DSAA Project Report

Members – Prazwal Chhabra (20171192), Sanchit Saini (20171191), Kunal Vaswani (20171064)

We tried various approaches before coming to our final one. The list of initial approaches are as follows -

- **Immse** – We calculated the mean square error between te images and accordingly gave the output as the frame with the least error with the slide. This method gave a lot of error as it essentially squares the pixels ignoring negative values and also if the images are a bit disoriented, it gives almost negligible similarity.
- **SSIM -** The SSIM index is a full refrence metric ,in other words, the measurement or prediction of image quality is based on an initial uncompressed or distortion-free image as reference. SSIM is a perception based model that considers image degradation as perceived change in structural information, while also incorporating important perceptual phenomena, including luminance masking and contrast masking terms. Accuracy increased but not to much extent.
- **Eigenfaces** – In this approach, we express each image in the original dataset as a vector and subtract from every image its mean. Then we compute the covariance matrix and find the k best eigenvectors (using best eigenvalues) to represent the images. Then we compute the projection of each image onto these eigenvectors.

  When comparing with test dataset, we project the test image to the k best eigenvectors and find the most similar image in the original dataset using the least mahalanobis distance among the images in the original dataset and output this

image as the most simialr image. This approach made the performance better but there were some drawbacks with it like same scaling, background issues and the most major issue was lighting which was prevalent in the frames.

Final Approach

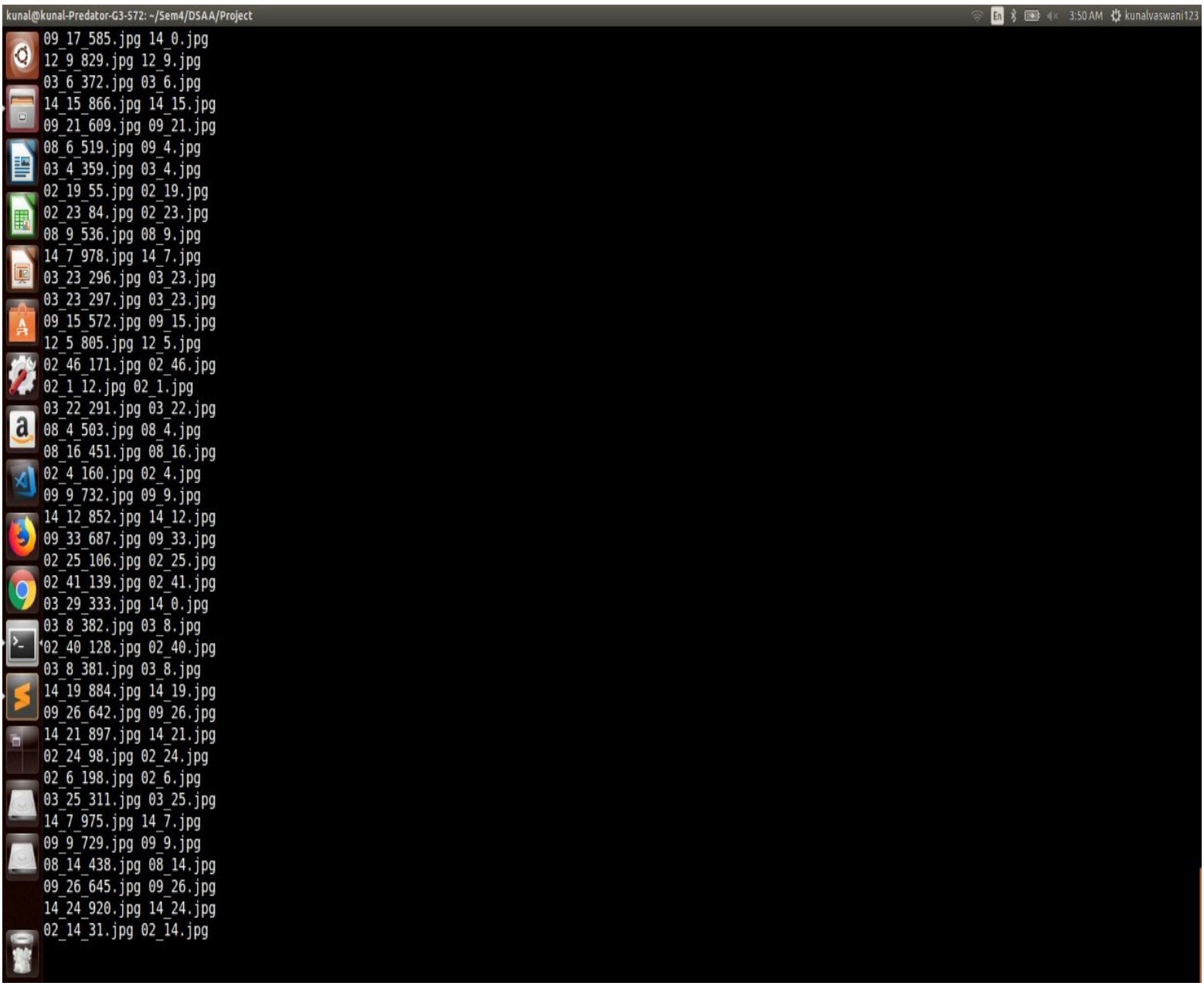
We have used a different version of normalized 2-D cross correlation.
In this approach, we first calculated the deviation of each entry of the image with its mean for both the images, i.e.
We calculated
imageA1 = imageA – imageA.mean()
imageB1 = imageB – imageB.mean()

Then we performed the point wise multiplication of these matrices and calculated the mean of the new obtained matrix (product).

Finally we checked whether the multiplication of the standard deviation of both the images is 0. If it is 0 we returned 0 as the result else we returned the (mean of the product divided by the multiplication of the standard deviations).

Another thing we did to improve the accuracy was to add **hazy noise** to the slides as we were not able to effectively denoise the frames. When hazy noise was added to the slides, it increased the structural similarity between the images thereby increasing the accuracy. It increased the accuracy by 10%.

Also we tried to vectorise the code as much as possible to make the process more efficient in terms of time taken and to make the code as short as possible.



This implementation produced an accuracy of **94.5 %** in the dataset provided to us.