VR Assignment 2 Report

P. Ramsai Koushik (IMT2021072)
Nilay Kamat (IMT2021096)
Barath.S.Narayan(IMT2021524)

February 15, 2024

Contents

1 Stitching Images to create Panorama

1.1 Overview

In the process of image stitching, maintaining continuity between images while accommodating variations in lighting conditions, scale, rotation angle, and capturing position is important. The initial phase involves identifying interest points through feature extraction using techniques such as SIFT or SURF. Once feature extraction is complete, matching algorithms, such as Brute Force Matcher or KNN Matcher, are employed to match these interest points. The resulting best-matched points are then utilized to compute the Homography matrix using OpenCV's `findHomography` function. After finding the best-matched points, we use the Homography matrix obtained to smoothly combine the images by warping one of them through OpenCV's `warpPerspective` function.

Steps followed for Image Stitching:

1. **Keypoint and Descriptor Extraction:** Extraction of keypoints and their descriptors(representations) from images using the SIFT algorithm, capturing distinctive features that define each image. The SURF algorithm is a patented algorithm, hence it is not possible to run the algorithm to extract the descriptors.

2. **Matching Interest Points:** Implementation of feature matching algorithms such as Brute Force Matcher or KNN Matcher, in order to find correspondences between keypoints detected across images.

3. **Draw Matched Interest Points:** Displaying matched interest points on the 2 images.

4. **Homography Matrix Computation:** Computation of the Homography Matrix that is used to describe the transformation between 2 image planes. Implementation is done using OpenCV's `findHomography` function and the usage of RANSAC to help provide an accurate estimate of the homography matrix.

5. **Stitching of the Images:** Implementation of Image combination using results from above steps and with the help of OpenCV's `warpPerspective` function. Using the **interest points** and **features** obtained from the **Keypoint and Descriptor Extraction** step, we perform **Interest Point Matching** on these interest points. Then, the Homography Matrix is computed and OpenCV's `warpPerspective` function is used to warp the images together by applying the Homography transformation.

1.2 Differences between SURF and SIFT

- SURF uses box filters whereas SIFT uses Gaussian filters.
- SURF scales the filter while keeping the image size the same, whereas SIFT scales the image.
- SURF uses a 64 dimensional vector for the descriptor whereas SIFT uses a 128 dimensional vector.
- SURF performs an initial filtering step based on intensity to speed up matching.
- SURF is generally faster than SIFT, making it more suitable for real-time applications.

1.3 Matching Algorithms

1.3.1 BruteForce Matcher

The brute force matcher as the name suggests takes one feature in the first set and is matched with all the other features in the second set using the Euclidean distance. The k best matches are returned which are selected by KNN and the ones with euclidean distance below a certain value is chosen while the others are discarded.

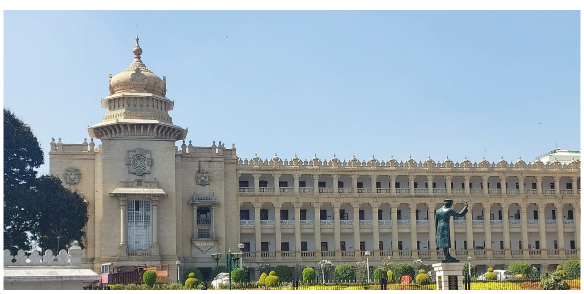
1.3.2 Flann Matcher

FLANN is a python library which stands for Fast Library for Approximate Nearest Neighbours. In general, FLANN works by first building an index of the data set using one of the supported indexing methods. Once the index is built, the user can perform nearest-neighbor searches by querying the index with a new point. The index returns the nearest neighbors to the query point, along with their distances. Both Randomized k-d Tree and Priority-based K-Means are efficient approximate nearest neighbor search algorithms that can be used in FLANN, the choice of algorithm depends on the specific requirements of the application. The main advantage of using this matcher is its speed and the fact that it optimizes the parameter on its own along with choosing the best algorithm for the dataset.

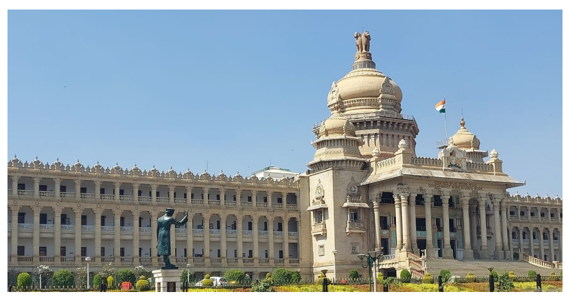
1.4 RANSAC

The Random Sample and Consensus Algorithm is used to detect and remove outliers to fit Machine Learning Models better. RANSAC is an iterative algorithm to fit the linear model, but unlike linear regression, it does not consider all the data present in the dataset. Instead, it only uses a subset of data called inliers to fit the curve and estimate parameters. Once it fits the curve, it calculates how many of the remaining data points are inliers (Ones which lie on the curve) and how many are outliers (Ones which lie outside the curve). The curve which has the least number of outliers is then chosen, and the set of the inlier points is considered the consensus set. Based on this the homography matrix is calculated to get the perspective transformation between the two images. The homography matrix then transforms the image so that it is suitable for stitching.

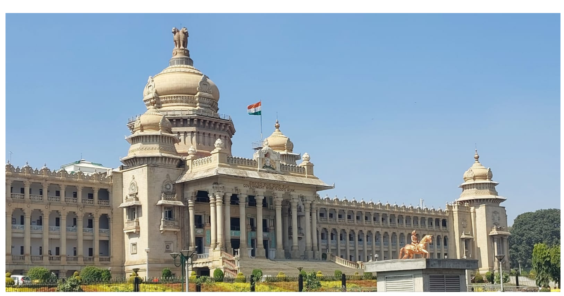
1.5 Results:



(a)

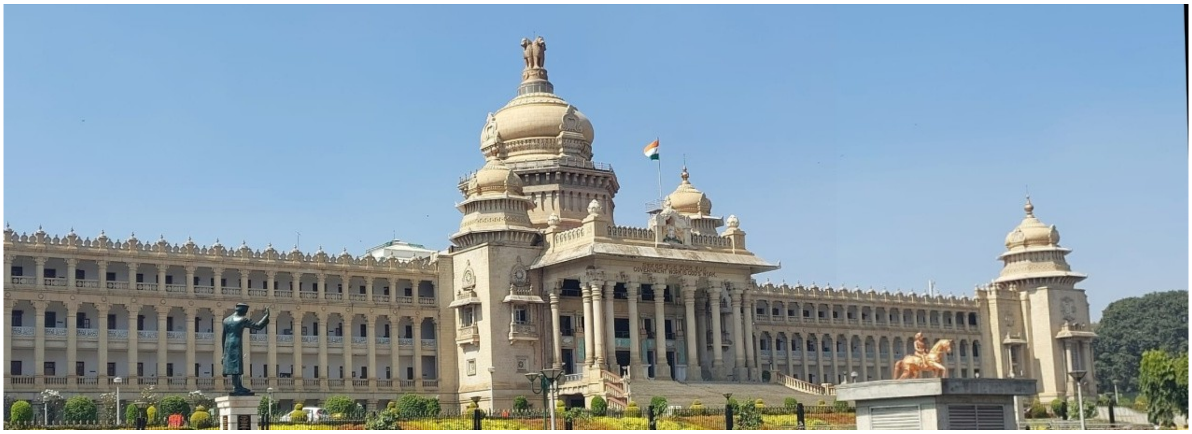


(b)



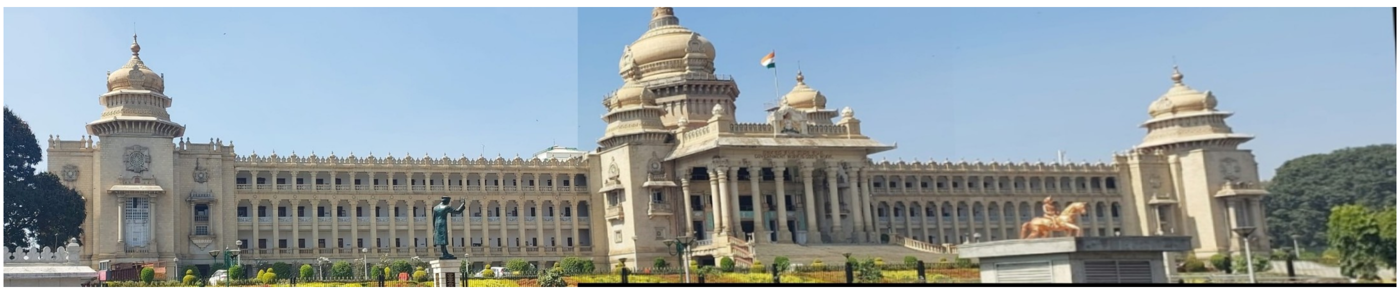
(c)

Figure 1: Snapshot showing (a) Left section of Vidhan Soudha, (b) Central section of Vidhan Soudha and (c) Right section of Vidhan Soudha.



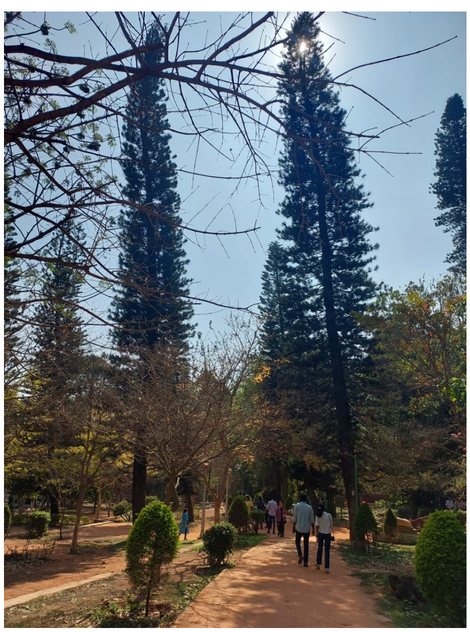
(a)

Figure 2: Snapshot showing result obtained after stitching 1 (b) and (c)

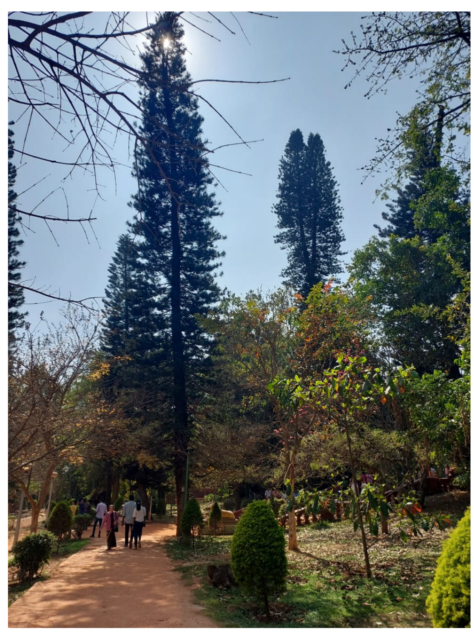


(a)

Figure 3: Snapshot showing result obtained after stitching 1 (a), (b) and (c)

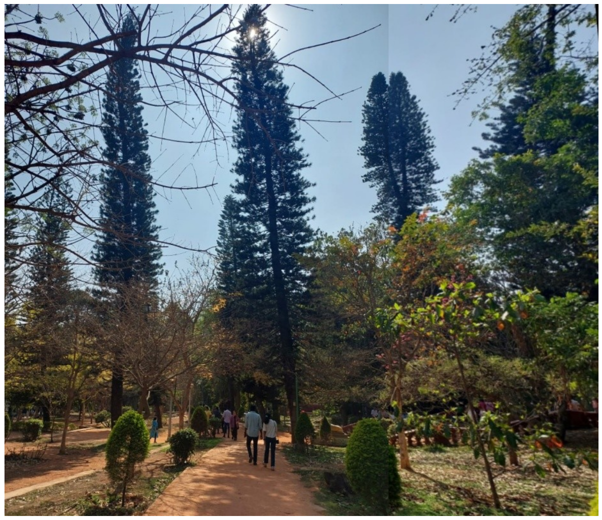


(a)



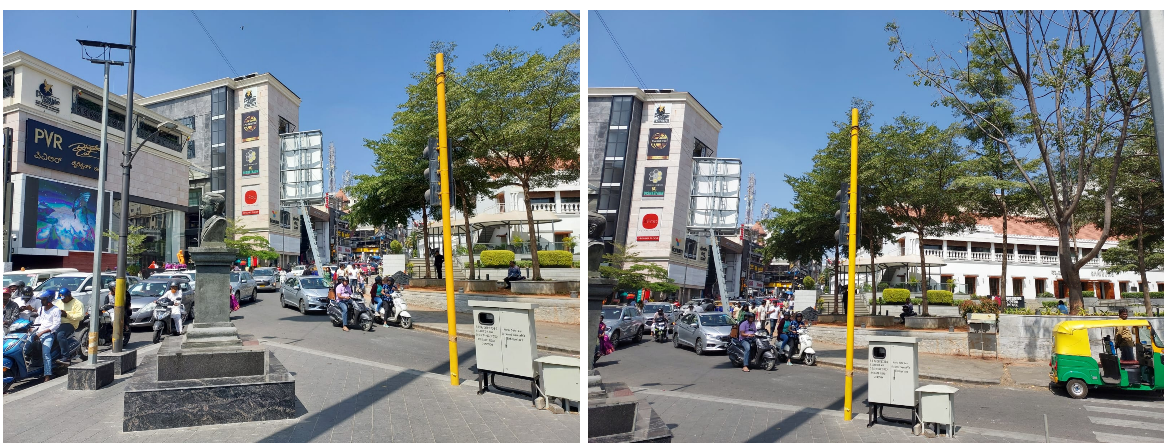
(b)

Figure 4: Snapshot showing (a) Left section of Cubbon Park, (b) Right section of Cubbon Park.



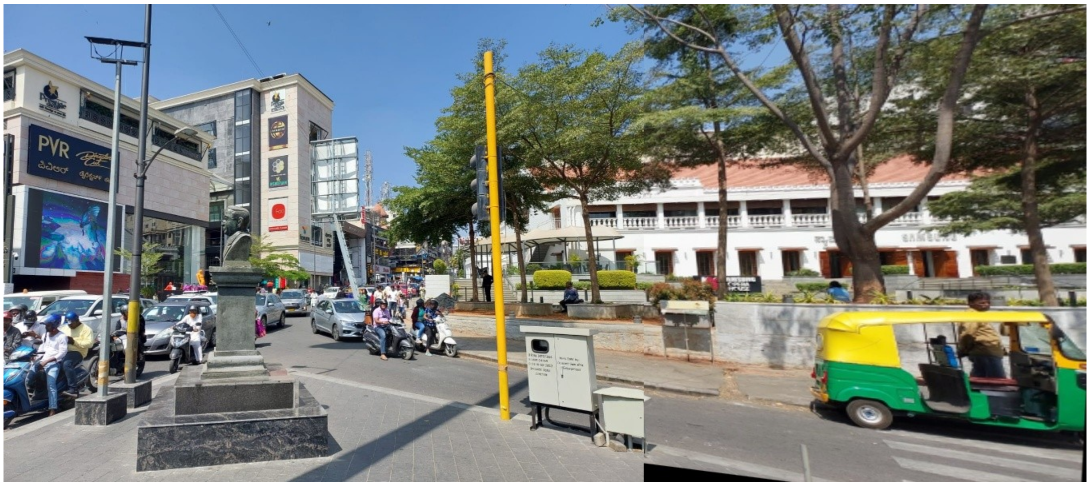
(a)

Figure 5: Snapshot showing result obtained after stitching images 4



(a) (b)

Figure 6: Snapshot showing (a) Left section of Brigade Road junction and (b) Right section of Brigade Junction.



(a)

Figure 7: Snapshot showing results obtained after stitching images 6.

1.6 Observations:

When the images are being stitched, due to the computation of the Homography matrix, we observe that the resulting image may have one or both of the original images at some deviation from the horizontal line. We can see this in 2 where the right edge of the image has a small blacked out area. This is due to the rotation of the right side image by a certain degree for effective stitching and to ensure the keypoints are overlapped correctly.

2 Classification of Images using Bag of Visual Words approach

2.1 Method Overview

The bag of visual words is a method extensively utilized in computer vision for image representation and classification. In this approach, images are characterized by a set of visual words, resembling a vocabulary, generated through the clustering of descriptors of keypoints in an image. Initially, keypoints and their respective descriptors are extracted from images using algorithms such as SIFT or SURF. But since each image might have different number of descriptors, these descriptors are then grouped into clusters employing KMeans clustering, where each cluster serves as a visual word. Following this, the frequency distribution of descriptors across these visual words is calculated for each image, producing a fixed-length feature vector. This method efficiently captures image characteristics, enabling effective image representation and subsequent classification using machine learning models.

Steps for classification of images:

1. **Keypoint and Descriptor Extraction:** Extract keypoints and their descriptors from images using the SIFT algorithm, capturing distinctive features that define each image.
2. **Descriptor Clustering for Feature Length Standardization:** Implement clustering techniques, such as KMeans, on the collected descriptors from all images, ensuring a standardized and fixed feature length for subsequent analysis.
3. **Frequency Distribution Calculation:** Calculate the frequency distribution of descriptors among clusters for each image, generating unique fixed-length feature vectors that succinctly represent the visual content of the images.

4. **Feature Vector Utilization in Classification:** Employ the computed frequency histograms of images as features to train diverse classification models, including KNN, logistic regression, and SVM, allowing the models to learn and discern patterns within the feature space.

5. **Model Testing and Accuracy Assessment:** Test the trained classification models on a separate test dataset, evaluating their performance and obtaining accuracy scores to quantify the models' effectiveness in accurately classifying images.

2.2 Classification of Bike vs Horse

2.2.1 Dataset Description

For this classification task, we were provided with a specific dataset that comprised two distinct folders of images – one containing pictures of bikes and the other featuring images of horses.

2.2.2 Observations and Results

2.2.3 Experimentation

We followed the 3-step approach , where in first step we extracted the descriptors of all the images, in the second step we cluster the descriptors using KMeans to create a vector of fixed length and in the third step we used various ML algorithms like SVC, Logistic Regression, K Nearest Neighbours to classify images.

For K Means we experimented on different **k**(number of clusters) values to find the optimum number of clusters to get the best accuracy. With regards to the ML models, we tried different hype-rparameters for **c** in SVC and neighbours in KNN.

The following are the accuracies for different combinations of hyper-parameters

Configuration	**SVC**	**SVC(linear kernal)**	**LR**	**KNN (n=5)**
k=10	75	75	75	77.77
k=100	88.88	88.88	97.22	88.88
k=200	88.88	94.44	**97.22**	94.44

Table 1: Accuracy(%) values for different configurations.

2.3 Classification of Cifar 10 Dataset

2.3.1 Dataset Description

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. The 10 classes in the dataset are airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. There are 50000 training images and 10000 test images.

The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

2.3.2 Observations and Results

2.3.3 Experimentation

We followed the 3-step approach , where in first step we extracted the descriptors of all the images, in the second step we cluster the descriptors using KMeans to create a vector of fixed length and in the third step we used various ML algorithms like SVC, Logistic Regression, K Nearest Neighbours to classify images.

For K Means we experimented on different **k**(number of clusters) values to find the optimum number of clusters to get the best accuracy. With regards to the ML models, we tried different hype-rparameters for **c** in SVC and neighbours in KNN.

The following are the accuracies for different combinations of hyper-parameters

Configuration	**SVC (c=12)**	**LR**	**KNN (n=20)**
k=10	21.20	21.08	18.90
k=50	22.63	25.54	18.99
k=100	23.46	**26.745**	16.31

Table 2: Accuracy(%) values for different configurations.