Computer Vision Assignment 4: Model Fitting

Charlotte Moraldo

1. Line fitting with RANSAC

For line fitting, we use the RANSAC algorithm. We randomly select 2 points in the dataset that we hope are inliers, and fit a polynomial (linear, degree 1). We then compute the distance between all points with the fitting line. If this distance is smaller than a certain predefined threshold, then the corresponding point is an inlier. We repeat this process several times (we set a maximum number of iteration). The best model is the one which has the most inliers. Below is what is obtained from the algorithm. The black line is the real model of the line, the green one is the least squares fitting, and the red line is obtained with RANSAC.

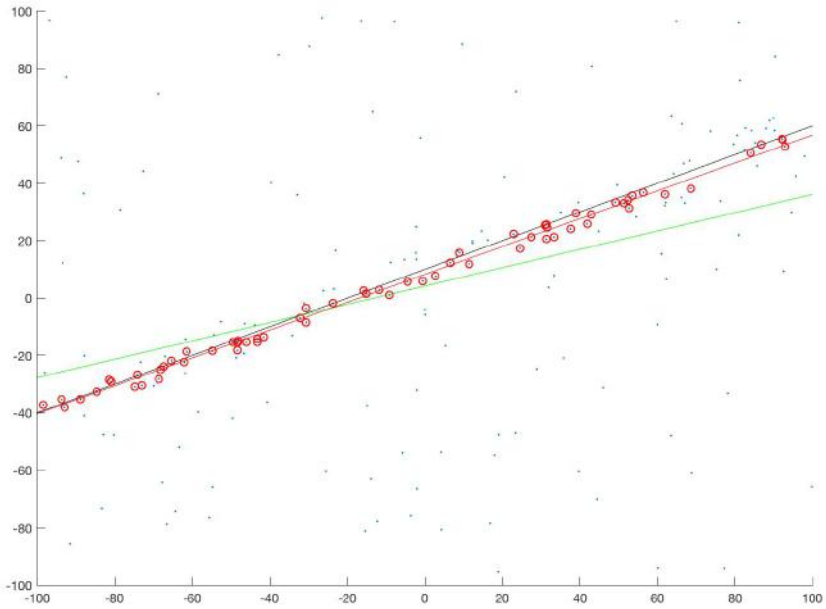

Figure 1: Line fitting with RANSAC.

As expected and as seen on the plot, the error from least squares fitting is the highest: $err_{LeastSquares} = 183.0052$. The RANSAC line is very close to the real line, and so is their error: $err_{RANSAC} = 39.2209$ and $err_{RealLine} = 38.1075$, respectively.

2. Fundamental Matrix

To compute the fundamental matrix, after normalizing the inputs, we compute the eight-point algorithm: $(xx', xy', x, yx', yy', y, x', y'1) * vec(F) = 0$. F is obtained with through SVD. To compute $\widehat{F}$, we take the SVD of F and set the last singular value to zero in order to inforce the singularity constraint $det(F) = 0$.

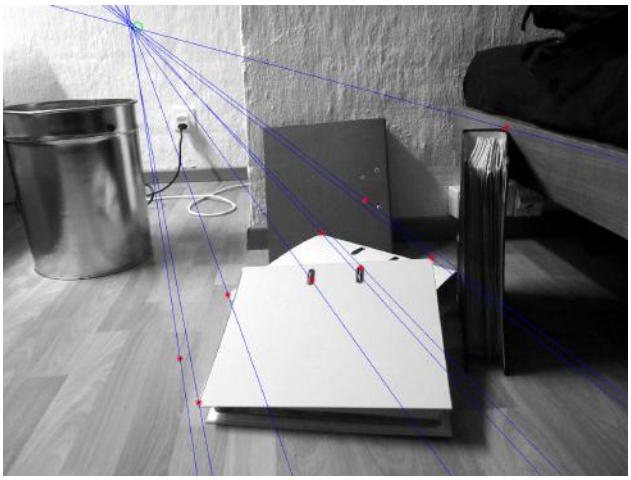

Figure 2: Epipolar lines with F, image 1

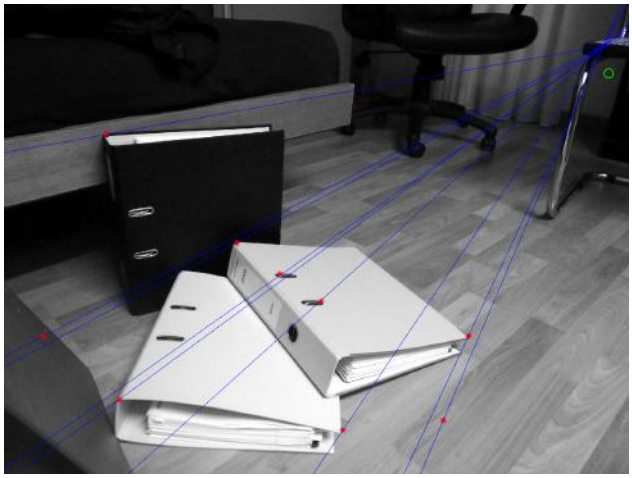

Figure 3: Epipolar lines with F, image 2

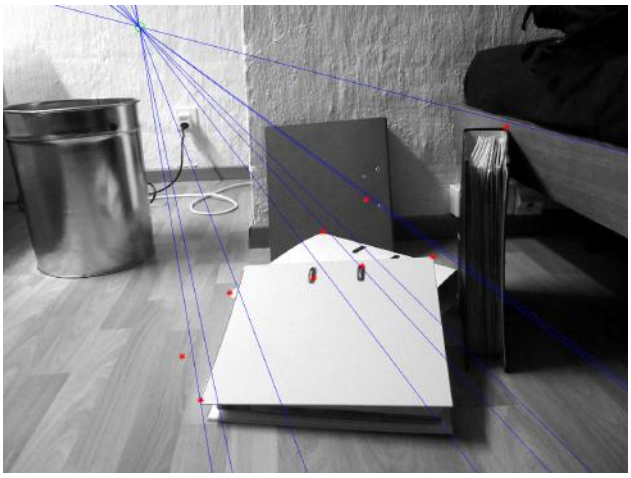

Figure 4: Epipolar lines with $\widehat{F}$, image 1

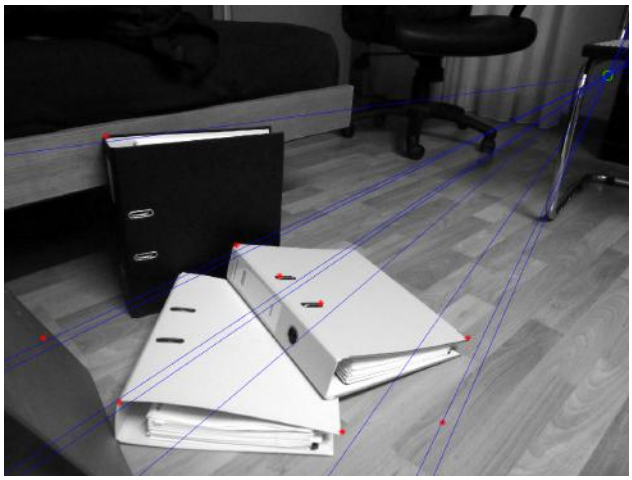

Figure 5: Epipolar lines with $\widehat{F}$, image 2

When computing the epipolar lines with the non-singular fundamental matrix F, we can see that they go through all the clicked points (in red), but don't intersect precisely at the epipole, in green (Fig. 2 and 3). However, when drawing the epipolar lines with the fundamental matrix $\widehat{F}$ (Fig.4 and 5), the lines are slightly shifted from the clicked points, but they intersect exactly at the epipole. The obtained fundamental matrix F is very close to $\widehat{F}$, with differences under 10^{-6}. We obtain:

$$F = \begin{bmatrix} -0.0000 & 0.0000 & -0.0000 \\ 0.0000 & 0.0000 & -0.0010 \\ -0.0001 & -0.0037 & 0.3095 \end{bmatrix}$$

3. Essential Matrix

Using the camera calibration matrix K which we are given, we can compute the 2D normalized points: $\widehat{\mathbf{x}} = \mathbf{K}^{-1}\mathbf{x}$. Similarly as previously, we can then apply the 8-point algorithm to obtain the essential matrix E. To compute $\widehat{E}$, we take the SVD of E, and force the first two singular values to be equal to their mean, and set the third one to zero. The fundamental matrix can then be derived: $\mathbf{F} = \mathbf{K}^{-1} \cdot \mathbf{E} \cdot \mathbf{K}$. The obtained matrices are:

$$E = \begin{bmatrix} -0.3480 & 5.0709 & 1.9379 \\ 5.8601 & 1.4109 & 2.4906 \\ 2.2150 & -2.4742 & -0.2102 \end{bmatrix} \qquad Eh = \begin{bmatrix} -0.7295 & 5.2708 & 1.8653 \\ 5.6229 & 1.0525 & 2.2843 \\ 2.3713 & -2.6938 & -0.2910 \end{bmatrix}$$

$$F = \begin{bmatrix} -0.0000 & 0.0000 & -0.0000 \\ 0.0000 & 0.0000 & -0.0010 \\ -0.0001 & -0.0037 & 0.3094 \end{bmatrix}$$

We observe that the camera calibration matrix F is almost the same as the one found in the previous section. The main difference observed is on the element (3,3), and is of order 10^{-4}.
As shown below, the epipolar lines computed with the obtained camera calibration matrix are very similar to those obtained in section 3.

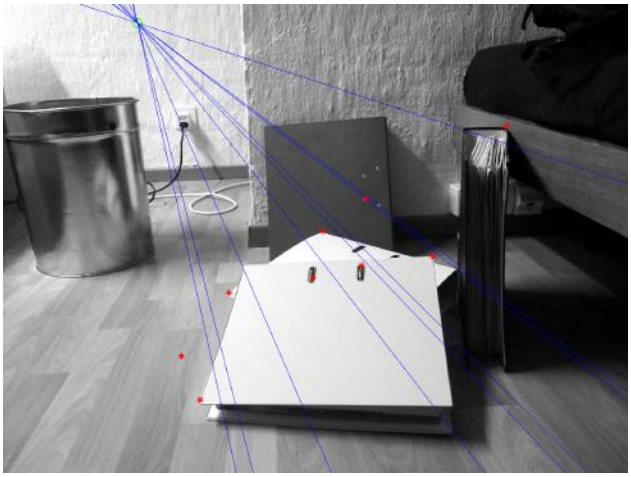

Figure 6: Epipolar lines with F, image 1

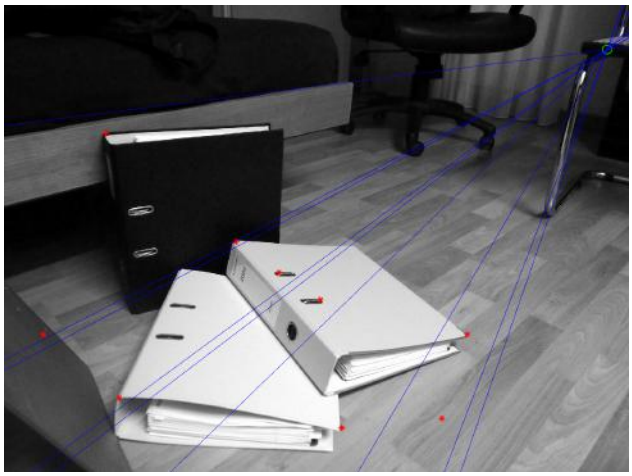

Figure 7: Epipolar lines with F, image 2

4. Camera Matrix

We decompose the essential matrix such that $E = [t]_x R$. To do so, we start by taking the SVD of E, such that $E = USV^T$. We define the following matrix:

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

We can then compute: $R_1 = UWV^T$, $R_2 = UW^TV^T$, and $t_1 = -t_2 = U(3,3)$. The first camera matrix is given by: $P_1 = [I\ 0]$. P_2 has 4 possible solutions: $[R_1\ t_1]$, $[R_1\ t_2]$, $[R_2\ t_1]$ or $[R_2\ t_2]$.
In order to compute which matrix is the right choice, we triangulate the 3D points with each possible P_2,

and then check that these points are in front of both cameras. In order to visualize this, we use the function `showCameras`, and we obtain:

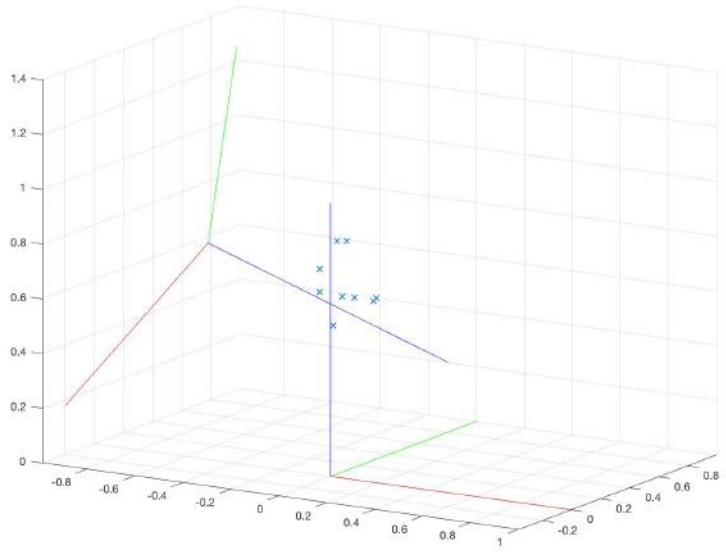

Figure 8: 3D graph of the cameras orientation and the clicked points

We therefore confirm that the choice for P_2 is the right one, as all the 3D points are in front of both cameras (the points have positive values on the blue axis).

5. Feature Extraction and Matching

I decided to use the VLFeat toolbox in order to perform the SIFT algorithm. Below are the results obtained with two different datasets:

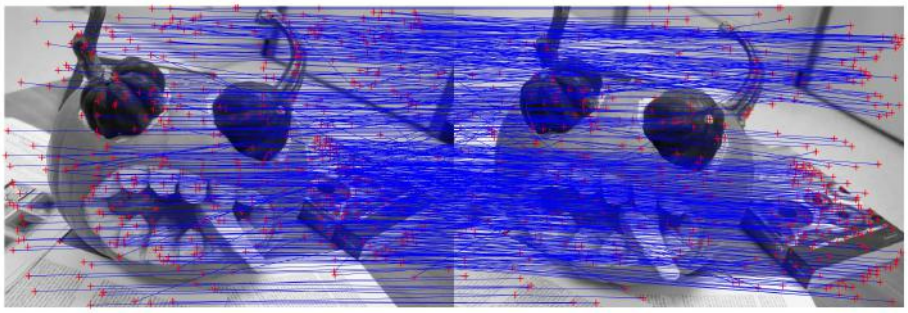

Figure 9: Dataset "pumpkin": matches of both images side to side

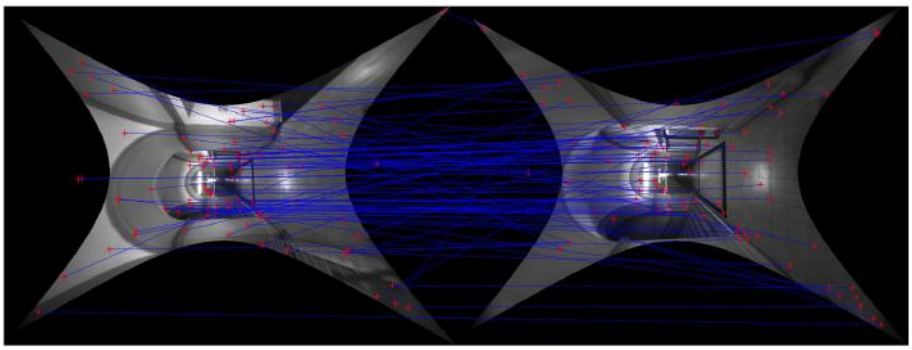

Figure 10: Dataset "hallway": matches of both images side to side

We can see that a lot of matches are found, but in those matches there is a lot of outliers. This can also be seen when computing the matches in the same image, to get a sense of motion. Below are the results obtained:

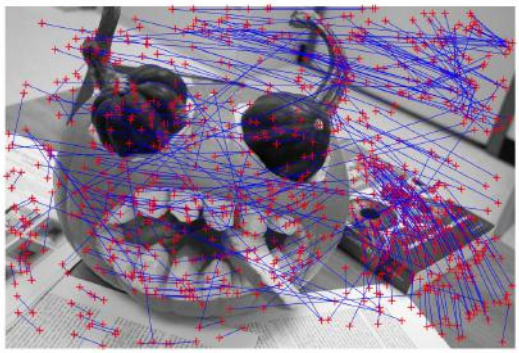

Figure 11: Dataset "pumpkin": matches in one image

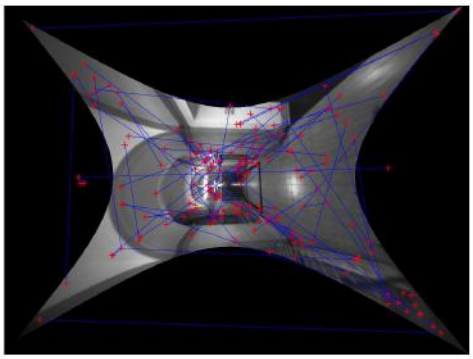

Figure 12: Dataset "hallway": matches in one image

The results of the dataset "pumpkin" aren't bad: when looking at Fig.11, we can get a little bit that sense of motion, we see that the camera has rotated around the pumpkin. However, a lot of outliers remain and better results must be achieved. In the case of the "hallway" dataset, we can also observe a lot of outliers. The movement of the camera isn't as well perceived due to the outliers that are located at the foreground of the picture.

6. 8-point RANSAC

6.1 Simple RANSAC

The RANSAC algorithm allows us to distinguish inliers (good matches) from outliers (bad matches), and to keep only the inliers. To do so, we use the distance between each point and their epipole as a threshold criteria.

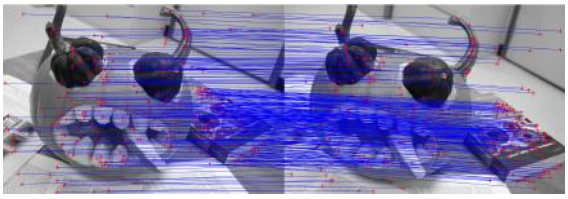

Figure 13: Inliers of the "pumpkin" dataset for a threshold of 5

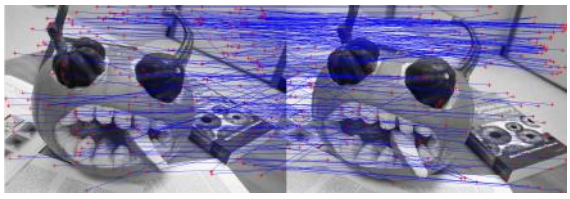

Figure 14: Outliers of the "pumpkin" dataset for a threshold of 5

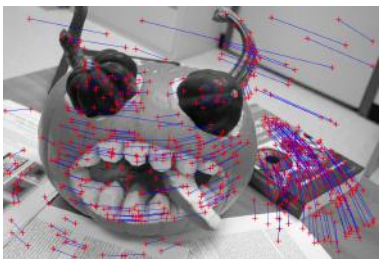

Figure 15: "Motion" when all inliers matches are on same image

When setting the threshold at 5 pixels, after applying the RANSAC algorithm, we obtain a number of 372 inliers. As seen on the images above, this result is very good: in the case of the single image, we now really have that sense of motion of the cameras rotating around the pumpkin. Below are some other examples of inliers and outliers, for a different dataset:

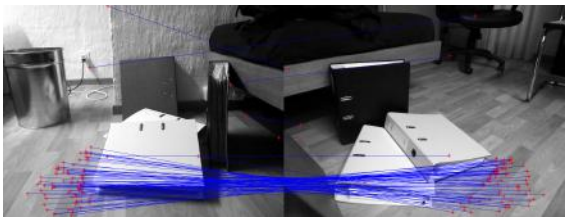

Figure 16: Inliers of the "rect" dataset

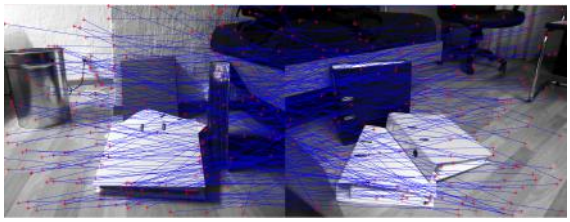

Figure 17: Outliers of the "rect" dataset

6.2 Adaptative RANSAC

The adaptative RANSAC algorithm stops after M trials if the following equation is satisfied: $p = 1 - (1 - r^N)^M$, where r is the inlier ratio, N is the number of samples drawn, and M is the number of iterations. Therefore, p represents the probability that at least one of the random samples of the N points from these M trials is free from outliers, and r is the best inlier ratio found so far.
The number of iterations needed varies depending on the threshold and the set of images: for the "hallway" dataset, it goes from 1409 with 1 pixel as a threshold to 272 with 3 pixels and for the "rect" dataset it goes from 12253 with 1 pixel as a threshold to 2190 with 3 pixels.