# AGE GROUP ESTIMATION FROM FACE IMAGES

TIONG PEI KEE

**A project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering (Hons) Electronic Engineering**

**Faculty of Engineering and Green Technology**
**Universiti Tunku Abdul Rahman**

**September 2015**

## DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

| | |
|---|---|
| Signature | : ___________________________ |
| Name | : Tiong Pei Kee |
| ID No. | : 10AGB03409 |
| Date | : ___________________________ |

## APPROVAL FOR SUBMISSION

I certify that this project report entitled **"AGE GROUP ESTIMATION FROM FACE IMAGES"** was prepared by **TIONG PEI KEE** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons) Electronic Engineering at Universiti Tunku Abdul Rahman.

Approved by,

| | | |
|---|---|---|
| Signature | : | _________________________ |
| Supervisor | : | Dr. Humaira Nisar |
| Date | : | _________________________ |

# ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. First and foremost, I would like to express my utmost gratitude to my supervisor Dr. Humaira Nisar for her supervision, invaluable advice and guidance throughout the development of the research.

In addition, I would also like to express my deepest appreciation to my loving parents and family members for their constant support and encouragement. This work would not have been possible without their support.

Last but not the least, I am grateful for the unselfish cooperation and assistance that my friends had given me to complete this task.

# AGE GROUP ESTIMATION FROM FACE IMAGES

## ABSTRACT

Age group estimation is useful in real-world applications such as security access control and human computer interaction. We have proposed an age group estimation algorithm based on the wrinkle features on the face image. During pre-processing stage, geometric normalization is performed to correct the out-of-plane rotated images. Then, conversion of image to grayscale image is performed, if needed; followed by noise removal using median filtering method. Wrinkle features are extracted from the regions of interest of a normalized image using Canny edge detection for age group estimation. Finally, the images are classified into three age groups: babies/ children, young adults and old adults. The average accuracy of the algorithm is 72.66% for good quality images and 44.92% for poor quality images.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF APPENDICES

# CHAPTER 1

# INTRODUCTION

The main focus of this project is age group estimation from face images. Some images may have been acquired under unconstrained condition. In order to achieve better performance, the input images of this proposed approach are limited to frontal and upright facial images that consists of single face with spectacles free and clear forehead region.

## 1.1 Background

Facial images consist of many facial attributes such as wrinkles, face angle and information of facial geometry. These features play an important role in applications of facial image analysis. There is a wide variety of information that can be obtained from a facial image analysis, including identity, race, age and gender. The identification characteristic of face images has already been implemented in real-world applications including passports and access control in a security area. However very limited amount of studies have been done on age group estimation from face images.

Kwon and Lobo (1994) presented a theory and practical computations for age classification from facial images. Their computations are based on skin wrinkle analysis and also cranio-facial changes in ratios of feature-position. They classified the facial images into three age groups: baby, young adult and senior adult.

Cootes, Edwards and Taylor (2001) proposed a new method of matching statistical models of appearance to images. In their proposed approach, statistical shape model and intensity model are learned separately using facial images. This field of study has been extended to facial aging by Cootes, Lanitis and Taylor (2002). They described how aging process affects facial appearance. Simulation of aging effects on face images can be performed using their proposed framework. In other words, the look of a particular individual in the past or in the future could be predicted.

AGES (AGing pattErn Subspace), was developed by Geng, Kate and Zhou (2007) to model the aging pattern of an individual. Aging pattern is defined as the sequence of a particular individual's face images sorted in time order. The projection in the subspace could reconstruct a previously unseen face image with least reconstruction error. The estimated age of an input face image is represented by its position in the aging pattern.

## 1.2 Problem Statement

The paper proposed by Hayashi et al. (2002) suggested the age and gender estimation based on wrinkle texture. However they had difficulties in extracting wrinkles in females with age 20 to 30 years old. Lin et al. (2012) proposed an automatic age estimation system in their research, but found out that there was no efficient method that can automatically align feature and points quickly and correctly. Eidinger, Enbar and Hassner (2013) found out that the performance of the system drops when they used their own Adience benchmark.

Therefore there is a need to develop an algorithm to extract suitable features correctly and accurately. At the same time, suitable databases have to be chosen so that it is able to train and test the system developed without affecting the performance.

## 1.3 Aim and Objectives

This project is to develop an algorithm to perform age group estimation from different face images. In order to perform this, the following objectives have to be achieved:

i. To perform pre-processing of face images
ii. To detect and localize face and regions of interest in a face image
iii. To perform feature (wrinkle) extraction from the face and classification of wrinkle feature to estimate age group of the face

## 1.4 Thesis Organization

Chapter 2 includes a brief review on the approaches and studies that have been done previously by other researchers. Methods and approaches used to obtain the desired output would be described in Chapter 3. Results of this proposed approach would be attached and discussed in Chapter 4. Conclusion, as well as some recommendations would be made in Chapter 5.

# CHAPTER 2

# LITERATURE REVIEW

This chapter is to establish the significance of the general field of study, and find a place where a new contribution and improvements could be made. Different methodologies used in field of age group estimation will be identified here. Literature review of this paper is classified into several parts, which include face detection, pre-processing, feature extraction, age group estimation and evaluation of performance. Table 2.1 at the end of this chapter shows a brief summary of literature review.

## 2.1 Face Detection

There are different types of face images and face detection methods that the previous researches worked on. Unconstrained and constrained face images could be used for estimation of age group or exact age. Some of them implemented Matlab built-in object function while there are researchers who developed their own algorithms to detect facial features.

The paper proposed by Han and Jain (2014) worked on age group estimation from face images acquired under unconstrained conditions. These images may have more than one face in the image or may be ill-posed, and they have to go through pre-processing stage such as pose and illumination correction in order to improve the performance of the system. Figure 2.1 shows the unconstrained face images from that Labeled Faces in the Wild (LFW) database used in their proposed approach and the face detection.

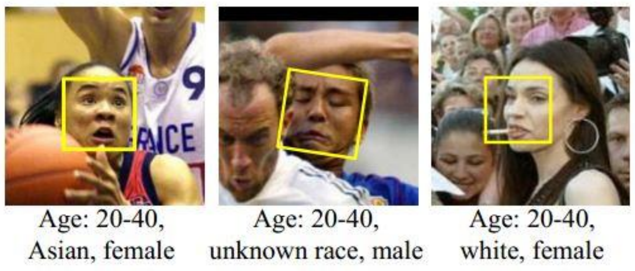


Figure 2.1: Examples of Unconstrained Face Images from the LFW database (Han and Jain, 2014)

Most of the previous papers used face images acquired under constrained and cooperative scenario, or face databases available online. These images are usually frontal upright face image that consists of only one face and free from glass, moustaches and beard.

Three different papers proposed by Jana, Pal and Chowdhury (2012), Eidinger, Enbar and Hassner (2013) and Jana, Datta and Saha (2013) respectively, applied Matlab built-in object function, which is Viola and Jones face detector to detect the face in the images. The Viola and Jones face detector is able to detect rectangular face area, eye pair, mouth, nose and chin. Viola and Jones face detector hardly copes with 45 degree face rotation both around the vertical and horizontal axis. Figure 2.2 shows the face image used in the paper presented by Jana, Datta and Saha, and the detected facial components.

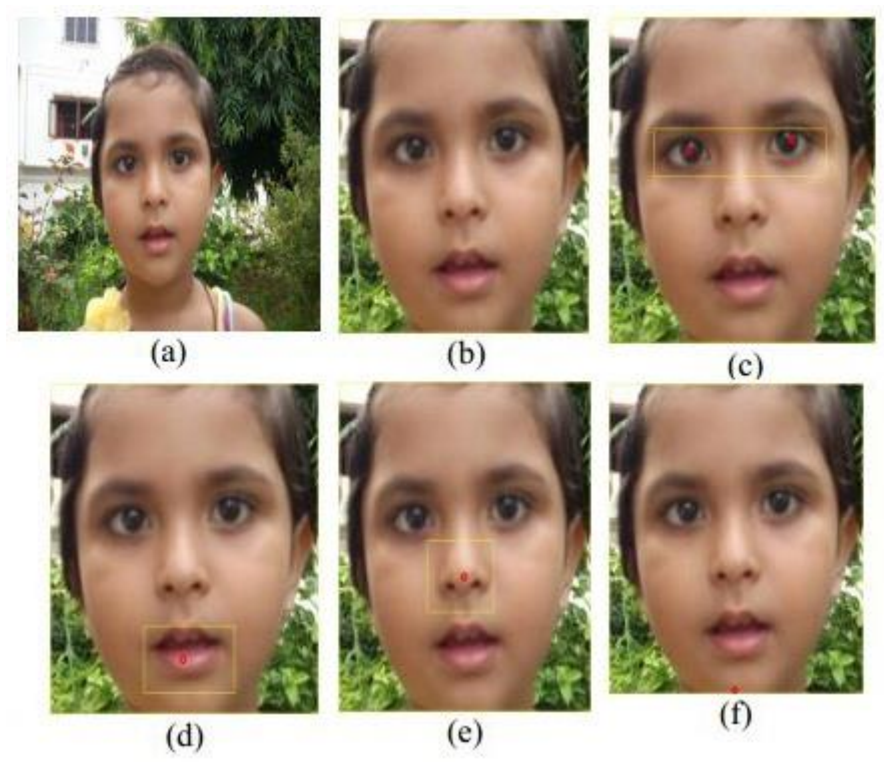


Figure 2.2: Face Image and Detected Facial Components (Jana, Datta and Saha, 2013)

In paper by Lin et al. (2012), the method proposed is to estimate age automatically based on facial images. Its face detection system is able to localize facial region by using searching windows of different sizes. These windows are applied to an input facial image to search for multi-scale facial candidates. There are 12 searching windows while their size is increased from the smallest (24 x 24) size with scale of 1.25.

## 2.2 Pre-Processing

The input images may have been acquired using different methods; they might be from scanner, digital camera or a frame shot from a video, which will result in difference in lighting and other factors. They may also have in-plane and out-of-plane rotation. There are various methods to process the images so that the performance of the system could be improved.

In the paper proposed by Roy et al. (2012), the size of the image was standardized to the form with 200 pixels height and 150 pixels width. In case that the input image is a colour image, it will be converted to grayscale image. Al-Qarni and Deravi (2012) used geometric normalization based on the eyes coordinates to pre-process the input images. The input face image was rotated, cropped and scaled so that feature extraction could be performed in the next stage.

One of the popular methods is to work on histogram of the input image. As in the paper proposed by Lin et al. (2012), lighting normalization, with histogram fitting method as fundamental, was used to transform the original histogram *H(l)* to the target histogram *G(l)*. The target histogram *G(l)* is determined by choosing the closest image histogram to the mean of face database. By applying histogram equalization, the input images that are too light or too dark are normalized to the target image. Figure 2.3 shows the lighting normalization done on three images and the resulting images. Another paper by Jana, Pal and Chowdhury (2012) used histogram equalization to adjust the contrast of the image.

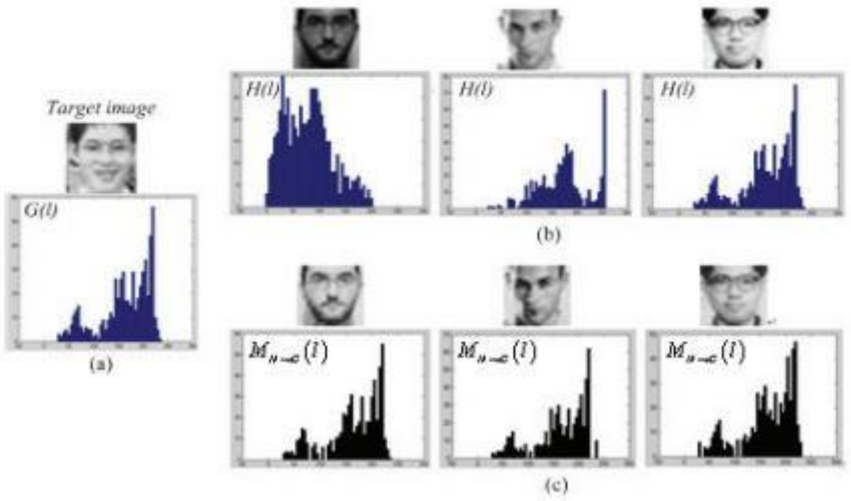


Figure 2.3: Lighting Normalization (a) Target Image (b) Input Images (c) Lighting Normalization Images (Lin et al. 2012)

Input images from databases can be either in colour or greyscale. Two papers by Jana, Datta and Saha (2013) and Han, Otto and Jain (2013) proposed the conversion of colour face image into greyscale face image. Jana, Datta and Saha applied a non-reflective similarity transformation to normalize each face image based on two eyes. Then different techniques were applied to extract the facial features.

In another paper by Han and Jain (2014), face normalization which included pose and illumination correction was done by applying 2D affine transformation and Difference of Gaussians filtering respectively. 2D affine transformation can solve in-plane rotations of face images and the resulting face images would be upright with the eyes in fixed positions. DoG filtering is able to improve the visibility of facial features in a blurred input images. Figure 2.4 shows two face images of a subject after pose and photometric correction are performed.

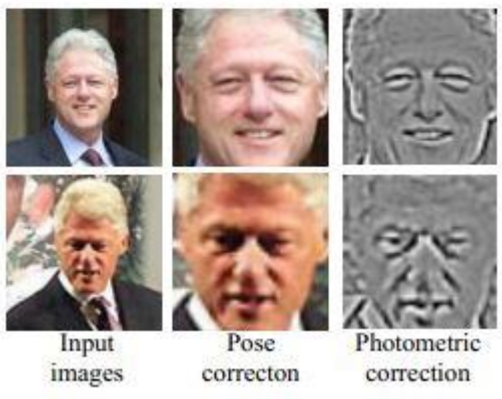


Figure 2.4: Pose and Photometric Correction done on Two Face Images of a Subject (Han and Jain, 2014)

## 2.3 Feature Extraction

In the paper proposed by TIN (2011), an algorithm that extract facial features fast and accurately was developed by applying training positions of the specific face region. The extracted facial features were expressed in column matrix and the average face for the same age group was computed. Euclidean distance was used to compute the face space and the construction of fundamental matrix A was done by using the difference in face space among the input and each face. Principal Component Analysis (PCA) was adopted to reduce the dimensionality of the vector space. Figure 2.5 shows the feature extraction method used by TIN.

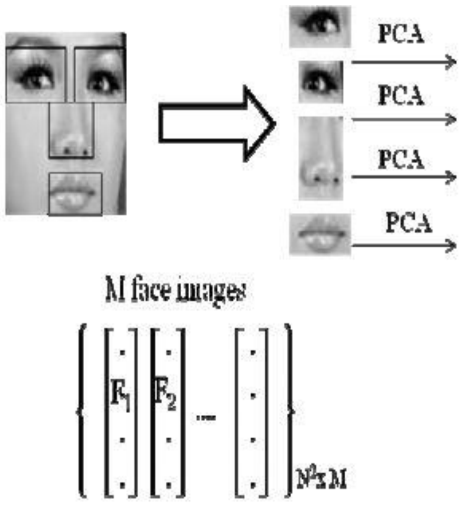


Figure 2.5: Feature Extraction (TIN, 2011)

Jana, Pal and Chowdhury (2012) divided the binary image of face into parts that contained left eye, right eye and mouth separately. Then the row and column number with the minimum row sum of grey level in each part to find the middle point of eye ball and mouth point, so that those points could form an isosceles triangle with face angle. Face angle is the important feature to perform age group estimation in this study.

Thukral, Mitra and Chellappa (2012) proposed extraction of geometric features based on landmark 2D points such as eye corners, nose and mouth. These landmark points are very sensitive to a small change in view, which causes affine transformation of the image. They suggested the method of treating the landmark points as a point in the Grassmanian manifold. The manifold mean of all landmark points on a face were computed and projected onto the tangent plane at the mean. Parameterization of any given face is done by velocity vector, which transforms the average face to the given face. These velocity vectors are used for regression for age groups.

In the paper presented by Al-Qarni and Deravi (2012), forehead region was localized based on the positions of eyes. The forehead region was treated as several small patches for texture analysis based on Gabor and LBP Operator. Hair/Skin patch classifier was developed to differentiate skin and hair in the forehead region, which will be able to detect usable skin regions. This classifier was trained by using the textures learned from Gabor features.

Roy et al. (2012) suggested a feature extraction technique in such a way that any unwanted and undesired variations in the face or background does not affect the process. Face triangle formation is the key part to perform age range estimation in this paper. Face mosacing technique was used to construct frontal image from its side-view image. Eyebrows and eyes are detected by assuming the eyebrow length is quarter of the face length. The characteristic that the mouth always located below and between the two eyes is used for its detection. The feature point is determined by scanning for the maximum number of black pixels in every part. Six coordinate points are used to form a face triangle.

In the paper proposed by Jana, Datta and Saha (2013), canny edge detection technique was applied to convert the input facial image into a binary image with wrinkle edges. In binary image, the white pixel of wrinkle regions is represented by 1 and black pixel is represented by 0. By summing up the white pixels, it is possible to calculate the percentage of area that wrinkles occupied in an input facial image. Figure 2.6 shows the conversion of image into binary image to extract the wrinkle features.

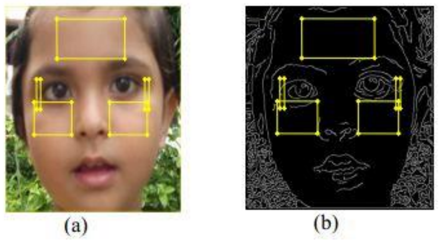


Figure 2.6: Conversion into a Binary Image using Canny Edge Detection Technique (Jana, Datta and Saha, 2013)

In the papers proposed by Han, Otto and Jain (2013), and Han and Jain (2014), the methods used were biologically inspired. The paper published in 2013 proposed the extraction of biologically inspired features from facial components. There are two layers of computational units in biologically inspired model, where simple units are followed by complex units. “MAX” pooling operator and “STD” normalization (MAX-STD) were applied to extract the complex unit features from the layer of simple unit. Another paper published later in 2014 used a biologically inspired feature (BIF) descriptor to extract features from the contextual facial region. In the first layer of BIF, Gabor filtering was applied to a face with 12 scales and 8 directions. “MAX” pooling operator was applied between every two successive scales to aggregate the features. The resulting second layer would have 6 scales and 8 directions. All the features were concatenated into a single feature vector at the end of this stage.

## 2.4 Age Group Estimation

Support Vector Machine (SVM) is one of the popular methods used in previous researches of age group estimation. It is a method developed to solve problems of classification and regression. This method has been applied in a few papers such as papers proposed by Lin et al. (2012), Eidinger, Enbar and Hassner (2013), Han, Otto and Jain (2013) and Han and Jain (2014).

In the paper proposed by Lin et al., SVM classifier was used to identify the age of the face which the image belongs to. In the paper proposed by Han, Otto and Jain, classification of each facial component into one of four disjoint age groups was done using a binary decision tree based on SVM. Also, RBF kernel was used for all classifiers and regressors of SVM. Three different SVM classifiers with RBF kernel were used in another paper by Han and Jain, to perform classifications of age group, gender and race.

Dropout-SVM was proposed in the paper by Eidinger, Enbar and Hassner, which was applied to avoid over-fitting. Over-fitting might due to shortage of data available for the research and the high dimensionality of standard face representations. In dropout-SVM, training is conducted by dropping-out the output of input-layer neurons. In linear SVM scheme, the method of dropping-out means randomly assign the value of zero to training features.

Neural Networks method of age estimation was proposed in the paper by Hewahi et al. (2010). In this method, the features were extracted from the input image of the system and classified in one of the four age classes, then further narrowed down to a more specific age range.

In the paper proposed by Chang, Chen and Hung (2010), ranking approach was used for age estimation. Age labels were treated as the rank order. The comparison with the face images with age was done by the design of query, which formed a binary classification problem to identify the preferred age class. K-Means clustering algorithm was used to classify the age ranges in the paper proposed by Jana, Datta and Saha (2013). Look up table that relates the feature vector to gender or age of a person was used to perform age estimation in the paper proposed by Hayashi et al. (2002).

Thukral, Mitra and Chellappa (2012) presented age group classification using multiple classifiers. Five classifiers were fused in this paper: μ-SVC, Partial Least Squares, Nearest Neighbor, Naive Bayes and Fished Linear Discriminant. The final classification of the test subjects into the age group is obtained by majority rule. In each age group, separate regression model are learned. Relevance Vector Machine (RVM), which is a Bayesian regression approach was used. The perceived homogeneity in the age group and the number of training data available decides the number and age range of age groups to be classified.

## 2.5 Evaluation of Performance

At the end of the proposed system, there are different methods used to evaluate its performance. One of the most common measures is by using mean absolute error (MAE). This evaluation method was adopted in papers proposed by Chang, Chen and Hung (2010), Lin et al. (2012), Thukral, Mitra and Chellappa (2012) and Han, Otto and Jain (2013). Mean absolute error denotes the average of the absolute errors between the estimated ages and the actual age, and its mathematical function is defined as in Equation 2.1 below.

$$MAE = \sum_{k=1}^{N} |\widehat{l_k} - l_k| / N \quad (2.1)$$

where

$\hat{l}_k$ = actual age of subject for image *k*,

$l_k$ = estimated age

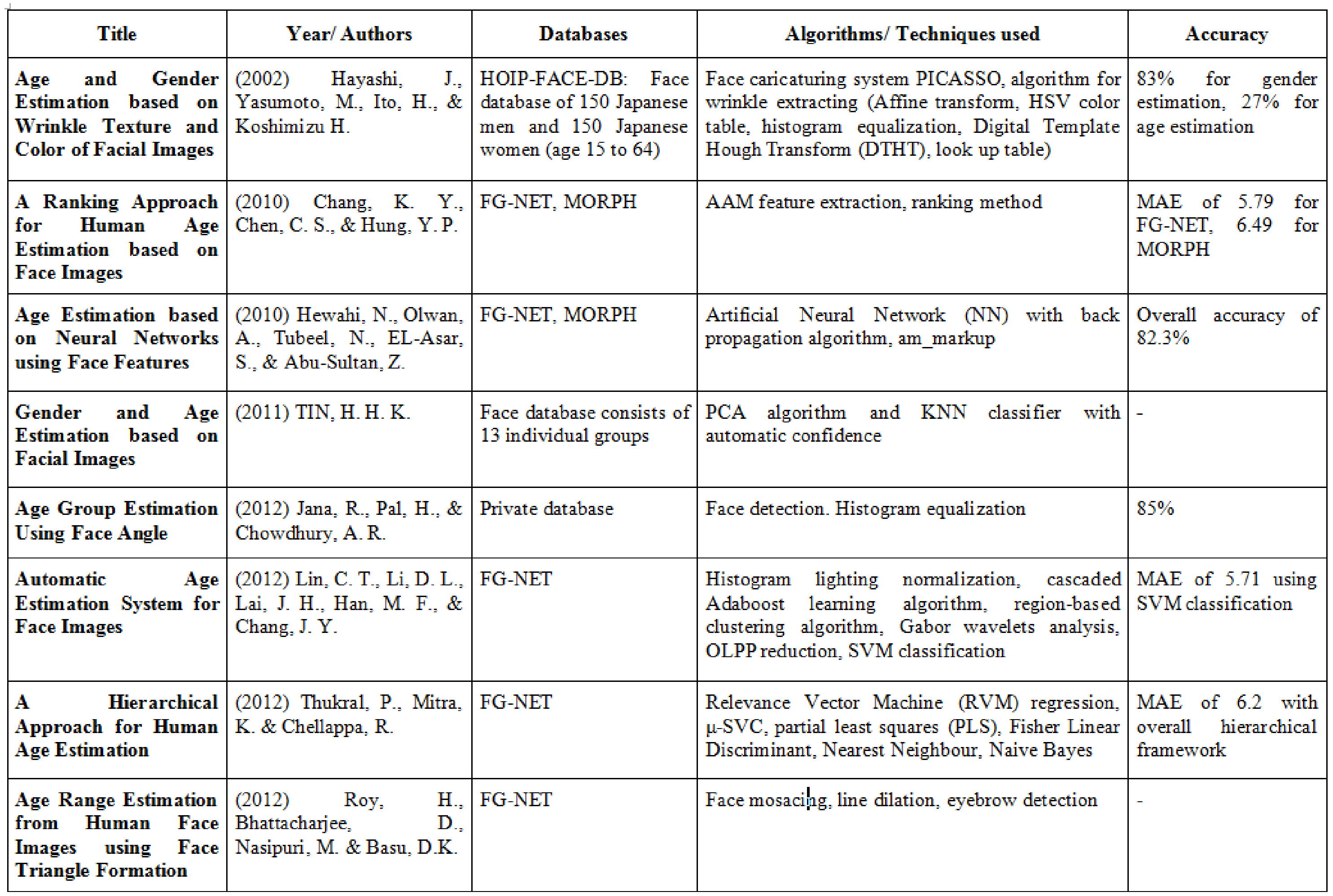
N = number of images in total

The lower the MAE, the more accurate the system is.

Another common method is by percentage of accuracy. This method was implemented in papers proposed by Hayashi et al. (2012), Hewahi et al. (2010), Jana, Pal and Chowdhury (2012), Jana, Datta and Saha (2013), Eidinger, Enbar and Hassner (2013) and Han and Jain (2014).

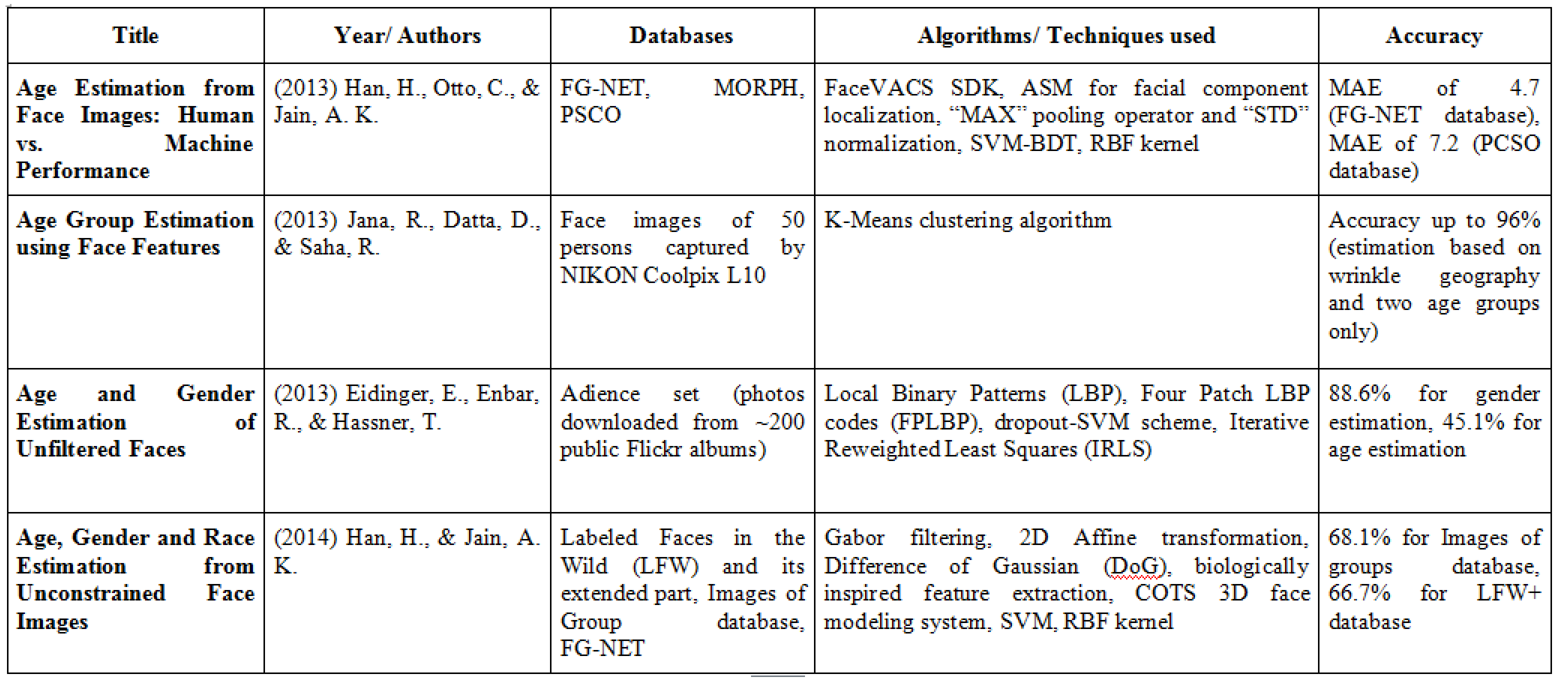
Table 2.1 shows the summary of the literature done in this project.

Table 2.1: Summary of Literature Review



| Title | Year/ Authors | Databases | Algorithms/ Techniques used | Accuracy |
|---|---|---|---|---|
| **Age and Gender Estimation based on Wrinkle Texture and Color of Facial Images** | (2002) Hayashi, J., Yasumoto, M., Ito, H., & Koshimizu H. | HOIP-FACE-DB: Face database of 150 Japanese men and 150 Japanese women (age 15 to 64) | Face caricaturing system PICASSO, algorithm for wrinkle extracting (Affine transform, HSV color table, histogram equalization, Digital Template Hough Transform (DTHT), look up table) | 83% for gender estimation, 27% for age estimation |
| **A Ranking Approach for Human Age Estimation based on Face Images** | (2010) Chang, K. Y., Chen, C. S., & Hung, Y. P. | FG-NET, MORPH | AAM feature extraction, ranking method | MAE of 5.79 for FG-NET, 6.49 for MORPH |
| **Age Estimation based on Neural Networks using Face Features** | (2010) Hewahi, N., Olwan, A., Tubeel, N., EL-Asar, S., & Abu-Sultan, Z. | FG-NET, MORPH | Artificial Neural Network (NN) with back propagation algorithm, am_markup | Overall accuracy of 82.3% |
| **Gender and Age Estimation based on Facial Images** | (2011) TIN, H. H. K. | Face database consists of 13 individual groups | PCA algorithm and KNN classifier with automatic confidence | - |
| **Age Group Estimation Using Face Angle** | (2012) Jana, R., Pal, H., & Chowdhury, A. R. | Private database | Face detection. Histogram equalization | 85% |
| **Automatic Age Estimation System for Face Images** | (2012) Lin, C. T., Li, D. L., Lai, J. H., Han, M. F., & Chang, J. Y. | FG-NET | Histogram lighting normalization, cascaded Adaboost learning algorithm, region-based clustering algorithm, Gabor wavelets analysis, OLPP reduction, SVM classification | MAE of 5.71 using SVM classification |
| **A Hierarchical Approach for Human Age Estimation** | (2012) Thukral, P., Mitra, K. & Chellappa, R. | FG-NET | Relevance Vector Machine (RVM) regression, µ-SVC, partial least squares (PLS), Fisher Linear Discriminant, Nearest Neighbour, Naive Bayes | MAE of 6.2 with overall hierarchical framework |
| **Age Range Estimation from Human Face Images using Face Triangle Formation** | (2012) Roy, H., Bhattacharjee, D., Nasipuri, M. & Basu, D.K. | FG-NET | Face mosaicing, line dilation, eyebrow detection | - |

Table 2.2: Summary of Literature Review (Continued)



| Title | Year/ Authors | Databases | Algorithms/ Techniques used | Accuracy |
|---|---|---|---|---|
| **Age Estimation from Face Images: Human vs. Machine Performance** | (2013) Han, H., Otto, C., & Jain, A. K. | FG-NET, MORPH, PSCO | FaceVACS SDK, ASM for facial component localization, "MAX" pooling operator and "STD" normalization, SVM-BDT, RBF kernel | MAE of 4.7 (FG-NET database), MAE of 7.2 (PCSO database) |
| **Age Group Estimation using Face Features** | (2013) Jana, R., Datta, D., & Saha, R. | Face images of 50 persons captured by NIKON Coolpix L10 | K-Means clustering algorithm | Accuracy up to 96% (estimation based on wrinkle geography and two age groups only) |
| **Age and Gender Estimation of Unfiltered Faces** | (2013) Eidinger, E., Enbar, R., & Hassner, T. | Adience set (photos downloaded from ~200 public Flickr albums) | Local Binary Patterns (LBP), Four Patch LBP codes (FPLBP), dropout-SVM scheme, Iterative Reweighted Least Squares (IRLS) | 88.6% for gender estimation, 45.1% for age estimation |
| **Age, Gender and Race Estimation from Unconstrained Face Images** | (2014) Han, H., & Jain, A. K. | Labeled Faces in the Wild (LFW) and its extended part, Images of Group database, FG-NET | Gabor filtering, 2D Affine transformation, Difference of Gaussian (DoG), biologically inspired feature extraction, COTS 3D face modeling system, SVM, RBF kernel | 68.1% for Images of groups database, 66.7% for LFW+ database |

# CHAPTER 3

# METHODOLOGY

## 3.1 Methodology Flow

This proposed approach performs wrinkle-based age group estimation from a given input image. The methodology flow begins with pre-processing of the input face image, followed by feature (wrinkle) extraction from the image and then classification of the images into three age groups: babies or children, young adults and old adults. The details of each stage will be discussed in the later sections and the methodology flow is shown in Figure 3.1.

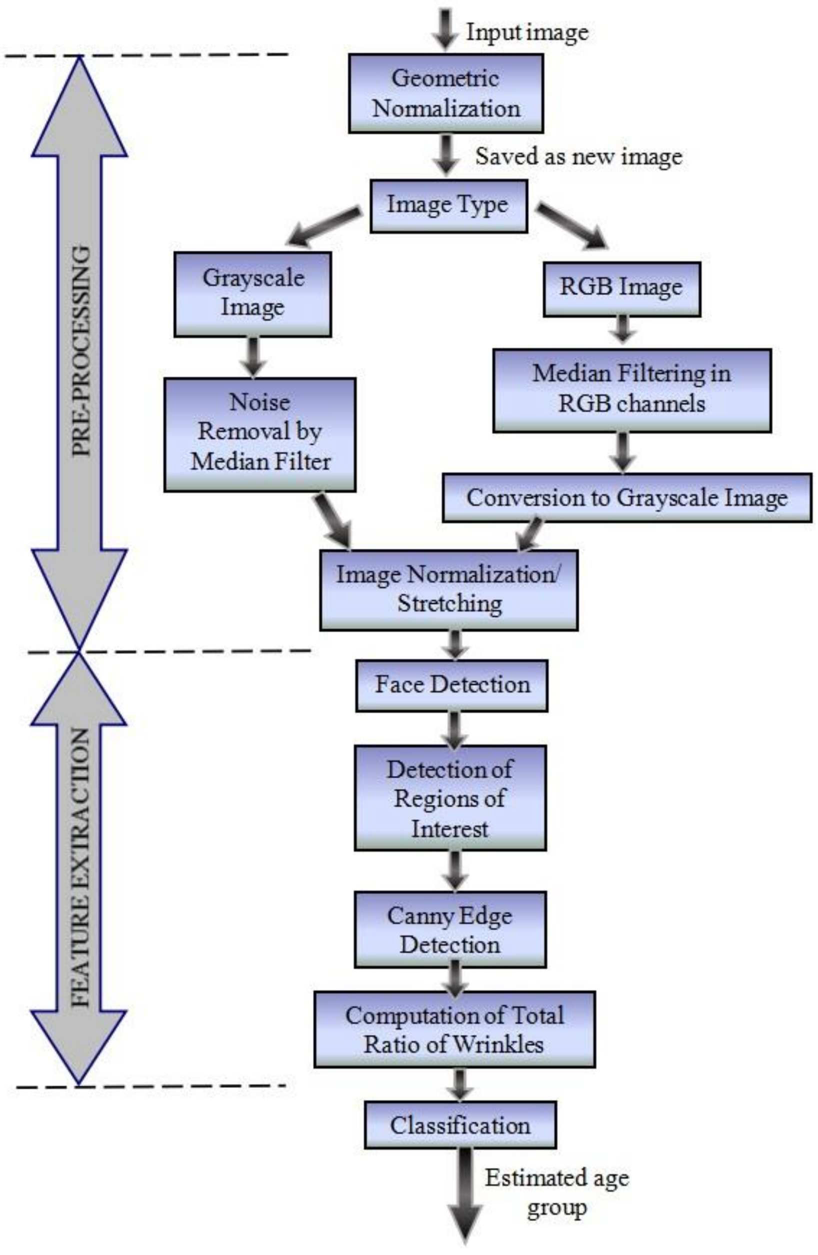

Input image
Geometric Normalization
Saved as new image
Image Type
Grayscale Image
RGB Image
Noise Removal by Median Filter
Median Filtering in RGB channels
Conversion to Grayscale Image
Image Normalization/ Stretching
Face Detection
Detection of Regions of Interest
Canny Edge Detection
Computation of Total Ratio of Wrinkles
Classification
Estimated age group
PRE-PROCESSING
FEATURE EXTRACTION

Figure 3.1: Overall Methodology Flow of the Proposed Approach

## 3.2 Input Images

### 3.2.1 Databases Used

FG-NET Aging database and MORPH-II database are used as both training set and testing set. FG-NET Aging database contains 1002 images of 82 individuals with age ranges from 0 to 69 years old. Figure 3.2 shows the age range distribution in FG-NET Aging database while Figure 3.3 shows some sample images from the database.

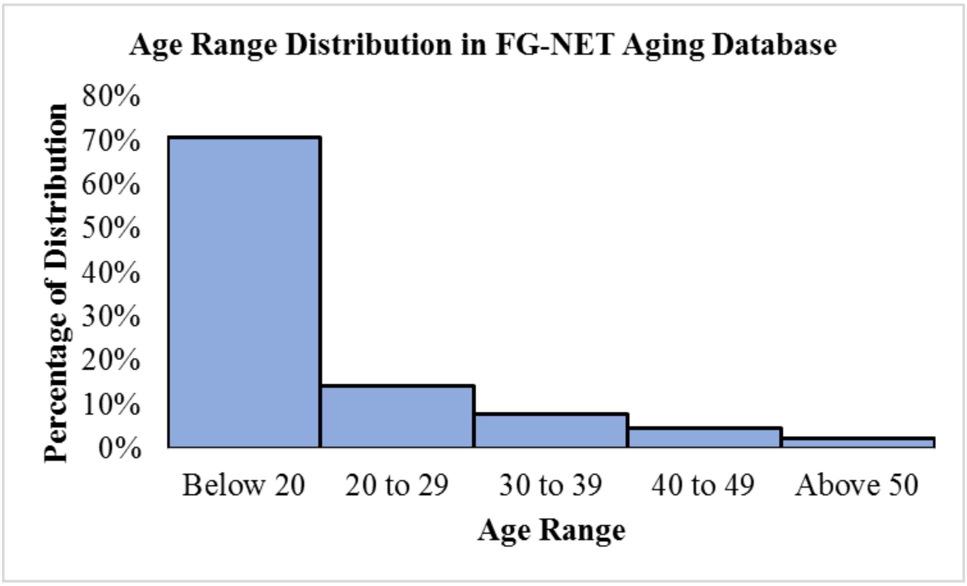


Figure 3.2: Age Range Distribution in FG-NET Aging Database

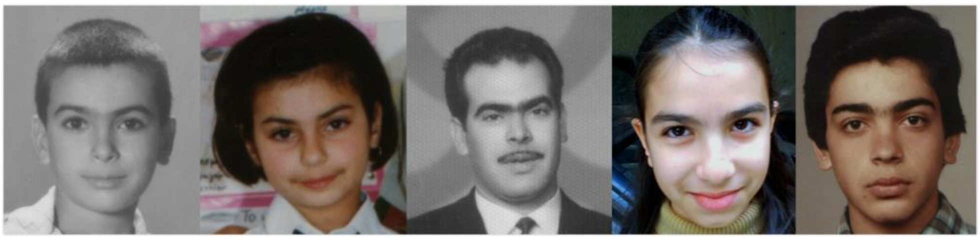


Figure 3.3: Sample Images in FG-NET Aging Database

Images for older age group are limited in FG-NET Aging database, therefore MORPH-II database is also used. MORPH-II database consists of 55134 images of more than 13000 individuals with youngest of 16 years old and oldest of 77 years old. Figure 3.4 shows the age range distribution in MORPH-II database; some sample images from MORPH-II database are shown in Figure 3.5.

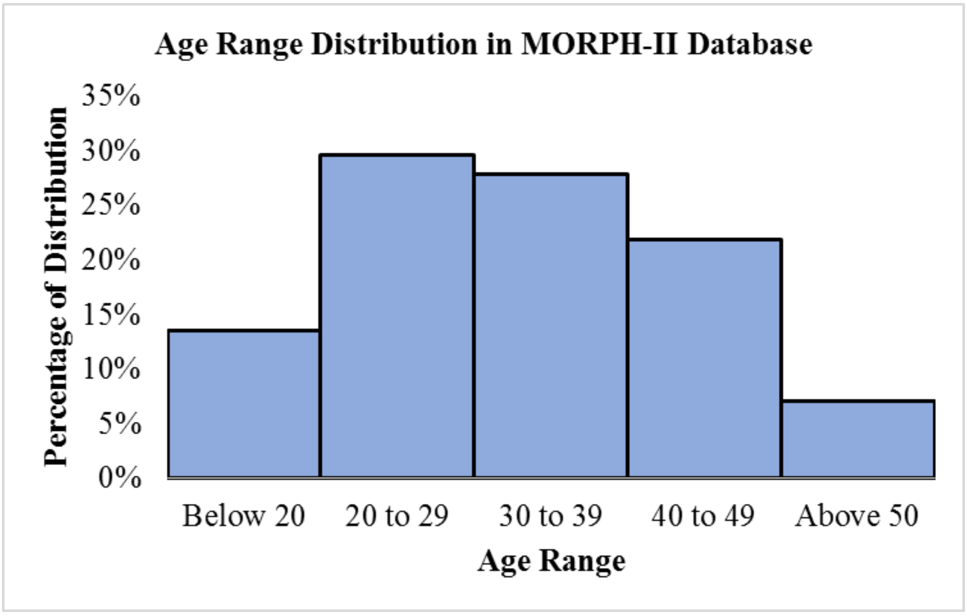


Figure 3.4: Age Range Distribution in MORPH-II Database

Figure 3.5: Sample Images in MORPH-II Database

Some of the images from both databases are poor in quality. Images with better quality are selected as training set and testing set, while images with poor quality are selected to form another testing set to draw a contrast in difference in performance between testing sets of different image quality.

Training set for the proposed algorithm is made up of 167 images; 91 images from FG-NET Aging database and 76 images from MORPH-II database. The youngest individual in the training set is less than one year old while the oldest individual is 64 years old. Figure 3.6 shows the age range distribution in the training set.

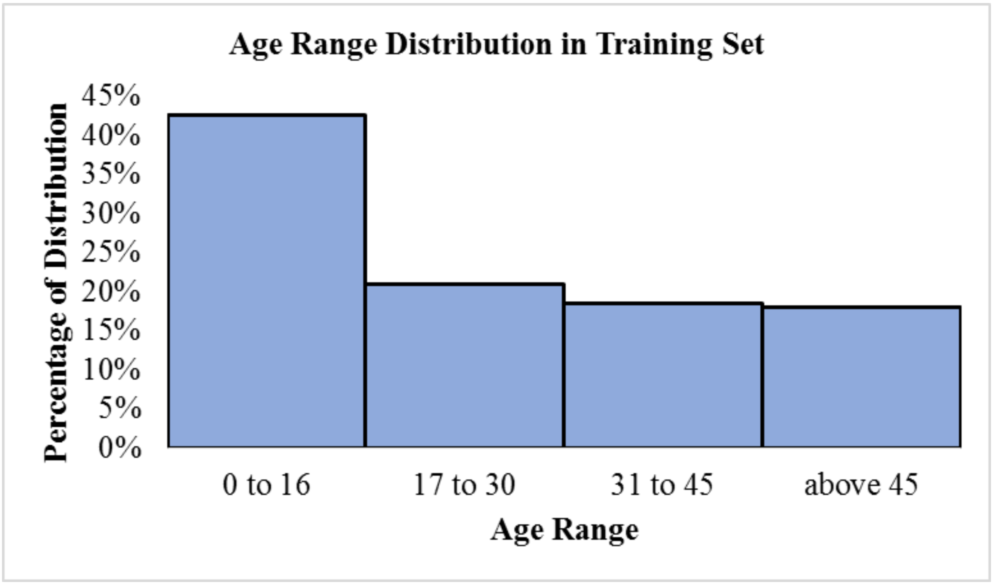


Figure 3.6: Age Range Distribution in Training Set

There are four different testing sets used for this algorithm. The first two testing sets are colour images of better quality and poor quality respectively, while another two testing sets are grayscale images of better quality and poor quality. Colour images are selected from both FG-NET Aging database and MORPH-II database while grayscale images are from FG-NET Aging database. Figure 3.7 to Figure 3.10 show the age range distribution in testing set I, II, III and IV respectively.

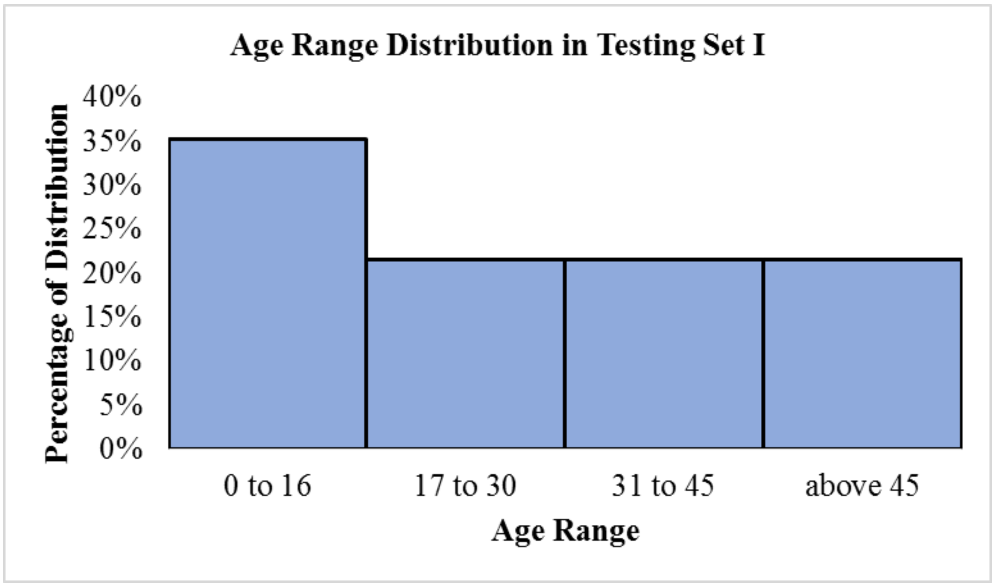


Figure 3.7: Age Range Distribution in Testing Set I

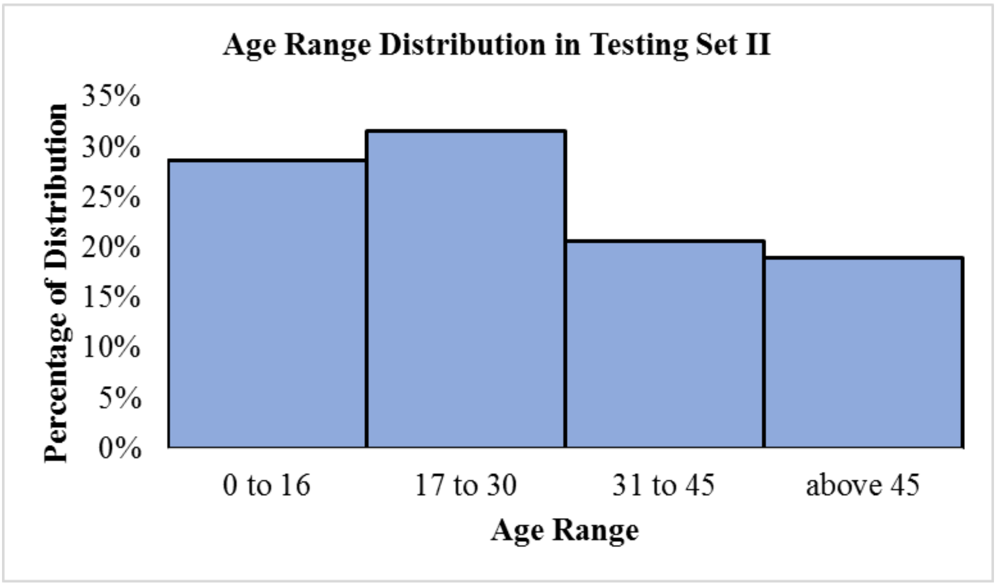


Figure 3.8: Age Range Distribution in Testing Set II

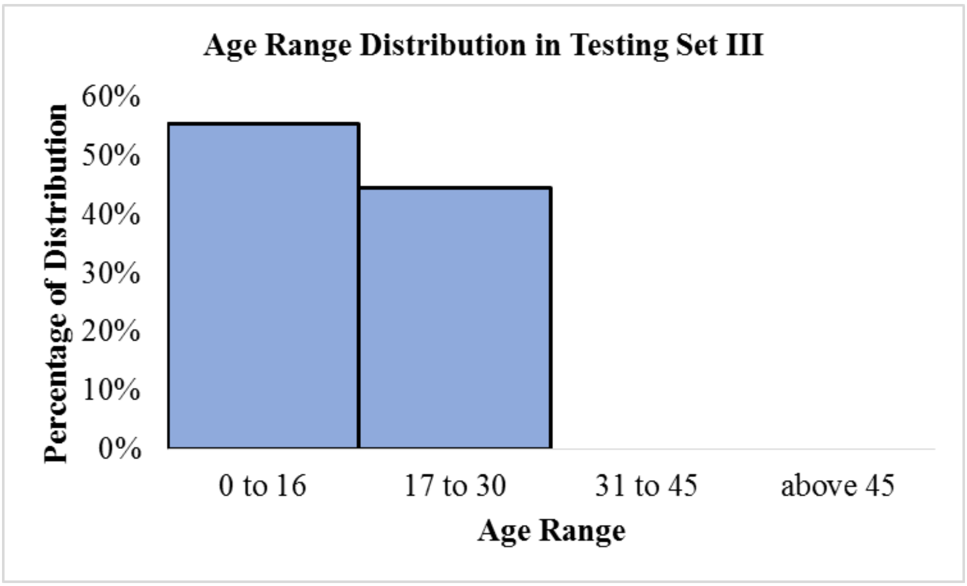


Figure 3.9: Age Range Distribution in Testing Set III

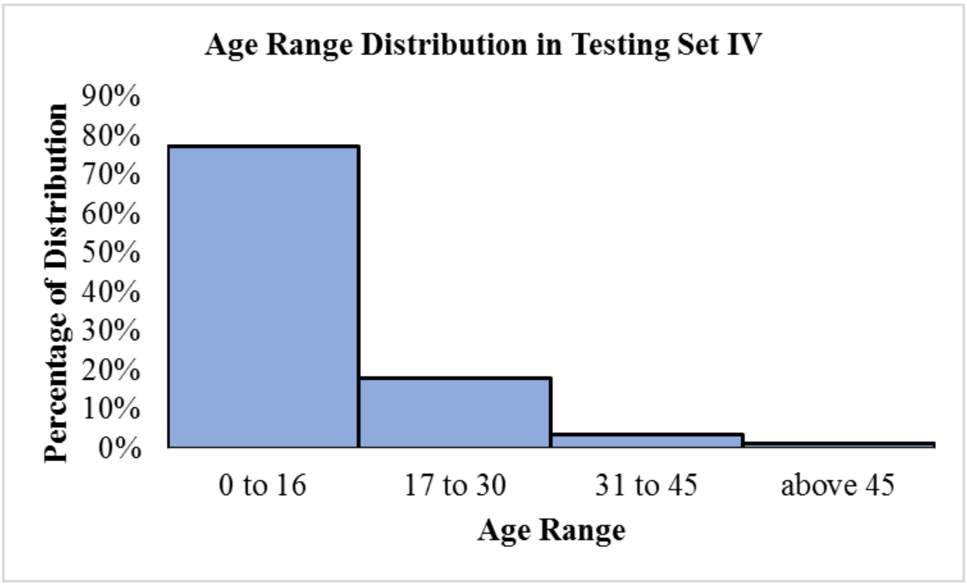


Figure 3.10: Age Range Distribution in Testing Set IV

### 3.2.2 Limitations of the Input Images

The input images for the proposed approach has to be frontal upright face images, with glass free and clear forehead region. These conditions have to be satisfied to ensure that the proposed approach could perform better.

## 3.3 Pre-Processing

Images from testing set and training sets are scanned photographs or images captured using cameras. The images could be either colour images or grayscale images. Some of them are not captured in controlled condition, which results in ill-posed images. There are also some unwanted noises in the images. Therefore, there is a need to go through several pre-processing steps before proceeding to feature extraction.

Pre-processing steps that would be carried out include geometric normalization, median filtering, conversion of colour images to grayscale images and image normalization. The details of these steps would be discussed in the later sections .

### 3.3.1 Geometric Normalization

Most of the images from both databases have out-of-plane rotated face. This adds difficulties in face detection and feature extraction later. Therefore, the algorithm is designed to allow user to perform rotation of images. The eyes of the individual will be on the same horizontal line after plane rotation.

In the proposed approach, rotation could be done by using the slider appearing in the user interface window. By dragging the slider to the right, the input image is rotated in a clockwise direction starting from a minimum angle of 0 degree until a maximum angle of 360 degree (goes back to original pose). Figure 3.11 shows the image being rotated at different angles, where the angle is controlled by the slider.

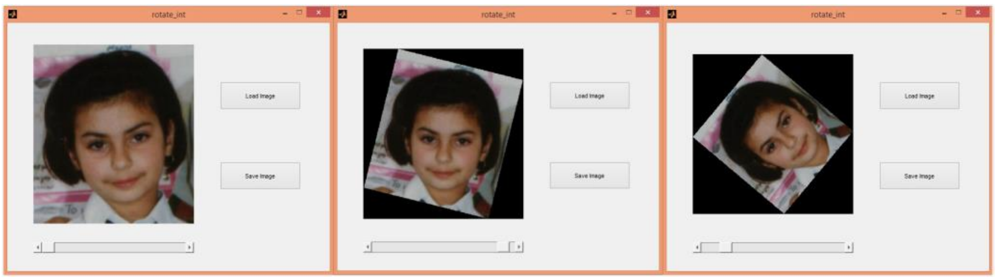


Figure 3.11: Image at Different Angles of Rotation

### 3.3.2 Median Filtering

Median filtering is one of the noise reduction methods widely used in various applications due to its ability to remove unwanted noises while at the same time not blurring the images. The edges of the images could be preserved.

Since the images are either grayscale or RGB images, median filtering is done by two different methods too. In RGB images, median filtering is performed on three different channels of the image. While in grayscale image, median filtering is performed directly. Figure 3.12 shows the image before and after noise removal by median filtering in three channels.

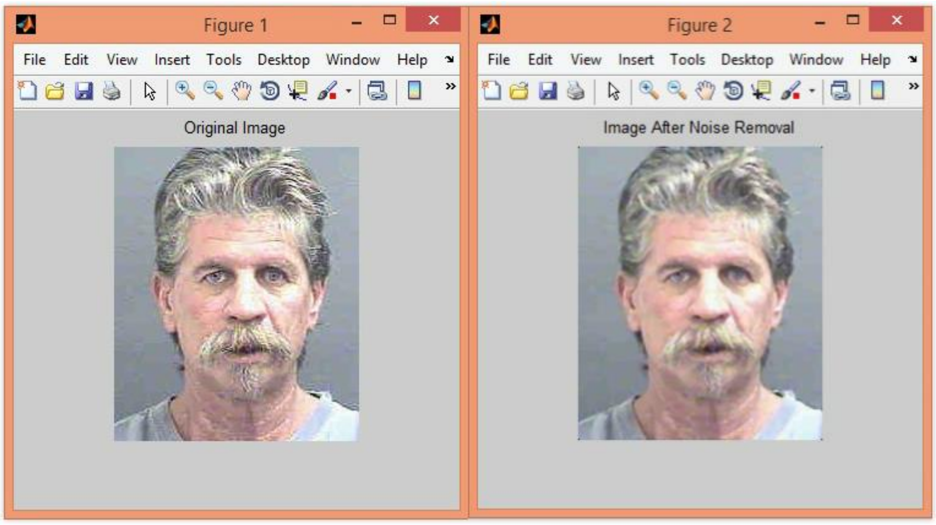


Figure 1
File Edit View Insert Tools Desktop Window Help
Original Image
Figure 2
File Edit View Insert Tools Desktop Window Help
Image After Noise Removal

Figure 3.12: Image Before and After Median Filtering

### 3.3.3 Conversion to Grayscale Image

Most of the images from the databases are colour images. However, image normalization and canny edge detection can only be performed on grayscale images. Therefore, colour images have to be converted to grayscale images before proceeding to the later steps. Figure 3.13 shows the conversion of image to grayscale image.

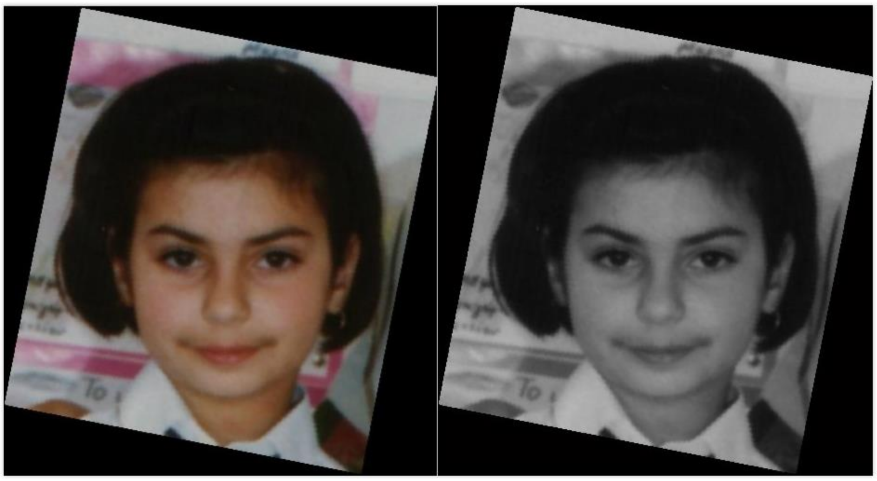


Figure 3.13: Conversion of Image to Grayscale Image

### 3.3.4 Image Normalization/ Stretching

Image normalization, also known as histogram stretching, is a technique of image enhancement. Image contrast is improved by stretching the range of its pixel intensity values to span a desired range of values. In this approach, the default range of pixel values is used, which is between 0 and 255.

The reason that image normalization is used instead of histogram equalization is image normalization gives a more natural look to the input image. The use of histogram equalization gives an image unrealistic and undesirable effects. When the image is further processed later, especially in canny edge detection step, it will result in undesired edges or lines in the output. Figure 3.14 shows histogram stretching done on the image.

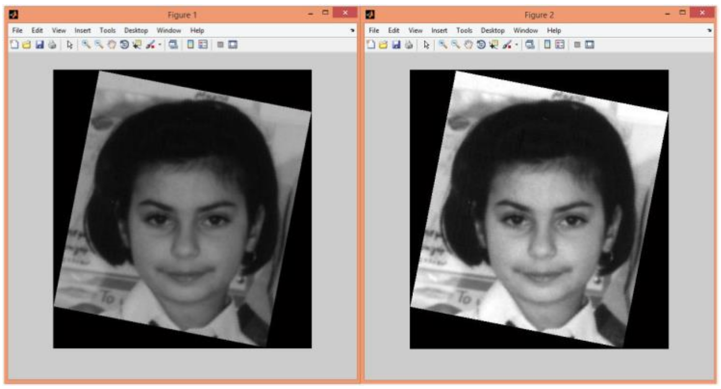


Figure 3.14: Image Normalization or Histogram Stretching

## 3.4 Feature Extraction

As human grows old, the aging process brings different changes in the face of an individual. The changes could be textural changes or change in geometric features. In order to estimate the age group of an individual, these features have to be extracted from the face images and classified appropriately. The stage of feature extraction begins with detection of face and regions of interest, followed by canny edge detection and computation of total ratio of wrinkles.

### 3.4.1 Face Detection

Since face detection is not the main concern here, Viola-Jones object detection framework will be used to detect the face from the input face images. Viola-Jones framework is an built-in MATLAB function, therefore it could be applied easily in the algorithm.

The limitation of Viola-Jones framework is that the input images have to be frontal upright image, the face of the individual must points towards the camera and should not be tilted to any side.

### 3.4.2 Age-Related Features

From birth to adulthood, human experiences craniofacial growth and development, which in turn leads to the change in geometric features. These changes are in terms of changes in the distances between eyes, nose, lip and chin. The changes can be expressed in face angle too. Nevertheless, these changes are limited to differentiate minors and adults only.

Textural information from the face is one of the key features to differentiate adult of different ages. When adult grows older, there are wrinkles slowly appear on the face.

In this proposed algorithm, wrinkle feature will be used to determine age group of the individual.

### 3.4.3 Detection of Regions of Interest

After the face is detected from the input face image, the coordinates and size of the detected face can be used to determine the locations of the regions of interest in the face image.

Forehead region, under-eye regions and end-of-the-eye regions are the regions of interest for this approach to determine age group of individuals. It was observed that these regions of interest occupy the detected face at specific ratio. Figure 3.15 and Figure 3.16 shows how the regions of interest are obtained and their coordinates and sizes.

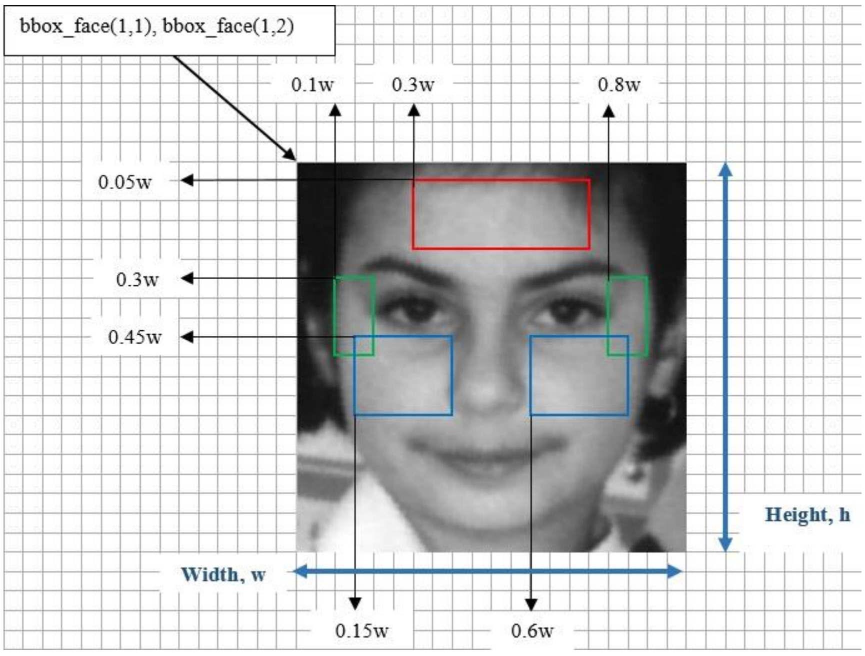


Figure 3.15: Coordinates of Region of Interest

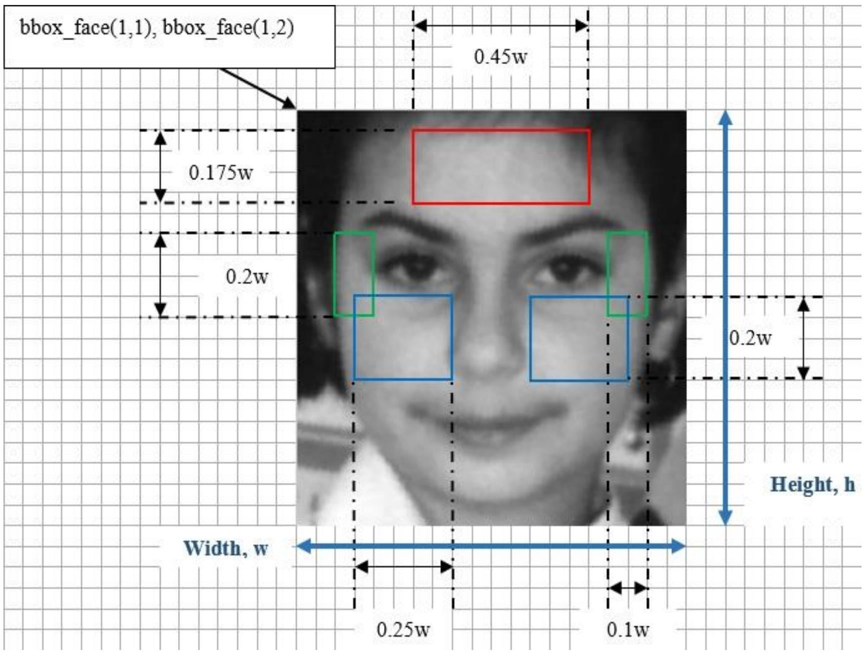


Figure 3.16: Size of Regions of Interest

### 3.4.4 Canny Edge Detection

Canny edge detection is a popular edge detection technique and it is used in this algorithm as a step to quantify the wrinkles. The application of canny edge detection transforms an input face image to a black and white image. Wrinkles will appear as fine white lines in the output image. Figure 3.17 shows the effect of canny edge detection being applied on the image.

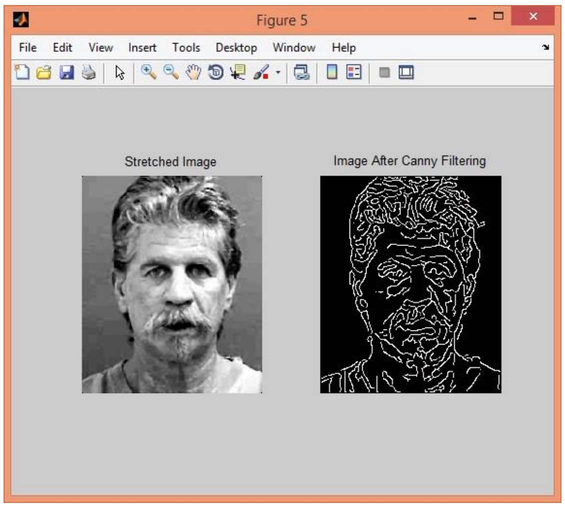


Figure 5
File Edit View Insert Tools Desktop Window Help
Stretched Image
Image After Canny Filtering

Figure 3.17: Canny Edge Detection on an Image

### 3.4.5 Computation of Total Ratio of Wrinkles

Before classification, the total number of black and white pixels in each region of interest is calculated. The ratio of white pixels to the total number of pixels is then calculated. Next, the total ratio of wrinkles is computed by summing up the ratio of white pixels in every region of interest.

## 3.5 Classification

The wrinkle features for every face image in training set are obtained in the form of quantitative data. The images would be classified into three age groups: babies or children, young adults and old adults, based on the quantitative wrinkle information obtained from the faces. The method of classification is based on the mean value of wrinkle feature for each age group. Table 3.1 shows the mean value of wrinkle feature for each age group.

**Table 3.1: Mean Value of Wrinkle Feature**

| Age Group in Years | Mean Value of Wrinkle Feature |
|---|---|
| 0 to 16 | 0.2701 |
| 17 to 45 | 0.4384 |
| Above 46 | 0.6590 |

The boundary value of the age group is determined by averaging the mean value of wrinkle feature of two age groups. Table 3.2 shows the boundary value that separates and classifies different age groups.

**Table 3.2: Boundary Value of Age Group**

| Age Group in Years | Boundary Value of Age Group |
|---|---|
| 0 to 16 | Less than or equal to 0.3543 |
| 17 to 45 | Greater than 0.3543 and less than or equal to 0.5487 |
| Above 46 | Greater than 0.5487 |

# CHAPTER 4

# RESULTS AND DISCUSSIONS

## 4.1 Results

The proposed wrinkle-based age group estimation algorithm has been developed based on a training set and applied to four different testing sets. The statistics of the testing sets has been discussed in the previous chapter.

Wrinkle feature is used and quantified to classify facial images into age groups. Wrinkle can be extracted from the facial image as long as the face is detected accurately from the face. Figure 4.1 shows one of the facial images and the detection of regions of interest, as well as the wrinkle feature visible on those regions.

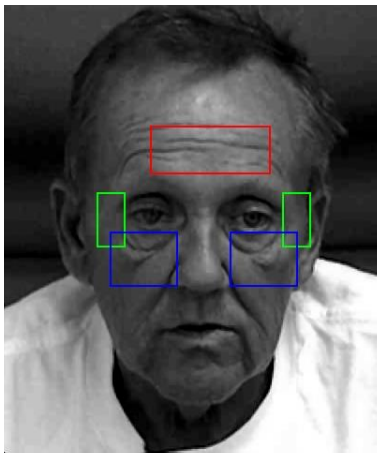


Figure 4.1: Detection of Regions of Interest

Figure 4.2 to Figure 4.6 show the magnified view of wrinkles on forehead region, end-of-eye region and under-eye region respectively.

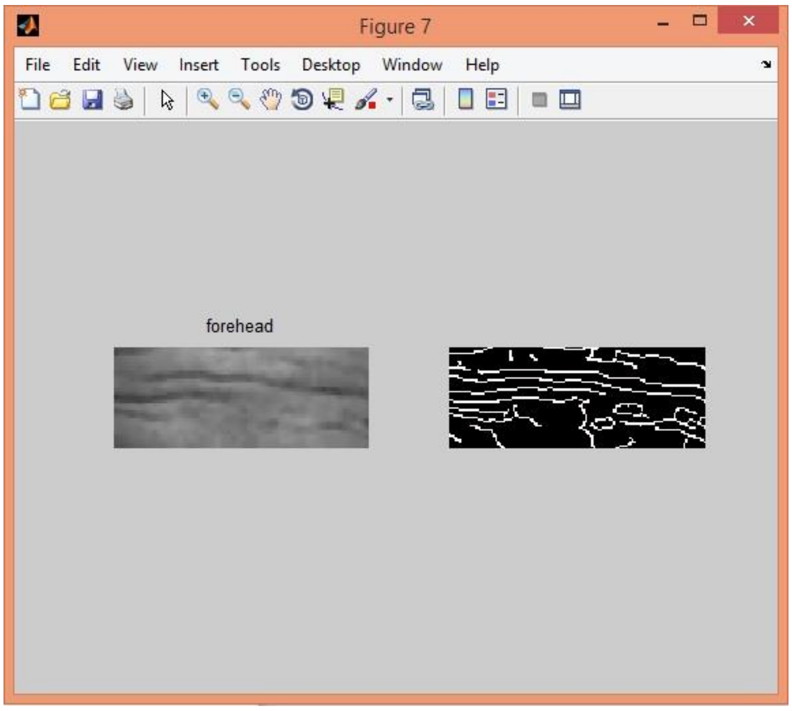


Figure 4.2: Wrinkles on Forehead Region

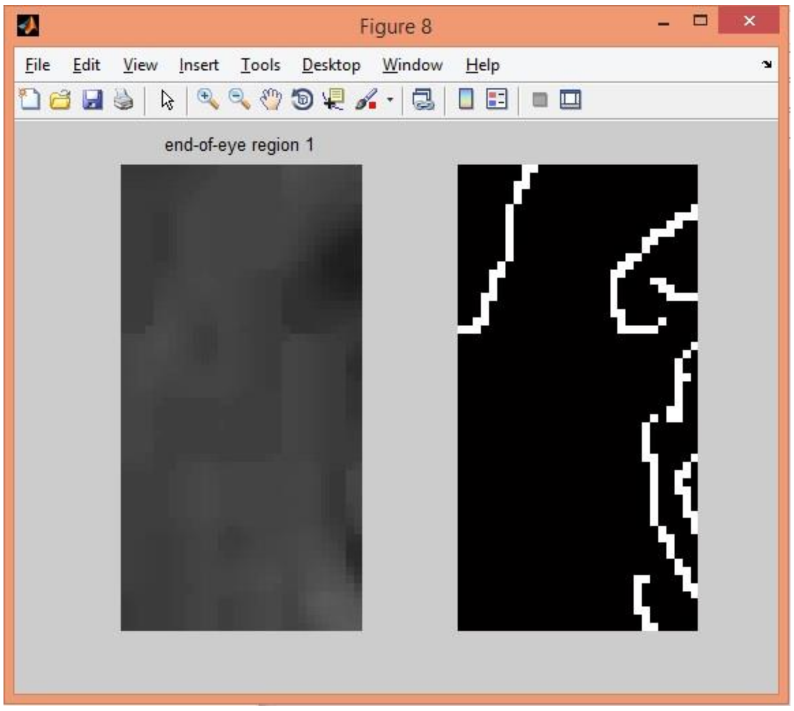


Figure 4.3: Wrinkles on End-of-eye Region (Right Eye)

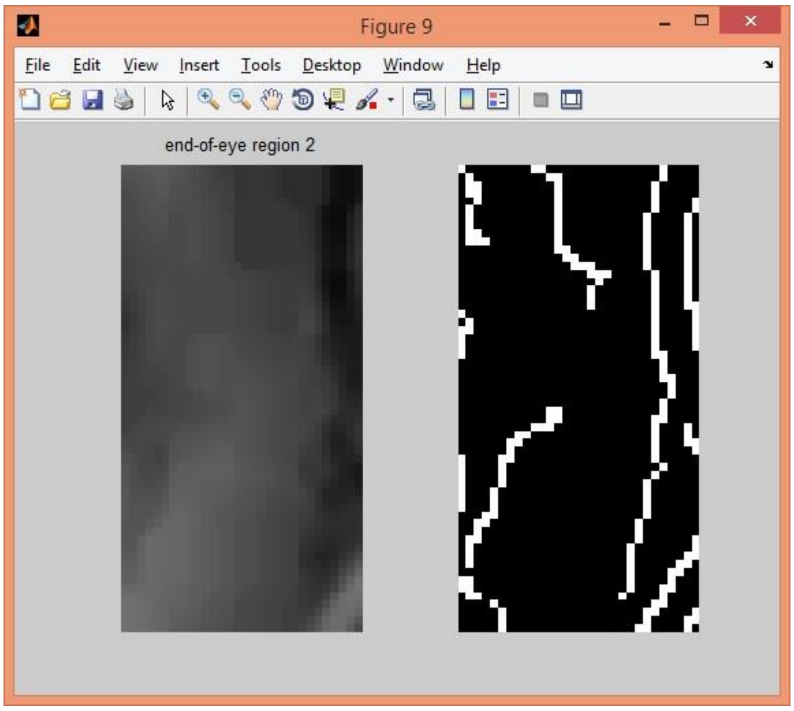


Figure 4.4: Wrinkles on End-of-eye Region (Left Eye)

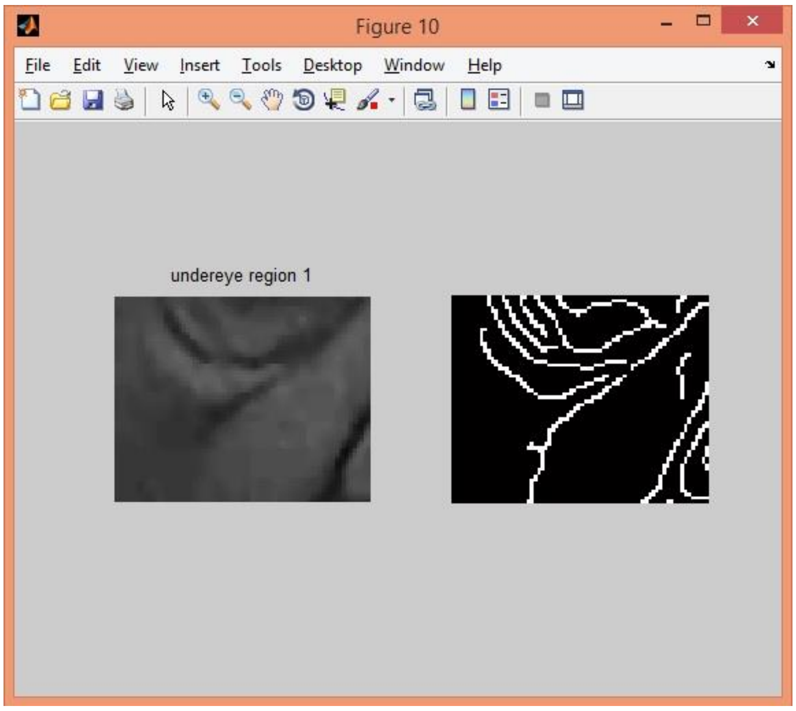


Figure 4.5: Wrinkles on Under-eye Region (Right Eye)

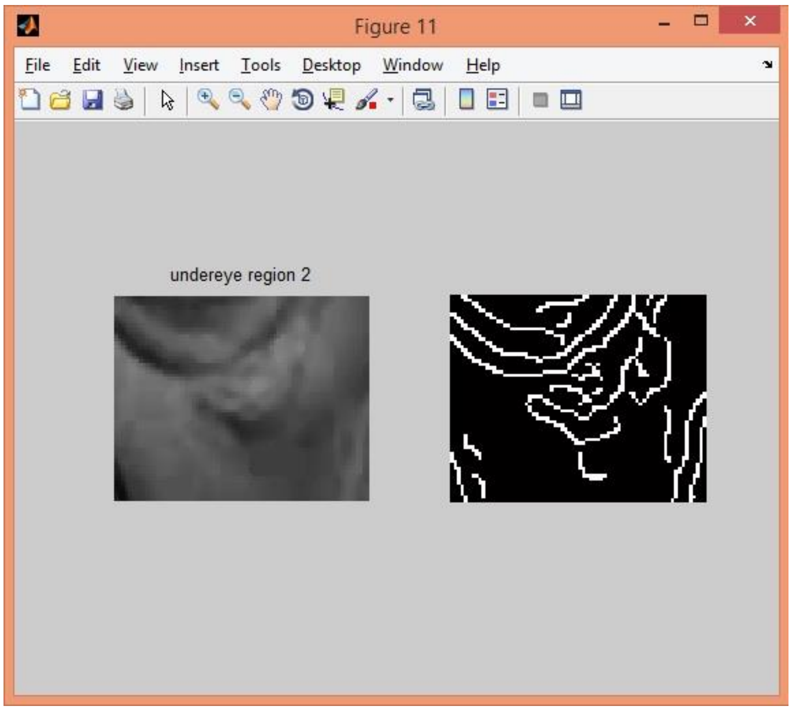
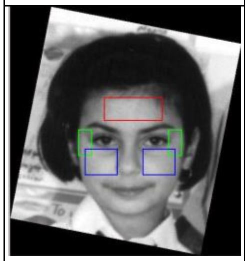
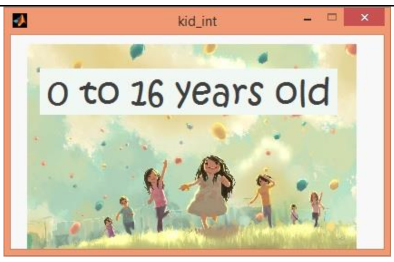
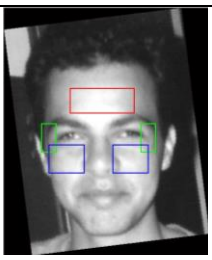
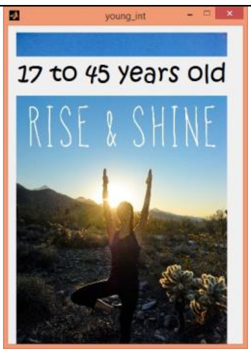
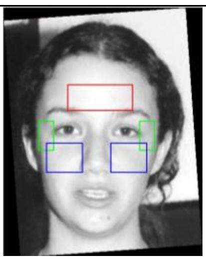
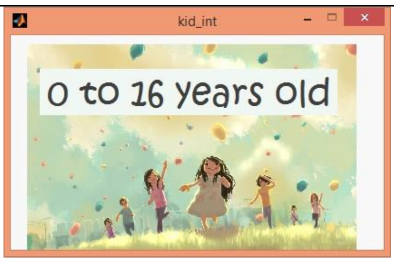
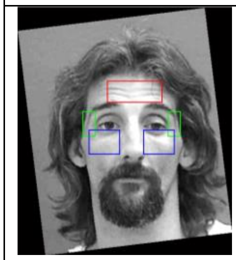
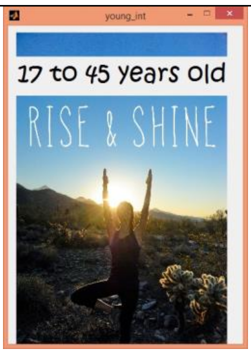
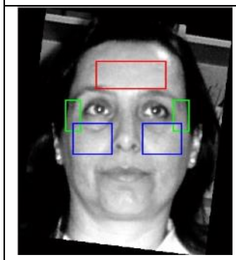
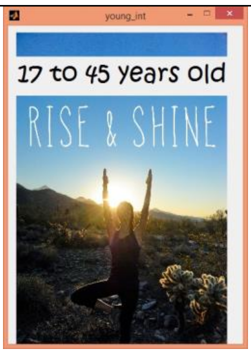
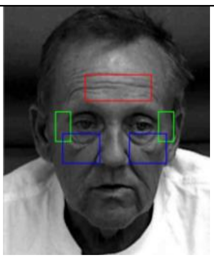
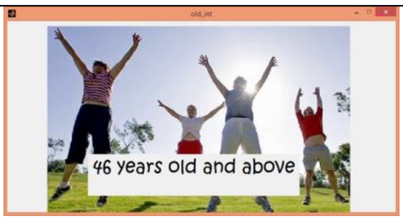


Figure 4.6: Wrinkles on Under-eye Region (Left Eye)

Table 4.1 shows some of the age group estimation results using the proposed algorithm.

**Table 4.1: Results of Age Group Estimation**

| Input Image | Estimated Age Group | Actual Age | Is it accurate? |
|---|---|---|---|
|  (009A09) |  kid_int 0 to 16 years old | 9 years old | Yes |
|  (022A18) |  young_int 17 to 45 years old | 18 years old | Yes |
|  (026A18) |  kid_int 0 to 16 years old | 18 years old | No |
|  (182413_03M36) |  young_int 17 to 45 years old | 36 years old | Yes |
|  (002A38) |  young_int 17 to 45 years old | 38 years old | Yes |

| | | | |
|---|---|---|---|
|  (041086_03M55) |  | 55 years old | Yes |

The performance of the proposed algorithm is tabulated in the Table 4.2.

**Table 4.2: Overall Performance of the Proposed Algorithm**

| Testing Set | Age Range (Years Old) | Total Number of Images | Number of Correct Estimation | Accuracy (%) | Average Accuracy (%) |
|---|---|---|---|---|---|
| I (Colour Images) | 0 to 16 | 49 | 48 | 97.96 | 72.66 |
| | 17 to 45 | 60 | 37 | 61.67 | |
| | 46 and above | 30 | 16 | 53.33 | |
| II (Colour Images) | 0 to 16 | 68 | 34 | 50.00 | 44.92 |
| | 17 to 45 | 124 | 54 | 43.55 | |
| | 46 and above | 44 | 18 | 40.91 | |
| III (Grayscale Images) | 0 to 16 | 20 | 16 | 80.00 | 61.11 |
| | 17 to 45 | 16 | 6 | 37.50 | |
| | 46 and above | - | - | - | |
| IV (Grayscale Images) | 0 to 16 | 65 | 30 | 46.15 | 45.24 |
| | 17 to 45 | 18 | 8 | 44.44 | |
| | 46 and above | 1 | 0 | - | |

## 4.2 Discussions

This proposed approach provides a method to estimate the age group of an individual, which is based on the wrinkle features on the face. Aging rate is different in every person, due to different lifestyles, health conditions and environments. Therefore, training set has to be chosen by taking these reasons into consideration.

The proposed approach is being trained and tested on datasets, which consists of 662 images of 180 individuals in total, with age 0 to 69 years old. These 662 images are further categorized into four different training sets, as images are different in their quality: some images are blur while some are clearer. The statistics of each dataset has been discussed in the earlier chapter.

Some pre-processing steps are performed on the input face image before the features are extracted. As some of the images are out-of-plane rotated, slider at the user interface can be used to rotate the image back to the upright position. Median filtering and histogram stretching (image normalization) are performed to improve the image so that the wrinkle feature on the face is more visible. Median filtering is used because it is able to preserve the edges of the image while removing the image noises. While histogram stretching provides a more natural look to the image and not causes any additional noises to the image after canny edge detection. The image will be converted to grayscale image if it is not a grayscale image because canny edge detection works on grayscale image.

Viola-Jones object detection framework is applied in this approach to detect and localize the face given a face image. From the detected face, an algorithm that can localize the regions of interest is applied. The detection of regions of interest works as long as the face is detected accurately. The detection is based on the geometric ratio of the face.

From the results obtained, testing set I, which consists of good quality colour images, achieves the highest accuracy (72.66%). Testing set III, which consists of good quality grayscale images, has accuracy in estimation slightly lower (72.66%)

than testing set I. Testing set II and testing set IV, which consist of poor quality colour and grayscale images respectively, have accuracy of approximately 45.00%.

Besides, the algorithm has better accuracy in estimating images from the first age group (0 to 16 years old). The accuracy of the algorithm is up to 97.96% when being tested with good quality images, followed by accuracy of 80.00% in grayscale good quality images. It can be said that the approach works best with good quality images, poor quality images could degrade the performance of the algorithm. Poor quality images could be either blurred images or having too much unwanted noises. In blurred images, the face is blurred out. The wrinkle feature is not visible and the face is smoothed due to the effect of blurring.

## 4.3 Comparison with How-Old.net

The popular How-Old.net website is also being tested on the same testing sets, and its performance is tabulated in Table 4.3.

**Table 4.3: Performance of How-Old.net**

| Testing Set | Age Range (Years Old) | Total Number of Images | Number of Correct Estimation | Accuracy (%) | Average Accuracy (%) |
|---|---|---|---|---|---|
| I (Colour Images) | 0 to 16 | 49 | 41 | 83.67 | 79.86 |
| | 17 to 45 | 60 | 45 | 75.00 | |
| | 46 and above | 30 | 25 | 83.33 | |
| II (Colour Images) | 0 to 16 | 68 | 40 | 58.82 | 55.27 |
| | 17 to 45 | 124 | 101 | 81.45 | |
| | 46 and above | 44 | 40 | 90.91 | |
| III (Grayscale Images) | 0 to 16 | 20 | 15 | 75.00 | 86.11 |
| | 17 to 45 | 16 | 16 | 100.00 | |
| | 46 and above | - | - | - | |

| | | | | | |
|---|---|---|---|---|---|
| IV (Grayscale Images) | 0 to 16 | 65 | 35 | 53.85 | 63.13 |
| | 17 to 45 | 18 | 18 | 100.00 | |
| | 46 and above | 1 | 0 | - | |

How-Old.net performs better than the proposed algorithm. Referring to the table, it is observed that its performance is better for estimation of age group from better quality images, like what the proposed algorithm did too. Figure 4.7 and Figure 4.8 shows the difference in performance using the proposed approach and How-Old.net.

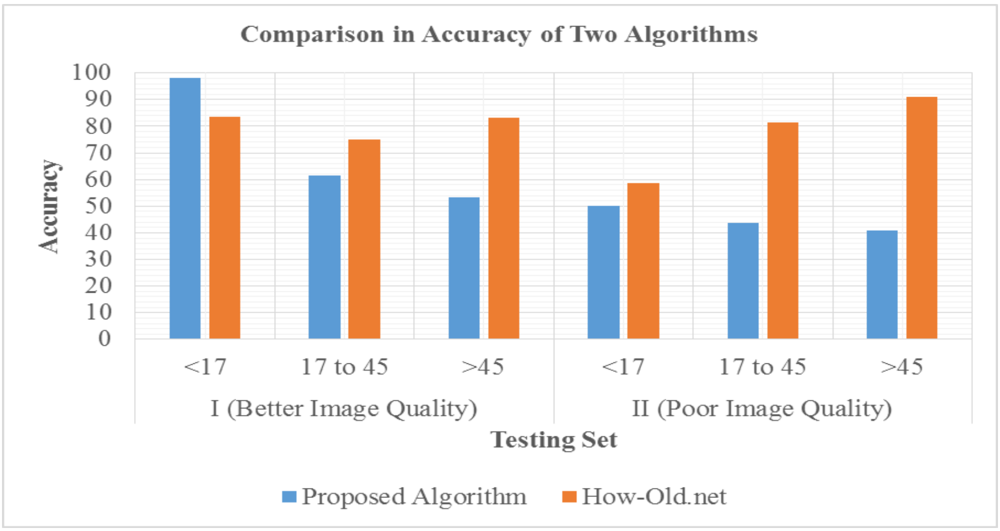


Figure 4.7 Comparison in Accuracy of Two Algorithms

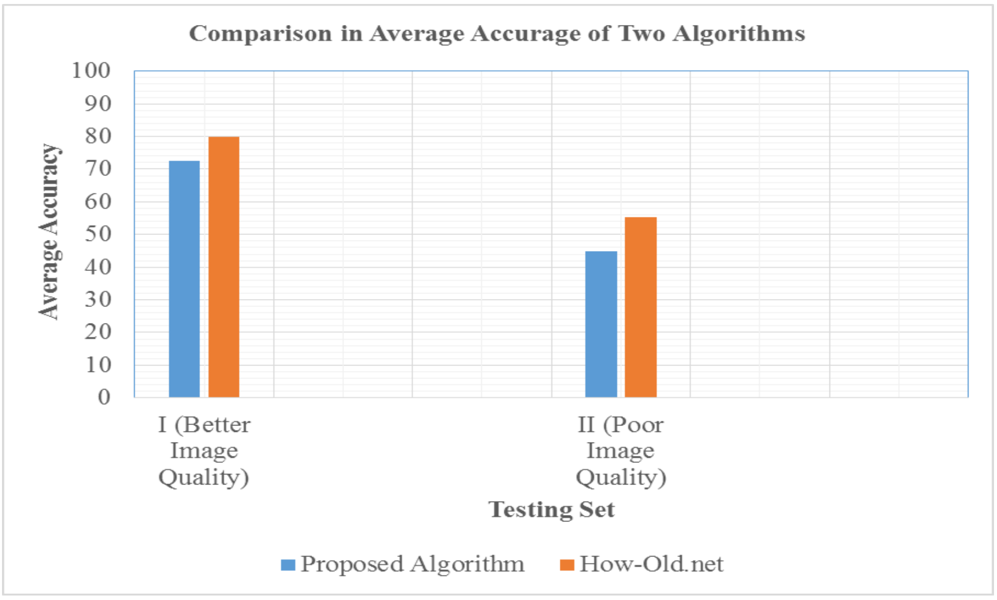


Figure 4.8 Comparison in Average Accuracy of Two Algorithms

Referring to Figure 4.7, it can be observed that the proposed algorithm works more accurately than How-Old.net in age group estimation on youngest age group of testing set with good quality images. However, How-Old.net outperforms the proposed algorithm on other age groups of the testing set, and testing set with poor quality image. Referring to Figure 4.8, How-Old.net has average accuracy higher than the proposed algorithm. The accuracy of two algorithms different by approximately 10%.

How-Old.net is developed based face detection API. This face detection API is available in Azure Machine Learning Gallery, where many finished intelligent services are available. Face API is capable of detecting and extracting information of faces in a given image, with maximum 64 faces. In face API, the positions of detected faces will be marked with rectangles, which come along with a series of face related attributes such as landmarks, pose, age and gender of individuals. Nevertheless, the detailed information of its working principle is unable to be obtained from their sites.

## 4.4 Comparison with Previous Researches

The difference between this proposed algorithm and two of the previous researches are tabulated in Table 4.4.

**Table 4.4: Comparison with Previous Researches**

| Paper/ Differences | Age Group Estimation using Face Features, 2013 | Proposed algorithm | Age Group Estimation using Face Angle, 2012 |
|---|---|---|---|
| Pose Correction | None | Plane rotation | None |
| Noise Removal | None | Median filtering | None |
| Image Enhancement | None | Histogram stretching | Histogram equalization |
| Features Used | Distances between eyes, nose, lip and chin, wrinkle feature | Wrinkle feature only | Face angle only |
| Databases | Own database of 50 persons | FG-NET, MORPH-II | Own database |
| Accuracy | 62% if based on wrinkle feature only | 72.66% | 85% |

In terms of image enhancement, the paper published in 2012 used histogram equalization to improve the image contrast while another paper did not apply any technique to improve the image contrast. In this proposed algorithm, image stretching is used to improve the image contrast. Both histogram equalization and image stretching have their own advantages and disadvantages, which makes them being used in different kind of applications. Histogram equalization, which is often used in x-ray applications, gives bone structure a clearer view. Similarly, it will enhance the eyes and mouth in the image if being applied, which makes the detection eyeballs and mouth easier. However, histogram equalization will cause the wrinkle feature to diminish when the image is over-saturated after processed. This is why image stretching is used in the proposed algorithm instead of histogram equalization.

The research published in year 2012 used only face angle as the feature to determine the age group of the individual. While the paper published in year 2013 used distances between eyes, lip, nose and chin as additional feature in determining the age group of the individual. In contrast to that, this proposed algorithm used only wrinkles to perform age group estimation. It is better to include more features in performing age group estimation. Taking more features into consideration for age group estimation makes an algorithm to be more robust to the changes in image quality and resolution. However there are insufficient publicly available aging databases with higher image quality and good head poses, while age-related features could be hardly obtained from poorer quality images.

Both previous researches used their own databases of images in their study, which are in the form of captured images. The paper published in year 2012 used an image database of 50 persons, while another paper in year 2013 did not mention the size of image database used. The proposed algorithm uses a relatively larger size of image database, which combines FG-NET Aging database and MORPH-II database, and has over 600 images with large variations in head pose, lighting and expression for training and testing. The estimation of age or age group is a difficult task due to the fact that various culprits cause different aging rate of human faces. Facial aging could be caused by diseases, environment, stress and unhealthy lifestyle such as smoking or stay up late. All these factors have to be taken into consideration when selecting images for training purpose. It would be better if an algorithm uses more images for training and testing, so that the result obtained provides better generalization.

In addition, both papers did not use apply technique for pose correction and removal of image noise. Plane rotation is implemented in the proposed algorithm using a slider in the user interface, so that the ill-posed images could be corrected. Median filtering is used to filter out noises in the image too. If the pose variation or out-of-plane rotation is not corrected by any method, it will lead to inaccuracy in the detection and extraction of facial features. Additionally, if the noises on the images are not removed, the algorithm might recognize the noises as part of the age-related features. All these will probably affects the overall performance of the algorithm.

## 4.5 Problems Faced and Solutions Taken

One of the problems in age group estimation is the difficulty to obtain sufficient and suitable images for training and testing purpose. It is hard to obtain publicly available databases with age-labeled images. FG-NET Aging database is one of the databases that could be downloaded by the public. Nevertheless, age distribution is not balanced. The images in older age groups are limited. Hence, a portion of images from MORPH-II database is combined with images from FG-NET Aging database to form new datasets, which are more balanced in the age distribution. These new datasets are then adopted to train and test the proposed algorithm.

Images in MORPH-II are captured under controlled condition; most of the images in FG-NET Aging database are out-of-plane rotated. If out-of-plane rotation is not corrected, it will lead to inaccuracy in detection of regions of interest. Hence, slider is used in the user interface to allow user to correct the image. By dragging the slider, user could correct the image to the correct pose.

In addition, some images in FG-NET Aging database are scanned photographs. These images, especially old photographs, have some lines or noises on it. The algorithm will transform those noises to form of white pixels, which then will be summed up as wrinkle feature. Median filtering is used to remove the noises in the images. The image quality can be slightly improved without blurring out the edges.

## 4.6 Weaknesses of the Algorithm

This wrinkle-based age group estimation is highly sensitive to the image quality. Any noise will add unnecessary lines to the processed image. Then these lines in the form of white pixels will be assumed as wrinkles. Besides, face wrinkles would be blurred out in a blurred image. This will decrease the accuracy in determining age group of older individuals. If an individual wears make up in the image for age group estimation, the wrinkles on the face would be covered too. Similarly if an individual has some scars on the face, the algorithm will transform the scar region into a region with white boundary. The wrinkles at the scar region might be covered.

As mentioned in the previous chapter, the regions of interest for this algorithm are located at forehead region, under-eye regions and end-of-the-eyes regions. These regions should not be covered by hair, beard or any accessories to ensure better performance. If these regions are covered, the edges of anything covering those regions would be detected and transformed into white edges, which will be assumed as wrinkle feature too.

Besides, the pose correction is done manually, where user determines the angle of plane rotation by dragging the slider. With naked eyes, the eyes of the individual in the face image might seems to be on the same horizontal line after plane rotation. In fact, the rotation might not ideally done. The rotation might be only one degree away from the correct pose but this difference could cause a great difference in the result obtained.

In addition, the method of classification for this proposed algorithm is simply based on the mean value of the wrinkle feature. The datasets used for this algorithm are huge and complicated to be classified by using simple mean value. Such separation or classification method gives poor generalization, which will actually degrade the performance of the algorithm.

# CHAPTER 5

# CONCLUSION AND RECOMMENDATIONS

## 5.1 Recommendations and Future Works

Performance of the proposed algorithm could be improved if some modifications could be made. As discussed in the previous chapter, wrinkle feature is not a robust method for the use of age group estimation. In future works, additional features could be included in the algorithm. When human ages from birth to adulthood, there is craniofacial growth and development. Craniofacial growth and development leads to a change in geometric ratio in human faces. Besides, babies have rounder face compared to children. Face angles of human from different age stages are different too. Such features can be used as separator to classify minors and adults.

Manual rotation and correction of image do not necessarily produce an ideal result. Hence, rotation of image using slider could be substituted by automatic pose correction. This technique can be implemented by detecting the centers of eyeballs. Then, plane rotation could be done by setting the algorithm to rotate the image automatically with reference to the pose variation detected.

As mentioned previously, some images in the database are scanned photographs. These old photographs have some lines or noises on it, will lead to unwanted lines in the binary image. The lost or deteriorated parts of these photographs can be reconstructed and restored using the technique of image inpainting. The technique will automatically fills in the defected regions with in-formation surrounding them.

Furthermore, classification method that provides better generalization should be used. It could be by Support Vector Machine (SVM) classification method or Neural Networks (NN) based classification method. The ideal separators between age groups could be determined, which will provide a higher accuracy and better performance of the algorithm.

## 5.2 Conclusion

In this approach, a wrinkle-based age group estimation method is thoroughly described. The proposed approach provides a method to classify individuals of different ages into three main age groups.

This approach able to detect and localize face from an input facial image. The approach provides a method to localize the regions of interest in the input facial image, which is based on the ratio of face. Extraction of wrinkle feature from the facial image is performed by applying Canny edge detection to the grayscale image. The wrinkle feature extracted is quantified by computing the ratio of white pixels in the regions of interest. The quantitative data is collected and used to classify images into age groups.

For better performance of the algorithm, the image should be frontal and upright facial image, which consists of single face with glass free and clear forehead region. The average accuracy of this algorithm is up to 72.66% if being tested on better quality images. More age-related features, such as geometric features and face angle can be introduced to the algorithm so that the accuracy can be improved.

# REFERENCES

Chang, K. Y., Chen, C. S., & Hung, Y. P., 2010. ‘A Ranking Approach for Human Age Estimation based on Face Images’. 2010 20th International Conference on Pattern Recognition (ICPR), pp. 3396-3399.

Eidinger, E., Enbar, R., & Hassner, T., 2014. ‘Age and Gender Estimation of Unfiltered Faces’. *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, October, pp. 2170-2179.

GRD, P., 2013. *Introduction to Human Age Estimation using Face Images*. Research Papers, Slovak University of Technology in Bratislava.

Hayashi, J., Yasumoto, M., Ito, H., & Koshimizu, H., 2002. ‘Age and Gender Estimation based on Wrinkle Texture and Color of Facial Images’. *Proceedings*. 16th International Conference on Pattern Recognition, vol. 1, pp. 405-408.

Hewahi, N., Olwan, A., Tubeel, N., EL-Asar, S., & Abu-Sultan, Z., 2010. ‘Age Estimation based on Neural Networks using Face Features’. *Journal of Emerging Trends in Computing and Information Sciences*, vol. 1, no. 3, October, pp. 61-67.

Han, H., & Jain, A. K., 2014. ‘Age, Gender and Race Estimation from Unconstrained Face Images’. MSU Technical Report, pp. 1-9.

Han, H., Otto, C., & Jain, A. K., 2013. ‘Age Estimation from Face Images: Human vs. Machine Performance’. 2013 International Conference on Biometrics (ICB), Madrid, pp. 1-8.

Jana, R., Datta, D., & Saha, R., 2013. ‘Age Group Estimation using Face Features’. *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 3, no. 2, August, pp. 130-134.

Jana, R., Pal, H., & Chowdhury, A. R., 2012. ‘Age Group Estimation using Face Angle’. *IOSR Journal of Computer Engineering (IOSRJCE)*, vol. 7, no. 5, November, pp. 35-39.

Lin, C. T., Li, D. L., Lai, J. H., Han, M. F., & Chang, J. Y., 2012. ‘Automatic Age Estimation System for Face Images’. *International Journal of Advanced Robotic Systems*, vol. 9, August, pp. 1-9.

Panis, G., Lanitis, A., Tsapatsoulis, N., & Cootes, T. F., 2015. ‘An Overview of Research on Facial Aging using the FG-NET Aging Database’. *IET Biometrics*. Available: https://www.dropbox.com/s/p4gpn4w0lebl37e/IET_BMT2015.pdf?dl=0

Ricanek, K., & Tesafaye, T., 2006. ‘MORPH: A Longitudinal Image Database of Normal Adult Age-Progression’. IEEE 7th International Conference on Automatic Face and Gesture Recognition, Southampton, UK, pp. 341-345.

Roy, H., Bhattacharjee, D., Nasipuri, M. & Basu, D. K., 2012. ‘Age Range Estimation from Human Face Images Using Face Triangle Formation’. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, vol. 2, no. 1, March, pp. 155-160.

Thukral, P., Mitra, K. & Chellappa, R., 2012. ‘A Hierarchical Approach for Human Age Estimation’. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, pp. 1529-1532.

TIN, H. H. K., 2011. ‘Gender and Age Estimation based on Facial Images’. *Acta Technica Napocensis Electronics and Telecommunications*, vol. 52, no. 3, October, pp. 37-40.

# APPENDICES

## Appendix A: Coding for Rotation Interface

```
function varargout = rotint(varargin)
% ROTINT MATLAB code for rotint.fig
%       ROTINT, by itself, creates a new ROTINT or raises the existing
%       singleton*.
%
%       H = ROTINT returns the handle to a new ROTINT or the handle to
%       the existing singleton*.
%
%       ROTINT('CALLBACK',hObject,eventData,handles,...) calls the local
%       function named CALLBACK in ROTINT.M with the given input arguments.
%
%       ROTINT('Property','Value',...) creates a new ROTINT or raises the
%       existing singleton*.  Starting from the left, property value pairs are
%       applied to the GUI before rotint_OpeningFcn gets called.  An
%       unrecognized property name or invalid value makes property application
%       stop.  All inputs are passed to rotint_OpeningFcn via varargin.
%
%       *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%       instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help rotint

% Last Modified by GUIDE v2.5 14-Aug-2015 11:21:13

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @rotint_OpeningFcn, ...
                   'gui_OutputFcn',  @rotint_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before rotint is made visible.
function rotint_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to rotint (see VARARGIN)

% Choose default command line output for rotint
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes rotint wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = rotint_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
angle=round(get(hObject,'Value'));
image=imread(handles.image);
image=imrotate(image, angle);
imshow(image);
[~, baseFileName, extension] = fileparts(baseFileName);
outputBaseFileName = sprintf('faceimage.jpg', baseFileName);
outputFullFileName = fullfile(folder, outputBaseFileName)
imwrite(newimage, outputFullFileName);


% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes on button press in load_image.
function load_image_Callback(hObject, eventdata, handles)
% hObject        handle to load_image (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles        structure with handles and user data (see GUIDATA)
[fn pn] = uigetfile('*.jpg','select jpg file');
handles.image=strcat(pn, fn);
axes(handles.axes1);
imshow(handles.image);

guidata(hObject,handles);


% --- Executes on button press in colour.
function colour_Callback(hObject, eventdata, handles)
% hObject        handle to colour (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles        structure with handles and user data (see GUIDATA)
close all;
if isfield(handles, 'image')
colourcanny;
end

% --- Executes on button press in grayscale.
function grayscale_Callback(hObject, eventdata, handles)
% hObject        handle to grayscale (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles        structure with handles and user data (see GUIDATA)
close all;
if isfield(handles, 'image')
graycanny;
end
```

## Appendix B: Coding for Image Type Interface

```
function varargout = imgtype_int(varargin)
% IMGTYPE_INT MATLAB code for imgtype_int.fig
%       IMGTYPE_INT, by itself, creates a new IMGTYPE_INT or raises the existing
%       singleton*.
%
%       H = IMGTYPE_INT returns the handle to a new IMGTYPE_INT or the handle to
%       the existing singleton*.
%
%       IMGTYPE_INT('CALLBACK',hObject,eventData,handles,...) calls the local
%       function named CALLBACK in IMGTYPE_INT.M with the given input
arguments.
%
%       IMGTYPE_INT('Property','Value',...) creates a new IMGTYPE_INT or raises the
%       existing singleton*.  Starting from the left, property value pairs are
%       applied to the GUI before imgtype_int_OpeningFcn gets called.  An
%       unrecognized property name or invalid value makes property application
%       stop.  All inputs are passed to imgtype_int_OpeningFcn via varargin.
%
%       *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%       instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help imgtype_int

% Last Modified by GUIDE v2.5 14-Aug-2015 12:01:14

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @imgtype_int_OpeningFcn, ...
                   'gui_OutputFcn',  @imgtype_int_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before imgtype_int is made visible.
function imgtype_int_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to imgtype_int (see VARARGIN)
% Choose default command line output for imgtype_int
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes imgtype_int wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = imgtype_int_OutputFcn(hObject, eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject      handle to figure
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
imshow('faceimage.jpg');


% --- Executes on button press in colour.
function colour_Callback(hObject, eventdata, handles)
% hObject      handle to colour (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
close all;
colourcanny;
if isfield(handles, 'image')
end


% --- Executes on button press in grayscale.
function grayscale_Callback(hObject, eventdata, handles)
% hObject      handle to grayscale (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
close all;
graycanny;
if isfield(handles, 'image')
end
```

# Appendix C: Coding for Colour Images

```
%%Read image file as input
image = imread('C:\Users\Pei Kee Tiong\Documents\MATLAB\faceimage.jpg');

%%Median filtering to remove noise
r=medfilt2(image(:,:,1), [3 3]);
g=medfilt2(image(:,:,2), [3 3]);
b=medfilt2(image(:,:,3), [3 3]);
newimage=cat(3,r,g,b);

%%Conversion to grayscale image
grayimage=rgb2gray(newimage);

%%Contrast strectching and histogram equalization
Stretched_image = imadjust(grayimage,stretchlim(grayimage),[ ]);

figure,
subplot(1,2,1), imshow(image);
title('Original Image');
subplot(1,2,2), imshow(newimage);
title('Image After Noise Removal');

%%Face detection and resize
FaceDetect = vision.CascadeObjectDetector;
bbox_face=step(FaceDetect,Stretched_image);
face=imcrop(Stretched_image,bbox_face);
outface=imresize(face,1.5,'bicubic');

figure,
subplot(1,2,1), imshow(grayimage);
title('Grayscale Image')
subplot(1,2,2),imshow(Stretched_image);
title('Stretched Image')

%%crop detected face
figure,
subplot(1,3,1), imshow(Stretched_image);
title('Input Image')
subplot(1,3,2), imshow(Stretched_image),
rectangle('Position',bbox_face,'LineWidth',2,'LineStyle','-','EdgeColor','r');
title('Detected Face')
subplot(1,3,3), imshow(outface);
title('Cropped Face')

%%Forehead detection
bbox_forehead(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.3;
bbox_forehead(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.05;
bbox_forehead(1,3)=bbox_face(1,4)*0.45;
bbox_forehead(1,4)=bbox_face(1,4)*0.175;

%%End-of-eye region detection
bbox0_endeye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.1;
bbox0_endeye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.3;
bbox0_endeye(1,3)=bbox_face(1,4)*0.1;
bbox0_endeye(1,4)=bbox_face(1,4)*0.2;
```

```
bbox1_endeye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.8;
bbox1_endeye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.3;
bbox1_endeye(1,3)=bbox_face(1,4)*0.1;
bbox1_endeye(1,4)=bbox_face(1,4)*0.2;

%%Under-eye region detection
bbox0_undereye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.15;
bbox0_undereye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.45;
bbox0_undereye(1,3)=bbox_face(1,4)*0.25;
bbox0_undereye(1,4)=bbox_face(1,4)*0.2;

bbox1_undereye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.6;
bbox1_undereye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.45;
bbox1_undereye(1,3)=bbox_face(1,4)*0.25;
bbox1_undereye(1,4)=bbox_face(1,4)*0.2;

%%show detection of regions of interest
figure,imshow(Stretched_image);
title('Detection of Regions of Interest');
rectangle('Position',bbox_forehead,'LineWidth',2,'LineStyle','-','EdgeColor','r');

rectangle('Position',bbox0_endeye,'LineWidth',2,'LineStyle','-','EdgeColor','g');
rectangle('Position',bbox1_endeye,'LineWidth',2,'LineStyle','-','EdgeColor','g');

rectangle('Position',bbox0_undereye,'LineWidth',2,'LineStyle','-','EdgeColor','b');
rectangle('Position',bbox1_undereye,'LineWidth',2,'LineStyle','-','EdgeColor','b');

%%Canny edge detection
BW=edge(Stretched_image,'canny');

%%Show image before and after canny edge detection
figure,
subplot(1,2,1), imshow(Stretched_image);
title('Stretched Image')
subplot(1,2,2), imshow(BW);
title('Image After Canny Filtering')

%%crop forehead, end-of-eye and undereye region
forehead=imcrop(Stretched_image,bbox_forehead);
endeye0=imcrop(Stretched_image,bbox0_endeye);
endeye1=imcrop(Stretched_image,bbox1_endeye);
undereye0=imcrop(Stretched_image,bbox0_undereye);
undereye1=imcrop(Stretched_image,bbox1_undereye);

%%crop forehead, end-of-eye and undereye region
BW_forehead=imcrop(BW,bbox_forehead);
BW_endeye0=imcrop(BW,bbox0_endeye);
BW_endeye1=imcrop(BW,bbox1_endeye);
BW_undereye0=imcrop(BW,bbox0_undereye);
BW_undereye1=imcrop(BW,bbox1_undereye);

%%show cropped regions with canny edge detection
figure,
subplot(1,2,1), imshow(forehead);
title('forehead');
subplot(1,2,2), imshow(BW_forehead);
```

```
figure,
subplot(1,2,1), imshow(endeye0);
title('end-of-eye region 1');
subplot(1,2,2), imshow(BW_endeye0);

figure,
subplot(1,2,1), imshow(endeye1);
title('end-of-eye region 2');
subplot(1,2,2), imshow(BW_endeye1);

figure,
subplot(1,2,1), imshow(undereye0);
subplot, title('undereye region 1');
subplot(1,2,2), imshow(BW_undereye0);

figure,
subplot(1,2,1), imshow(undereye1);
title('undereye region 2');
subplot(1,2,2), imshow(BW_undereye1);


%%Calculate pixels
foreheadwhite=sum(sum(BW_forehead));
foreheadblack=sum(sum(BW_forehead== 0));
foreheadtotal=foreheadwhite+foreheadblack;

endeye0white=sum(sum(BW_endeye0));
endeye0black=sum(sum(BW_endeye0== 0));
endeye0total=endeye0white+endeye0black;

endeye1white=sum(sum(BW_endeye1));
endeye1black=sum(sum(BW_endeye1== 0));
endeye1total=endeye1white+endeye1black;

undereye0white=sum(sum(BW_undereye0));
undereye0black=sum(sum(BW_undereye0== 0));
undereye0total=undereye0white+undereye0black;

undereye1white=sum(sum(BW_undereye1));
undereye1black=sum(sum(BW_undereye1== 0));
undereye1total=undereye1white+undereye0black;

foreheadratio=foreheadwhite/foreheadtotal;
endeye0ratio=endeye0white/endeye0total;
endeye1ratio=endeye1white/endeye1total;
undereye0ratio=undereye0white/undereye0total;
undereye1ratio=undereye1white/undereye1total;
totalratio=foreheadratio+endeye0ratio+endeye1ratio+undereye0ratio+undereye1ratio;

if (totalratio<=0.3543)
        kid_int;
end
if ((totalratio>0.3543) && (totalratio<=0.5487))
        young_int;
end
if (totalratio>0.5487)
            old_int;
end
```

# Appendix D: Coding for Grayscale Images

```
%%Read image file as input
image = imread('C:\Users\Pei Kee Tiong\Documents\MATLAB\faceimage.jpg');

%%Median filtering to remove noise
newimage=medfilt2(image, [3 3]);

%%Contrast strectching and histogram equalization
Stretched_image = imadjust(newimage,stretchlim(newimage),[ ]);

figure, imshow(image);
title('Original Image');
figure, imshow(newimage);
title('Image After Noise Removal');

%%Face detection and resize
FaceDetect = vision.CascadeObjectDetector;
bbox_face=step(FaceDetect,Stretched_image);
face=imcrop(Stretched_image,bbox_face);
outface=imresize(face,1.5,'bicubic');

figure,
subplot(1,2,1), imshow(newimage);
title('Image after noise removal')
subplot(1,2,2),imshow(Stretched_image);
title('Stretched Image')

%%crop detected face
figure,
subplot(1,3,1), imshow(Stretched_image);
title('Input Image')
subplot(1,3,2), imshow(Stretched_image),
rectangle('Position',bbox_face,'LineWidth',2,'LineStyle','-','EdgeColor','r');
title('Detected Face')
subplot(1,3,3), imshow(outface);
title('Cropped Face')

%%Forehead detection
bbox_forehead(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.3;
bbox_forehead(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.05;
bbox_forehead(1,3)=bbox_face(1,4)*0.45;
bbox_forehead(1,4)=bbox_face(1,4)*0.175;

%%End-of-eye region detection
bbox0_endeye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.1;
bbox0_endeye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.3;
bbox0_endeye(1,3)=bbox_face(1,4)*0.1;
bbox0_endeye(1,4)=bbox_face(1,4)*0.2;

bbox1_endeye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.8;
bbox1_endeye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.3;
bbox1_endeye(1,3)=bbox_face(1,4)*0.1;
bbox1_endeye(1,4)=bbox_face(1,4)*0.2;
```

```
%%Under-eye region detection
bbox0_undereye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.15;
bbox0_undereye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.45;
bbox0_undereye(1,3)=bbox_face(1,4)*0.25;
bbox0_undereye(1,4)=bbox_face(1,4)*0.2;

bbox1_undereye(1,1)=bbox_face(1,1)+bbox_face(1,4)*0.6;
bbox1_undereye(1,2)=bbox_face(1,2)+bbox_face(1,4)*0.45;
bbox1_undereye(1,3)=bbox_face(1,4)*0.25;
bbox1_undereye(1,4)=bbox_face(1,4)*0.2;

%%show detection of regions of interest
figure,imshow(Stretched_image);
title('Detection of Regions of Interest');
rectangle('Position',bbox_forehead,'LineWidth',2,'LineStyle','-','EdgeColor','r');

rectangle('Position',bbox0_endeye,'LineWidth',2,'LineStyle','-','EdgeColor','g');
rectangle('Position',bbox1_endeye,'LineWidth',2,'LineStyle','-','EdgeColor','g');

rectangle('Position',bbox0_undereye,'LineWidth',2,'LineStyle','-','EdgeColor','b');
rectangle('Position',bbox1_undereye,'LineWidth',2,'LineStyle','-','EdgeColor','b');

%%Canny edge detection
BW=edge(Stretched_image,'canny');

%%Show image before and after canny edge detection
figure,
subplot(1,2,1), imshow(Stretched_image);
title('Stretched Image')
subplot(1,2,2), imshow(BW);
title('Image After Canny Filtering')


%%crop forehead, end-of-eye and undereye region
forehead=imcrop(Stretched_image,bbox_forehead);
endeye0=imcrop(Stretched_image,bbox0_endeye);
endeye1=imcrop(Stretched_image,bbox1_endeye);
undereye0=imcrop(Stretched_image,bbox0_undereye);
undereye1=imcrop(Stretched_image,bbox1_undereye);

%%crop forehead, end-of-eye and undereye region
BW_forehead=imcrop(BW,bbox_forehead);
BW_endeye0=imcrop(BW,bbox0_endeye);
BW_endeye1=imcrop(BW,bbox1_endeye);
BW_undereye0=imcrop(BW,bbox0_undereye);
BW_undereye1=imcrop(BW,bbox1_undereye);

%%show cropped regions with canny edge detection
figure,
subplot(1,2,1), imshow(forehead);
title('forehead');
subplot(1,2,2), imshow(BW_forehead);

figure,
subplot(1,2,1), imshow(endeye0);
title('end-of-eye region 1');
subplot(1,2,2), imshow(BW_endeye0);
```

```
figure,
subplot(1,2,1), imshow(endeye1);
title('end-of-eye region 2');
subplot(1,2,2), imshow(BW_endeye1);


figure,
subplot(1,2,1), imshow(undereye0);
subplot, title('undereye region 1');
subplot(1,2,2), imshow(BW_undereye0);


figure,
subplot(1,2,1), imshow(undereye1);
title('undereye region 2');
subplot(1,2,2), imshow(BW_undereye1);


%%Calculate pixels
foreheadwhite=sum(sum(BW_forehead));
foreheadblack=sum(sum(BW_forehead== 0));
foreheadtotal=foreheadwhite+foreheadblack;

endeye0white=sum(sum(BW_endeye0));
endeye0black=sum(sum(BW_endeye0== 0));
endeye0total=endeye0white+endeye0black;

endeye1white=sum(sum(BW_endeye1));
endeye1black=sum(sum(BW_endeye1== 0));
endeye1total=endeye1white+endeye1black;

undereye0white=sum(sum(BW_undereye0));
undereye0black=sum(sum(BW_undereye0== 0));
undereye0total=undereye0white+undereye0black;

undereye1white=sum(sum(BW_undereye1));
undereye1black=sum(sum(BW_undereye1== 0));
undereye1total=undereye1white+undereye0black;

foreheadratio=foreheadwhite/foreheadtotal;
endeye0ratio=endeye0white/endeye0total;
endeye1ratio=endeye1white/endeye1total;
undereye0ratio=undereye0white/undereye0total;
undereye1ratio=undereye1white/undereye1total;
totalratio=foreheadratio+endeye0ratio+endeye1ratio+undereye0ratio+undereye1ratio;

fprintf('Total ratio of white wrinkle using canny filter is %f', totalratio);

if (totalratio<=0.3543)
        kid_int;
end
if ((totalratio>0.3543) && (totalratio<=0.5487))
        young_int;
end
if (totalratio>0.5487)
            old_int;
end
```

# Appendix E: Coding for Result Interface (Children)

```
function varargout = kid_int(varargin)
% KID_INT MATLAB code for kid_int.fig
%        KID_INT, by itself, creates a new KID_INT or raises the existing
%        singleton*.
%
%        H = KID_INT returns the handle to a new KID_INT or the handle to
%        the existing singleton*.
%
%        KID_INT('CALLBACK',hObject,eventData,handles,...) calls the local
%        function named CALLBACK in KID_INT.M with the given input arguments.
%
%        KID_INT('Property','Value',...) creates a new KID_INT or raises the
%        existing singleton*.  Starting from the left, property value pairs are
%        applied to the GUI before kid_int_OpeningFcn gets called.  An
%        unrecognized property name or invalid value makes property application
%        stop.  All inputs are passed to kid_int_OpeningFcn via varargin.
%
%        *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%        instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help kid_int

% Last Modified by GUIDE v2.5 15-Aug-2015 22:53:25

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @kid_int_OpeningFcn, ...
                   'gui_OutputFcn',  @kid_int_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before kid_int is made visible.
function kid_int_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to kid_int (see VARARGIN)
% Choose default command line output for kid_int
handles.output = hObject;
kidback=imread('C:\Users\Pei Kee Tiong\Desktop\Final Year Project\Figures\Childhood.jpg');
imshow(kidback);
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes kid_int wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = kid_int_OutputFcn(hObject, eventdata, handles)
% varargout   cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

## Appendix F: Coding for Result Interface (Young Adults)

```
function varargout = young_int(varargin)
% YOUNG_INT MATLAB code for young_int.fig
%       YOUNG_INT, by itself, creates a new YOUNG_INT or raises the existing
%       singleton*.
%
%       H = YOUNG_INT returns the handle to a new YOUNG_INT or the handle to
%       the existing singleton*.
%
%       YOUNG_INT('CALLBACK',hObject,eventData,handles,...) calls the local
%       function named CALLBACK in YOUNG_INT.M with the given input arguments.
%
%       YOUNG_INT('Property','Value',...) creates a new YOUNG_INT or raises the
%       existing singleton*.  Starting from the left, property value pairs are
%       applied to the GUI before young_int_OpeningFcn gets called.  An
%       unrecognized property name or invalid value makes property application
%       stop.  All inputs are passed to young_int_OpeningFcn via varargin.
%
%       *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%       instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help young_int

% Last Modified by GUIDE v2.5 16-Aug-2015 11:26:27

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @young_int_OpeningFcn, ...
                   'gui_OutputFcn',  @young_int_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before young_int is made visible.
function young_int_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to young_int (see VARARGIN)
% Choose default command line output for young_int
handles.output = hObject;
youngback=imread('C:\Users\Pei Kee Tiong\Desktop\Final Year
Project\Figures\YoungAdult.jpg');
imshow(youngback);
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes young_int wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = young_int_OutputFcn(hObject, eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject      handle to figure
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

## Appendix G: Coding for Result Interface (Old Adults)

```
function varargout = old_int(varargin)
% OLD_INT MATLAB code for old_int.fig
%       OLD_INT, by itself, creates a new OLD_INT or raises the existing
%       singleton*.
%
%       H = OLD_INT returns the handle to a new OLD_INT or the handle to
%       the existing singleton*.
%
%       OLD_INT('CALLBACK',hObject,eventData,handles,...) calls the local
%       function named CALLBACK in OLD_INT.M with the given input arguments.
%
%       OLD_INT('Property','Value',...) creates a new OLD_INT or raises the
%       existing singleton*.  Starting from the left, property value pairs are
%       applied to the GUI before old_int_OpeningFcn gets called.  An
%       unrecognized property name or invalid value makes property application
%       stop.  All inputs are passed to old_int_OpeningFcn via varargin.
%
%       *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%       instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help old_int

% Last Modified by GUIDE v2.5 16-Aug-2015 11:28:06

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @old_int_OpeningFcn, ...
                   'gui_OutputFcn',  @old_int_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before old_int is made visible.
function old_int_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to old_int (see VARARGIN)
% Choose default command line output for old_int
handles.output = hObject;
oldage=imread('C:\Users\Pei Kee Tiong\Desktop\Final Year Project\Figures\OldAdult.jpg');
imshow(oldage);
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes old_int wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = old_int_OutputFcn(hObject, eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject       handle to figure
% eventdata    reserved - to be defined in a future version of MATLAB
% handles       structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```