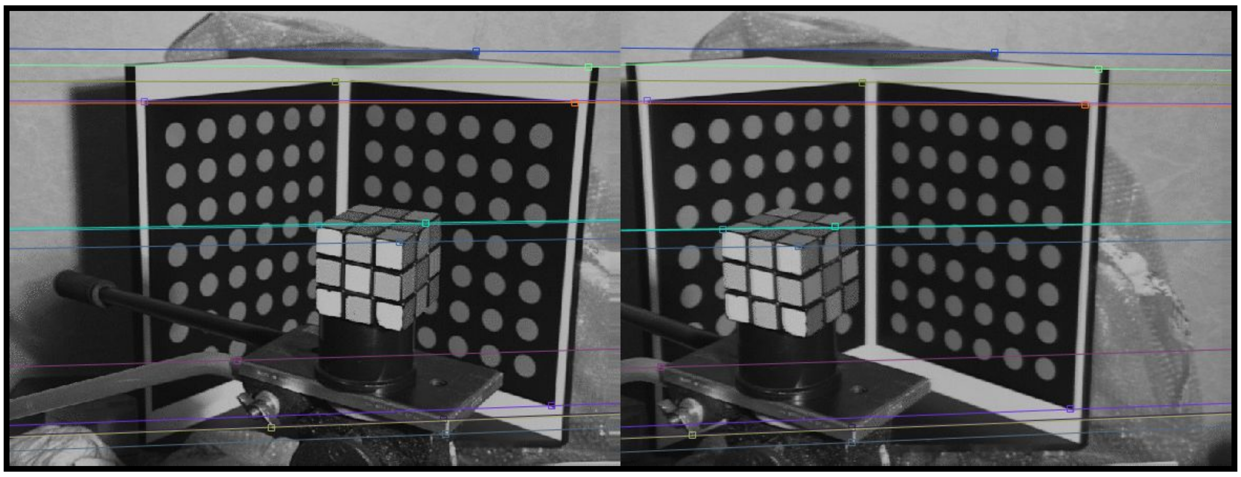
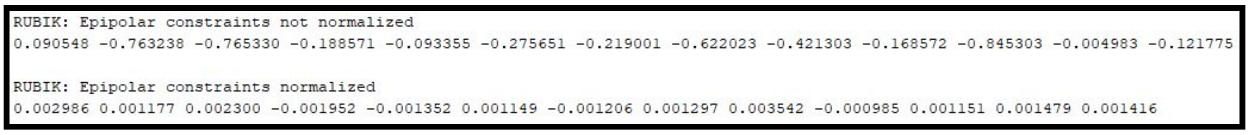


Figure 3.7 - Rubik *stereo images, normalized points (second algorithm)*

As for the previous case, the estimate of F obtained by applying the second version of algorithm is more precise, even though it is not that evident at first glance.
However, this difference becomes much more visible if we have a look at the epipolar constraint (*Figure 3.8*): in the first version of the algorithm it does not hold with an adequate precision, whereas in the second version the precision looks reasonable.



```
RUBIK: Epipolar constraints not normalized
0.090548 -0.763238 -0.765330 -0.188571 -0.093355 -0.275651 -0.219001 -0.622023 -0.421303 -0.168572 -0.845303 -0.004983 -0.121775

RUBIK: Epipolar constraints normalized
0.002986 0.001177 0.002300 -0.001952 -0.001352 0.001149 -0.001206 0.001297 0.003542 -0.000985 0.001151 0.001479 0.001416
```

Figure 3.8 - Rubik *epipolar constraint*

Moreover, the right epipole related to the first version does not correctly represent the intersection of the epipolar lines (hence, the image is not included in this report). This can also be considered as a consequence of the missing normalization, since in the second version left and right epipoles are correctly computed (*Figure 3.9*).

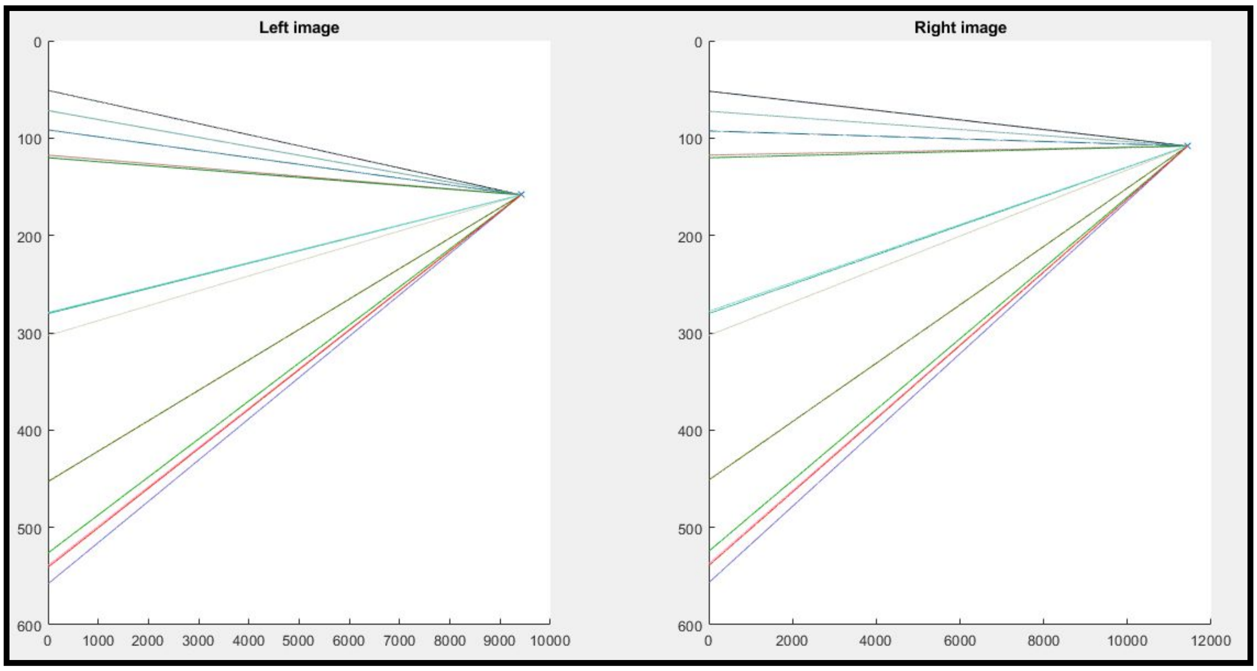


Figure 3.9 - Rubik *epipoles and epipolar lines in left and right image (second version)*

4. Conclusions

The results of this experiment were coherent with the ones expected. For both the sets of stereo images, matrix F was correctly estimated and in both cases it was evident that the normalized version of the algorithm allowed to obtain a far more precise estimation: as for *Mire* set of images, the resulting epipolar lines were significantly different in the two cases, whereas for the *Rubik* case, not performing the normalization of the points led to a bad precision of the epipolar constraint, as well as to a wrong calculation of one of the two epipoles.
These outcomes are a consequence of the fact that matrix F, estimated with this approach, is not invariant to point transformations. Performing a normalization (i.e. translation and scaling) of the points before applying the *8 points algorithm* (and de-normalizing matrix as a final step) is definitely an advisable procedure, as shown in this experiment.