Assignment 2

Mayank Chadha — IMT2020045

Instructor: Prof. Dinesh Babu

Date: February 15, 2023

1 Play With Panorama

1.1 Assumptions

Keep 1*.jpg* and 2*.jpg* in the same directory as the code, the program stitches both of them and generates *output.jpg*.

1.2 SIFT vs SURF

- SURF is fundamentally faster, by an enormous amount, than SIFT if you were to count the FLOPS of two well-written implementations.
- SIFT computes an image pyramid by convolving the image several times with large Gaussian kernels, while SURF approximates that using integral images.
- In SIFT, Lowe approximated Laplacian of Gaussian with the Difference of Gaussian for finding scale-space. SURF goes a little further and approximates LoG with Box Filter. One significant advantage of this approximation is that convolution with a box filter can be easily calculated with the help of integral images. And it can be done in parallel for different scales.
- SIFT has good results than SURF in scale area. SURF has good results in rotation, blur, warping, RGB noise, and time consumption than SIFT. In illumination, both have the same effect on find detect feature points.

1.3 FLANN and RANSAC

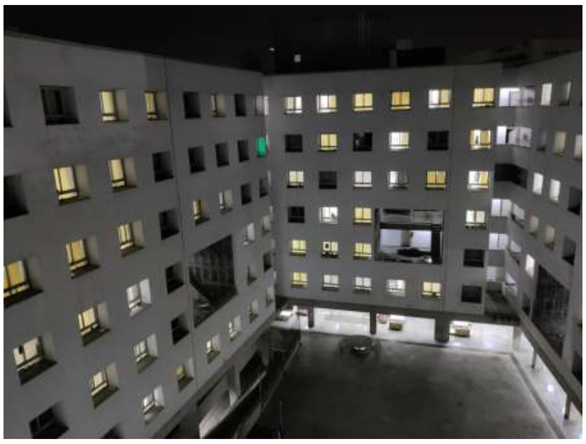
1.3.1 FLANN

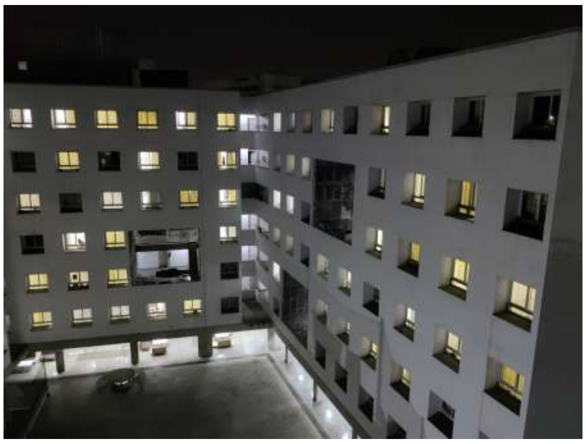
FLANN stands for Fast Library for Approximate Nearest Neighbors. It contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features. These algorithms project the high-dimensional features to a lower-dimensional space and then generate the compact binary codes. Benefiting from the produced binary codes, fast image search can be carried out via binary pattern matching or Hamming distance measurement, which dramatically reduces the computational cost and further optimizes the efficiency of the search. FLANN will find a good match, but not necessarily the best one.

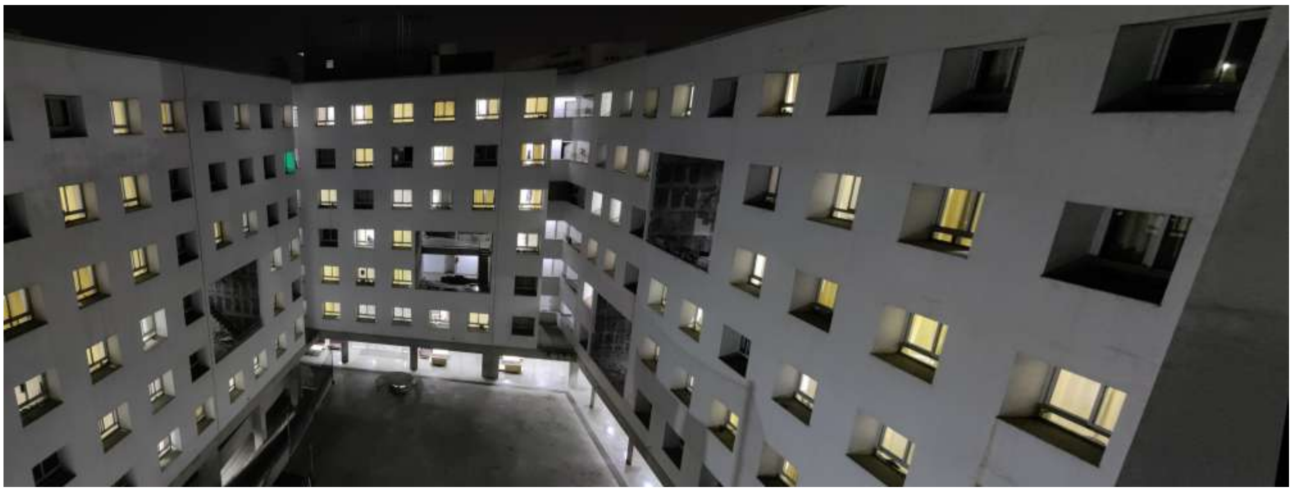
1.3.2 RANSAC

Random sample consensus (RANSAC) is an iterative method to estimate the parameters of a mathematical model from a set of observed data that contains outliers when outliers are to be accorded no influence on the values of the estimates. Therefore, it also can be interpreted as an outlier detection method. It is a non-deterministic algorithm in the sense that it produces a reasonable result only with a certain probability, with this probability increasing as more iterations are allowed.

1.4 Results


Left Image


Right Image


Stitched Image

2 Image Classification

2.1 Bike vs Horse

2.1.1 Assumptions

Keep the folder Assignment2_BikeHorses as provided by Sir in the same directory as the code before running.

2.1.2 Procedure

In computer vision, the bag of visual words (BOVW) is a powerful technique that allows us to represent images as sets of features. These features are comprised of key points and descriptors. Key points are salient points in an image that remain stable under various transformations such as rotation, scaling, or translation. Descriptors are the mathematical representation of these key points. By using key points and descriptors, we can construct vocabularies and represent each image as a frequency histogram of features in the image. This histogram can be used to identify similar images or predict the category of the image.

2.1.3 Approach

1. Loading the training data: We begin by loading the images and their corresponding labels for the training data.
2. Image Descriptors: Next, we extract image descriptors that can be used to identify the images. We utilize SIFT descriptors to extract features from the images.
3. Descriptor Lists: We then create a list of descriptors that defines a certain class of images.
4. We create a list of descriptors that define a certain class of images.
5. Visual Vocabulary: After extracting features and descriptors, we build a visual dictionary (vocabulary) by using k-means clustering to create clusters from the descriptors. The center of each cluster is used as a visual vocabulary.
6. Bag of Visual Words: Finally, for each image, we create a frequency histogram from the visual vocabulary to determine the frequency of each word in the image. These histograms are our bag of visual words (BOVW).

2.1.4 Observations

For the keypoint detection algorithm SIFT, I used k-means clustering with $k = 128$ and extracted the features from the data. Then, I applied three different classification algorithms to the extracted features. When I increased the number of clusters from 128 to 512 for all three algorithms, I observed only a slight improvement in accuracy, which varied across multiple runs due to the randomness of the machine-learning models.

Moreover, I found that the runtime almost tripled when I increased the number of clusters. Therefore, it is important to carefully consider the tradeoff between the number of clusters and the runtime when applying k-means clustering for feature extraction. In my specific case, increasing the number of clusters did not yield a significant improvement in accuracy, but it did increase the runtime considerably.

Model	Parameters	Accuracy
SVM	C = 0.005, kernel='linear'	0.972
LR	max_iter = 1000	0.972
KNN	n_neighbors = 5	0.944

Table 1: Accuracy Score with corresponding models

2.2 CIFAR - 10

2.2.1 Assumptions

Keep the cifar-10-batches-py directory in the same directory as the code.

2.2.2 Procedure and Approach

In order to perform a classification task, we follow a standard approach similar to that of binary classification. However, there is a slight difference in the way data is loaded for this specific task. The dataset is provided in the form of batch files, which are processed using an “unpickle” method from the Python “pickle” library. This method extracts a dictionary consisting of labels, filenames, and images represented as NumPy arrays of pixel values. This enables us to load and process the data efficiently for our classification task.

2.2.3 Observations

I experimented with the number of clusters (k) in k-means clustering. Specifically, I initially set k to be 128, and then increased it to 512. I found that although increasing the number of clusters had little impact on the accuracy of the model, it significantly increased the runtime, which almost tripled.

It's important to note that due to the randomness inherent in machine learning models, the reported accuracy may differ between runs. Additionally, I did not use all six data batches in my experiments because concatenating them and running the code on the concatenated data was taking a lot of time and had the potential to cause a kernel crash. As a result, the reported accuracy may be lower than it would be if all of the data were used.

K-nearest neighbors (KNN) caused my kernel to crash, so I was unable to report the accuracy that it produced.

Model	Parameters	Accuracy
SVM	C = 0.005, kernel='linear'	0.2537
LR	max_iter = 1000	0.2582

Table 2: Accuracy Score with corresponding models