**CHAPTER 5**

**CONCLUSION AND RECOMMENDATION**

CONCLUSION

The reliability and efficiency in a face recognition algorithm is of the most importance. That is why it is very crucial for such an algorithm to adapt to certain conditions to perform better. One recognizable asset of the Eigenface algorithm is that it does recognition purely via mathematical means that only requires low-level processing and its ability to compress large datasets into a small number of eigenfaces.

To minimize the processing cost for the solution, a mathematical solution for has been implemented. Adjustment of brightness, also darkness, of the images will take place when input images are taken. Adding first-hand solution to the illumination issue makes the algorithm flexible to different lighting conditions and since it is the common issue for eigenface, we try to approach each and every images taken with this solution to avoid any inconsistency.

For a given set of images, due to high dimensionality of the images, the space is relatively large. By using Principal Component Analysis’ (PCA) Dimension Reduction, we are try to lessen the dimensionality for the algorithm to make it easier to attempt recognition. However, the Dimension Reduction won’t be able to maintain its purpose due to the increasingly growing number of training faces. Devoted to efficiency, it is very important to dismiss the Dimension Reduction when this period comes to prevent enlargement of the processing level.

Such procedures are implemented and improved so that the results would yield more reliable results.

RECOMMENDATION  
 To improve the results of the Eigenface algorithm, a normalization procedure could be added to determine the facial metrics and normalize the photos such that facial features are held in constant positions. Localizing of the features would keep head positions fairly consistent and yield better eigenfaces.

Creating a certain method to extract facial features to further expand the techniques of recognition, improving the accuracy even further.x