CV Assignment-2(A and B)

By:- Kwanit Gupta (B19EE046)

Q.1 You are given a scene image containing a logo, and a gallery containing reference logos for 10 business brands. Find out which business brand is present in the scene. Try out three different approaches and compare them.

(a) Submit your implementation. The file name of your python file should be logoMatch.py and it should take the scene image, logo image, and approach name as input and either show the matched region or say not enough match points found.

(b) Show your results qualitatively in the report and write down your observation

Solution:-

The basic pipeline for the matching was to determine the key feature points and pair them according to their preferential positions. Also, this process involved grayscaling them so that the blob detection or other key components can be performed simultaneously. As far as the keypoint pairing is concerned, Brute Force Matcher with 2-Norm was implemented. The following 3 key feature point detectors were implemented:-

1. KAZE:-
    a. KAZE (Kernel Adaptive Feature Detectors and Extractors) is a feature detection and description algorithm.
    b. It is an extension of the SIFT algorithm and aims to improve its speed and robustness.
    c. KAZE uses nonlinear scale space for feature detection and description, which allows it to handle images with large variations in scale and viewpoint.
    d. It uses an extended set of descriptors that incorporate information about the orientation of the image gradient.
    e. KAZE is known for its robustness to blur, noise, and affine distortion.
2. AKAZE:-
    a. AKAZE (Accelerated-KAZE) is an improved version of KAZE that further enhances its speed and performance.
    b. It uses a novel technique called Fast Explicit Diffusion (FED) to compute the nonlinear scale space, which is faster and more accurate than the method used in KAZE.
    c. AKAZE also introduces new keypoint detection and feature description methods that improve its performance on challenging image datasets.

d. It is known for its high performance on benchmark datasets like Oxford and Mikolajczyk.

3. SIFT:-

a. SIFT (Scale-Invariant Feature Transform) is a classic feature detection and description algorithm.
b. It is known for its robustness to scale, rotation, and affine distortion, which makes it suitable for object recognition and tracking tasks.
c. SIFT uses the Difference of Gaussian (DoG) pyramid to detect scale-invariant key points and extract descriptors using local image gradients.
d. It is a popular algorithm that has been extensively used in computer vision applications like image retrieval, 3D reconstruction, and robot navigation.

Regarding Example-1, these implementations showed promising results, with lesser key points in other labels, but more focused key points for the case of LEVIS and NESCAFE.

(a) KAZE (b) AKAZE

(c) SIFT

(a) KAZE (b) AKAZE

(c) SIFT

Regarding Example-2, these implementations showed lesser secure results, because one of the approaches, didn’t even realize its original logo’s image too, considered like others.

(a) KAZE (b) AKAZE (c) SIFT

Possible Limitations:-

All 3 seemed to be highly sensitive to resizing the logos and scenic images, as well as the color composition (which happened in the case of Example-1, between LEVIS and NESCAFE).

1. KAZE and AKAZE:-

    a. Limited scale invariance: KAZE is less scale-invariant compared to SIFT and AKAZE.
    b. High computational cost: KAZE has a higher computational cost compared to SIFT and AKAZE due to its more complex feature detection algorithm.
    c. Memory requirements: KAZE requires more memory than SIFT and AKAZE because it generates more key points.
    d. Sensitivity to image rotation: KAZE and AKAZE are sensitive to image rotation, which can result in false matches.

e. Dependence on thresholding: The performance of KAZE and AKAZE is highly dependent on the selection of threshold values for feature detection and matching.

2. SIFT:-

    a. Computationally intensive: SIFT is computationally intensive and may not be suitable for real-time applications.
    b. Memory requirements: SIFT requires a significant amount of memory due to its generation of large numbers of key points.
    c. Limited scale invariance: SIFT is less scale-invariant compared to AKAZE and some other modern feature detectors.

Q.2 Implement Hough Transform for line detection from scratch. Compare the result of openCV implementation vs your implementation (both speed and performance-wise) on the picture of your choice.

(a) Submit your implementation. The file name of your python file should be HoughTrans.py.
(b) Show your results qualitatively in the report and write down your observation.

Solution:-

The explanation behind the scratch implementation was as follows:-

- The input image is first passed through the Canny edge detection algorithm to obtain the edges in the image.

- An array of theta values ranging from -90 to 90 degrees is generated and converted to radians.

- The height and width of the image are retrieved and used to calculate the diagonal length of the image.

- An array of rho values is generated by evenly spaced values from -diagonal to diagonal.

- A 2D accumulator array is created with dimensions of rho and theta ranges initialized with zeros.

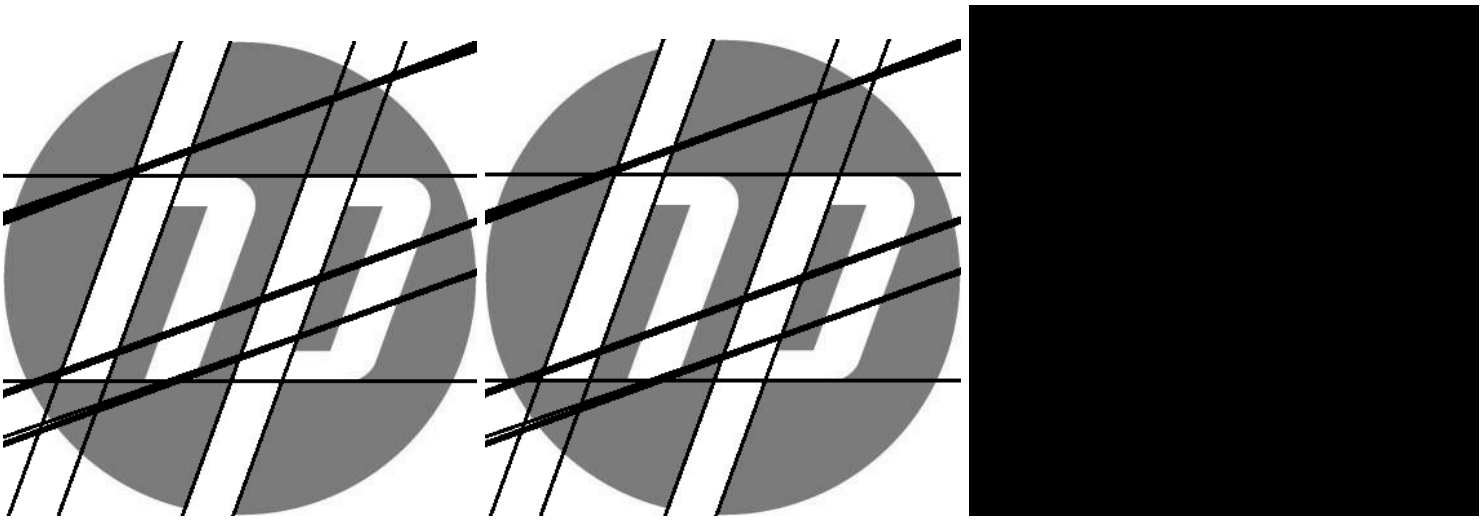
- A loop is executed over each pixel in the edge-detected image, and for each edge pixel, a nested loop iterates over each theta value. For each theta value, the rho value is calculated and incremented in the accumulator array.

- A threshold value is calculated as 80% of the maximum value in the accumulator array.

- The indices of the values in the accumulator array that exceed the threshold value are retrieved.

- The rho and theta values corresponding to the indices are extracted and returned as a list of lines.

- The draw_lines function is called to draw the lines on the original input image using the rho and theta values.



(a) Scratch (b) OpenCV's (c) Difference of both

There is no difference observed spatially between both implementations, but according to the implementation's time, there's a considerable difference. i.e 4.385 seconds and 0.004 seconds.
Q.3 A manufacturing company in Bangalore, came up with the following problem statement: They have one reference design image for a part of equipment and a probe image of either faulty or perfect. You need to identify the faulty image and show the defective region.

(a) Submit your implementation. The file name of your python file should be intelligentMatch.py, it should take two images (reference and probe) and outputs: faulty or perfect and if faulty it shows the region because of which it comes to the conclusion that is faulty.

(b) Show your results qualitatively in the report and write down your observation

Solution:-

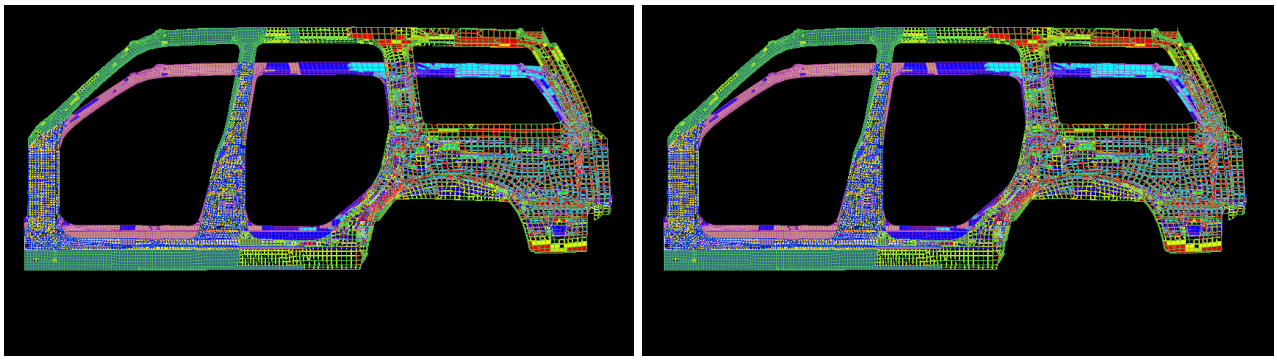
Since the images were of different sizes, so for the sake of implementation, I brought them to the same size and then applied the following 3 different variations of the same pixel-wise approach:-

1. Normal Difference, i.e X1-X2
2. Ratio, i.e X1/X2

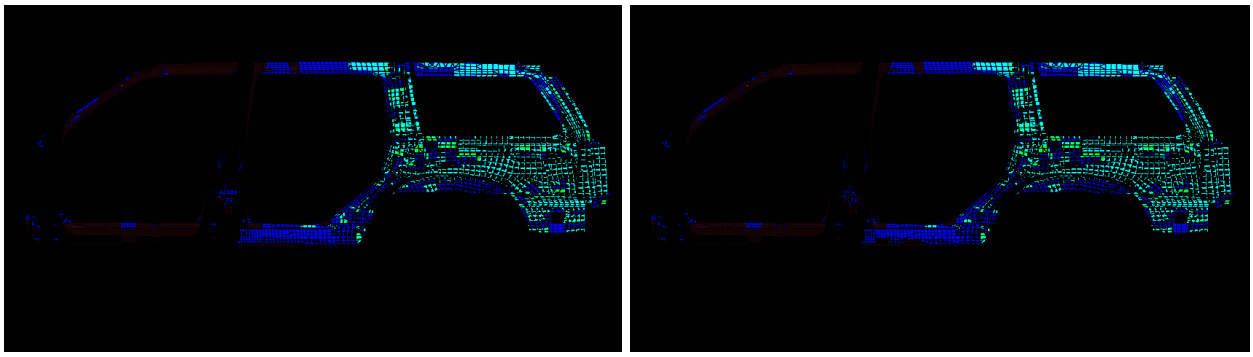
3. Log Ratio, i.e log(X1/X2)

The reason behind the last 2 approaches is to consider the faulty pixels as different pixel values than the reference images. So at that position, either the resultant pixel will be close to 0 or greater than 1. The output accordingly will highlight the possible variations. Also, I went with the variation of exponentially attenuating the resultant matrix output so that the strength of faults or perfection will highlight accordingly.

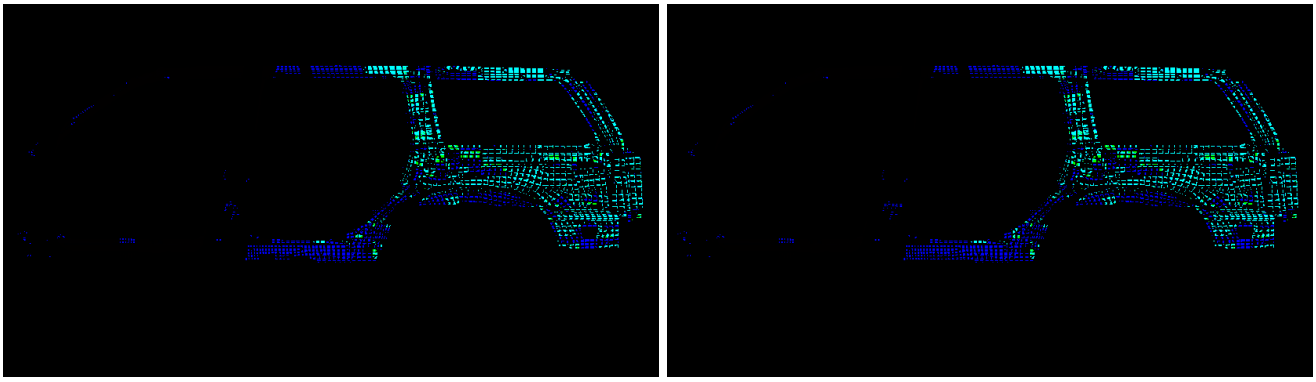
For Example-1 and Example-2, outputs were as following:-



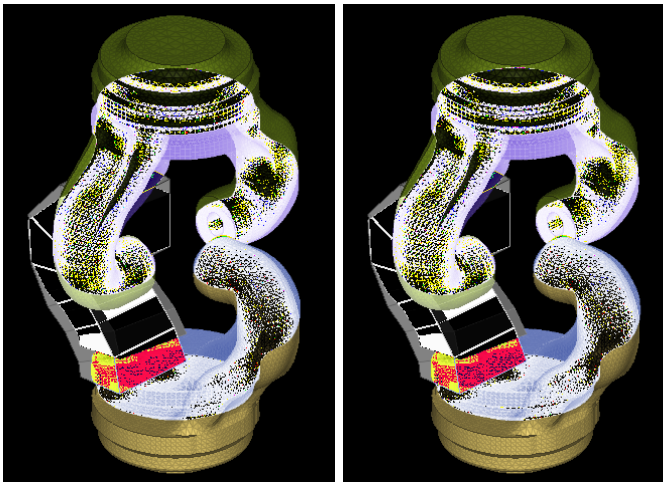
(a) Normal Difference of Example-1 (Perfect v/s Faulty)



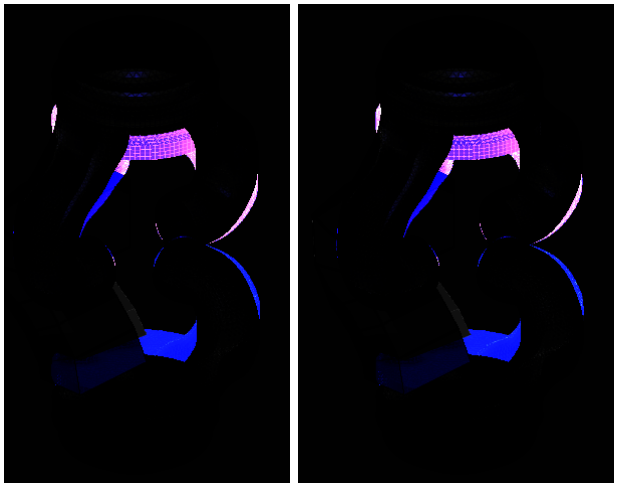
(b) Squared Ratio of Example-1 (Perfect v/s Faulty)



(c) Log Ratio of Example-1 (Perfect v/s Faulty)



(d) Normal Difference of Example-2 (Perfect v/s Faulty)



(e) 5-Powered Ratio of Example-2 (Perfect v/s Faulty)

The difference is not that visible with bare eyes, but because of the high resolution of images, some of the pixels were barely visible and differentiated in both cases. Even attenuating the pixel values via power greater or lesser than 1 is not sufficient that much (a little bit, but not that greatly). Also, even grayscaling them and then applying the above techniques didn't help a lot in finding out the exact locations of the faults.

Q.4 What is a Homography Matrix? Write down steps to compute Homography Matrix in detail with clear illustrative figures.

Solution:-

Assuming that we want to project a 3D World's Object into the 2D Image Plane, a Homography Matrix will be a 3x3 transformation matrix that explains the transition between two images from the same or different viewpoints, depending on the camera's alignment and the user. It also tries to adjust the transformation parameters to minimize the distance between the transformed

points and the actual points in the second image. It is used in various computer vision and image processing applications as follows:-

1. Align images, Perform image stitching, and Create panoramic images.

2. Augmented reality in projecting virtual objects onto real-world scenes.

3. Robotics, Remote-Sensing, and 3D Reconstruction.

4. Feature detection, Optical flow, and Camera calibration to improve its accuracy and applicability.

It is calculated by selecting the corresponding pairs of feature points in two images, formulating the homography equations, and solving for the matrix using linear algebra techniques such as Singular Value Decomposition (SVD) or Constrained Least Squares Optimization. Following will be the appropriate assumptions and then the steps to calculate the Homography Matrix:-

1. Assumptions:-

    a. The two images are related by a planar projective transformation.

    b. The corresponding points in the two images are known and can be accurately matched.

    c. There are no significant lens distortions in the images.

Correspondence Point Selection:

- Select a set of corresponding pairs of feature points in two images that you want to align. This can include either

    - Non-Synchronous techniques (in this case, additional work will be to establish pair-wise matching of key points for both images) like
        - Harris Corners
        - Applying filters and detectors for Blob Points
        - Hough Transforms

    - Synchronized techniques like
        - SIFT (Scale Invariant Feature Transform)
        - SURF (Speeded-Up Robust Features)
        - ORB (Oriented FAST and Rotated BRIEF)

Formulating the Homography Equations:

- This means that for each corresponding point pair (X_i, X'_i), there exists a homography matrix H such that X'_i = H*X_i (transformation of one feature point to another).

- Rewrite this equation as follows:-

$$[x_2\ y_2\ 1] = [x_d\ y_d\ z_d] = \begin{vmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{vmatrix} * [x_1\ y_1\ 1]$$

  where the h_ij's are the elements of the Homography Matrix, [x_d y_d z_d] represents the appropriate 3-dimensional vector and [x_i y_i 1] represents the homogeneous co-ordinates for 2D Plane-related points.

- Expand this matrix equation to get three equations in terms of the h_ij for each point pair in the following manner:-

  $x_2 = x_d / z_d = (x_1*h_{11} + y_1*h_{12} + h_{13}) / (x_1*h_{31} + y_1*h_{32} + h_{33})$
  $y_2 = y_d / z_d = (x_1*h_{21} + y_1*h_{22} + h_{23}) / (x_1*h_{31} + y_1*h_{32} + h_{33})$

  Note:- These pair-wise equations will be happening for a certain number of feature point pairs, otherwise the system will become overdetermined and solutions may or may not come optimal/robust.

Solving the Homography Equations:

- With the required number of feature point pairs, collect these paired equations and stack them into a matrix A as follows:-

  $x_2*(x_1*h_{31} + y_1*h_{32} + h_{33}) = (x_1*h_{11} + y_1*h_{12} + h_{13})$
  $y_2*(x_1*h_{31} + y_1*h_{32} + h_{33}) = (x_1*h_{21} + y_1*h_{22} + h_{23})$

$$\begin{vmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_2*x_1 & -x_2*y_1 & -x_2 \\ \cdots & & & & & & & & \end{vmatrix} * \begin{vmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{vmatrix} = [0] \text{ (Vector of N\_pairs x 1)}$$

  Note that the leftmost matrix with 9 elements per row is matrix A and the center vector is the flattened version of Homography Matrix H alongside the rightmost vector being a

zero vector. This matrix has dimensions N_pairs x 9, where N_pairs is the number of point pairs.

- Solve the system of equations Ah = 0 using linear algebra techniques. For Example:-
    - Constrained Least Squares.Optimization, where the goal is to attain minimum Ah and constraint being ||h||**2 = 1. It can be seen as follows:-

        1. Min ||Ah||**2 = h*A_T*A*h such that ||h||**2 -1 = h_T*h - 1 = 0

        2. Define a loss function L(h,l) = h*A_T*A*h - l(h_T*h - 1)

        3. Since the extrema of the loss function come at the zero point of 1st derivative, we can take dL/dh = 0, i.e A_T*A*h = l*h (Being an EigenValue Problem).

        4. Now utilize techniques like Power Iteration, SVD (Singular Vector Decomposition), etc. to figure out the lowest eigenvalue corresponding to the above-constrained equation.

        5. The eigenvector corresponding to that will be the flattened version of Homography Matrix H. Then rearrange it accordingly.

However, Homography Matrix presents its own set of limitations:-

- Planar assumption: The homography matrix assumes that the two planes being transformed are perfectly planar. In reality, this may not always be the case, and any deviations from planarity can lead to errors in the transformation.

- Affine transformation limitation: The homography matrix is limited to affine transformations only. It cannot handle non-linear transformations, which may be required in some applications.

- Lack of scale and orientation information: The homography matrix only provides information about the transformation of points between two planes. It does not provide any information about the scale or orientation of the planes. This can be problematic in applications where this information is important.

- Degenerate cases: There are some cases where the homography matrix becomes degenerate, leading to errors in the transformation. For example, if all the points on one plane are collinear, or if the two planes are parallel, the homography matrix cannot be calculated.

- Limited to 2D images: The homography matrix is limited to 2D images and cannot be used for 3D objects. This makes it unsuitable for applications where 3D information is required.

- Limited to perspective projection: The homography matrix is only applicable to perspective projection and cannot be used for other types of projections, such as orthogonal or oblique projections.

Q.5 What is stereo matching? Write down three applications of stereo matching. You can use web/books, but write the answer in your own words.

Solution:-

Stereo-matching is the process of finding corresponding points in two or more images captured from different viewpoints. It involves calculating the disparity, which is the difference in the pixel coordinates of corresponding points in each image. Stereo matching can be formulated as an optimization problem, where the objective is to minimize the difference between the left and right images subject to a set of constraints. To solve this problem, various approaches can be used like Dynamic Programming, Graph Cut, Block Matching, etc. It does come with the following assumptions:-

- The scene is static, and the left and right images were captured from different viewpoints.

- The cameras are calibrated, and their intrinsic and extrinsic parameters are known.

- The image noise is small, and the images are well-aligned.

- The disparities are spatially smooth and vary slowly over the image.

- The disparities are within a certain range, which depends on the baseline and the focal length of the cameras.

In the course, Horizontal Stereo Matching was mentioned where the left and right images differed by distance b, and that too in the x-direction.

Some of the Applications are as follows:-

- Geological surveying: Stereo matching is used in geological surveying for mapping and terrain modeling. By computing the 3D geometry of the terrain, geologists can analyze its features and plan their exploration or mining activities more efficiently.

- Industrial automation: Stereo matching is used in industrial automation for quality inspection and control, robot guidance, and object recognition. By computing the 3D

position and orientation of objects, automation systems can operate more efficiently and accurately.

- Cultural heritage restoration: Stereo matching is used in cultural heritage restoration to restore damaged or deteriorated artifacts or structures. By reconstructing the 3D geometry of the object or structure, restoration experts can plan and execute restoration work more accurately.

- Surveillance and security: Stereo matching is used in surveillance and security applications for object detection and tracking. By computing the 3D position of the objects in the scene, their movements can be tracked and monitored.

It also exhibits various limitations such as:-

- Lack of texture or occlusions can cause errors in feature matching and disparity estimation.

- The accuracy of the results depends on the quality of the images and the matching algorithm used.

- It can be computationally expensive, especially for high-resolution images.

Q.6 Write down steps to stitch images to create the panorama. Use the three Taj Mahal Images provided with this assignment to create one panorama. Show panorama into one.

Solution:-

Following are the steps, with the assumption that multiple images can come to be stitched:-

- Capture images: Take a series of overlapping images of the scene you want to stitch. It is important to keep the camera settings, focus, and exposure consistent across all the images.

- Feature detection and matching: Extract and match the key feature points in the images using algorithms such as:-

    - Non-Synchronous techniques (in this case, additional work will be to establish pair-wise matching of key points for both images) like
        - Harris Corners
        - Applying filters and detectors for Blob Points
        - Hough Transforms

    - Synchronized techniques like
        - SIFT (Scale Invariant Feature Transform)

    - SURF (Speeded-Up Robust Features)
    - ORB (Oriented FAST and Rotated BRIEF)

These algorithms can identify distinctive features such as corners, edges, and blobs in the images.

- Calculate homography matrix: Calculate the homography matrix H, which maps the key features in one image to the corresponding features in another image. The homography matrix represents the transformation between the two images.

- Blending: Blend the warped images together using any variety of techniques like:-
    - Poisson Blending
    - Alpha Blending
    - Laplacian Pyramid Blending, etc.

to create a seamless transition between them. This step involves adjusting the pixel values at the boundaries between the images to create a smooth transition.

- Repeat: Repeat steps 3-5 for all pairs of adjacent images until all images are aligned and blended. If the merging of the images happening from left to right, then constructing iterative left panoramic images and traversing to right images. Vice versa if otherwise, but if no convention is followed (random stitching), then identifying the coordinates of the feature points and then stitching accordingly in either direction.

- Crop and resize: Crop and resize the stitched panorama image to remove any black borders and adjust the size as needed.

The results eventually came out as follows, after merging the images.