Dongheng Lin
Feb 25th, 2024

CS 549: Computer Vision
Assignment 1

1 Image Stitching (Python)

1.1 Putative Matches

We started by detecting key points in both images using the Scale-Invariant Feature Transform (SIFT) algorithm provided by OpenCV. Specifically, we used the provided functions *cv2.SIFT_create()* for initializing the SIFT detector and *detectAndCompute* for finding keypoints and computing descriptors. Then we compute the Euclidean distance between every descriptor in one image to every descriptor in the other using the *scipy.spatial.distance.cdist* function with 'sqeuclidean' as the metric. This step identifies potential matches by evaluating how similar the feature descriptors are across the two images.

After computing distances between every descriptor in one image to every descriptor in the other, the next step is to filter these to identify the most promising matches, known as putative matches. This filtering can be done based on a distance threshold or by selecting the top N matches with the smallest distances. The code snippet *get_best_matches(img1, img2, num_matches)* for finding the best matches does the following:

- Sorting and Selecting Matches: The distances between descriptors are sorted, and the indices of the sorted distances are used to select the top N matches. This is achieved using *np.argsort* on the flattened distance matrix, followed by *np.unravel_index* to convert flat indices to pairs of indices, and finally selecting the top N pairs.
- Constructing Match Pairs: For each selected pair of indices, the corresponding keypoints from both images are retrieved. The keypoints are then used to form a match, with each match represented by the (x, y) coordinates of the corresponding keypoints in both images.
- Given that the images pairs may have different deviations to each other, we picked top 300 keypoint pairs in each of the images instead of setting a hard threshold. The result of putative match pairs are visualized in Figure 1

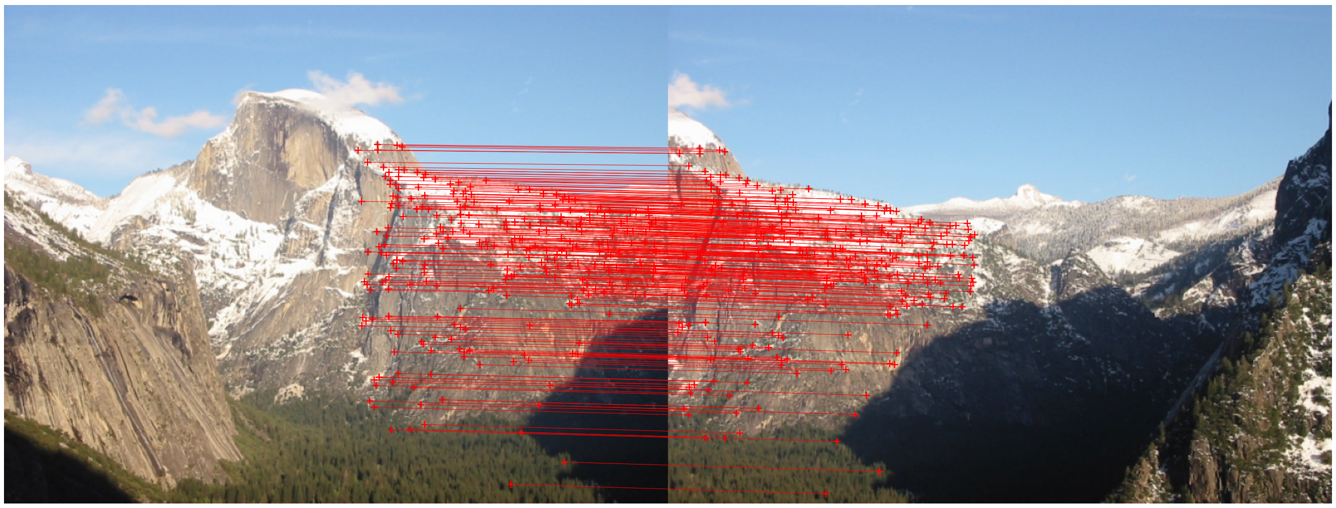


Figure 1: Putative Match Pairs

1.2 Homography Estimation and RANSAC

The Random Sample Consensus (RANSAC) algorithm is employed to robustly estimate a homography that maps points from one image to another. The implementation details of the corresponding function *ransac(matches, num_iterations, threshold, outliers_ratio)* are as follows:

- In each iteration, a set of random sample of matches is selected to estimate a homography matrix. The *compute_homography* function is called with these matches to compute the matrix based on the direct linear transform (DLT) algorithm, involving the construction of a system of equations and solving it using Singular Value Decomposition (SVD).
- For each match, the homography is used to transform the point from the first image and compare it to its corresponding point in the second image. Matches are considered inliers if the distance between the transformed point and its match is below a specified threshold.
- The homography that results in the highest number of inliers is selected as the best model. The RANSAC loop includes mechanisms to break early if an exceptionally good model is found or to continue iterating to possibly find a better model.

Under hyperparameters of *num_iterations = 1000, threshold = 0.5, outliers_ratio = 0.5* we got following results as it is visualized in Figure 2, and quantitative results are in TABLE.

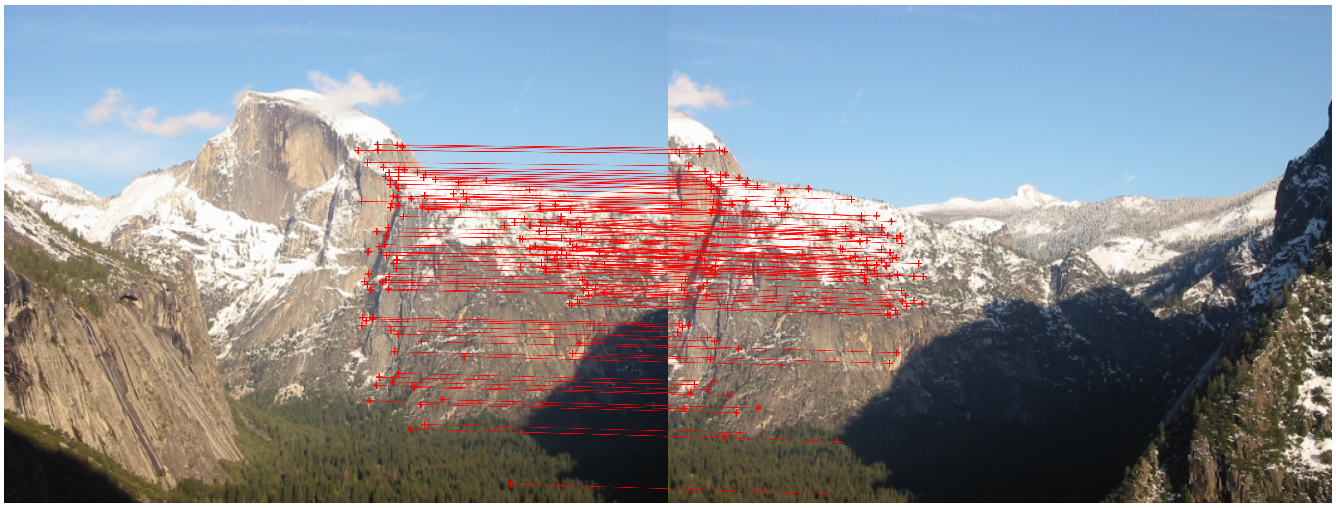


Figure 2: RANSAC Matched Pairs

1.3 Image Warping

With the estimated homography, one image can be warped onto the perspective of the other, creating a panoramic stitch. The warping process adjusts the geometry of one image to align with the other, based on the homography matrix. The *warp_images* function computes the size of the output panorama to accommodate both images. It then warps one of the images using the estimated homography and overlays it onto the other image. The function calculates the corners of both images in the panorama's coordinate system, ensuring that the entire content of both images is visible. A translation matrix *H_translation* is applied to adjust the position of the warped image, ensuring no part of it is outside the panorama's bounds. Finally, the images are composited into a single output, with the first image placed onto the adjusted (warped and translated) second image.

1.4 Extra Credit

To complete this task, we implemented another *stitch_images* function extends the stitching process to handle three images. It involves detecting and matching features across all three images and sequentially estimating homographies

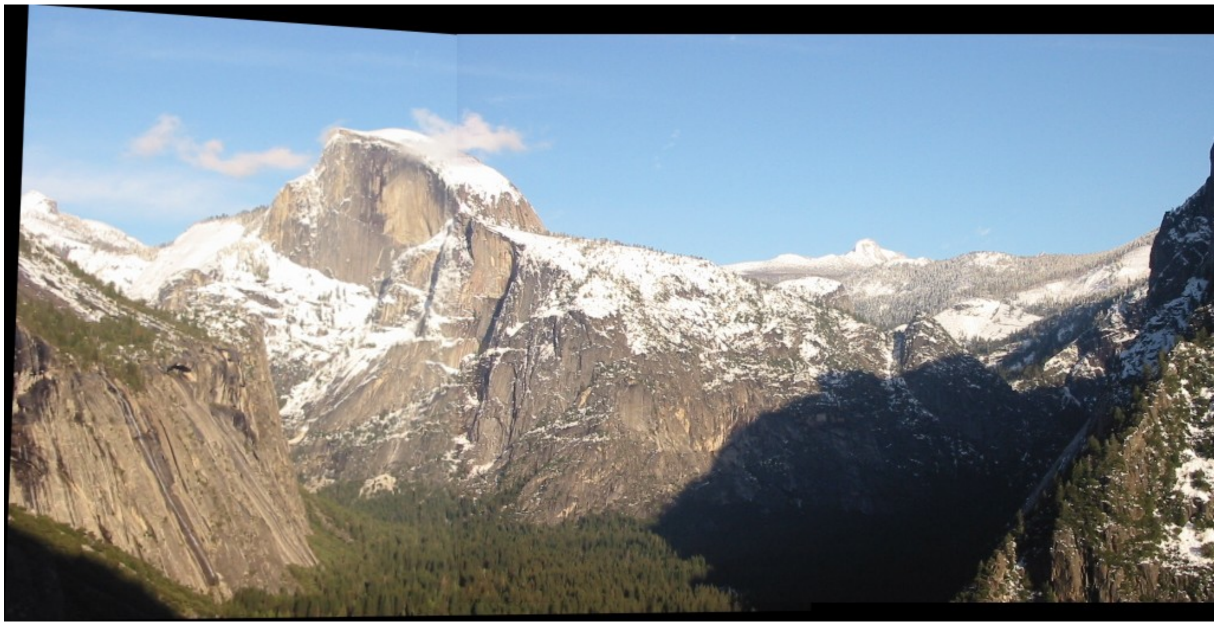


Figure 3: RANSAC Stitiched Images

between consecutive pairs of images. Good matches are filtered using a Lowe's ratio test to ensure robustness. Specifically, this test is a method for filtering out less reliable keypoint matches based on the ratio of distances between the best and second-best match. We empirically set the ratio to be 0.75.

The function also calculates the cumulative homography for each image relative to the first and warps them into a common coordinate system, creating a single panoramic image. Adjustments are made to accommodate the dimensions of the resulting panorama and to ensure that all images are correctly positioned within it.

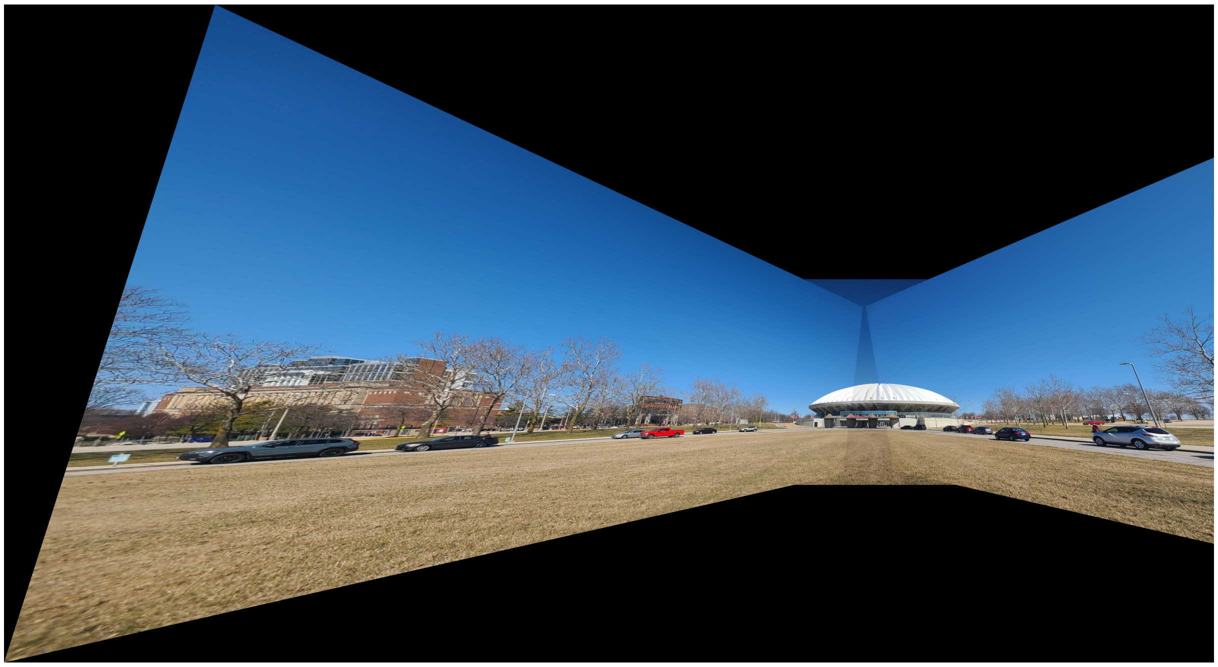


Figure 4: State Farm Stitich

2 Fundamental Matrix Estimation, Camera Calibration, Triangulation

2.1 Fundamental Matrix Estimation

We implemented 2 types of Fundamental Matrix Estimation which are unormalized and normalized version.

The unnormalized algorithm directly uses the input point correspondences to construct a system of equations that, when solved, yields the fundamental matrix. Specifically, for each pair of matching points, a row is added to matrix A that encodes the epipolar constraint. The fundamental matrix, F, is then estimated by solving the equation $AF = 0$ using Singular Value Decomposition (SVD). This process involves decomposing matrix A into U, S, and V, where F is found as the last column of V, reshaped to a 3×3 matrix. However, F must satisfy a rank-2 constraint, necessitating a further SVD of F itself, zeroing the smallest singular value in S, and recomputing F. This method, while straightforward, can be sensitive to noise since it doesn't normalize the input data.

The normalized algorithm, on the other hand, pre-processes the input point correspondences to improve the numerical stability and robustness of the estimation process. This normalization involves translating and scaling the points in both images such that their centroid is at the origin and their average distance from the origin is $\sqrt{2}$. After this preprocessing step, the unnormalized algorithm is applied to estimate F. Finally, F is denormalized to map back to the original image coordinates. This normalization step is critical for reducing the sensitivity of the estimation process to noise and improving the accuracy of the estimated fundamental matrix.

Here are the corresponding outputs for each version of the function, as it is shown, we find that the normalized version is generally more robust as it have smaller residual.

Table 1: Comparison of Residuals for Non-Normalized and Normalized Methods

Metric	**Non-Normalized Method**	**Normalized Method**
Residual in Frame 1	0.1491	0.0595
Residual in Frame 2	0.1792	0.0667
Combined Residual	0.1642	0.0631

$$\text{Non-Normalized Fundamental Matrix} = \begin{bmatrix} -1.32341616 \times 10^{-6} & 1.36640519 \times 10^{-5} & -6.82803870 \times 10^{-4} \\ -2.88178174 \times 10^{-5} & 2.66440807 \times 10^{-7} & 4.09069255 \times 10^{-2} \\ 5.62362952 \times 10^{-3} & -3.72771609 \times 10^{-2} & -9.98451273 \times 10^{-1} \end{bmatrix}$$

$$\text{Normalized Fundamental Matrix} = \begin{bmatrix} -1.05691112 \times 10^{-7} & 1.34452638 \times 10^{-6} & -1.15723794 \times 10^{-4} \\ -3.33385198 \times 10^{-6} & 2.55967775 \times 10^{-8} & 5.06486890 \times 10^{-3} \\ 7.01855207 \times 10^{-4} & -4.60282534 \times 10^{-3} & -1.20958631 \times 10^{-1} \end{bmatrix}$$

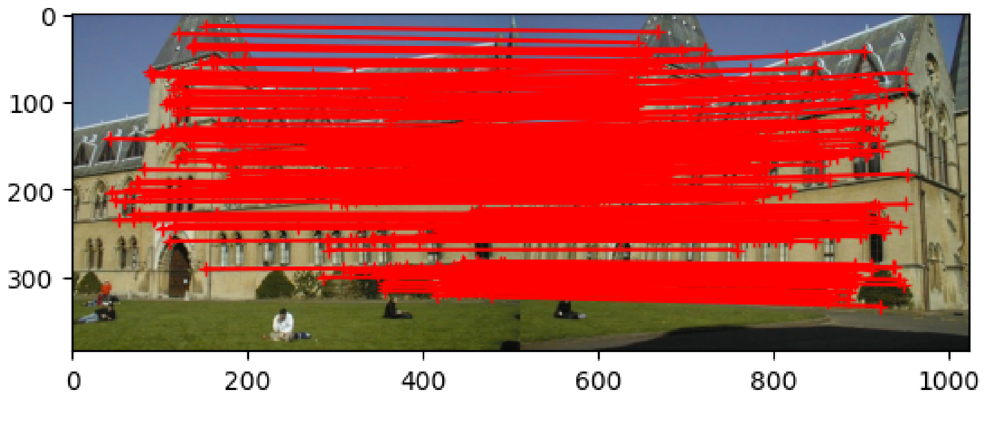


Figure 5: Matched Points Provided

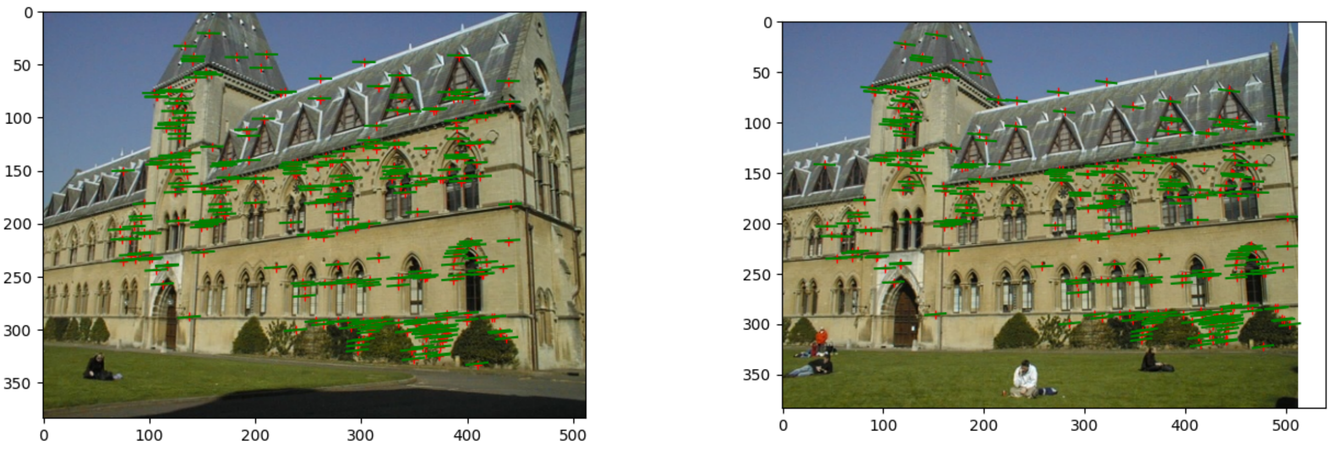


Figure 6: Unnormalized Fundamental Matrices

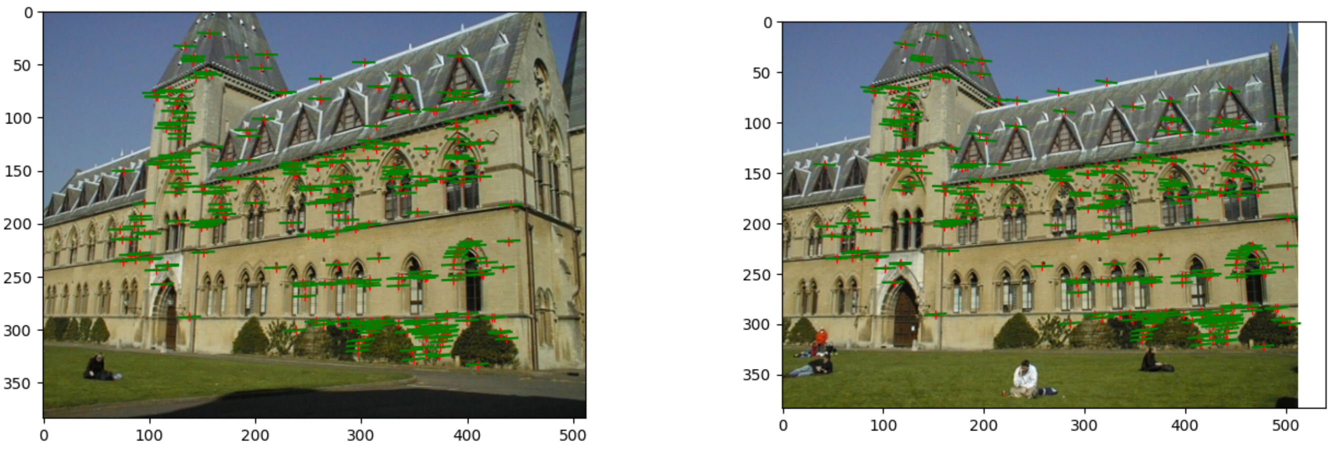


Figure 7: Normalized Fundamental Matrices

2.2 Camera Calibration

For the camera calibration, we estimated the camera projection matrices using 2D-3D point correspondences. The estimated camera projection matrices for the lab dataset are as follows:

Lab 1 Camera Projection Matrix:

$$\begin{bmatrix} 3.09963996e-03 & 1.46204548e-04 & -4.48497465e-04 & -9.78930678e-01 \\ 3.07018252e-04 & 6.37193664e-04 & -2.77356178e-03 & -2.04144405e-01 \\ 1.67933533e-06 & 2.74767684e-06 & -6.83964827e-07 & -1.32882928e-03 \end{bmatrix}$$

Lab 2 Camera Projection Matrix:

$$\begin{bmatrix} 6.93154686e-03 & -4.01684470e-03 & -1.32602928e-03 & -8.26700554e-01 \\ 1.54768732e-03 & 1.02452760e-03 & -7.27440714e-03 & -5.62523256e-01 \\ 7.60946050e-06 & 3.70953989e-06 & -1.90203244e-06 & -3.38807712e-03 \end{bmatrix}$$

The residual errors for both cameras are well below the threshold of 20, indicating a good fit of the projection matrices to the observed data.

- residual in lab1: 13.54583289329173
- residual in lab2: 15.544953452595896

2.3 Camera Centers

The camera centers were calculated from the projection matrices. The camera centers were determined by calculating the null space of the camera projection matrix. The 3D locations of the camera centers for both lab and library datasets are reported below:

Lab 1 Camera Center: $[305.83, 304.20, 30.14]$
Lab 2 Camera Center: $[303.10, 307.18, 30.42]$
Library 1 Camera Center: $[7.29, -21.52, 17.73]$
Library 2 Camera Center: $[6.89, -15.39, 23.41]$

2.4 Triangulation

Triangulation was used to reconstruct the 3D positions from matching pairs of 2D points. The function *riangulation(matches, proj1, proj2)* constructs a system of equations based on the camera projection matrices and the 2D point correspondences. This system essentially represents the geometric constraints that any point in 3D space projected onto the camera's image plane must satisfy. The system is set up as follows:

- For each point match, four equations are constructed (two from each camera view). These equations relate the 3D point's coordinates to its 2D projections using the camera projection matrices.
- The equations for each point are derived from the projection equation $x = PX$, where x is the homogeneous 2D point in the image, P is the camera projection matrix, and X is the homogeneous 3D point in the world.
- The system of equations for each point is over-determined (there are more equations than unknowns), so the function solves it using Singular Value Decomposition (SVD). SVD is a technique that can solve linear equations, including over-determined systems, by finding the least squares solution that minimizes the error (the difference between the observed projections and those predicted by the 3D point estimate).

The results are as follows:
Mean 3D reconstution error for the lab data: 0.00025
2D reprojection error for the lab 1 data: 10.899446020058848
2D reprojection error for the lab 2 data: 1.5485148022902966
2D reprojection error for the library 1 data: 24.662071196868606
2D reprojection error for the library 2 data: 28.649537735260804

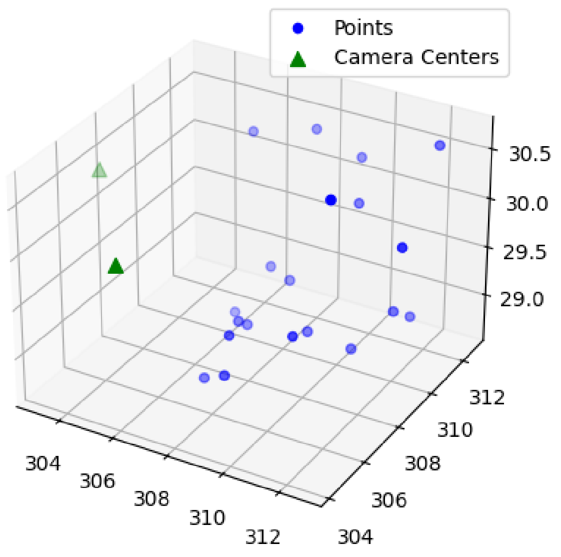


Figure 8: Triangulation of Lab

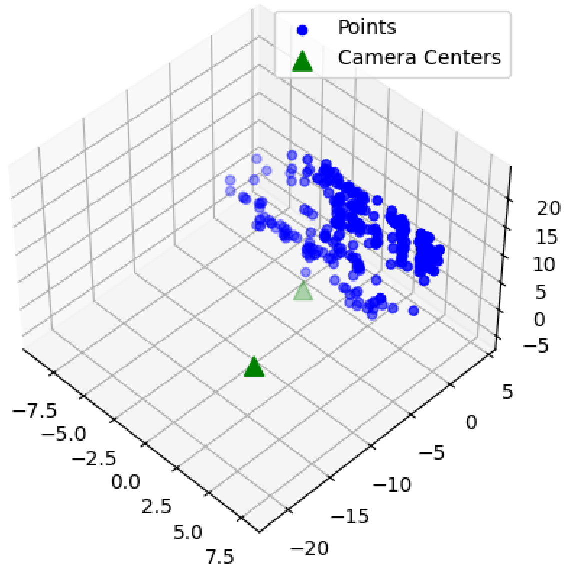


Figure 9: Triangulation of Library

2.5 Extra Credit

In this section, we apply a combination of RANSAC and SIFT to estimate the fundamental matrices without relying on ground-truth matches. The method involves detecting key points and computing their descriptors using SIFT, followed by identifying the best matches based on descriptor distances. These matches are then used with RANSAC to robustly estimate the fundamental matrix, effectively handling outliers and inaccuracies in the match selection process.

2.5.1 SIFT Feature Matching

Using the SIFT algorithm, we extracted and matched features between two images of a lab setup. This process involved computing the Euclidean distance between SIFT descriptors in both images and selecting the best matches based on these distances. The top matches were visualized to verify the accuracy of the matching process.

2.5.2 RANSAC for Robust Fundamental Matrix Estimation

RANSAC was employed to estimate the fundamental matrix using the matches obtained from the SIFT algorithm. The algorithm iteratively selected random subsets of matches, estimated the fundamental matrix, and then computed residuals to identify inliers. The process aimed to maximize the number of inliers while minimizing the residual error.

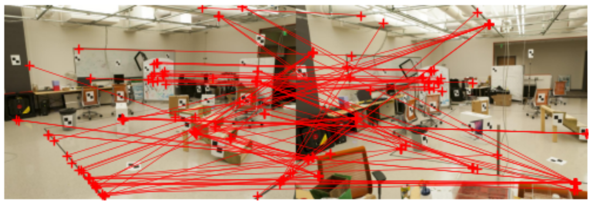
2.5.3 Results

The estimated fundamental matrix and the visualization of inliers demonstrate the effectiveness of the combined use of SIFT and RANSAC for feature matching and fundamental matrix estimation. The results are summarized below:

- Number of inliers: *[Include the number of inliers here]*
- Average residual for the inliers:
  - Lab Residual in Frame 1 (Normalized Method): 0.00726
  - Lab Residual in Frame 2 (Normalized Method): 0.00689
  - Lab Residual Combined (Normalized Method): 0.00707
  - lib residual in frame 1 (normalized method) = 0.07194691905949735
  - lib residual in frame 2 (normalized method) = 0.09127589331708358
  - lib residual combined (normalized method) = 0.08161140618829046

These results indicate a high degree of accuracy in the estimated fundamental matrices, with low residual errors suggesting a good fit to the observed data. The combination of SIFT and RANSAC proves to be a powerful tool for feature matching and fundamental matrix estimation in the absence of ground-truth matches.

The inlier matches and the effect of the estimated fundamental matrices were visualized in separate figures, illustrating the correspondence between points across the two images and the adherence of these points to the estimated epipolar geometry. The figures underscore the precision of our feature matching and fundamental matrix estimation process, more intermediate results (inliers selected, etc.) can be refered in the jupyter notebook submitted.



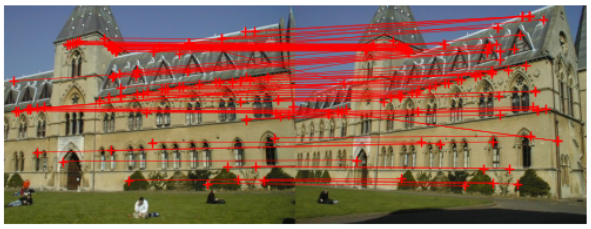


Figure 10: SIFT Visualization

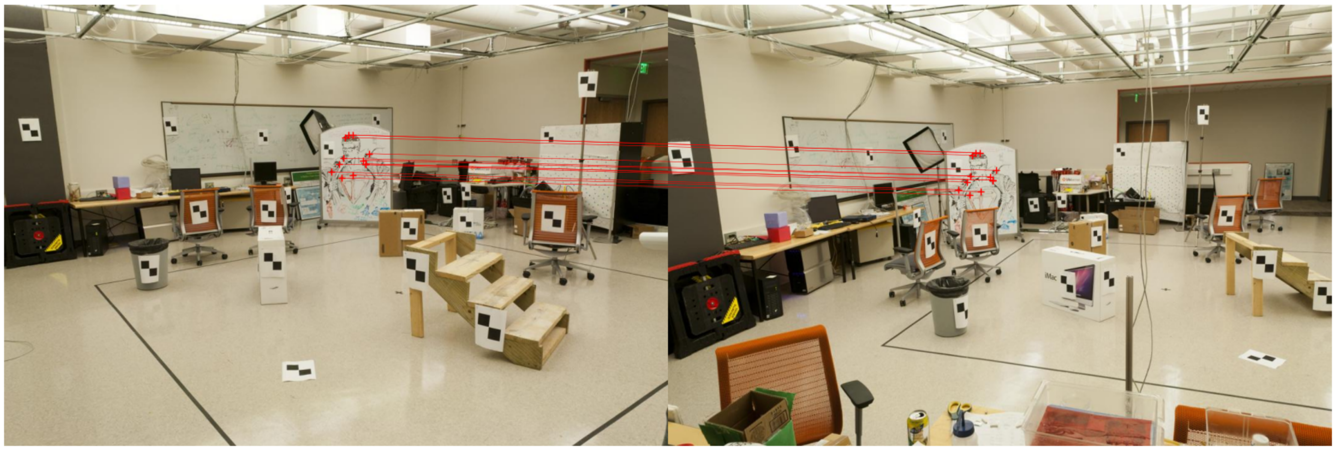


Figure 11: Lab Match Points Selected by RANSAC on Fundamental Matrices

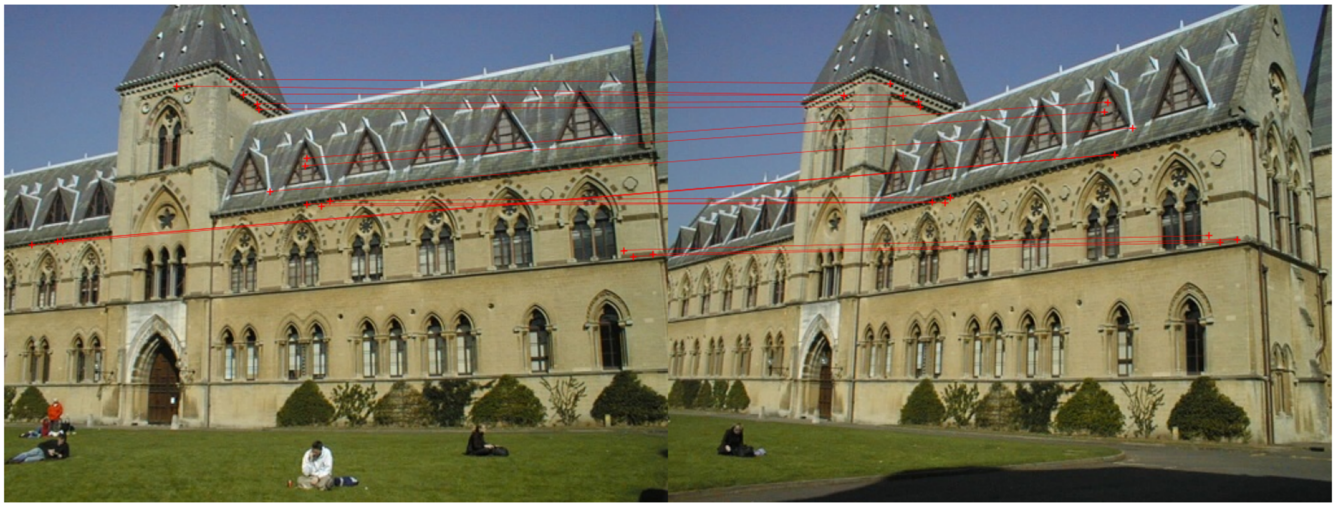


Figure 12: Library Match Points Selected by RANSAC on Fundamental Matrices

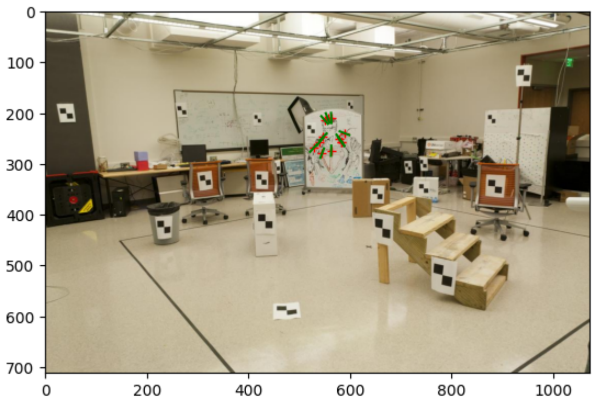
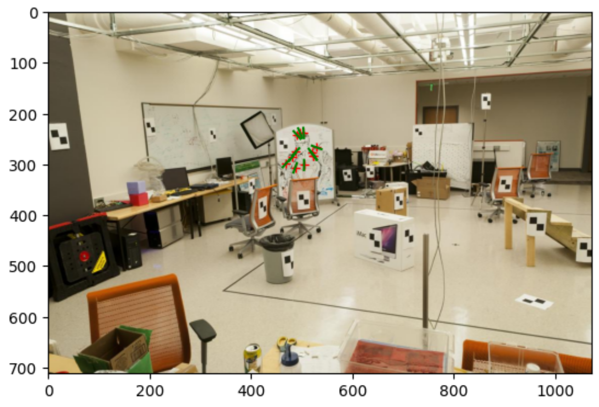



Figure 13: Residual Visualization of RANSAC Selected Match Points in Lab

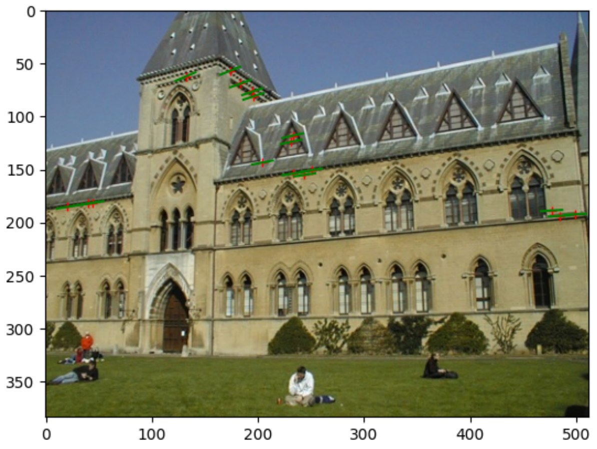
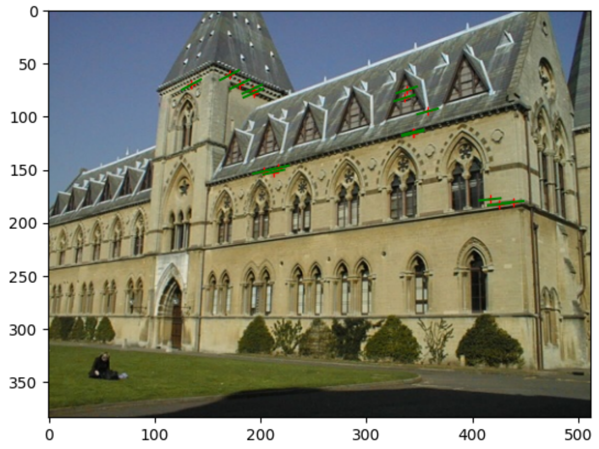



Figure 14: Residual Visualization of RANSAC Selected Match Points in Library

3 Single-View Geometry

3.1 Vanishing Points

We estimated three major orthogonal vanishing points based on the directions: vertical (Y-direction), towards the back of the building (Z-direction), and horizontal to the left (X-direction). For each direction, at least three lines were marked and used to compute their intersection using least squares, which represents the vanishing point. Mathematically, this involves setting up a system of equations based on the lines' equations and solving for the point of intersection. The computation takes advantage of the fact that the cross product of two lines (represented as vectors) in homogeneous coordinates yields a point (also in homogeneous coordinates) where the lines intersect.

The pixel coordinates for the vanishing points are as follows:

- X-direction Vanishing Point: (955.18, 94.56, 0.00103)
- Y-direction Vanishing Point: (-828.90, 559.37, 0.00384)
- Z-direction Vanishing Point: (113.72, 993.51, 0.00027)

The vanishing points and the lines used for their estimation are plotted on the image plane, as shown in the figures below.

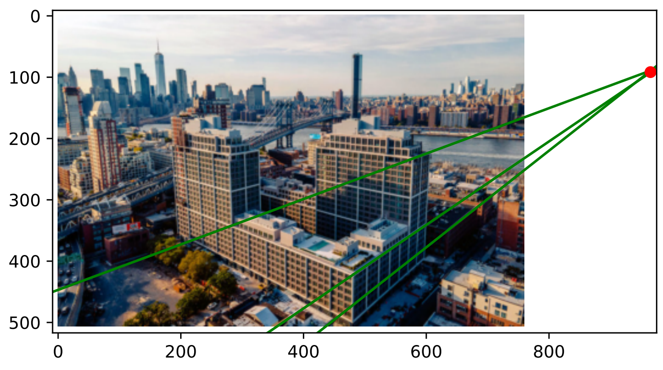


Figure 15: Lines and vanishing point in the X-direction.

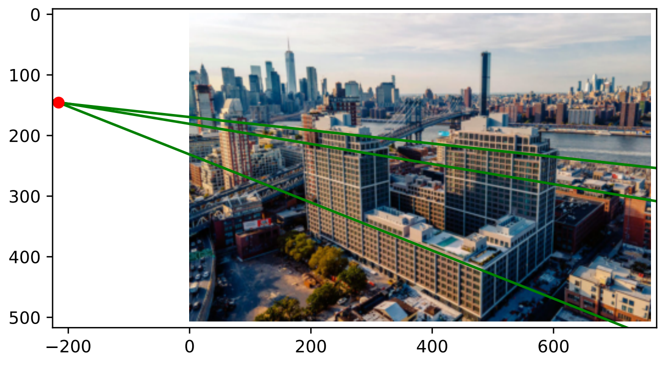


Figure 16: Lines and vanishing point in the Y-direction.

3.2 Horizon

The horizon line was estimated using the vanishing points corresponding to the horizontal directions (X and Z). By taking the cross product of these two vanishing points, we obtain the equation of the line in homogeneous coordinates that represents the horizon. This line is where the ground meets the sky in the image. The normalization of the line's parameters ensures that the equation conforms to the constraint $a^2 + b^2 = 1$ are reported as:

$$\text{Horizon line equation:} \; -0.046x - 0.999y + 135.682 = 0$$

From the Figure 18, we see that the estimated horizon is a bit deviated from the real horizon, which is due to the inaccuracy of manual annotation of the lines.

3.3 Camera Calibration

The calibration of the camera involved solving for the camera's intrinsic parameters, including the focal length and the optical center. This was achieved by leveraging the geometric property that the vanishing points corresponding to orthogonal directions in the scene are themselves orthogonal when projected onto the image plane. By formulating and solving a set of equations that express this orthogonality constraint in terms of the camera's intrinsic parameters, we were able to solve for the focal length and the principal point of the camera. The intrinsic matrix K was then constructed using these values.

- Focal length (f): 326654.63
- Optical center (u, v): (261.25, 218.98)

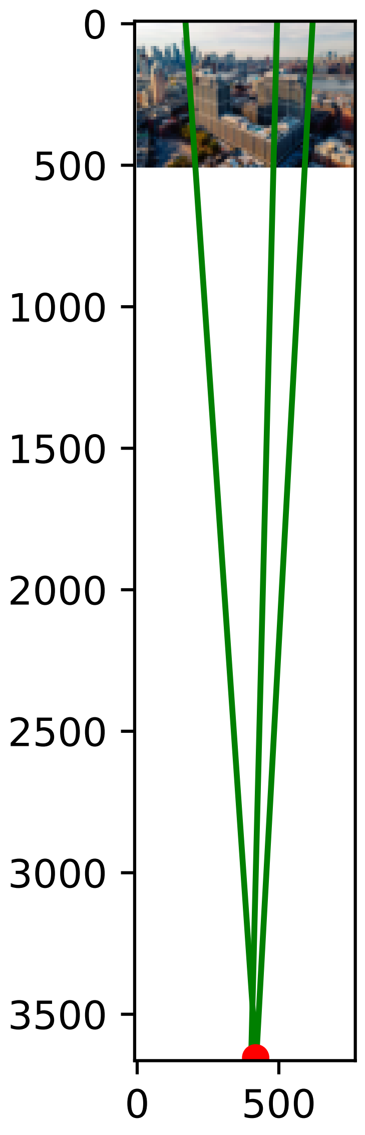


Figure 17: Lines and vanishing point in the Z-direction.

The intrinsic matrix K is given by:

$$\begin{bmatrix} 326654.63 & 0 & 261.25 \\ 0 & 326654.63 & 218.98 \\ 0 & 0 & 1 \end{bmatrix}$$

3.4 Rotation Matrix

The rotation matrix of the camera was computed to determine the orientation of the camera relative to the scene. This involved first normalizing the vanishing points and then using the intrinsic matrix K to transform these points back into the scene's coordinate system. The vectors obtained in this way represent the directions of the axes of the scene's coordinate system as seen from the camera. Normalizing these vectors and arranging them as columns in a matrix gives the rotation matrix, which describes the rotation needed to align the camera's coordinate system with that of the scene.

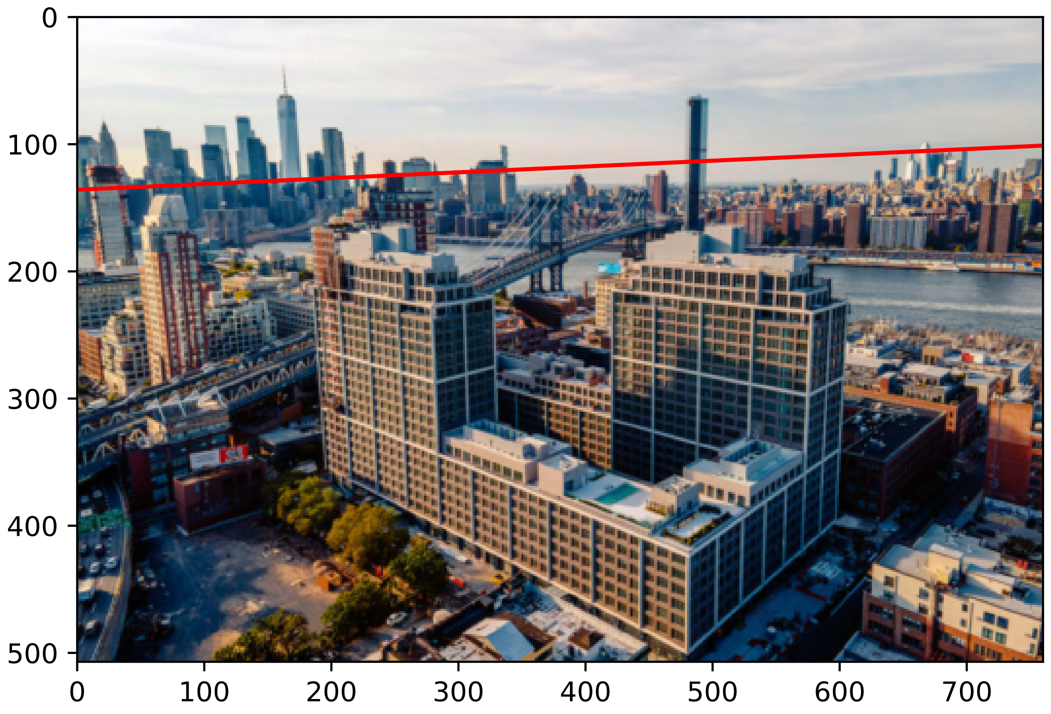


Figure 18: Ground horizon line plotted on the image.

$$\begin{bmatrix} 2.15565587e-03 & -1.46069200e-03 & 4.80560917e-04 \\ -3.89642158e-04 & -2.24366579e-04 & 1.05149923e-02 \\ 9.99997601e-01 & 9.99998908e-01 & 9.99944600e-01 \end{bmatrix}$$

This matrix indicates the orientation of the camera relative to the scene's coordinate system.

3.5 Extra Credit

For the extra credit, the project focused on automating line fitting and vanishing point estimation to further explore single-view geometry. The implementation can be broken down into several key steps:

1. **Edge Detection and Line Fitting:** The process begins with the application of the Canny edge detector to identify edges within the image. Following this, the Hough Transform is used to detect straight lines from the edge-detected image, which are then represented as edgelets – small line segments characterized by their endpoints.

2. **Computing Line Equations:** Each edgelet is converted into a line equation in the form $Ax + By + C = 0$, enabling the calculation of intersections, i.e., vanishing points, from combinations of these lines.

3. **RANSAC for Vanishing Points:** The project utilizes the Random Sample Consensus (RANSAC) algorithm to robustly estimate vanishing points from the detected lines. This involves selecting random subsets of lines, computing intersections for candidate vanishing points, and counting inliers to identify the most probable vanishing points accurately.

3.6 Rotation Matrix and Homography Computation

As we have already estimated the vanishing points automatically from the image, we can use the 3 vanishing points to estimate the Camera Parameters as we have implemented above. The results are as follows:

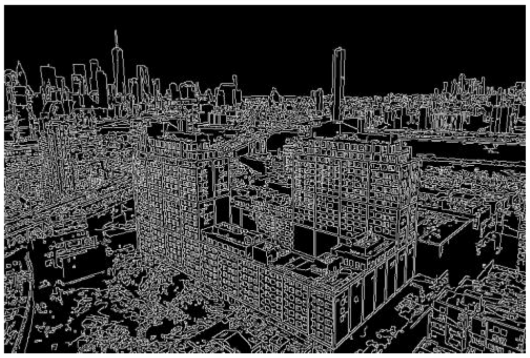


Figure 19: Edgelets Detected

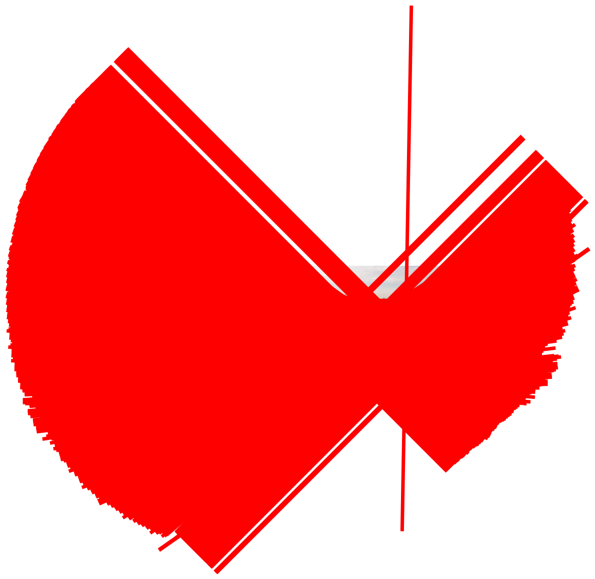


Figure 20: Estimated Line Space

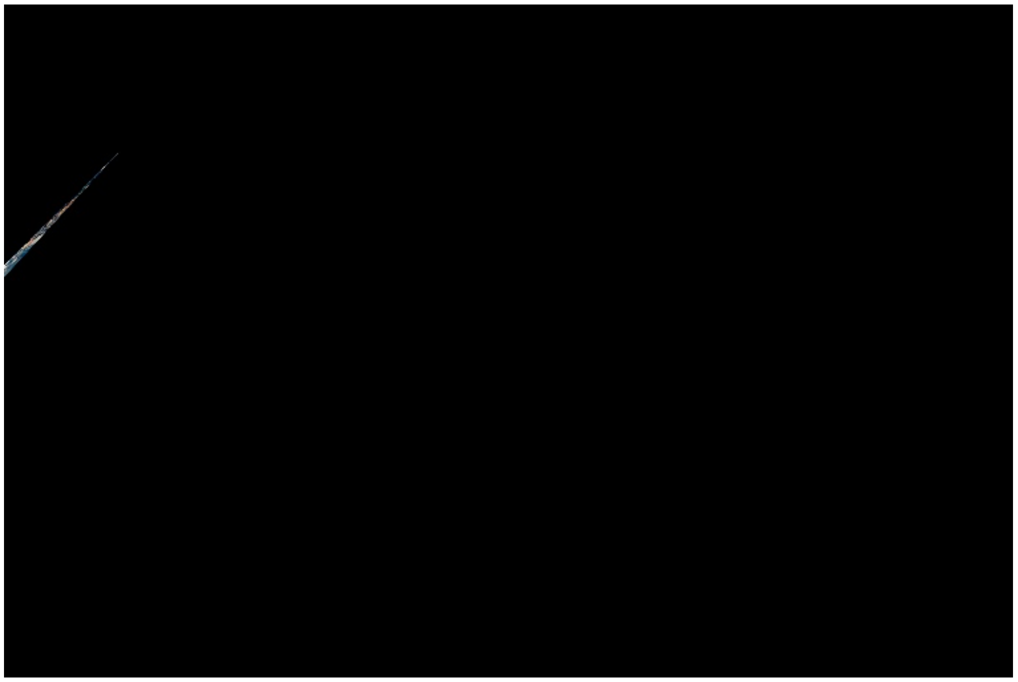


Figure 21: Result after transformation

- Focal length (f): -6.18867590200397
- Optical center (u, v): (-1.84925401516265, 1.81374503440052)

The intrinsic matrix K is given by:

$$\begin{bmatrix} -6.18867590200397 & 0 & -1.84925401516265 \\ 0 & -6.18867590200397 & 1.81374503440052 \\ 0 & 0 & 1 \end{bmatrix}$$

Applying transform with regards to the matrix did not produce correct image, we suspect it is due to the difficulty of finding reliable lines to predict vanishing points among many of them.