Computer Vision
Model Fitting

Student: Yuchang Jiang

Date: 28 Oct. 2020

1. Line Fitting

1.1 Implementation

The minimal number of points required for this problem is two so the first step in *ransacLine.m* is to randomly select two point among data. Considering the line as $y = kx + b$, these two parameters can be solved as: $k = \frac{y_2 - y_1}{x_2 - x_1}$ and $b = y_2 - kx_2$. Then the orthogonal (shortest) distance between data points and fitted line are computed for all points. For distance calculation, this formula is used: $distance = \frac{|kx+b-y|}{\sqrt{k^2+1}}$. Then based on chosen threshold, points with distance smaller than threshold are selected as inliers. Among all iterations, parameters, k, b from iteration with the largest number of inliers are selected. Here I used 300 iterations with threshold equals to 3.

1.2 Result and Discussion

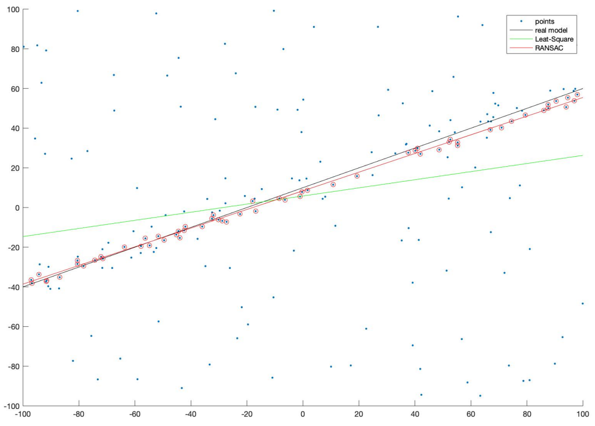

Figure 1: Result of RANSAC for line fitting

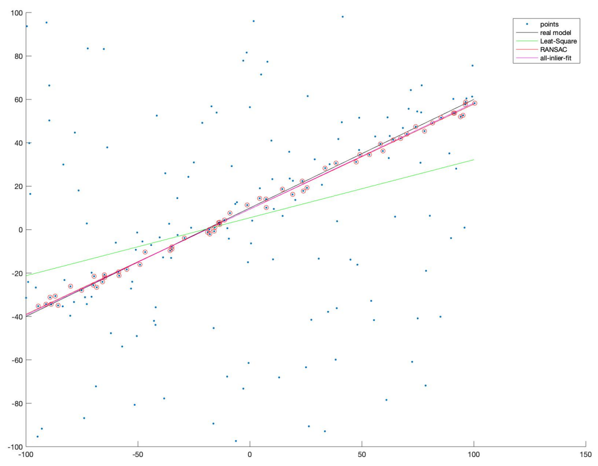

Figure 2: Result of RANSAC for line fitting (use all inliers)

As shown in Figure 1, the resulted line from RANSAC, least squared methods and the real line are plotted. It is obvious that RANSAC line is closer to real line and less affected by noisy points. This solution of RANSAC is decided by two randomly selected data points. In Figure 2, all inliers from the RANSAC solution that has the most inliers are fitted to find line. This line decided by all inliers is very similar to the line decided by only two points, which proves the usefulness of RANSAC. The quantitative result is shown in Table 1: it is clear least square method is largely affected by outliers and has a large error while RANSAC method has a small error (close to the error of real line).

error(real)	error(least-square)	error(RANSAC)	error(all inlier)
35.8304	158.4140	38.7928	35.8255

Table 1: Table of error comparison

2. Fundamental matrix estimation

2.1 Implementation

To solve fundamental matrix, at least eight points are selected manually and they are normalized: at first, calculate centroid and shift all points towards centroid.

$$x_{centroid} = \sum_{i=1}^{N} x_i \qquad y_{centroid} = \sum_{i=1}^{N} y_i \tag{2.1}$$

where N is the number of data points. With centroids, T', can be formed:

$$T' = \begin{bmatrix} 1 & 0 & -x_{centroid} \\ 0 & 1 & -y_{centroid} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

With this matrix, all data points can be shifted towards centroid: $xy_{shifted} = T' \cdot xy$. Then scale factor can be obtained as $s_{xy} = \frac{\sqrt{2}}{xy_{meandist}}$ and final transition matrix can be formed: $T = s_{xy} \cdot T'$. Then data points can be transformed: $xy_{normalised} = T \cdot xy$.

To solve fundamental matrix, A matrix in $Af = 0$ is formed and each row in A matrix uses the correspondence pair of points: $[x_1x_2, x_1y_2, x_1, y_1x_2, y_1y_2, y_1, x_2, y_2, 1]$. Then singular value decomposition is performed on A and the last column of V is taken as f, which can be reshaped into 3x3 fundamental matrix. To enforce the singularity of fundamental matrix, the last singular value of F matrix is replaced by 0 to form new F matrix. With fundamental matrix, the epipoles and epipolar lines can be computed accordingly: $Fe = 0, e'^T F = 0, l' = Fx, l = F^T x'$.

2.2 Result and Discussion

Three datasets: ladybug, rect and pumpkin are used to solve fundamental matrix. As shown from Figure 3 to Figure 8, clicked points, epipoles and epipolar lines based on singular (blue lines) and non-singular (green lines) F matrix are plotted. For ladybug dataset, it is clear that camera was moving along the optical axis and the epipole is in the center of image. The epipolar lines from singular and non-singular matrix are overlapped a lot and most clicked points lie on epipolar lines. As for epipole, it seems to be the intersection of blue lines (epipolar lines of singular F) so it is important to enforce singularity.

As for other two datasets, rect and pumpkin, the result is worse than ladybug. For rect, the clicked points do not lie on epipolar lines and epipolar lines from two F matrix do not overlap. This can be caused by errors of manual point selection: it is harder to find corresponding points in rect than ladybug. The similar problem also appears in pumpkin dataset. This indicates that the accuracy of corresponding points are essential for fundamental matrix and a bad pair can impose large influence, which makes RANSAC necessary for this case.

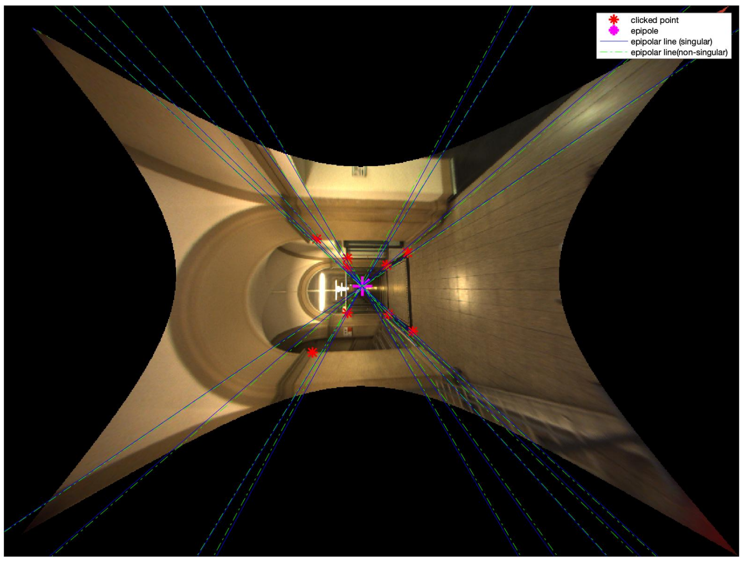

Figure 3: Result of image 1 in ladybug

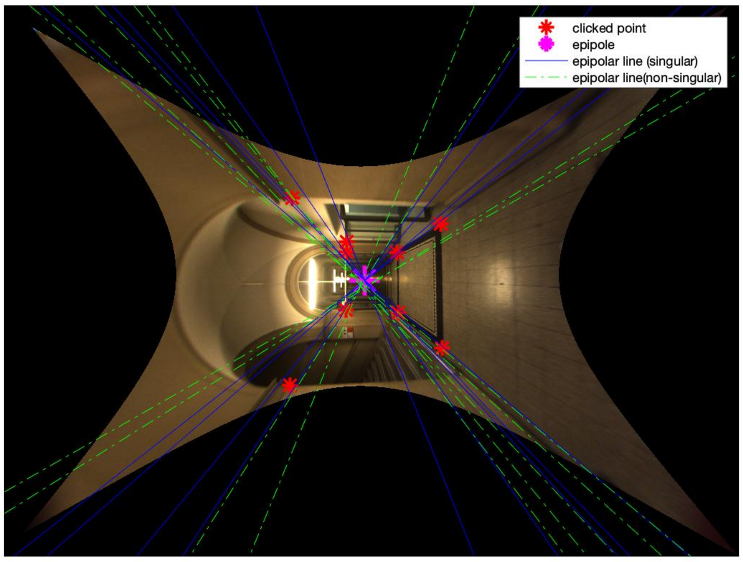

Figure 4: Result of image 2 in ladybug

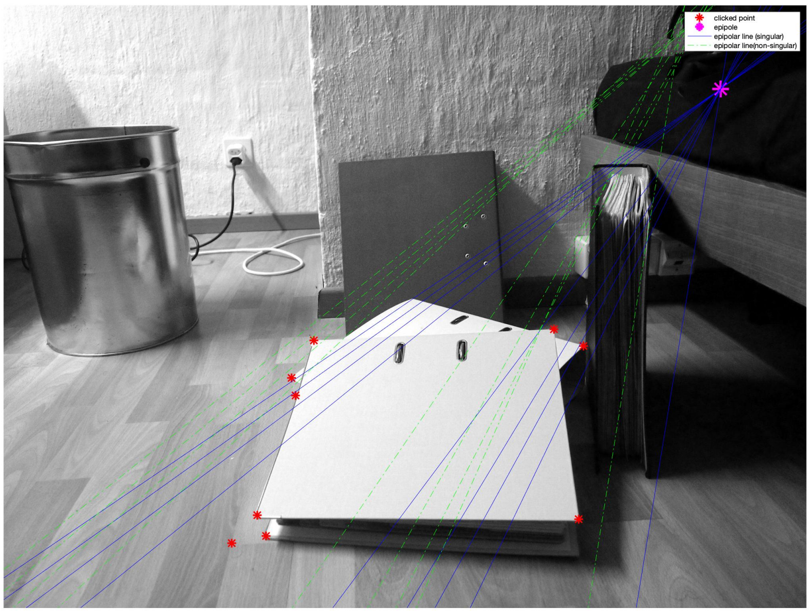

Figure 5: Result of image 1 in rect

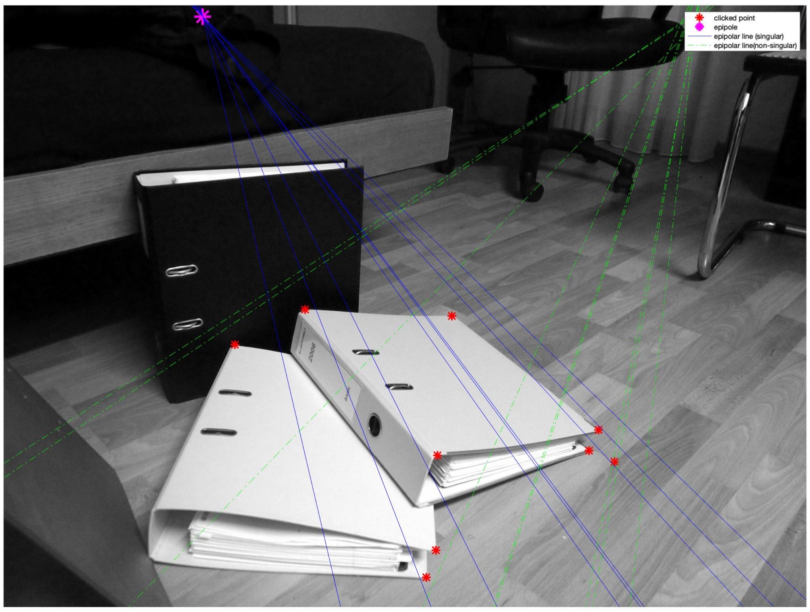

Figure 6: Result of image 2 in rect

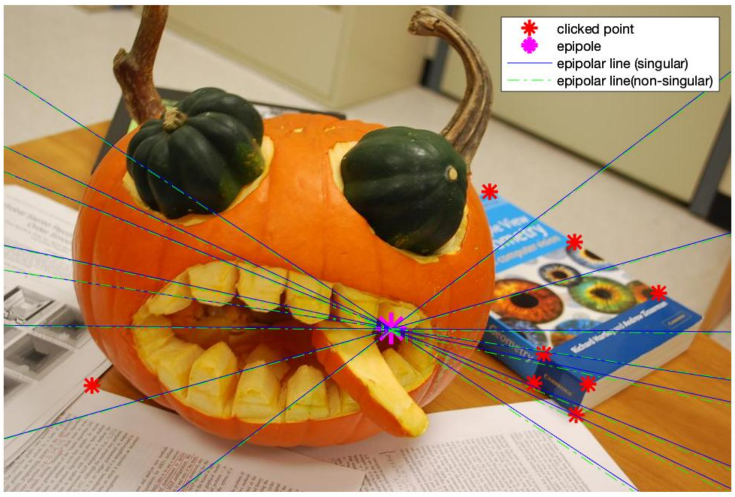

Figure 7: Result of image 1 in pumpkin

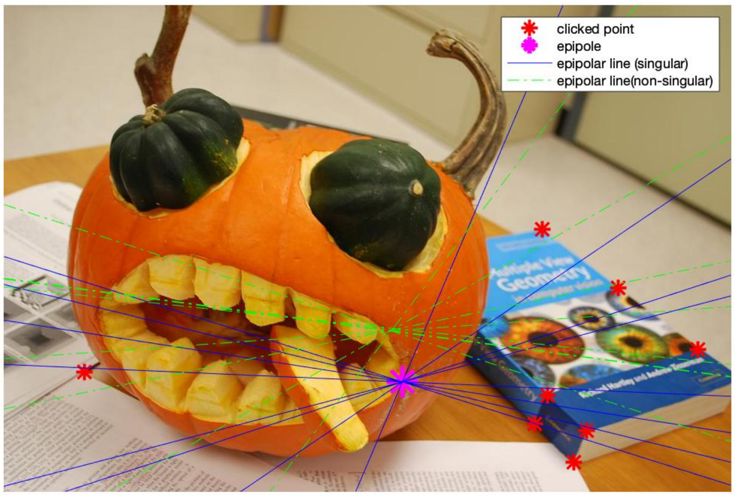

Figure 8: Result of image 2 in pumpkin

3. Feature extraction and matching

3.1 Implementation

Here I just downloaded VLFeat package and extracted features for three dataset: laydbug, rect and pumpkin.

3.2 Result and Discussion

As shown from Figure 9 to Figure 11, there are many bad matches/outliers for feature extraction result. Thus, RANSAC is needed when solving fundamental matrix based on matches.

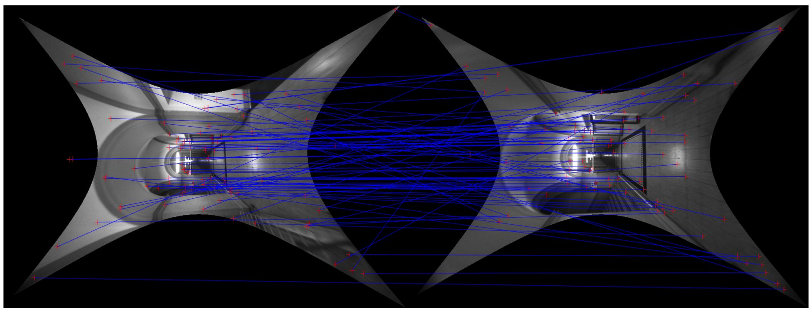

Figure 9: Result of matches for ladybug

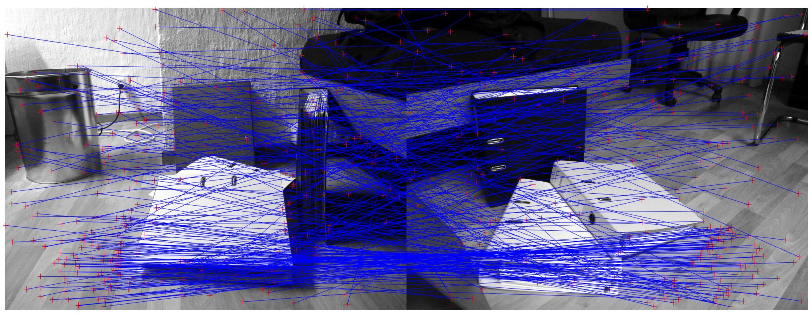

Figure 10: Result of matches for rect

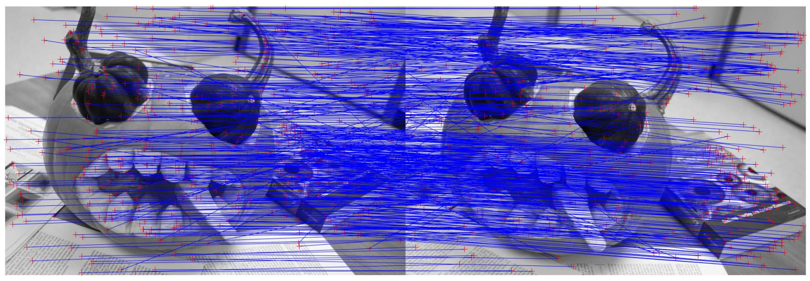

Figure 11: Result of matches for pumpkin

4. Eight-point RANSAC

4.1 Implementation

Here two kinds of RANSAC are tried: simple RANSAC and adaptive RANSAC.

4.1.1 Simple RANSAC

At the first step, the minimal number of pairs required (8 in this case) is randomly selected. Then fundamental matrix is computed using function implemented in the 'Fundamental matrix estimation' problem. Then for points in image 1 and 2, compute their orthogonal(shortest) distance to resulted epipolar lines and. The final distance is defined as the average of distances in two images. Then filter out inliers based on determined threshold and count inlier number. The solution with most inliers is chosen. Here 1000 iterations is used for RANSAC loop.

4.1.2 Adaptive RANSAC

Instead of using fixed 1000 iterations, adaptive RANSAC will terminate iterations if the probability of having at least one inlier among random sample is larger than a threshold, for example 99% so in the loop, outlier ratio, probability of having at least one inlier among random sample and the sample count will be computed in each iteration. This is generally more efficient than the simple version. (Code is in *ransac8pF_adaptive.m*)

4.2 Result and Discussion

The threshold is a key parameter for RANSAC so experiments with various threshold is tried out for simple RANSAC(dataset pumpkin is chosen for this experiment). As shown in Table 2, a larger threshold will allow more points counted as inliers, which will lead to a larger mean error of inliers. As for experiments with all datasets, the threshold = 5 is chosen and the corresponding visualizations with highest RANSAC score are shown in

Figure 12, Figure 13 and Figure 14. Comparing to feature extraction result in the last problem, there are fewer matches and also fewer obvious outliers.

Threshold	total inlier count (N)	mean error of inliers (dist)	$\frac{N}{dist}$
1	37	0.4535	81.5905
3	96	1.4165	67.7732
5	111	2.5980	42.7244
9	202	4.2188	47.8804
20	304	9.5707	31.7637

Table 2: Table of varying threshold

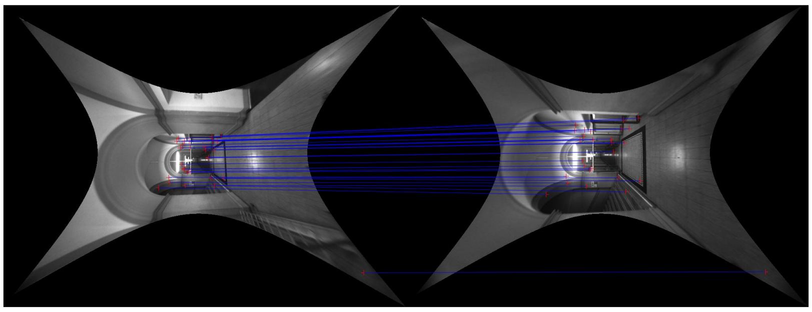

Figure 12: Result of simple RANSAC for ladybug

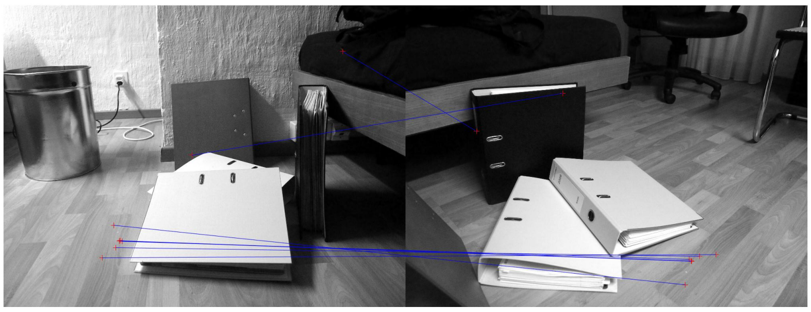

Figure 13: Result of simple RANSAC for rect

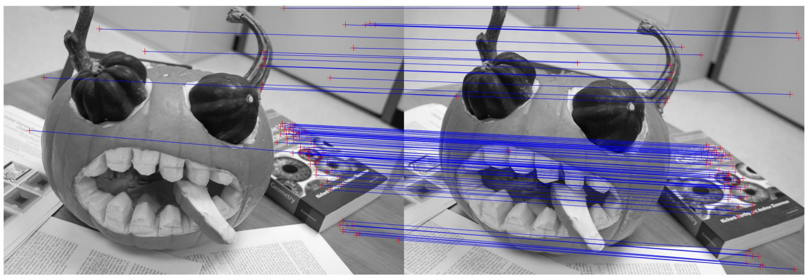

Figure 14: Result of simple RANSAC for pumpkin

As for adaptive RANSAC, threshold = 5 is used for three datasets. The resulted number of trials for each dataset is summarized in Table 3. It is noticed that the number of trials has a positive relationship with the outlier ratio. The dataset, rect, has the highest outlier ratio and largest number of trials. Because the relative motion in ladybug and pumpkin seems to be orthogonal and parallel movement but the motion in rect is more complicated, the difficulty to find good matches is much larger for rect. These resulted number of trials also suggests that adaptive RANSAC may not be more efficient than simple RANSAC in certain cases.

The resulted plots of matches for adaptive RANSAC are shown in Figure 15, Figure 16 and Figure 17. Generally, there are more matches in the result of adaptive RANSAC than the simple version.

	ladybug	rect	pumpkin
number of trials	4218	761138	10042

Table 3: Table of result of adaptive RANSAC

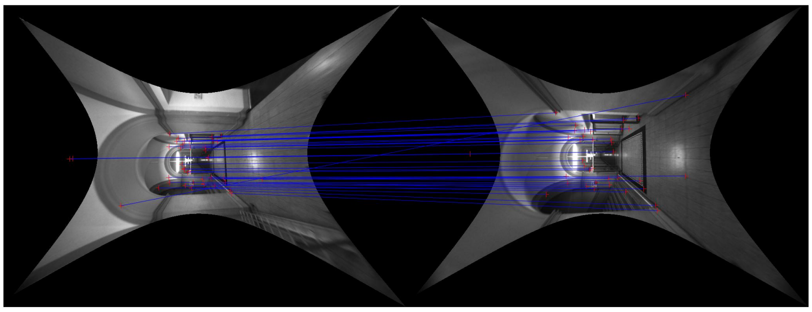

Figure 15: Result of adaptive RANSAC for ladybug

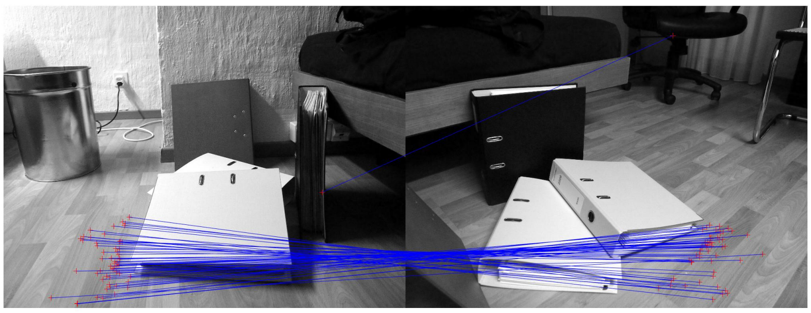

Figure 16: Result of adaptive RANSAC for rect

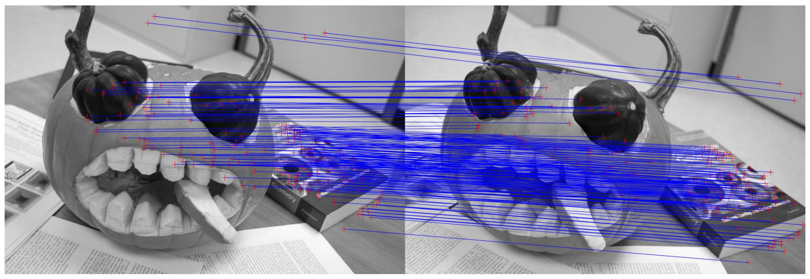

Figure 17: Result of adaptive RANSAC for pumpkin