University of Burgundy

Assignment III

Epipolar Geometry & 8-Point Algorithm

Kimberly Grace Sevillano Colina

1 Introduction

In this exercise, we will focus on estimating the fundamental matrix (F) for an uncalibrated camera. We will implement the widely adopted 8-Point Algorithm for estimating the fundamental matrix from correspondences. Although the fundamental matrix has only 7 degrees of freedom and requires 7 constraints (which is addressed by the 7-Point Algorithm), the 8-Point Algorithm offers a linear formulation as opposed to the non-linear nature of the 7-Point Algorithm. It is important to note that the 8-Point Algorithm is also valid and often used for computing the essential matrix in the calibrated setting, instead of the 5-Point Algorithm.

2 Part I - Environment Setup & Useful Commands

2.1 Environment Setup

To set up the conda environment for working on the local machine, follow the steps below:

- Create a new conda environment: `conda create -n epipolar`
- Activate the environment: `conda activate epipolar`
- Install Jupyter Notebook: `conda install jupyter`
- Navigate to the workspace directory of the course and run Jupyter: `jupyter - notebook`

2.2 Input Images

The input files for this task are taken from the folder images/ in the file `CV_Assignment_3_Epipolar.zip`. We have the left and right images taken by the same camera and we are also provided with two text files with the coordinates of the corresponding points between the images.

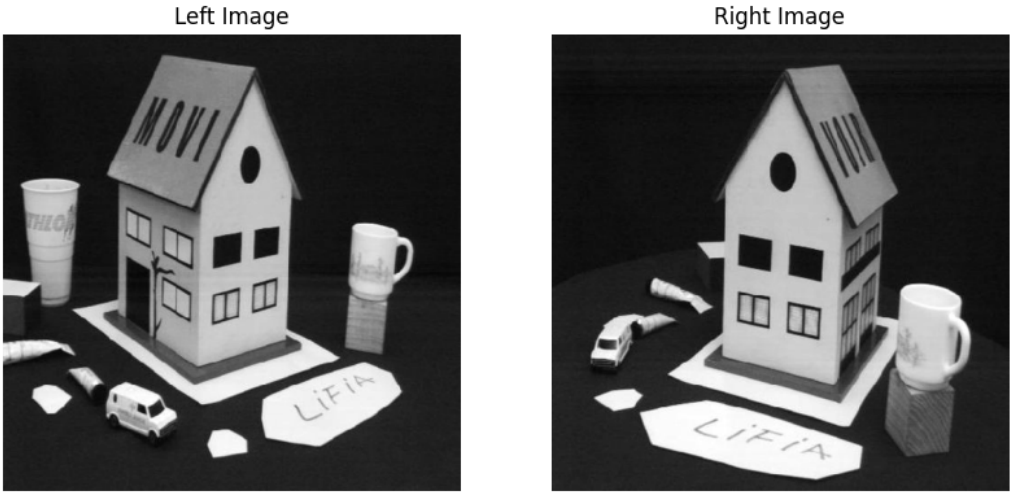

Figure 1: These are all the images that are shown in the folder images.

3 Part II - Normalized 8-Point Algorithm

Algorithm 1 Normalized 8-Point Algorithm (Determine the fundamental matrix F)

INPUT: $n \geq 8$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$
OUTPUT: F (fundamental matrix)

1: **Normalization:** Transform the image coordinates according to $\hat{\mathbf{x}}_i = \mathrm{T}\mathbf{x}_i$ and $\hat{\mathbf{x}}'_i = \mathrm{T}'\mathbf{x}'_i$, where T and T′ are normalizing transformations consisting of a translation and scaling.
2: **Find the fundamental matrix** $\hat{F}'$ corresponding to the matches $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ by
   (a) **Linear solution:** Determine $\hat{F}$ from the singular vector corresponding to the smallest singular value of $\hat{\mathrm{A}}$, where $\hat{\mathrm{A}}$ is composed from the matches $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$.
   (b) **Constraint enforcement:** Replace $\hat{F}$ by $\hat{F}'$ such that $\det \hat{F}' = 0$ using the SVD.
3: **Denormalization:** Set $F = \mathrm{T}'\mathrm{T}\hat{F}'\mathrm{T}$. Matrix F is the fundamental matrix corresponding to the original data $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$.
4: **return** F

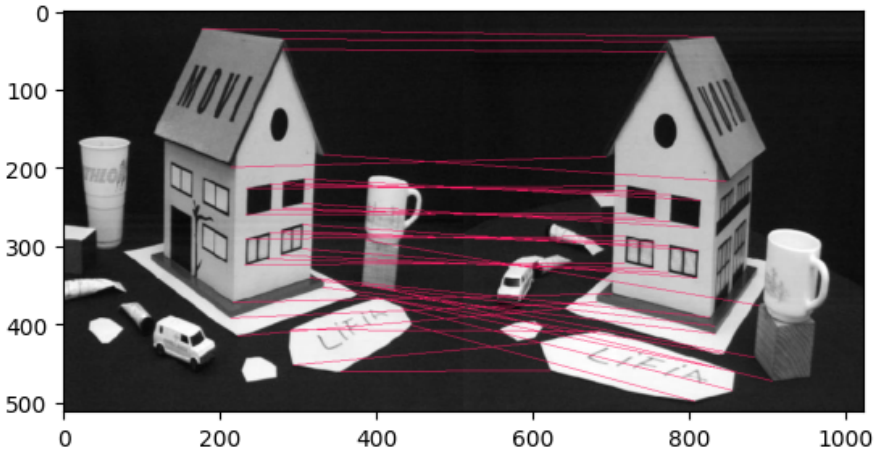

Figure 2: The image shows the correspondences between the two images

3.1 Data Conditioning

In the data conditioning step, we first normalize the keypoints to improve the numerical stability of our algorithm. The process consists of translating the keypoints such that their centroid is at the origin and scaling them so that their average distance to the origin is $\sqrt{2}$. The transformation matrix T can be computed as the product of the scaling matrix S and the translation matrix M:

$$T = SM \tag{1}$$

First, we calculate the mean coordinates of the keypoints:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} kps_i \tag{2}$$

Then, we center the keypoints by subtracting the mean coordinates:

$$kps'_i = kps_i - \mu \tag{3}$$

Next, we compute the average distance of the centered keypoints to the origin:

$$\bar{d} = \frac{1}{N} \sum_{i=1}^{N} ||kps'_i|| \tag{4}$$

We determine the scaling factor based on the average distance:

$$s = \frac{\sqrt{2}}{\bar{d}} \tag{5}$$

Now, we can construct the translation matrix M and the scaling matrix S:

$$M = \begin{bmatrix} 1 & 0 & -\mu_x \\ 0 & 1 & -\mu_y \\ 0 & 0 & 1 \end{bmatrix}, \quad S = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Finally, we obtain the normalized coordinates by applying the transformation matrix T to the original keypoints:

$$kps''_i = Tkps_i \tag{7}$$

3.2 Linear Estimation

In the linear estimation step, we construct a matrix A using the normalized correspondences between the two sets of keypoints. The matrix A is created by stacking the outer product of corresponding points in both images:

$$A = \begin{bmatrix} u_1u'_1 & v_1u'_1 & u'_1 & u_1v'_1 & v_1v'_1 & v'_1 & u_1 & v_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_Nu'_N & v_Nu'_N & u'_N & u_Nv'_N & v_Nv'_N & v'_N & u_N & v_N & 1 \end{bmatrix} \tag{8}$$

where (u_i, v_i) and (u'_i, v'_i) represent the corresponding keypoints in the first and second images, respectively. The matrix A should have at least 8 point correspondences to satisfy the constraint for the 8-point algorithm.

We then perform Singular Value Decomposition (SVD) on the matrix A:

$$A = U\Sigma V^T \tag{9}$$

The last column of the matrix V^T, corresponding to the smallest singular value, is reshaped into a 3x3 matrix to form the estimated fundamental matrix F:

$$F = \text{reshape}(V^T[:, -1], (3, 3)) \tag{10}$$

3.3 Enforce Rank-2 Condition

To ensure that the estimated fundamental matrix F satisfies the rank-2 condition, we perform the following steps. First, we compute the Singular Value Decomposition (SVD) of the matrix F:

$$F = U\Sigma V^T \tag{11}$$

Next, we set the smallest singular value in the diagonal matrix Σ to 0:

$$\Sigma' = \Sigma - \text{diag}(0, 0, \sigma_{min}) \tag{12}$$

Finally, we reconstruct the rank-2 fundamental matrix F_{rank2} by multiplying the modified singular value matrix Σ' with the original matrices U and V^T:

$$F_{rank2} = U\Sigma' V^T \tag{13}$$

Enforcing the rank-2 condition ensures that the resulting fundamental matrix F_{rank2} better satisfies the geometric constraints associated with the epipolar geometry.

3.4 Denormalize Estimated Fundamental Matrix

After enforcing the rank-2 condition on the estimated fundamental matrix, we need to denormalize it to obtain the final fundamental matrix F_{denorm}, which relates the original image coordinates. The denormalization is done using the transformation matrices T_{p1} and T_{p2} obtained during the data conditioning step for both sets of keypoints.

The denormalization process can be represented as follows:

$$F_{denorm} = T_{p2}^T F_{rank2} T_{p1} \tag{14}$$

By denormalizing the fundamental matrix, we ensure that it is in the same coordinate system as the original image points, allowing us to accurately describe the epipolar geometry between the two images.

3.5 RANSAC

RANSAC algorithm is used to estimate the fundamental matrix robustly, dealing with outliers. The steps are:

1. Randomly select 8 point correspondences: $p1$ and $p2$.
2. Estimate F_{lin} and enforce rank-2 condition to obtain F_{rank2}.
3. Count inliers using the Sampson distance (d_s) and a threshold (t).
4. Update the optimal fundamental matrix with higher inliers.
5. Repeat for a specified number of iterations.

The Sampson distance (d_s) is calculated as follows:

$$d_s = \frac{(\boldsymbol{p_2}^T F_{rank2} \boldsymbol{p_1})^2}{(\boldsymbol{p_1}^T F_{rank2})_{[0]}^2 + (\boldsymbol{p_1}^T F_{rank2})_{[1]}^2 + (\boldsymbol{p_2}^T F_{rank2}^T)_{[0]}^2 + (\boldsymbol{p_2}^T F_{rank2}^T)_{[1]}^2} \tag{15}$$

If $d_s < t$, the point is considered an inlier. After RANSAC, denormalize the fundamental matrix.

4 Part III - Fundamental Matrix Estimation Tests

4.1 Test 1: Visualization of Epipolar Lines

In this subsection, we present Test 1 results comparing the visualization of epipolar lines without RANSAC and with RANSAC. The goal is to evaluate the impact of using RANSAC on the estimation accuracy.

The first image displays the epipolar lines without RANSAC, while the second image shows the results with RANSAC applied. The comparison highlights the improvement in epipolar lines estimation when RANSAC is employed, emphasizing its benefits in fundamental matrix estimation tasks.

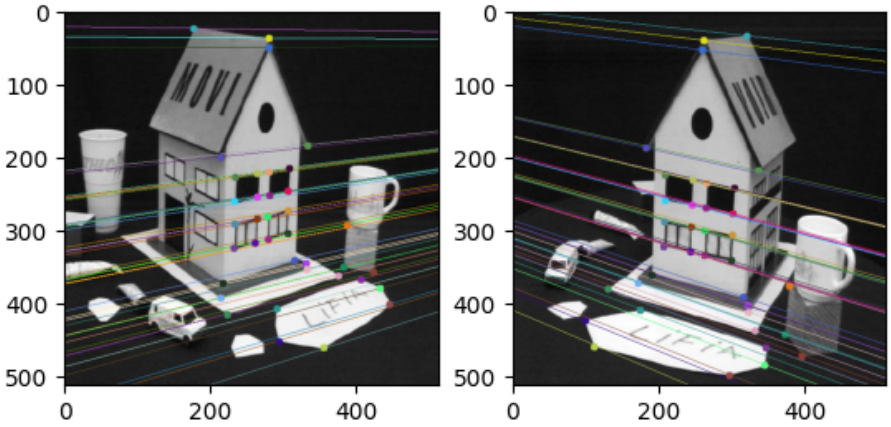

Figure 3: The image shows the visualization of epipolar lines without Ransac.

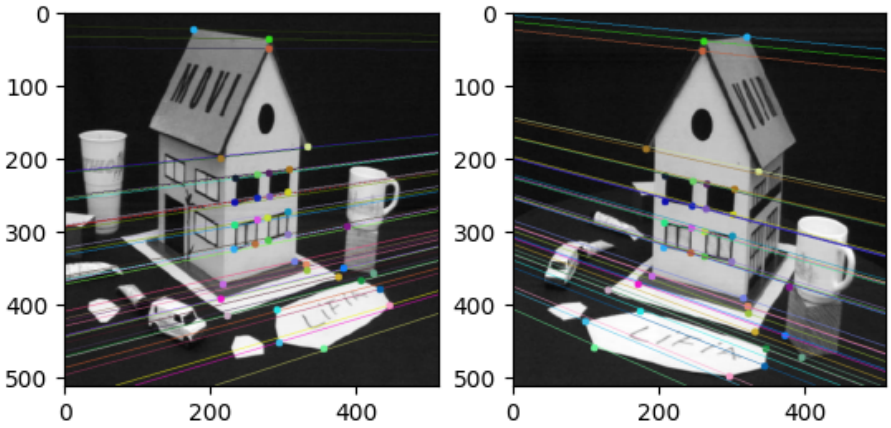

Figure 4: The image shows the visualization of epipolar lines with Ransac.

4.2 Test 2: Estimation Error

In this subsection, we analyze the results of Test 2, which focus on the estimation error. The error is calculated as the mean epipolar distance, which represents the average distance from each point to its corresponding epipolar line. The results indicate that the mean epipolar distance for the fundamental F_denorm matrix is 0.9336, while the distance for the RANSAC-enhanced fundamental F_ransac matrix is 0.9505.

Although the error of F_ransac is slightly higher than that of F_denorm in this particular case, RANSAC is generally known for its robustness to noise and outliers. In scenarios with higher noise levels, RANSAC is expected to outperform the standard method. The current results do not undermine the effectiveness of RANSAC, as its performance may still be superior in the presence of increased noise or outliers.

```
In [35]:
1 avg_distance_F_denorm = avg_epipolar_dist(kps_1, kps_2, F_denorm)
2 print("Average epipolar distance in F_denorm:", avg_distance_F_denorm)

Average epipolar distance in F_denorm: 0.9335650623616543
```

Figure 5: The image shows the error when calculating the average epipolar distance without Ransac.

```
In [36]:
1 avg_distance_F_ransac = avg_epipolar_dist(kps_1, kps_2, F_ransac_denorm)
2 print("Average epipolar distance in F_ransac:", avg_distance_F_ransac)

Average epipolar distance in F_ransac: 0.9505061565558812
```

Figure 6: The image shows the error when calculating the average epipolar distance with Ransac.

4.3 Test 3: Performance with Noise

4.3.1 Adding Noise to Pixel Coordinates and Effect of Noise on Estimation Error with and without RANSAC

The estimation errors for both methods, with and without RANSAC, are shown for various noise standard deviations (std_values) in the following results:

From the results, we can observe that as the noise level (standard deviation) increases, the estimation errors for both methods also increase. However, the method with RANSAC demonstrates a slightly better performance in handling higher noise levels compared to the method without RANSAC.

```
In [38]:
print("Errors without RANSAC:", errors_no_ransac)
print("Errors with RANSAC:", errors_ransac)

Errors without RANSAC: [0.9335650623616543, 1.06547029750088, 1.2288395448379674, 1.8098986773299528, 2.8062284994689333]
Errors with RANSAC: [0.9505061565558812, 1.1054399363436083, 1.3863189297663652, 1.6671796959309177, 2.7701327685547588]
```

Figure 7: The errors when calculating the average epipolar distance with and without ransac after adding noise to the pixel coordinates, zero-mean Gaussian noise with an increasing standard deviation value (std_values) of [0, 0.5, 1, 1.5, 2].

When noise is low (std_values = 0, 0.5), the estimation errors are similar for both methods. However, as the noise level increases (std_values = 1, 1.5, 2), the estimation error for the method without RANSAC increases at a faster rate than the method with RANSAC. This indicates that RANSAC is more robust against the noise in the data, as it can better handle outliers caused by noise.

In conclusion, using RANSAC for fundamental matrix estimation improves the robustness against noise, resulting in a more accurate estimation, especially when the noise level is high

5 Part IV - Estimation of the Fundamental Matrix with Detected and Described Features

5.1 Feature Detection and Extraction using SIFT

In this subsection, we implement feature detection and extraction using SIFT. The code creates a SIFT object with a specified contrast threshold and computes keypoints and descriptors for both grayscale images.

```
sift = cv2.SIFT_create(contrastThreshold=0.04)
kp1, des1 = sift.detectAndCompute(grayscale_left, None)
kp2, des2 = sift.detectAndCompute(grayscale_right, None)
```

5.2 Matching Descriptors using Brute Force Matching

In this subsection, we match descriptors using Brute Force Matching. The code creates a BFMatcher object and performs k-Nearest Neighbors matching between the two sets of descriptors.

```
bf = cv2.BFMatcher()
matches = bf.knnMatch(des1, des2, k=2)
```

5.3 Outlier Removal using SIFT Ratio and Cross Validation Tests

In this subsection, we perform outlier removal using SIFT ratio and cross validation tests. Before the removal, matches are drawn between the two images. The ratio test is then applied, followed by cross validation to further refine the matches. After the outlier removal, the improved matches are visualized.

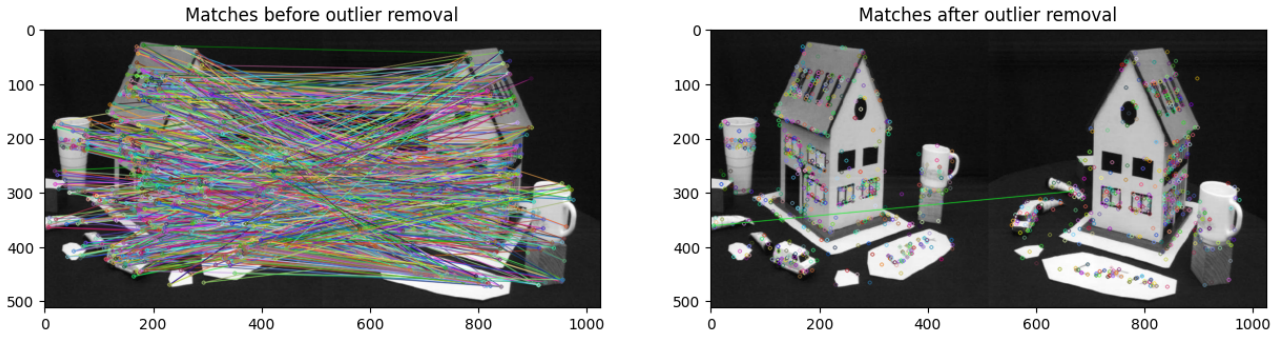

Figure 8: Comparison between matches before and after outlier removal.

5.4 Comparison and Improvement

When comparing the results of using ground-truth correspondences and SIFT correspondences, we can see that the average epipolar distance error is much higher in the case of SIFT. This indicates that the correspondences estimated by SIFT are not as accurate as the ground-truth correspondences.

```
In [46]:  1 avg_distance_F_SIFT = avg_epipolar_dist_sift(kp1, kp2, F_ransac_denorm_sift, matches_to_use)
          2 print("Average epipolar distance in F_denorm sift :", avg_distance_F_SIFT)

Average epipolar distance in F_denorm sift : 85.42800259780047
```

Figure 9: The image shows the error when calculating the average epipolar distance with the SIFT process.

To improve this result, we could consider the following approaches:

- Image Rectification: Image rectification is the process of transforming images so that epipolar lines become horizontally aligned. This makes finding correspondences easier and can reduce the error in the fundamental matrix estimation. We can use the cv2.stereoRectifyUncalibrated() function to rectify the images before computing correspondences and the fundamental matrix.
- Projective Transformation: If the images have significantly different perspectives, we might consider applying a projective transformation to align the images before computing correspondences and the fundamental matrix. We can use the cv2.findHomography() function to find the homography matrix and cv2.warpPerspective() to apply the projective transformation to the images.