Project Report

Automatic Panorama Stitching

Project Id - 2

Project Title: Automatic Panorama Stitching

Team Name: Team Shinigami

Team Members: Aditya Aggarwal (20161129)

Prakyath Madadi (20161236)

Github Link:

https://github.com/adityaaggarwal97/DIP_Panorama-Stitching

Introduction:

In this project we have implemented a fully automatic tool for creating panoramas given the input set of images. Previous approaches have used human assistance in ordering the images/aligning the images or some restrictions on the input dataset in order to establish matching images. In this project we formulate stitching as a multi-image matching problem, and use invariant local features to find matches between all of the images. This makes our method independent of image orientations, scale and illumination,noise and even the ordering.

Problem Statement:

So the formal problem statement of the project can be:

“Given a set of images from multiple scenes in jumbled unordered way, construct all possible panoramas of the different scenes.”

The process of panorama image stitching is fully automated. We also allow multi row stitching, that is create a 2D panorama, where the images have matchings across x and y directions.

Motivation:

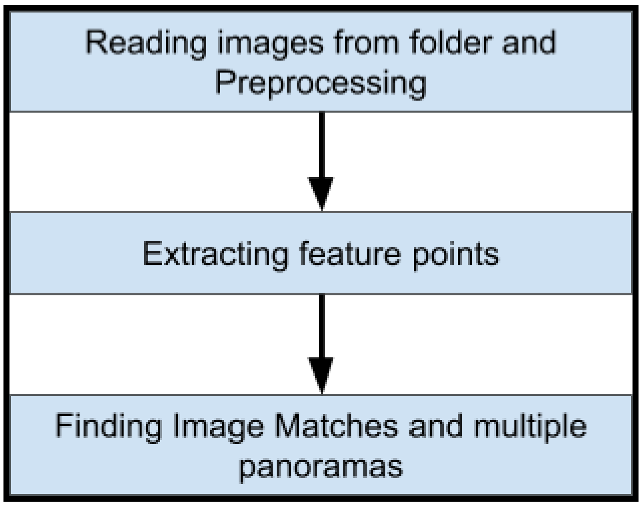
A decade ago creating a panoramic photograph required either buying an expensive camera setup or a lot of time spent in stitching the images together. But now, with inexpensive cameras and phones in everyone’s pockets, taking images and creating panoramas has become a little easy. However panoramas clicked through these devices still has some constraints like:

1. Cameras should move in a fixed direction.
2. There should exist a minimum level of overlap.
3. Camera should be steady while being moved across the scene.
4. Scene should be static.

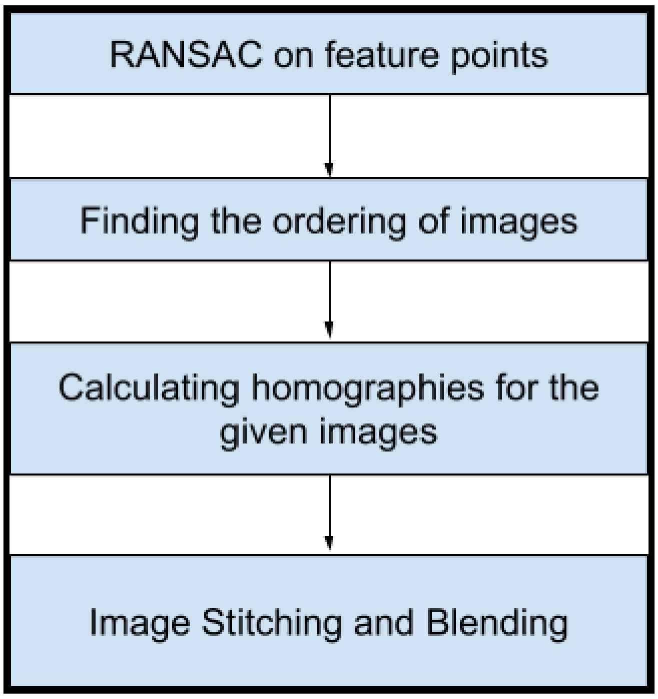
These constraints make the process of taking panoramas challenging and a difficult task. However clicking multiple images of the same scene with small overlap still remains very easy. So an offline tool for creating panoramas is very important in today’s world. Also creating panoramas offline has number of perks to it like higher resolution images with more details, wider angle photos in both x and y direction and considerably lesser noise. Therefore panoramic image stitching has an extensive research literature , which motivated us to work on this project.

Overview:

The algorithm takes in the name of directory with multiple input images and then returns the constructed panorama image of the scene. Input images can be of multiple scenes along with noise images and even unordered. So the pipeline for the algorithm is explained in the flowchart:


Reading images from folder and Preprocessing
Extracting feature points
Finding Image Matches and multiple panoramas

After detecting multiple panoramas and removing noise images, each set of labelled component is then used to create a panorama and an overview of this pipeline is given.


RANSAC on feature points
Finding the ordering of images
Calculating homographies for the given images
Image Stitching and Blending

Action Plan:

Preprocessing:

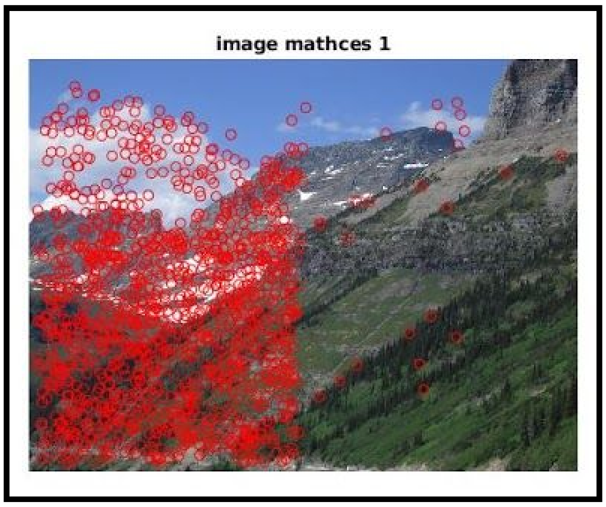
After reading all the images in an array the images are resized to 800 X 800 dimensions. This particular dimensions are used because it gives us a balance between the number of feature points and time taken to run the algorithm. As we increase the dimensions of image, number of feature points increases but at the same time memory requirements and time also increase. On the other side, on decreasing the size of images number of feature points are too less to reconstruct the images for a good panorama.

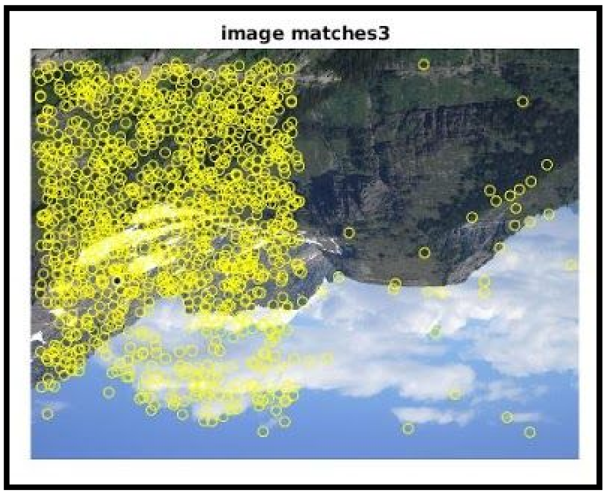
Extracting Feature Points:

First essential step in this pipeline involves finding the feature points. For this, SURF (Speeded-up Robust Features) feature detector is used. SURF features are located at scale-space maxima/minima of LOG which is approximated by a BOX filter.

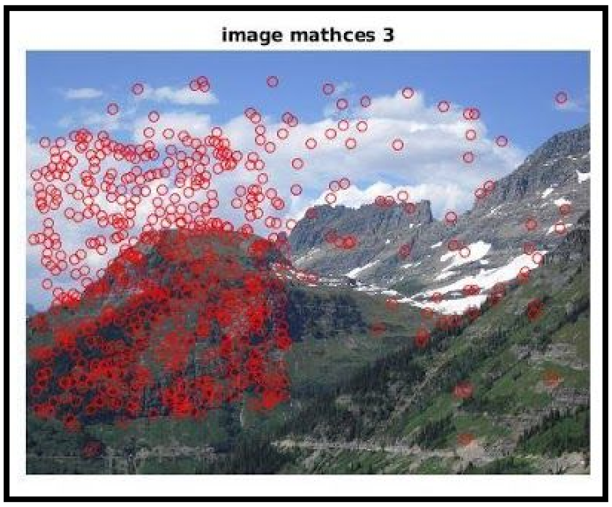
For feature description, SURF uses Wavelet responses in horizontal and vertical direction. A neighbourhood of size 20sX20s is taken around the keypoint where s is the size. It is divided into 4x4 subregions. For each subregion, horizontal and vertical wavelet responses are taken and a vector is formed which gives us SURF feature descriptor.

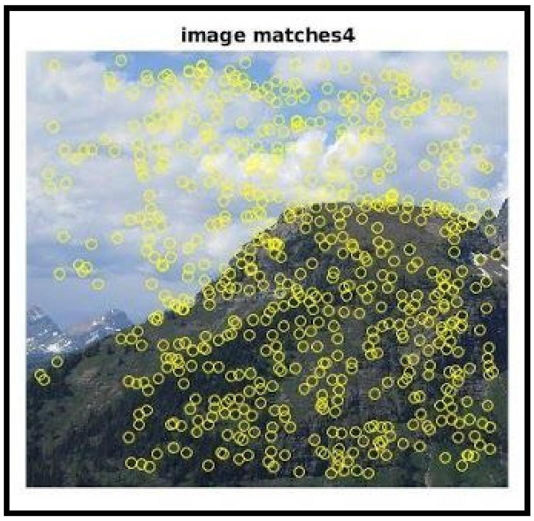
SURF was our preferred feature descriptor over SIFT and Harris Corner detector because it is faster than SIFT and at the same time scale, rotation and illumination invariant.



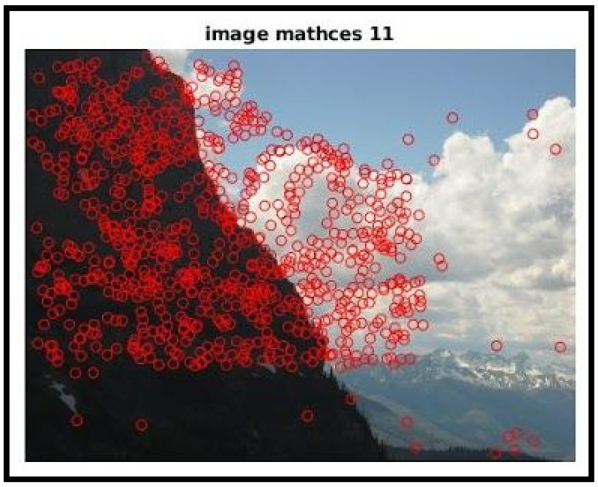


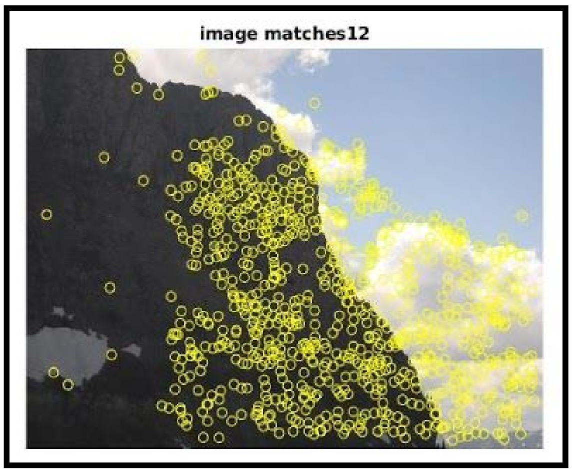
Rotation Invariant





Scale Invariant



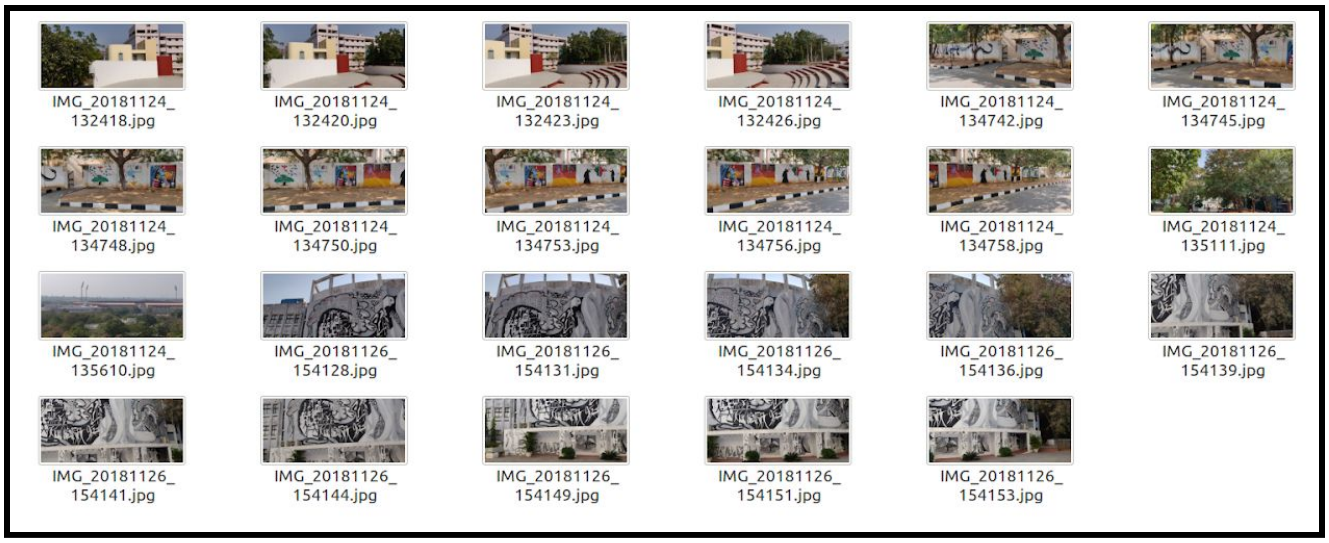


Illumination Invariant

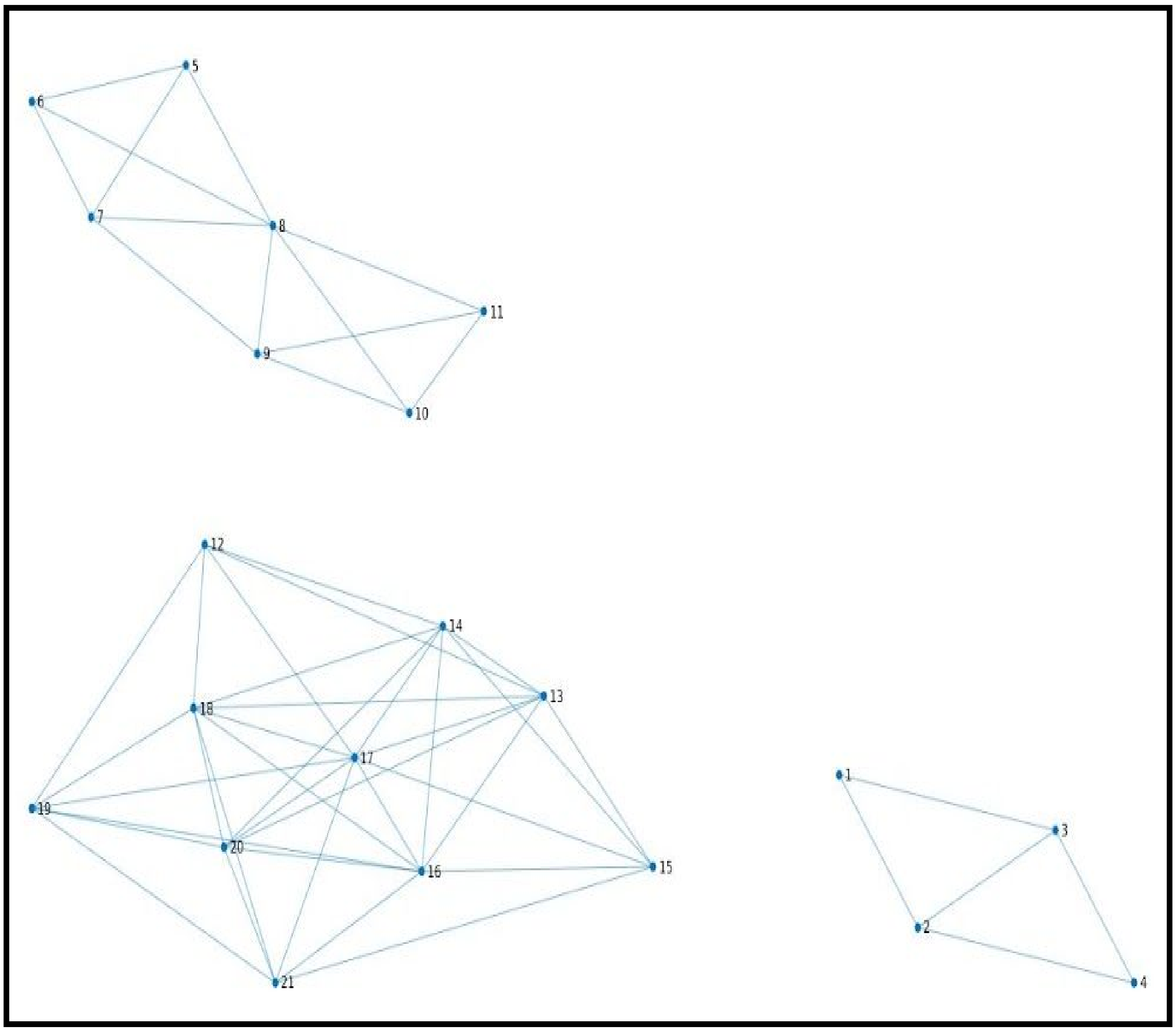
Finding Image Matches:

After finding feature points in all the images, these images are compared with each other to find the matched pair of images. A matched pair of image is defined as a pair of images with a minimum number of 300 matched points with 0.9 threshold. All such pairs with the matching points are stored and then used to find the number of different panoramas. For detecting number of panoramas, we find all the connected components in the given pairs and label it as a part of a single panorama. Hence all the different connected components will correspond to different panoramas.

Below is a sample case of a set of 22 images consisting of 20 images from 3 different scenes and 2 noise images. Connected Component graph of this sample looks something like this:



Input Images



Connected Components Graph

RANSAC:

RANSAC is a robust estimation method that uses random image correspondences to estimate transformation matrices like the homography matrix, Fundamental matrix, etc. While finding the transformation matrices, it also finds the inlier and outlier correspondences. The inlier correspondances are then used to find the homography between the two images using bundle adjustment. We have used fundamental matrix estimation to extract the inlier correspondences for all the image matches from the previous step. We randomly select 4 correspondences and the estimate of fundamental matrix using direct linear transformation method (DLT). Given four correspondences ‘x’ and ‘x1’ in first and second image, the fundamental matrix(F) is the one which satisfies the equation:

x.F.x1’ = 0

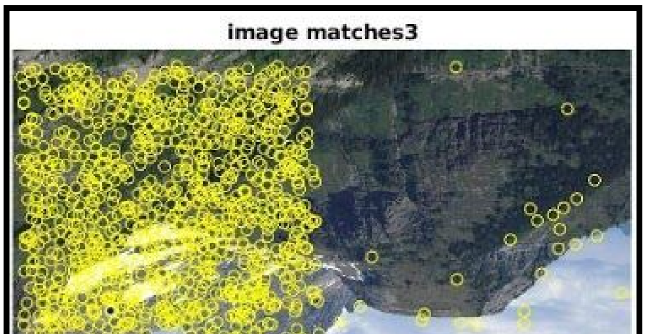
We repeat the process for n trials (n=500) and select the solution which has the maximum number of inliers (points which are consistent with the estimate). Given the probability that a feature match is correct between a pair of matching images (the inlier probability) is ‘pi’, the probability of finding the correct matrix after n trials is:

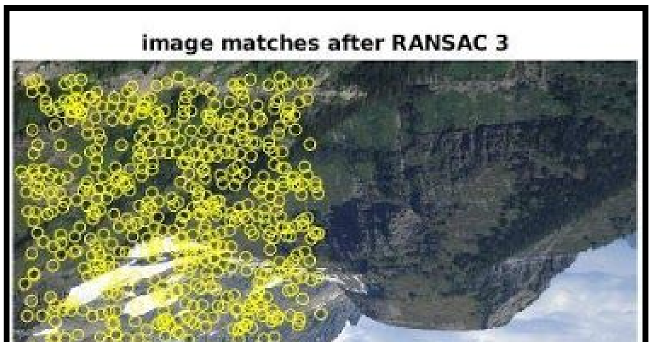
p(F is correct) = 1 − (1 − (pi) r) n .

After a large number of trials the probability of finding the correct homography is very high. For example, for an inlier probability pi = 0.5, the probability that the correct homography is not found after 500 trials is approximately 1×10^{-14} .

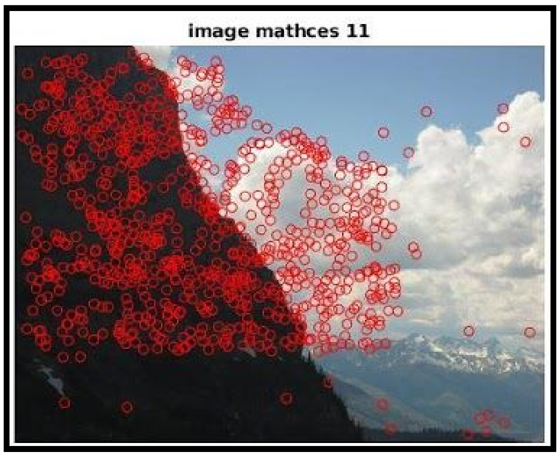
Ransac Output

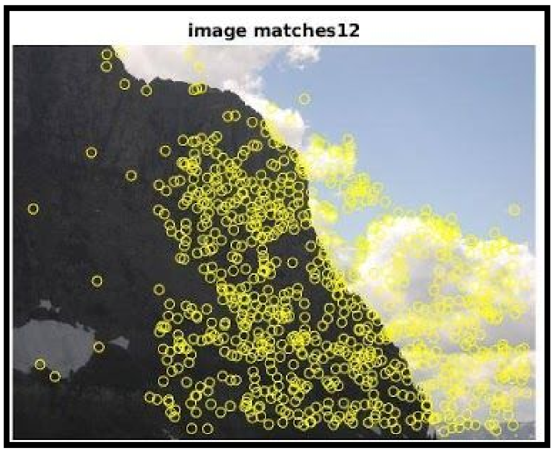
Rotation invariance

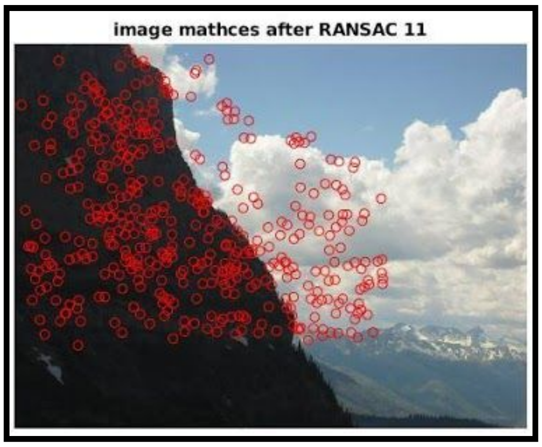


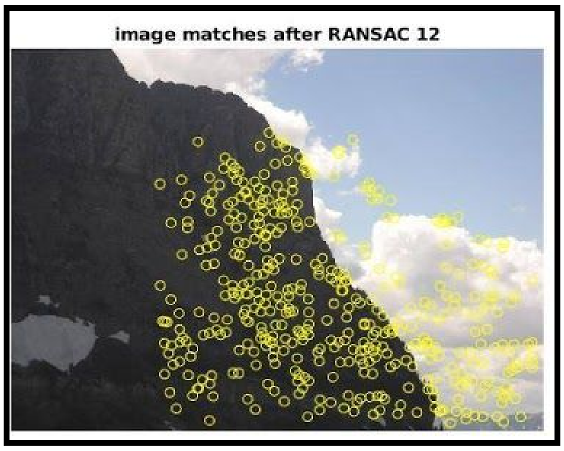


Illumination invariance

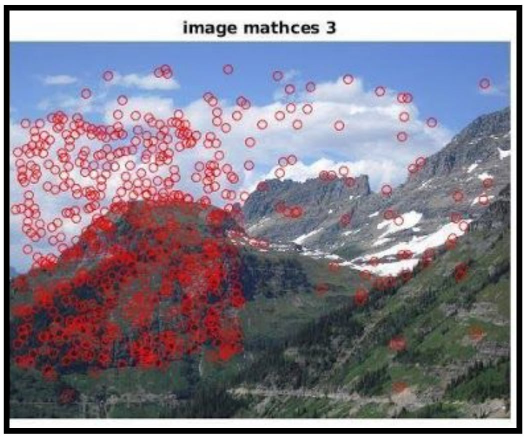


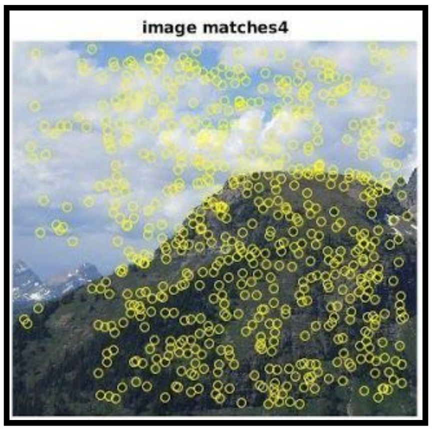


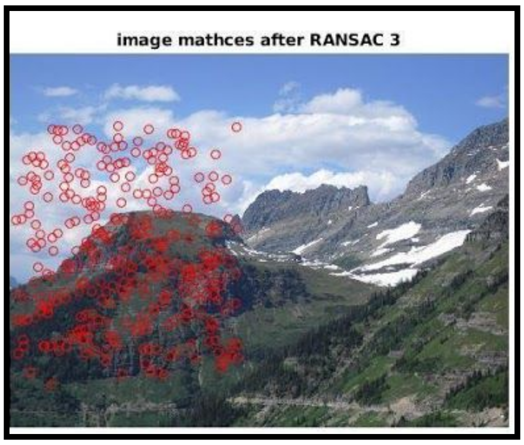


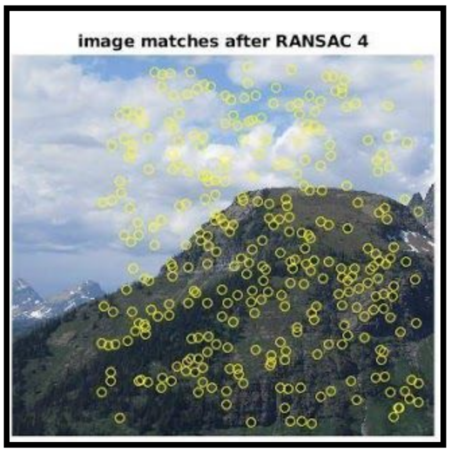


Scale Invariance





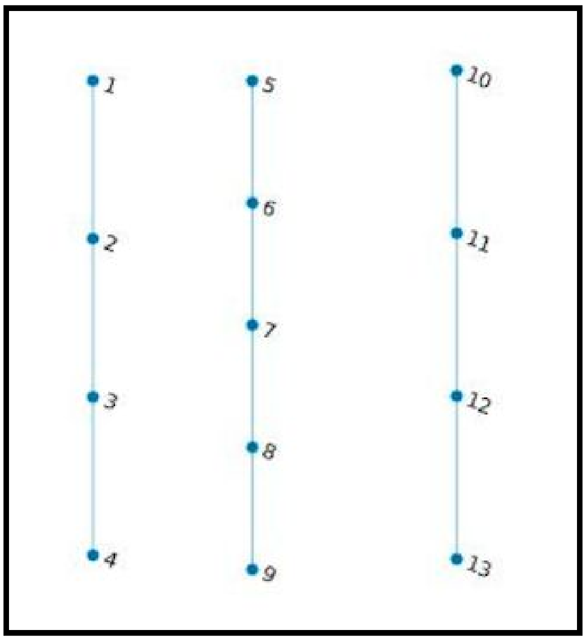




Finding the ordering of images :

After receiving the output of RANSAC inliers we find a way in which the images should be ordered for homography estimation. For each image I we find another image I' which is the best matched image i.e. with maximum number of inlier points and add it to the tree. Similarly then we consider all the neighbors of I' and take its best possible matched image. We keep on repeating this process till we form a linear unconnected tree. This process ensures that we consider the best matches at every instance which ensures maximum accuracy.

Ordering of the images on the output of the previous data looks like this:



Unconnected Tree after ordering

Bundle Adjustment:

After finding the best image matches we need to estimate the homographies between these images. Given the inlier correspondences between two images, we initially estimate the homography between the images using DLT by randomly selecting 4 points. We then minimize the nonlinear least square error in estimation of homographies using bundle adjustment. Bundle adjustment uses the Levenberg Marquardt (LM) algorithm to update the parameters. The objective function we use is a robustified sum squared projection error. That is, each feature is projected into all the images in which it matches, and the sum of squared image distances is minimised with respect to the camera parameters.

Given a correspondence u ki ↔ u lj (u ki denotes the position of the kth feature in image i), the reisual is

$$\mathbf{r}_{ij}^{k} = \mathbf{u}_{i}^{k} - \mathbf{p}_{ij}^{k}$$

where p^{k}_{ij} is the projection from image 'j' to image 'i' of the point corresponding to u ki.

$$\tilde{\mathbf{p}}_{ij}^{k} = \mathbf{K}_i \mathbf{R}_i \mathbf{R}_j^T \mathbf{K}_j^{-1} \tilde{\mathbf{u}}_j^l .$$

The error function is the sum over all images of the robustified residual errors

$$e = \sum_{i=1}^{n} \sum_{j \epsilon \mathcal{I}(i)} \sum_{k \epsilon \mathcal{F}(i,j)} h(\mathbf{r}_{ij}^{k})$$

where n is the number of images, I(i) is the set of images matching to image i, F(i, j) is the set of feature matches between images i and j. We use a Huber robust error function

$$h(\mathbf{x}) = \begin{cases} |\mathbf{x}|^2, & \text{if } |\mathbf{x}| < \sigma \\ 2\sigma|\mathbf{x}| - \sigma^2, & \text{if } |\mathbf{x}| \geq \sigma \end{cases}.$$

This error function combines the fast convergence properties of an L2 norm optimisation scheme for inliers (distance less than σ), with the robustness of an L1 norm scheme for outliers (distance greater than σ). We use an outlier distance $\sigma = \infty$ during initialisation and $\sigma = 2$ pixels for the final solution. This is a non-linear least squares problem which we solve using the Levenberg-Marquardt algorithm. Each iteration step is of the form:

$$\mathbf{\Phi} = (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{C}_p^{-1})^{-1}\mathbf{J}^T\mathbf{r}$$

where Φ are all the parameters, r the residuals and $J = \partial r/\partial\Phi$. We encode our prior belief about the parameter changes in the (diagonal) covariance matrix Cp

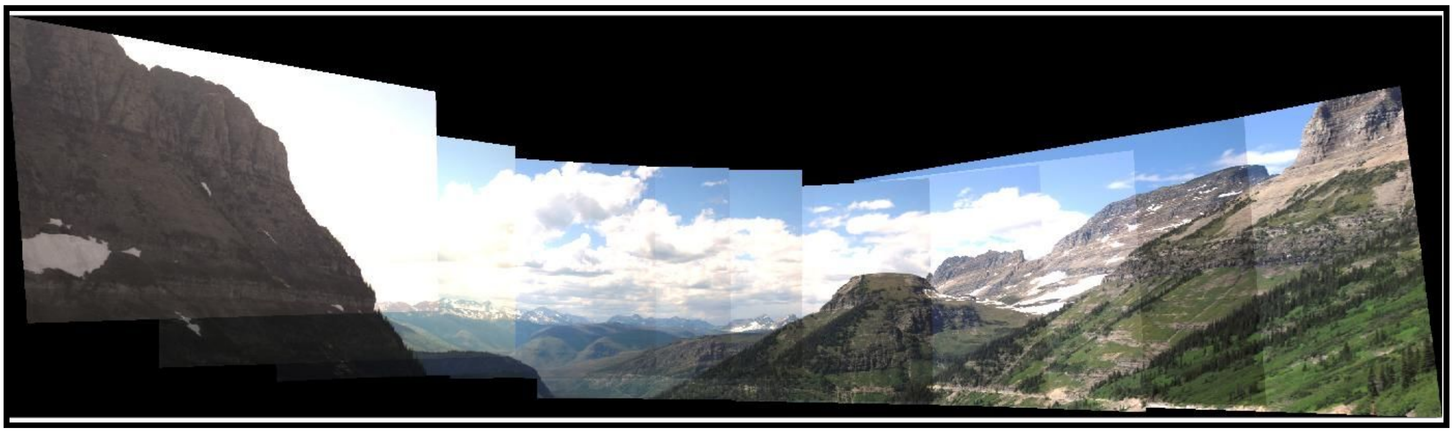
$$\mathbf{C}_p = \begin{bmatrix} \sigma_\theta^2 & 0 & 0 & 0 & 0 & \cdots \\ 0 & \sigma_\theta^2 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \sigma_\theta^2 & 0 & 0 & \cdots \\ 0 & 0 & 0 & \sigma_f^2 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \sigma_\theta^2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \end{bmatrix}$$

This is set such that the standard deviation of angles is $\sigma_\theta = \pi/16$ and focal lengths $\sigma_f = \bar{f}/10$ (where $\bar{f}$ is the mean of the focal lengths estimated so far).

Image Stitching with Blending:

After finding out the homographies of all the best image matches, we fix the frame about which we are about to construct the panorama. We find all the homographies about the fixed frame. All the image matches are stitched one by one starting from the the fixed frame. The fixed frame is most probably the image with highest number of image matches.

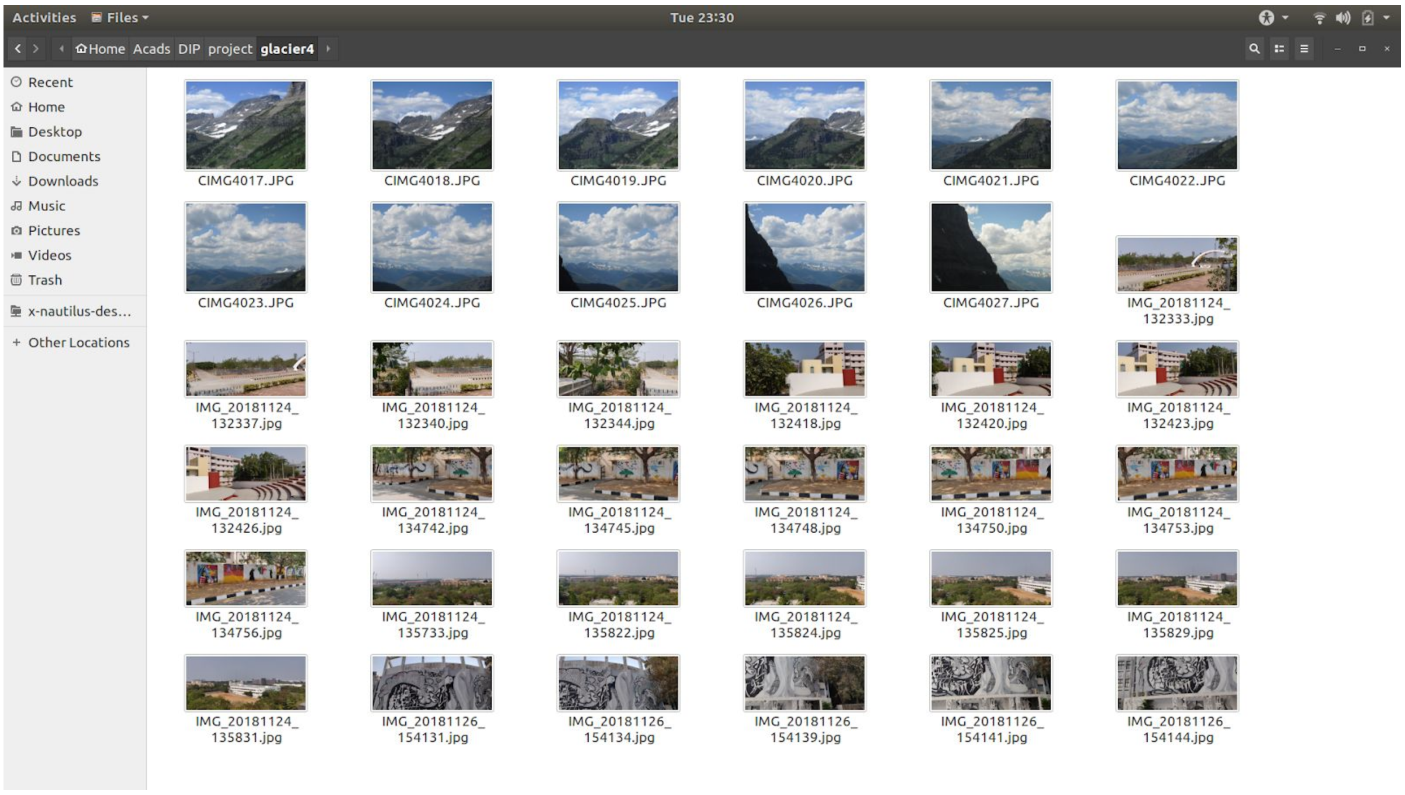
To stitch two images, we fix the frame of one image and then project the second image onto the frame of the first image using the homography matrix solved using bundle adjustment. Initially we define an empty array which is large enough to accommodate both the images. The projection of the the image will have an overlap with the first image. The pixel intensity in the overlap region is dealt with image blending, We used the multi band image blending algorithm to stitch the images. In the multi band blending algorithm, the frequency bands are extracted forming a pyramidal band and then smoothed across the various frequency bands.



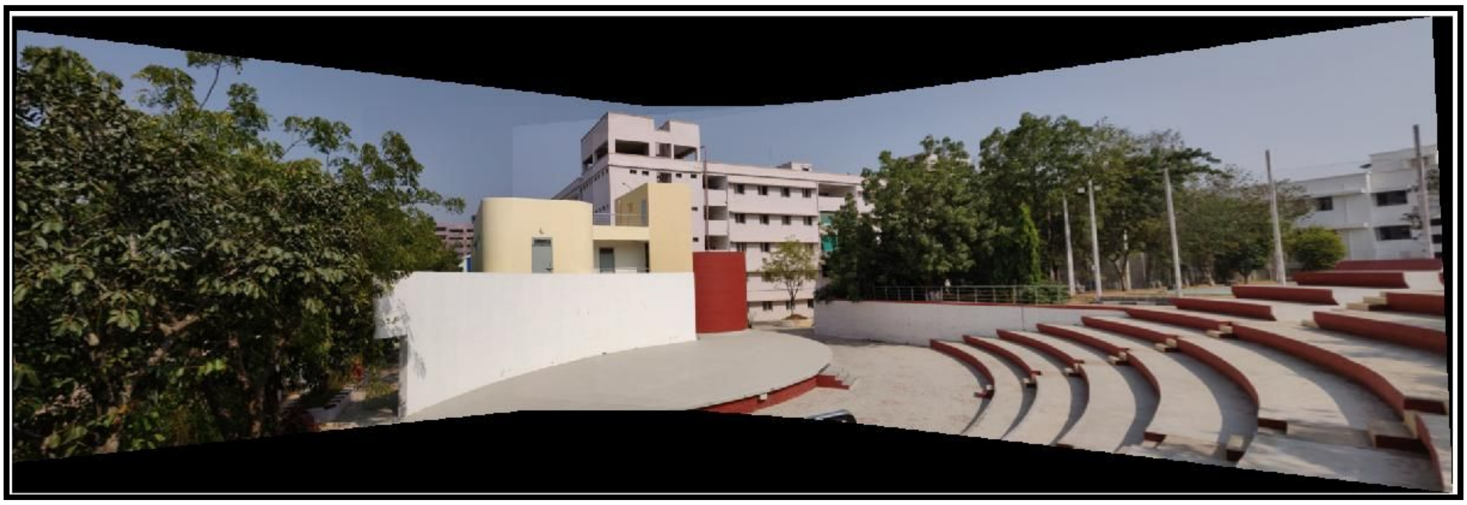
Output on Glacier Dataset

Results

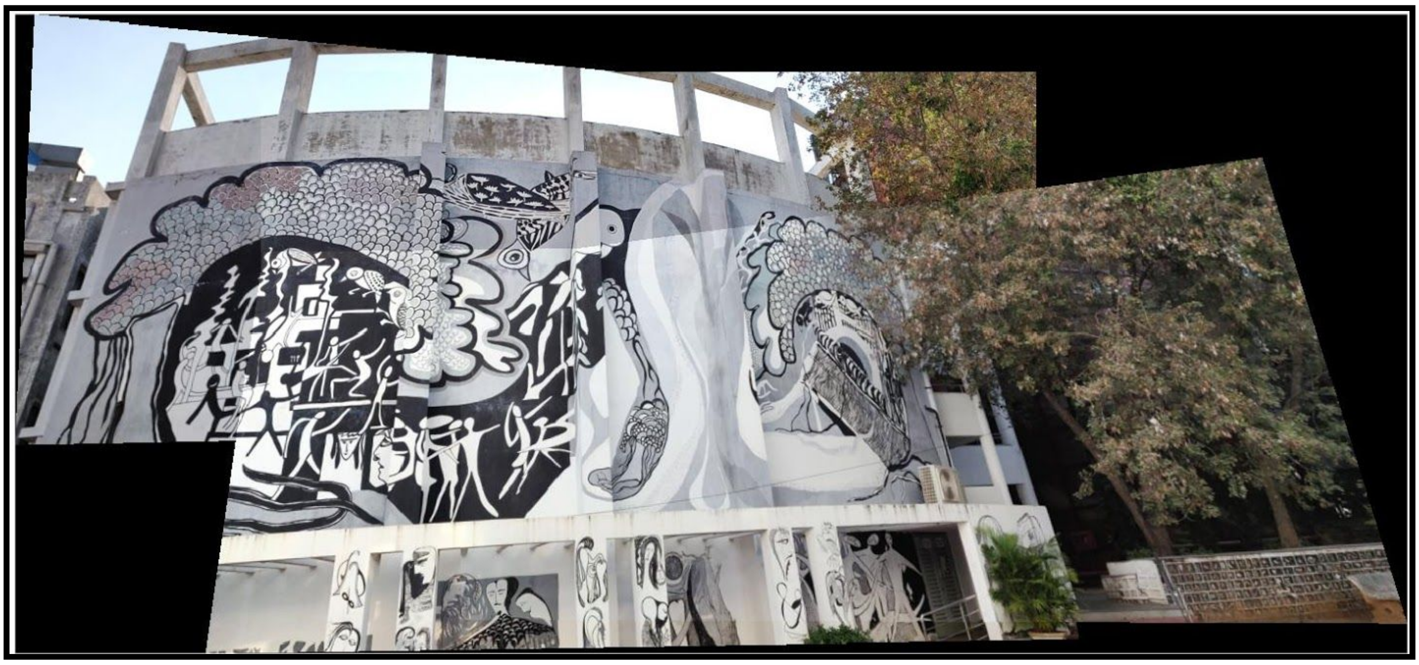
Input Dataset



Amphitheatre near Bakul Residency



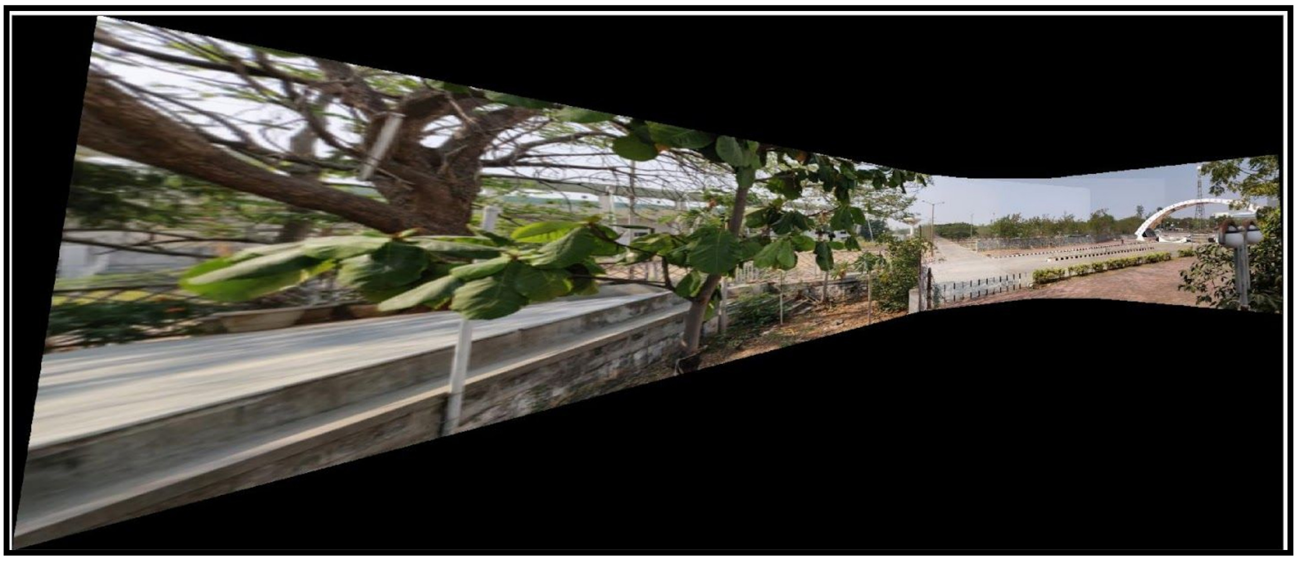
Himalaya Building in IIIT



Football Ground in IIIT (From T-HUB)

Wall Painting near NBH

Failure cases



Result Analysis

Following were some of the observations during the project:

- Varying the thresholds in the SURF features detection can improve or make the results worse depending upon the parameters.
- Finding connected components for multiple scenes helped to create multiple panoramas and remove noise images.
- Although SURF uses a feature descriptor to find image matches, still there were a number of false matches which were removed by using RANSAC (Random sample consensus).
- Levenberg–Marquardt algorithm used for minimizing non-linear least square errors can be take a very long time if the initializations are very poor.
- While stitching multiple images (around 7-8) a lot of memory requirement was observed which acted as one of the constraint to the algorithm.

Some of the cases where the project falls short are:

- If the image set consists of mixed photos of closeby and faraway objects of a single scene, then the panorama constructed will be stretched because of the nearby objects which will give rise to a homography.
- If the field of view is very large, the wavy effect will be high and panorama will not be accurate.

Future Milestones:

- **Image straightening**: The constructed panorama can be projected onto a common axis to remove the wavy effect. This is possible if the camera parameters are known.
- **360 degree panorama:** Project the images onto a cylindrical background to represent the 360 degree view.

Task Assessment

As discussed in the project proposal, project was supported to implement stitching for only one given scene. However within the given time frame it was possible to extend it to constructing multiple panoramas by finding connected components.
There were equal contributions from both the team members. It was the joint effort that helped us to complete the project within the given time frame.

Specific topics covered by individuals are:

Prakyath: Finding Image Matches, RANSAC, Bundle Adjustment, Image Stitching and Blending.

Aditya: Feature Extraction, Finding multiple panoramas, Ordering the Images, Bundle Adjustment.

Github Link:

https://github.com/adityaaggarwal97/DIP_Panorama-Stitching

Acknowledgement

- Paper link
- https://github.com/ppwwyyxx/OpenPano
- LM algorithm
- SURF
- Multiple view geometry in computer vision (Richard Hartley)