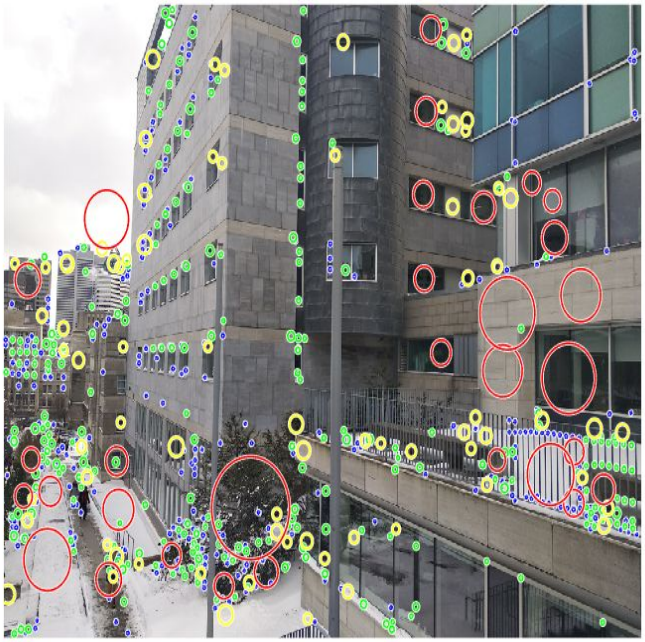
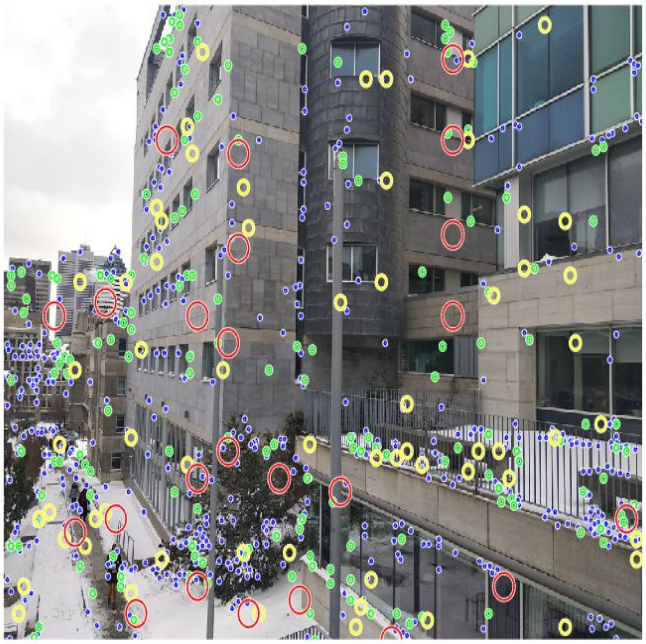
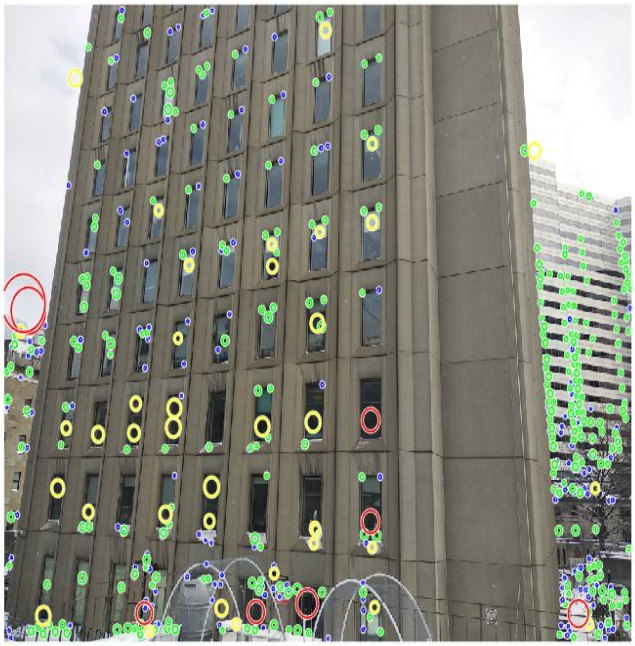
Q1 SIFT features computed by the given implementation are shown below for a sample image from both vertical and horizontal dataset.



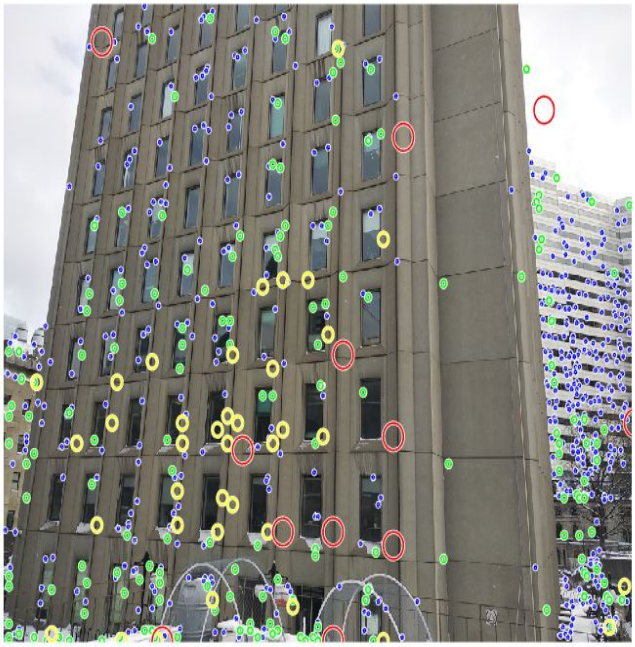
GIVEN IMPLEMENTATION



ASSIGNMENT 2 IMPLEMENTATION



PROVIDED IMPLEMENTATION

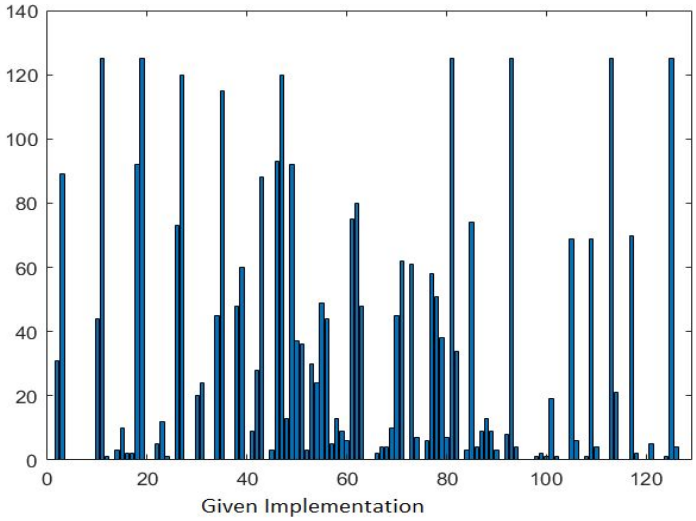


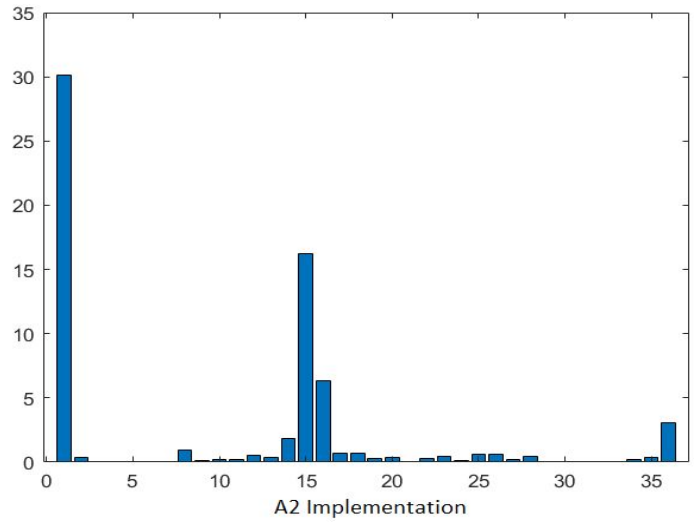
ASSIGNMENT 2 IMPLEMENTATION

As we can see, the features extracted are very different for both the implementations. The provided implementation was able to find 792 features in the first horizontal image with a threshold of 5, while in the basic implementation from assignment 2, only 738 matches were found at a small threshold of 1. Also the features found in case of the first implementation are much robust to scene changes like illumination etc and more significant in terms of being unique regions of intensity extrema, while my implementation had a number of edge pixels,and was not as robust to scene changes.

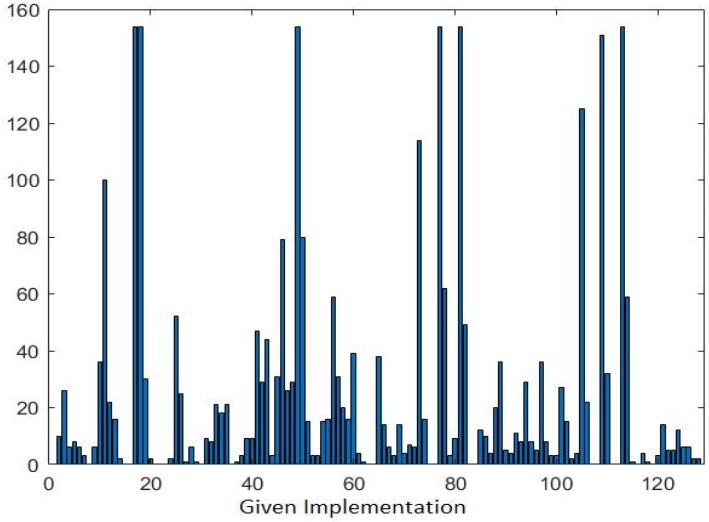
A 5*5 region of Interest where both methods had seemingly similar sets of feature matches was taken in the image to compare the features extracted using them, histograms were plotted for the two sets of features extracted for two points using both the implementations.

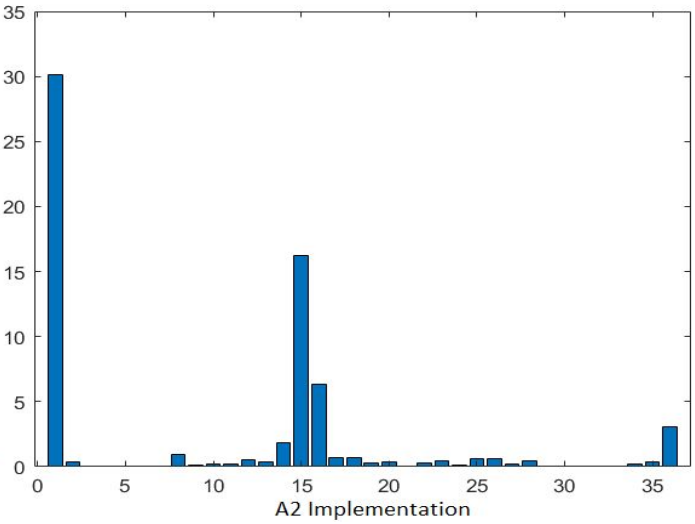
The location for first feature was (804.3,190.7) in the feature space of provided implementation and (801,189) in the feature space of A2 implementation. Their histograms are as shown





The location for second feature was (456.7,347) in the feature space of provided implementation and (461,348) in the feature space of A2 implementation. Their histograms are as shown





We can see the provided solution is much more robust.

- It seems to have higher scale features than my A2 based implementation which only took features at 4 scales. Also this implementation has multiple levels of octaves(4) and is thus much more accurate at a higher threshold than ours which had only one level.
- The given implementation also combines interpolation of nearby data points using Taylor series expansion to obtain more accurate positions as we can see in the above two cases where the positions of features are decimal points and hence more accurate than our simple integer positions.
- The low contrast features are eliminated in this implementation using second order Taylor series based thresholding.
- Features along edges are eliminated using eigenvalues of the hessian matrix, and weak features along edges will have low eigenvalues along the edge and are therefore suppressed. Our naive implementation had none of these second order keypoint extraction and was thus prone to certain poor features.
- We also see the feature histograms obtained in the provided implementation are 128 dimensional while we only had 36 dimensional features. Our method was based on simply shifting the histogram based on the principal orientation. While in the given implementation,the values for the features recomputed in a rotated square neighbourhood. The 128 dimensional features are intuitively better descriptors.

Q2.

Using the SIFT feature vectors computed for sequence of images, feature matches were found. Two different feature matching strategy were used. The naive matchFeatures function provided in MATLAB which computes the absolute differences or the squared differences under the hood and the more robust feature matching that was used for assignment 2 which used Bhattacharya Distance metric between the two histogram vectors to decide on a match. For the Bhattacharya distance based matching we had to normalize the feature vectors before and calculations, while the features were directly passed to the matchFeatures function. The threshold for matchFeatures was kept at the default value of 1,which means for two features to match the distance needs to be below 1. For Bhattacharya distance, we kept the threshold at 0.15 and did not localise the feature matching area.

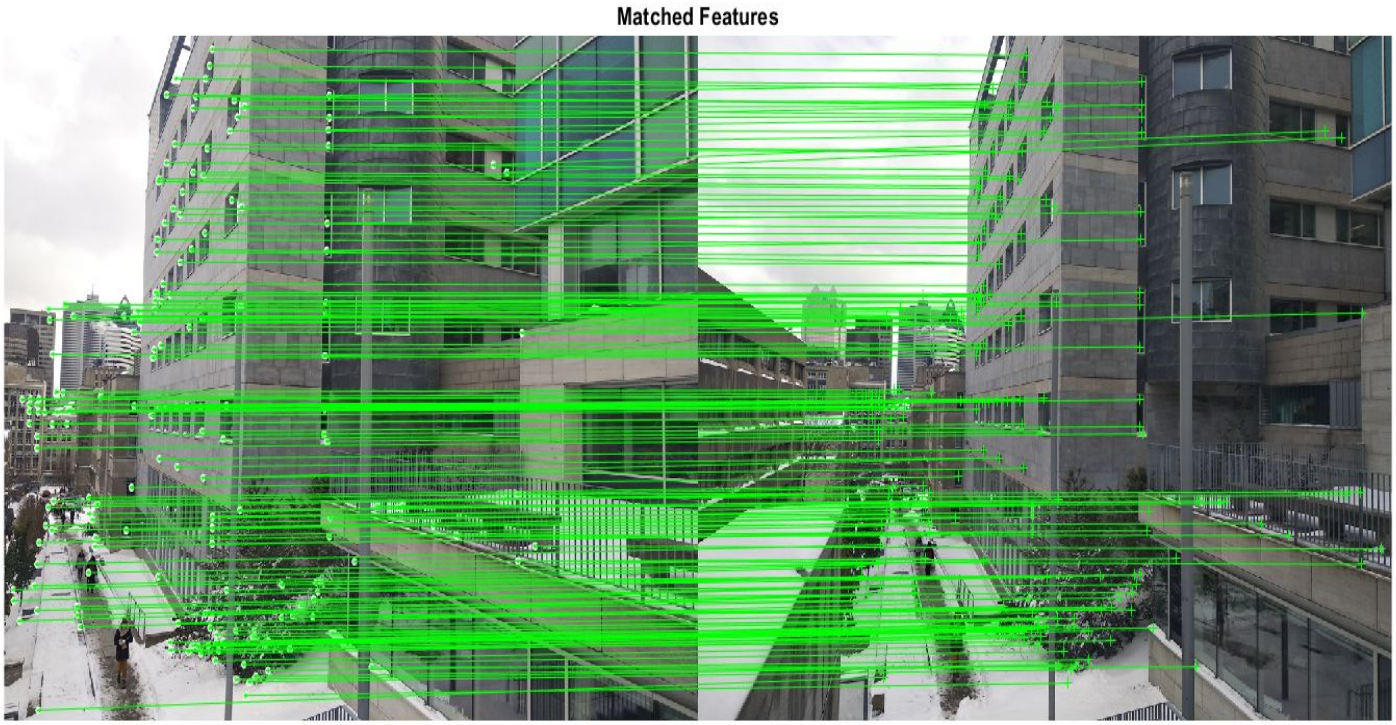
For the results shown below we used images 1 and 2 from both horizontal and vertical case.

In case of the horizontal image sequence 205 matches were found using matchFeatures while 283 matches were found using Bhattacharya distance based method.

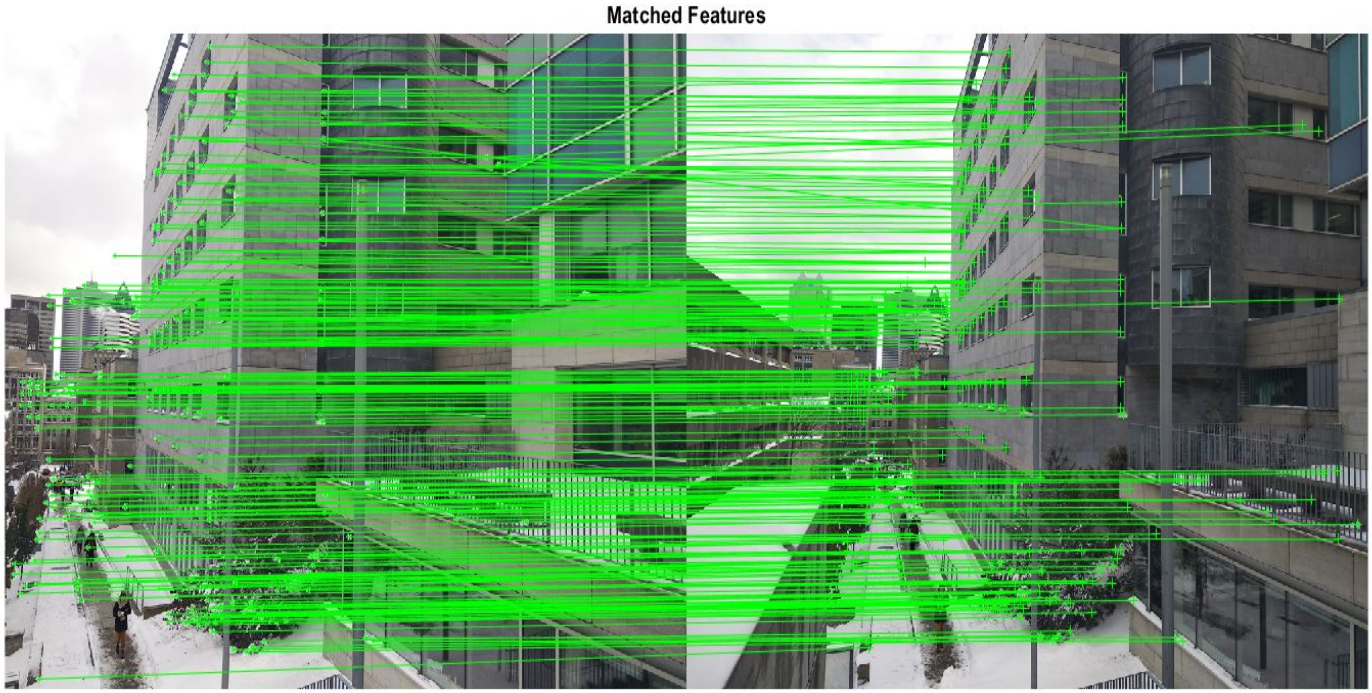
Bhattacharya distance is given by

$$d_{\text{Bhattacharyya}}(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}}$$

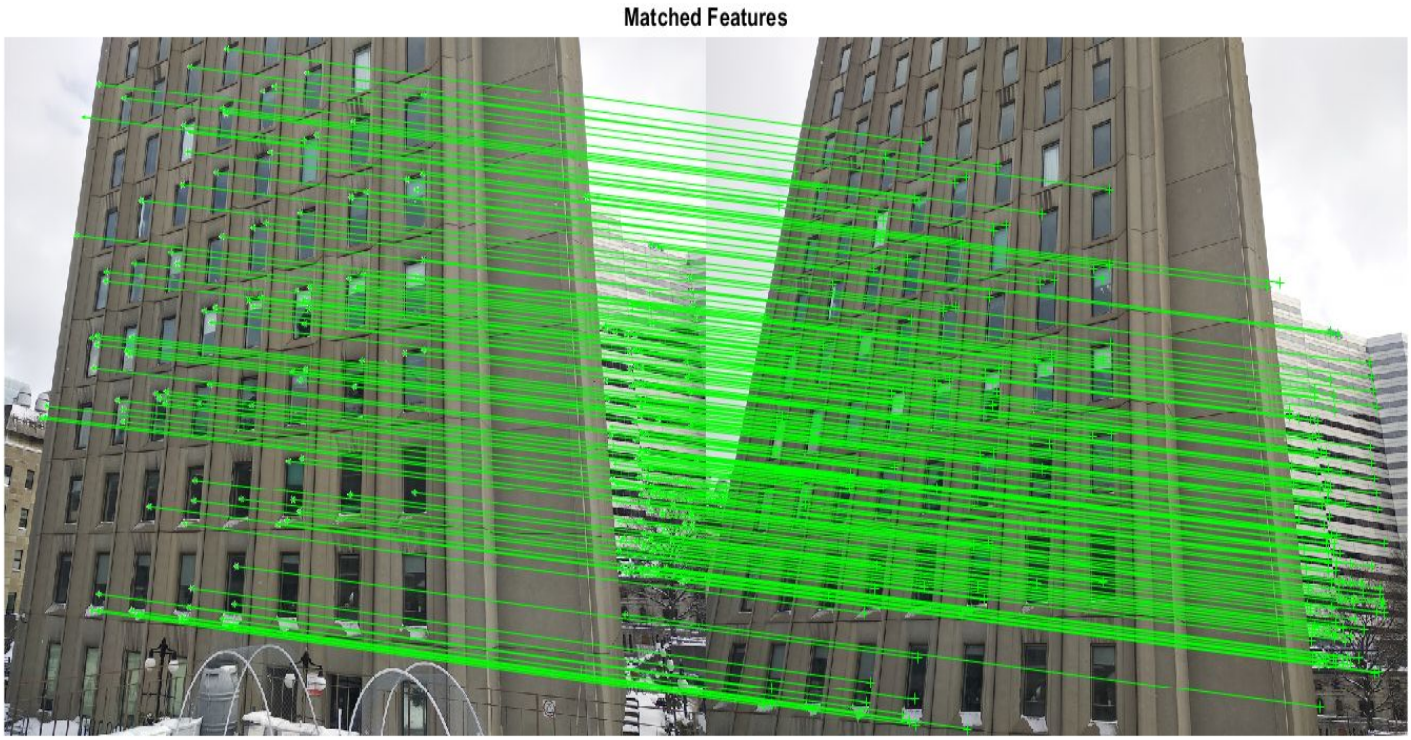
The bhattacharya distance based matches were richer in terms of feature capture as we can see from the results and they were more robust to certain changes in exposure etc between the images, and when combined with a localised search approach are highly accurate. But since the RANSAC based model we are going to use for using these features to compute homographies is robust to outliers and needs only four matches to compute the homography, we decided to use matchFeatures ahead as it is nearly as accurate as Bhattacharya distance based matching and is more efficient.



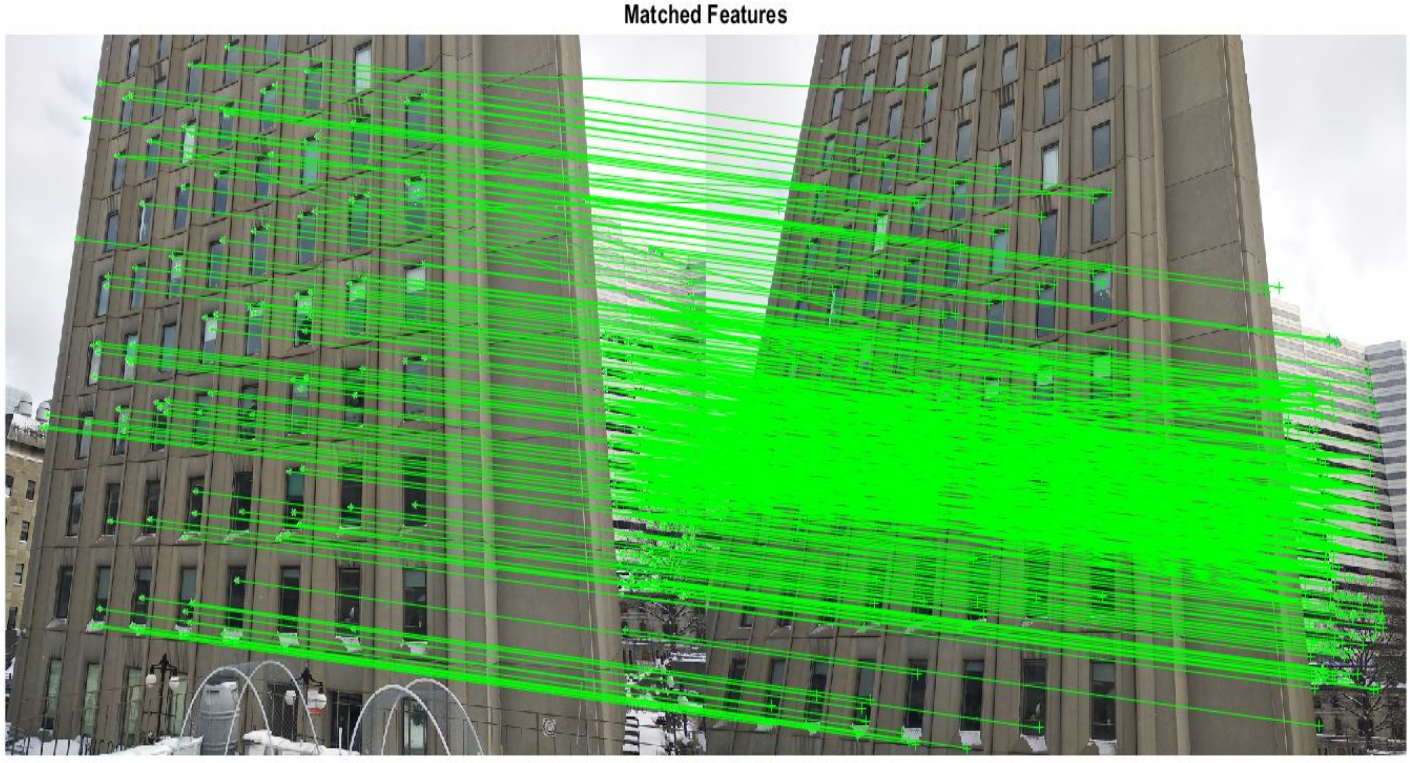
Horizontal 1 and 2 images using matchFeatures



horizontal 1 and 2 images using Bhattacharya distance



Vertical 1 and 2 images using featureMatch



vertical 1 and 2 images using Bhattacharya distance

Q3. In order to find the best approximate Homography between successive images,we need to solve for H matrix in the below equation.

$$\begin{bmatrix} w\tilde{x}_i \\ w\tilde{y}_i \\ w \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Once we have obtained the feature =matches in the two images we can use them to solve for the homography using a least squares based RANSAC algorithm which has been described below

The RANSAC Procedure is implemented as follows:-

Algorithm to determine the number of iterations of RANSAC needed:

- Let the number of matches found be N_M, i.e., between 2 consecutive Images, we find "N_P" pairs of corresponding feature point matches.
- Let the total number of Key-Points estimated be N_P. N_P is the maximum of the number of Features found in each Image.
- Hence, the number of inliers "w" = numberOfMatches / numberOfKeyPoints; w = N_M/N_P.
- We assume a 98% probability of success, i.e., the probability of estimating the correct Homography is 0.98 for our process.
- Also, let n = 4, since 4 points are needed for estimating a Homography.
- Therefore, the probability that we get all inliers = w^n.
- Also, the probability that we get at-least 1 outlier, is given as:- (1 – w^n).
- For RANSAC, the "N" number of iterations can be found by:

$$(1 - p) = (1 - w^n)^N;$$

$$N = \log(1 - p) / \log(1 - w^n).$$

4 Points algorithm:

- Thus, we run the RANSAC algorithm for "N" times.
- For each iteration, we choose 4 random points with replacement from the number of matched feature points.
- We use these 4 points to solve for the Homography Matrix 'H'.
- This can be solved as a Least Squares Estimation problem for Ax = b.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -\tilde{x}_1 x_1 & -\tilde{x}_1 y_1 & -\tilde{x}_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -\tilde{y}_1 x_1 & -\tilde{y}_1 y_1 & -\tilde{y}_1 \\ : & : & : & : & : & : & : & : & : \\ x_N & y_N & 1 & 0 & 0 & 0 & -\tilde{x}_N x_N & -\tilde{x}_N y_N & -\tilde{x}_N \\ 0 & 0 & 0 & x_N & y_N & 1 & -\tilde{y}_N x_N & -\tilde{y}_N y_N & -\tilde{y}_N \end{bmatrix} \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ : \\ 0 \\ 0 \end{bmatrix}$$

Where A is the 2N*9 matrix on the right hand side
x = The Homography Matrix which has 9 Parameters,laid out as a vector in the equation
We need 9 parameters but have only 8 DOFs,this is because, we can divide by the last scalar to make it 1.And b is a vector of zeros.

- A homography H that puts the points into exact correspondence can be obtained by finding the null space of this system of equations.
  That is, we have 9 column vectors each with 8 elements. These column vectors cannot be linearly independent and so there must be a linear combination of them (the H_{ij}) that sums to the zero vector. We find H by solving for the null space of the matrix A

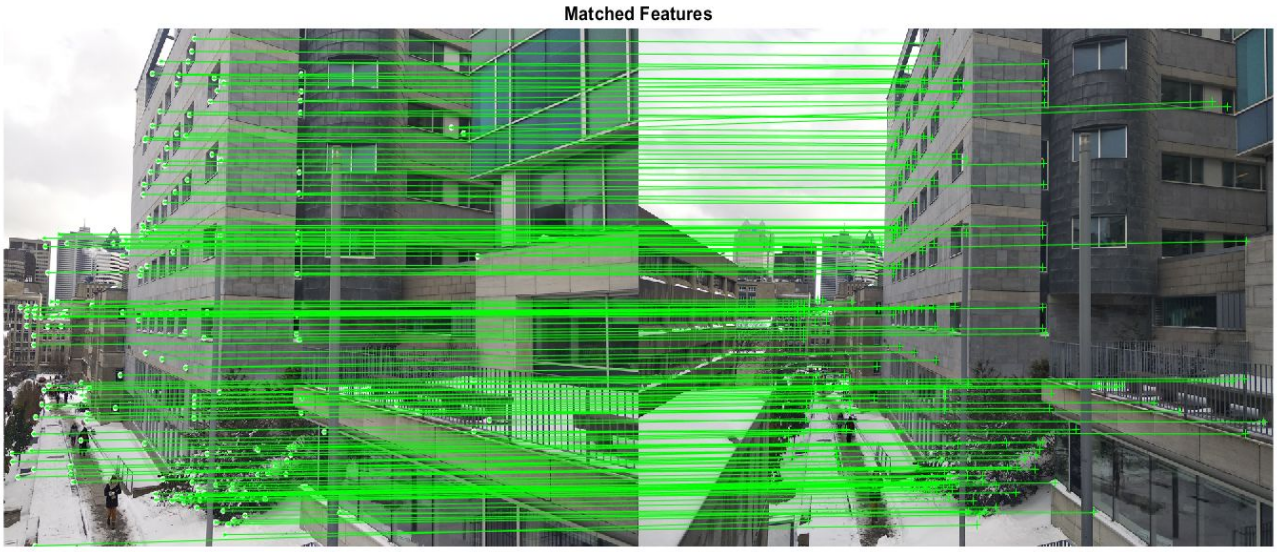
- The solution for x can be found by finding the Singular Valued Decomposition (SVD) of A and using the last column of the V Matrix found from the SVD.
- For a given H, we then find the consensus set, i.e., the number of matches that agree to the estimated Homography.
- We use the Euclidean distance between the transformed points (using H) and the corresponding match, and threshold it at a distance to estimate inliers and outliers. In our process, we use a threshold value of 0.5.

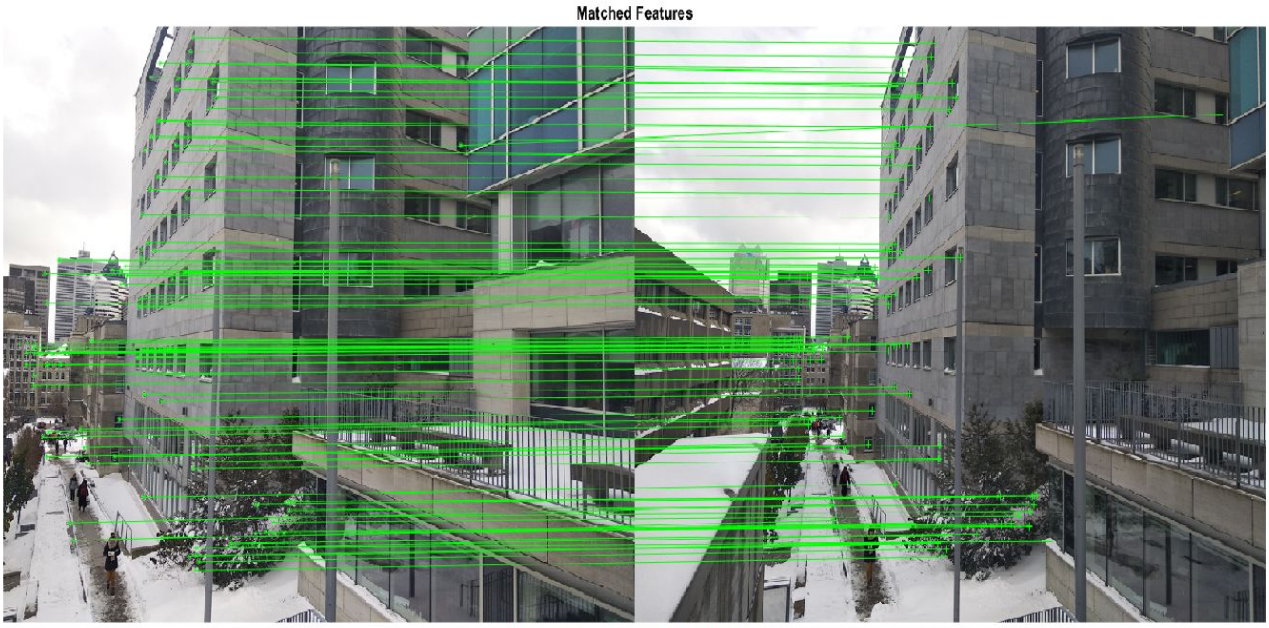
$$dist_{\mathbf{H}_i}(x, y, \tilde{x}, \tilde{y}) \;=\; \| \, (\tilde{x}, \tilde{y}) - (x', y') \, \|_2 \; .$$

For the inliers found, we estimate the Consensus Set using the distance metric

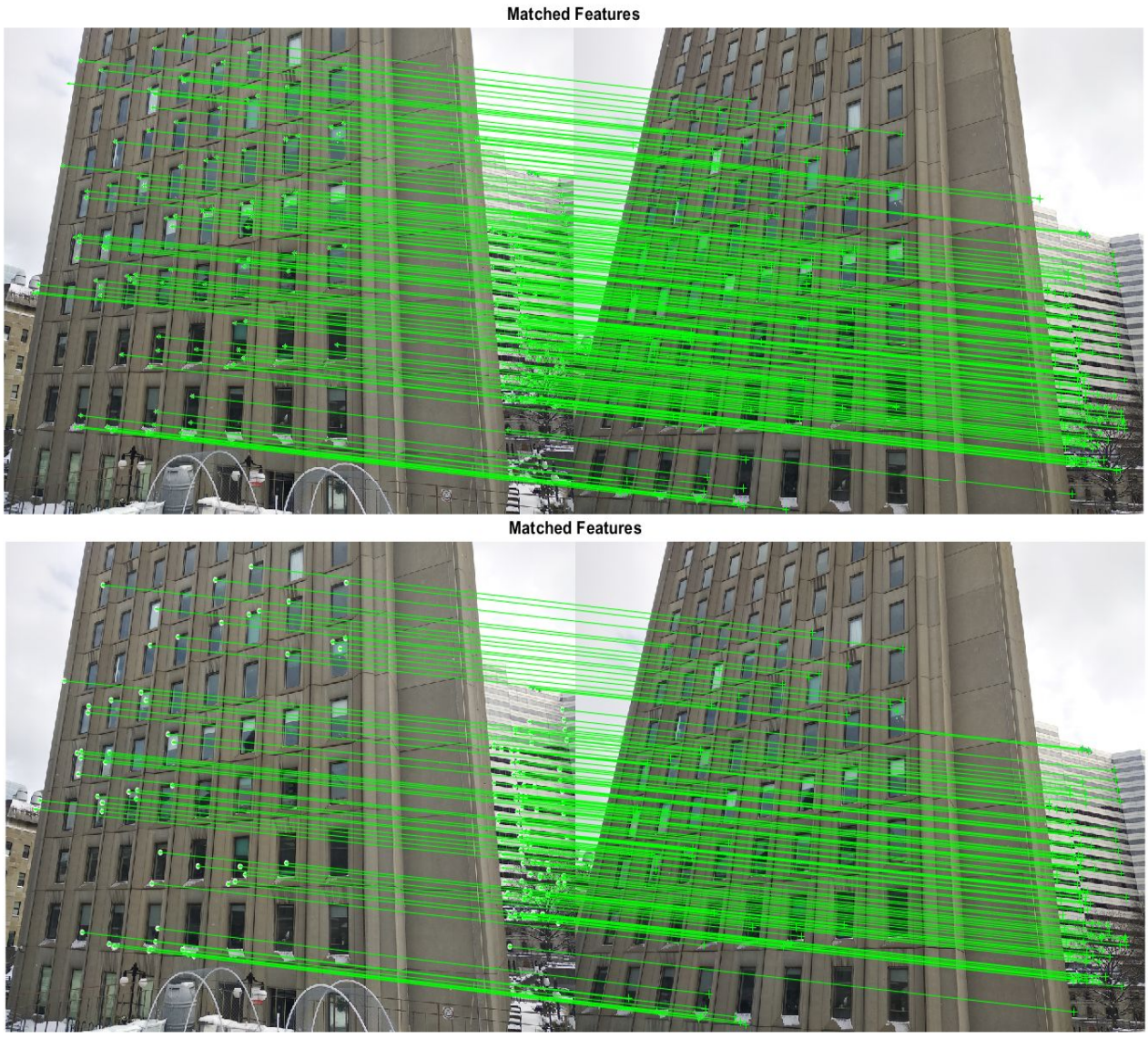
- This process is repeated until “N” iterations and the homography that leads to the largest consensus set is selected and the consensus set is retained.

- After the RANSAC Loop, described, we get a Consensus Set of Matching Keypoints. We use the points in this Consensus Set to estimate the final Homography for the successive pair of Images by using least squares on the points based on the same principle described above. The figure below shows the matches obtained naively compared to the matches before and after finding the Consensus Set using RANSAC.

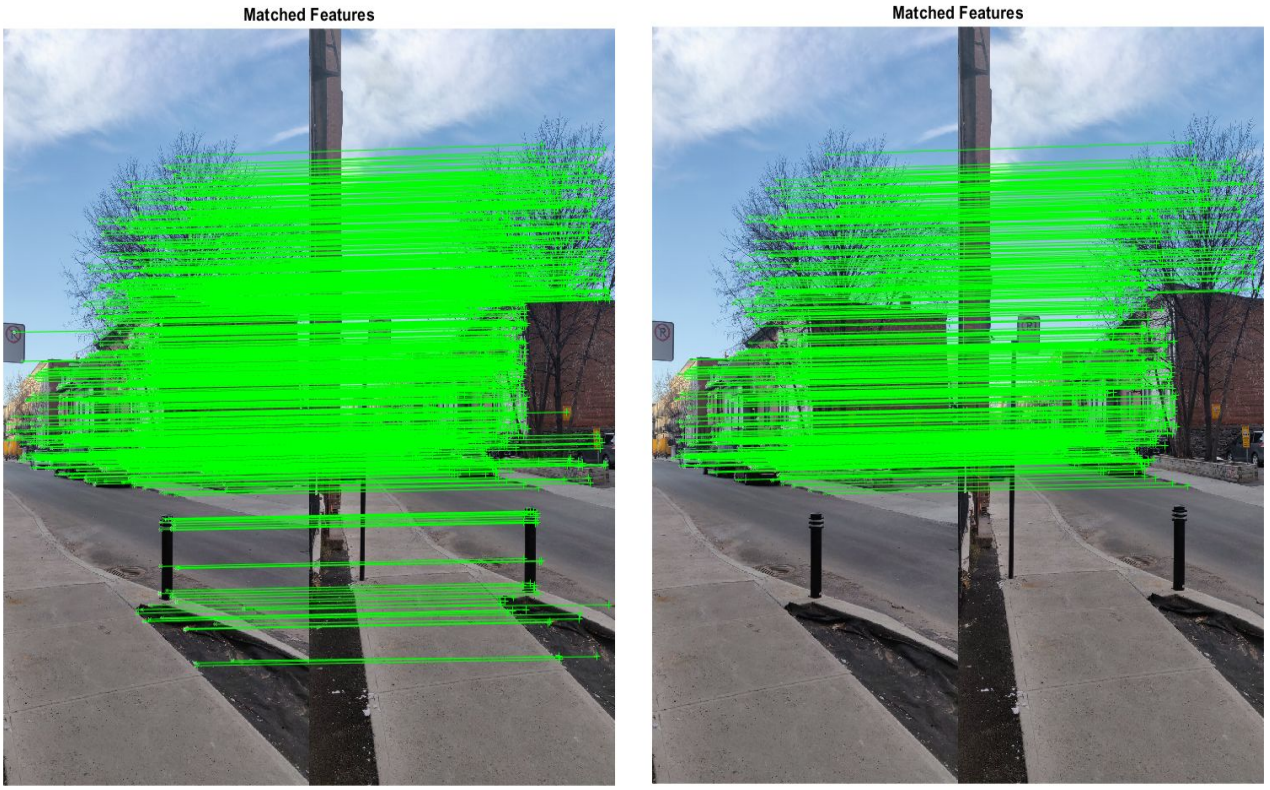




matches before and after RANSAC



matches before and after RANSAC



MATCHES BEFORE AND AFTER RANSAC

Q4. Once we have solved the sequential image registration problem, we use the matched features from consecutive images to learn geometric transformations between them in order to project them into a panoramic image. This process is called Image Stitching. In order to perform image stitching, an empty panorama is created, then the images are aligned and blended based on the learned homography after which they are warped on to the panorama canvas. Image blending involves executing the adjustments figured out in the calibration stage, combined with remapping of the images to an output projection. Colors are adjusted between images to compensate for exposure differences.

Alignment may be necessary to transform an image to match the view point of the image it is being composted with. Alignment in simple terms is a change in the coordinates system so that it adopts a new coordinate system which outputs image matching the required viewpoint.Here we have used projective transformation for alignment. Image rectification is an alignment step that could make this process easier, and feature matching faster.

Results of image stitching are shown below

Figure Horizontal Images 0,1,2 stitched

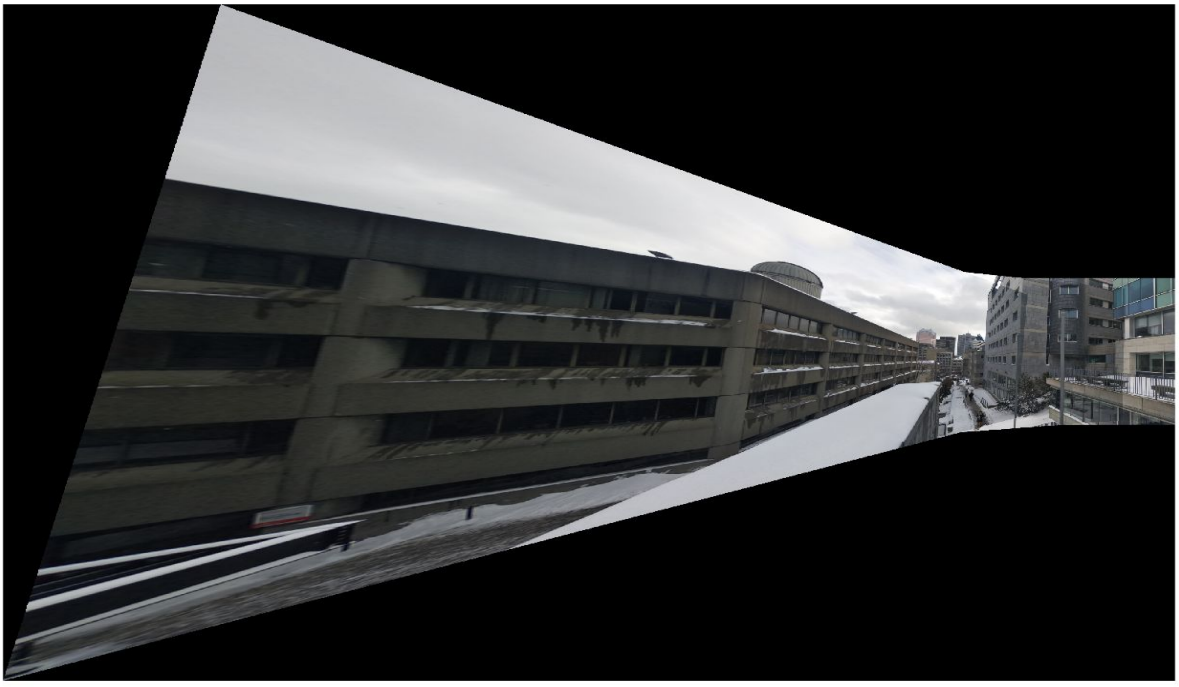


Figure Horizontal Images 1,2,3 stitched

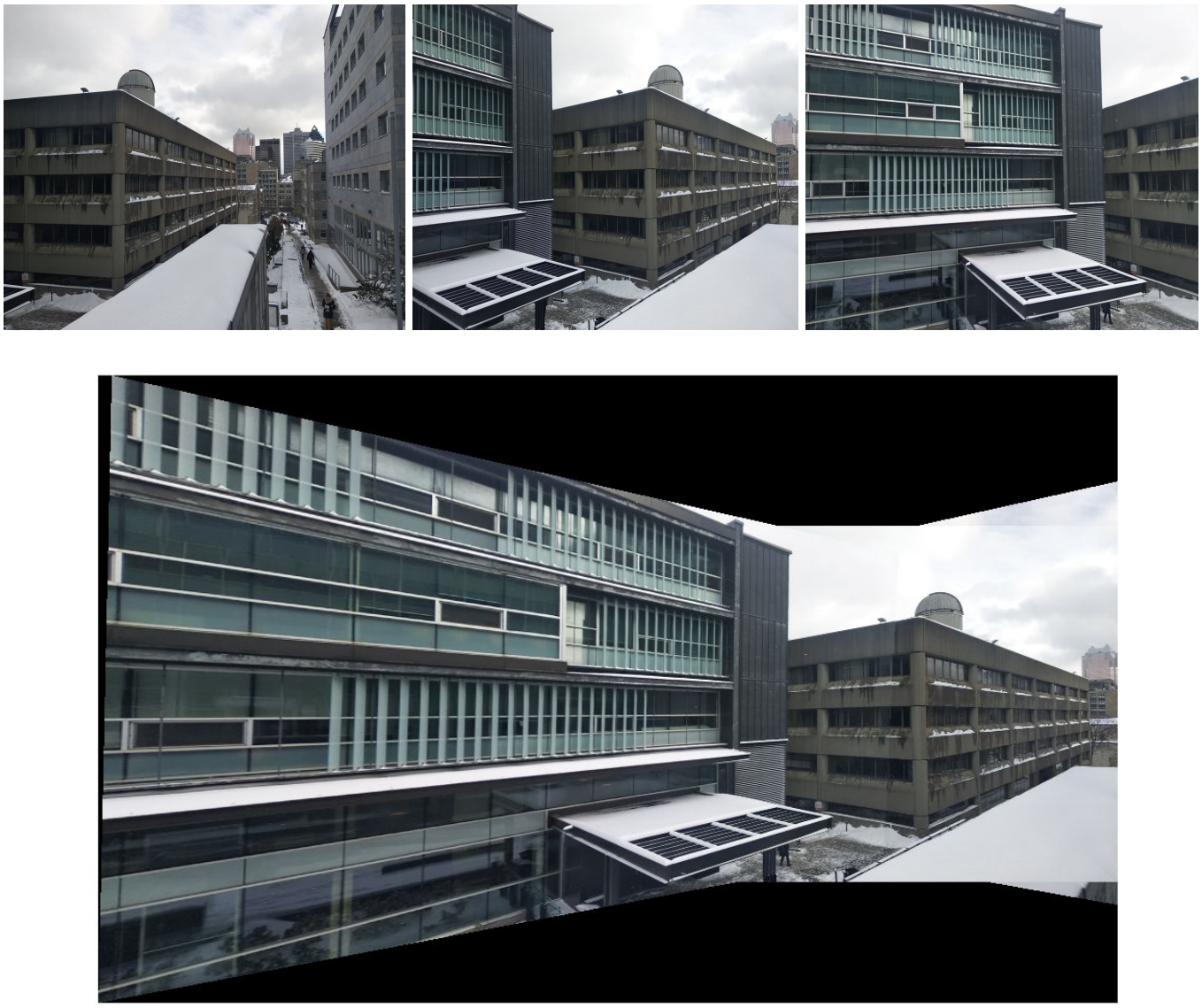


Figure Horizontal Images 3,4,5 stitched

Nishant Mishra(260903177)



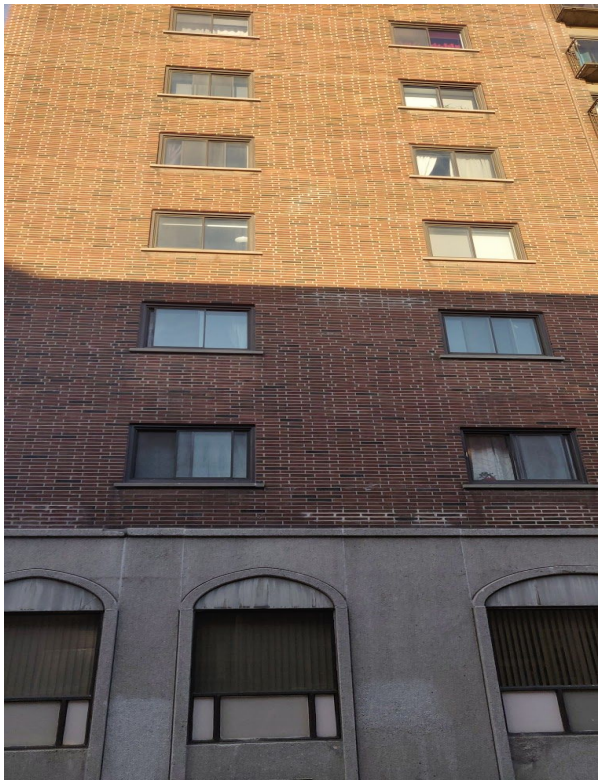



Figure vertical Images 0 to 5 stitched

Q5. Figures below show the stitching algorithm applied to pictures taken from mobile camera

Nishant Mishra(260903177)

Figure Horizontal Images 0 to 7 taken using phone stitched



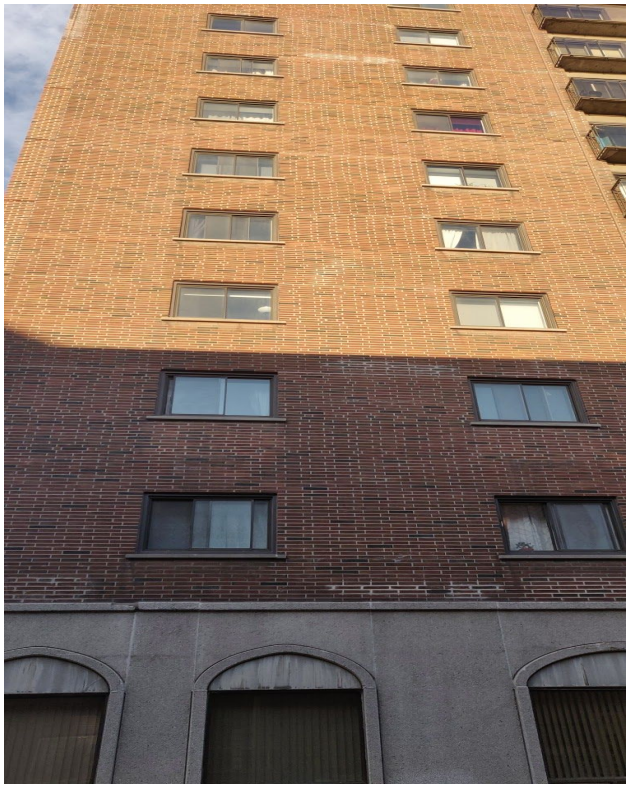


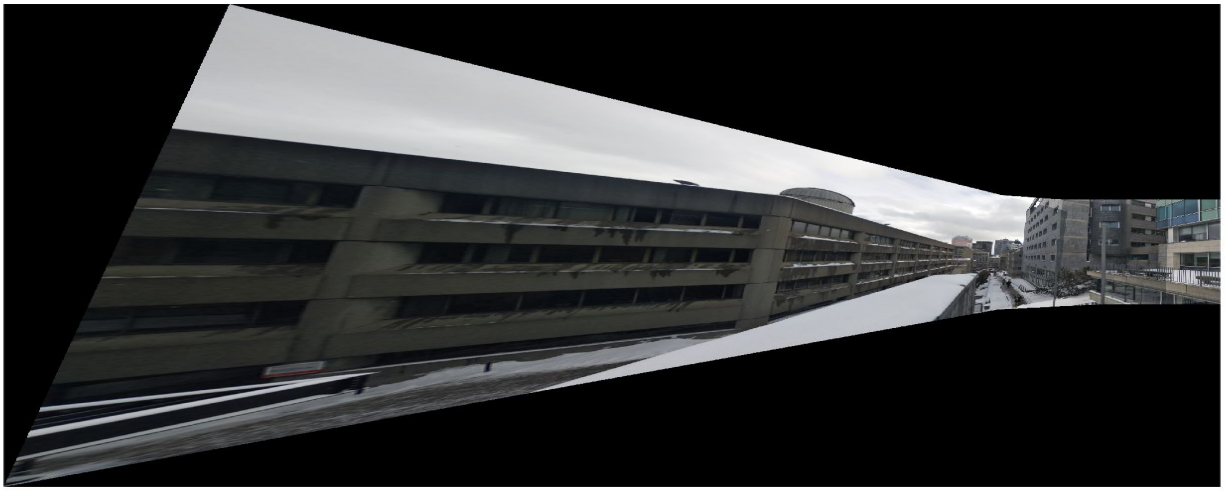
Figure set of vertical images to be stitched

Figure vertical images 0 to 6 taken using phone camera stitched

Q6. As we can see the image stitching was not perfect, a number of artifacts were introduced.

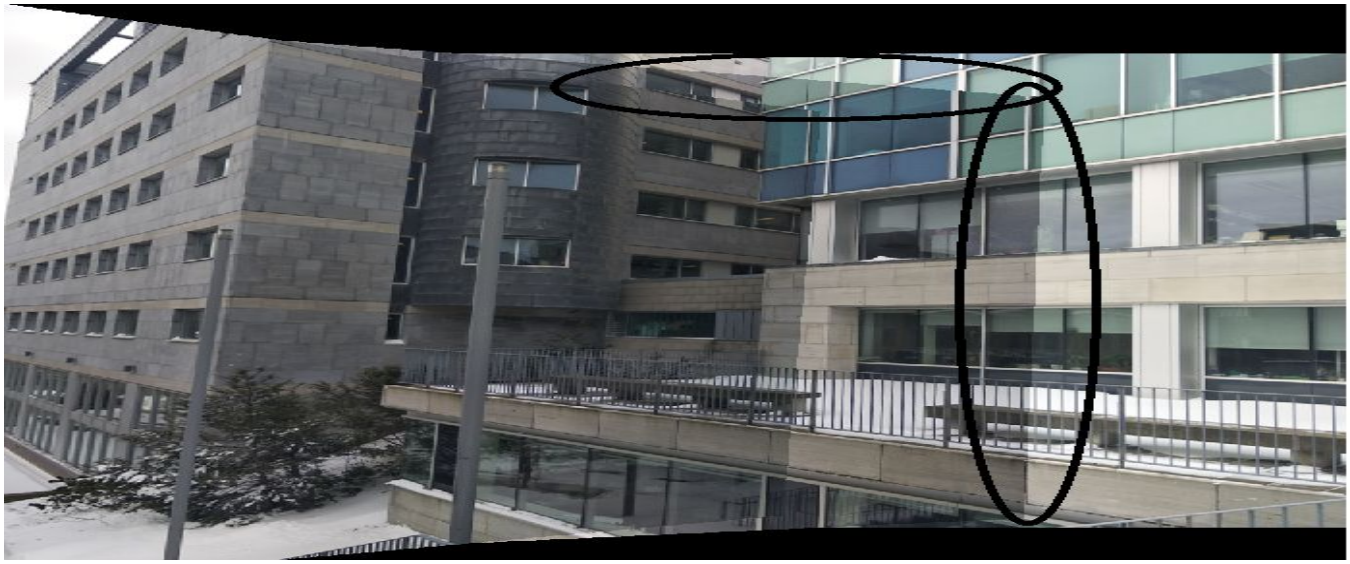
Straightening the image

Perspective warping leads to non rectified image stitching. Image registration using the steps of Q3 and 4 gives the relative rotations between the cameras, but there remains an unknown 3D rotation to a chosen world coordinate frame. We have assumed that there is no relative rotation between the sequence of frames and if we simply assume that R = I for one of the images, we typically find a wavy effect in the output panorama as seen below



This is because the real camera was unlikely to be perfectly level and un-tilted. We can correct this wavy output and automatically straighten the panorama by making use of the fact that the camera X vectors (horizontal axis) typically lie in a plane . The idea is that it is rare for people to twist the camera relative to the horizon. By finding the null vector of the covariance matrix of the camera X vectors, we can find the "up-vector" u (normal to the plane containing the camera centre and the horizon). Applying a global rotation such that up-vector u is vertical (in the rendering frame) can effectively remove the wavy effect.

We also noticed how while blending two images with significant differences in intensity or large translations of features, new, ghost edges were formed in the panorama,as shown below.



These are caused due to misregistration of features,parallax,lens distortion or differences in exposure and large differences in images leading to blowing up of homographies. In order to deal with these issues,usually two methods are used

- Gain Compensation This compensation is basically minimizing intensity difference of overlapping pixels. Image blending algorithm allots more weight to pixels near the center

of the image. This simplifies the computation and gives some robustness to outliers, which might arise due to small misregistrations between the images

- Even after gain compensation, some image edges are still visible due to a number of unmodelled effects, such as vignetting (intensity decreases towards the edge of the image), parallax effects due to unwanted motion of the optical centre, mis-registration errors due to mismodelling of the camera, radial distortion and so on. Due to these reasons a blending strategy called multi band blending is used.
- Multiband blending allows blending the overall color of the images and mitigate the differences in the scale and/or the exposure existing between the juxtaposed images to remove sharp differences in the exposure between pictures and get a panorama as homogeneous as possible. The idea behind multi-band blending is to blend low frequencies over a large spatial range, and high frequencies over a short range.

REFERENCES

- Matthew Brown and David G. Lowe. 2007. Automatic Panoramic Image Stitching using Invariant Features. Int. J. Comput. Vision 74, 1 (August 2007), 59-73.
- S. Suen; E. Lam; K. Wong (2007). "Photographic stitching with optimized object and color matching based on image derivatives". *Optics Express*. **15** (12): 7689–7696. doi:10.1364/OE.15.007689. PMID 19547097