FACE DETECTION AND LIP LOCALIZATION

A Thesis
presented to
the Faculty of California Polytechnic State University,
San Luis Obispo

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Electrical Engineering

By

Benafsh Nadir Husain

August 2011

COMMITTEE MEMEBERSHIP

TITLE:	Face Detection and Lip Localization
AUTHOR:	Benafsh Husain
DATE SUBMITTED:	August 2011
COMMITTEE CHAIR:	Dr. Xiaozheng (Jane) Zhang Associate Professor of Electrical Engineering, Advisor
COMMITTEE MEMBER:	Dr. John Saghri Professor of Electrical Engineering,
COMMITTEE MEMBER:	Dr. Fred DePiero Professor of Electrical Engineering, Associate Dean, College of Engineering

ABSTRACT

Face Detection and Lip Localization

Benafsh Husain

Integration of audio and video signals for automatic speech recognition has become an important field of study. The Audio-Visual Speech Recognition (AVSR) system is known to have accuracy higher than audio-only or visual-only system. The research focused on the visual front end and has been centered around lip segmentation. Experiments performed for lip feature extraction were mainly done in constrained environment with controlled background noise. In this thesis we focus our attention to a database collected in the environment of a moving car which hampered the quality of the imagery.

We first introduce the concept of illumination compensation, where we try to reduce the dependency of light from over- or under-exposed images. As a precursor to lip segmentation, we focus on a robust face detection technique which reaches an accuracy of 95%. We have detailed and compared three different face detection techniques and found a successful way of concatenating them in order to increase the overall accuracy. One of the detection techniques used was the object detection algorithm proposed by Viola-Jones. We have experimented with different color spaces using the Viola-Jones algorithm and have reached interesting conclusions.

Following face detection we implement a lip localization algorithm based on the vertical gradients of hybrid equations of color. Despite the challenging background and image quality, success rate of 88% was achieved for lip segmentation.

ACKNOWLEDGMENTS

The end of this thesis marks the culmination of my academic career at Cal Poly. My experiences at Cal Poly have inspired and shaped me to be thoroughly involved in my subject and in love with what I do.

I dedicate this thesis to my family, my mom Tehmina and my brother Talib. Both of you have supported me in every way possible.

I also want to thank Chiweng Kam, without whose guidance, motivation and positive attitude it would have been impossible to have seen this thesis through.

I must also express my gratitude for Dr. Xiaozheng (Jane) Zhang for her guidance throughout this endeavor, her review of my work, and for the freedom in direction she allowed me to take on this project. I must also thank Dr John Saghri and Dr Fred DePiero for serving on my thesis committee and for fostering my interest in image processing and pattern recognition.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1. Background

Speech and speaker recognition has become a popular field of research with its multitude of applications. Voice recognition softwares are commonly seen in car navigation systems and mobile phones. Research in speech recognition aims to increase the ease with which humans interact with machines.

Early research on speech technology was primarily based on identification of the audio signal and hence the result depended on the quality of the audio signal available. Research now aims to integrate the audio information with the additional visual information, in order to eliminate the shortcomings of a distorted audio signal and enhance the accuracy of speech and speaker recognition [1]. The first work on audio-visual speech recognition (AVSR) system was published by Petajan in 1984 [2]. Petajan applied gray-scale thresholding to obtain prominent regions such as the nose and lips. The method of nostril tracking was used to track the faces over a sequence of image frames. Moreover, he demonstrated a method of combining the audio and visual signals and showed improved speech recognition results over the audio-only solution.

One of the widely accepted models for AVSR System is depicted in Figure 1.1 [3]. In this model, audio and video signals are separated and independently processed to extract relevant features. Once the relevant information is picked from each signal; they are fused together and then used for an improved speech or speaker recognition system. The preprocessing of audio signals includes signal enhancement, tracking environmental and channel noise, feature estimation and smoothing [3]. Preprocessing of the video signals typically consists of solving the challenging problems of detecting and tracking the face along with the lips. Feature extraction is an important topic and is discussed in length in the following sections. Lastly, in the case of audio-visual fusion, there are several general approaches existing based on their discrimination ability and reliability. Fusion approaches are broadly classified as early integration (pre-mapping fusion), intermediate integration (midst-mapping) and late integration (post-mapping) [3].

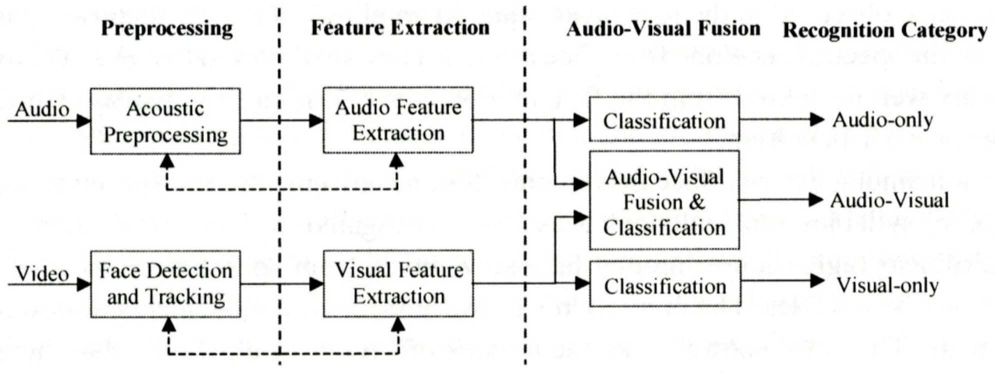


Figure 1.1: Block Diagram for Audio-Visual Speech Recognition System [3]

The aim of this thesis is to restrict ourselves to the visual front end of the entire system, more specifically, face detection and lip segmentation. The visual front end consists of preprocessing of the images (frames of the sequence) and feature extraction (lip region).The stage of feature detection and extraction is most essential in case of visual speech recognition. Without a robust detection algorithm, next stages of the system would automatically have low accuracy. Lip segmentation in unconstrained environment has proven to be a challenge. Various background noises such as light conditions, mechanical vibrations, and poor image quality lower the accuracy of segmentation algorithms. In this thesis, we address the issues introduced by such images and aim to achieve a high accuracy.   Figure 1.2 illustrates an example of the visual front-end. The algorithm starts by face detection within the image. Then, lip segmentation is applied to face detected image. The example utilized AAM (Active Appearance Model), which combines the lip shape model with pixel statistical distribution around the mouth, to locate the lip contour. Visual features are then extracted for the recognizer. The applications of AVSR also include biometric speaker recognition, speech-to-text technology.

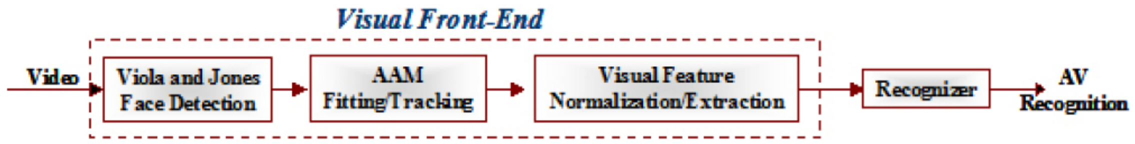

Visual Front-End
Video
Viola and Jones Face Detection
AAM Fitting/Tracking
Visual Feature Normalization/Extraction
Recognizer
AV Recognition

Figure 1.2: Sample Visual Front-End of the AVR system [3]

The in-car environment proves to be a challenge for both the audio and the visual feature extraction. The background noise and the mechanical vibrations introduce immense noise in audio signal. The non-uniformed lighting condition is particularly troublesome for the visual signal processing. The speech recognition system in modern cars today use only audio command signals which in certain cases are highly corrupted. Therefore, fusion of both the signal information becomes valuable in increasing the accuracy of the system. In this thesis, we utilize the AVICAR database for experimentation. The AVICAR database consists of 50 male and 50 female subjects in a car environment. Each subject travels at 5 different speeds and at each speed read scripts containing isolated digits, isolated letters, phone numbers, and sentences [4].

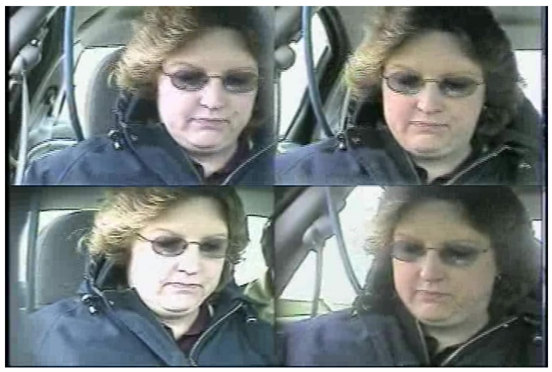


Figure 1.3: Sample Image of the AVICAR Database [4]

A video sequence of 86 subjects at all speeds is available [4]. For the purpose of converting the video into still shots, the freeware, ‘Free Video to JPG Converter 1.8.7’, was downloaded from the internet [5].

1.2. Previous Work in this Area

1.2.1. Face Detection

Research in the field of face detection gained immense popularity and several techniques have been experimented and published in the literature. Table 1-1 summarizes the 4 broad categories for face detection approaches [6]. See paper [6] for reference to the representative works listed in Table 1-1. There are clear overlaps in methods over the categories.

Table 1-1: Categorization of methods for Face Detection [6]

Approach	Representative Works
Knowledge-based	
	Multi-resolution rule-based method
Feature invariant	
- Facial Features	Grouping of edges
- Texture	Space Gray-Level Dependence Matrix (SGLD) of face pattern
- Skin Color	Mixture of Gaussian
- Multiple Features	Integration of skin color, size and shape
Template matching	
- Predefined face templates	Shape template
- Deformable templates	Active Shape Model (ASM)
Appearance-based method	
- Eigenface	Eigenvector decomposition and clustering
- Distribution-based	Gaussian distribution and multilayer perceptron
- Neural Network	Ensemble of neural networks and arbitration schemes
- Support Vector Machine (SVM)	SVM with polynomial kernel
- Naïve Bayes Classifier	Joint statistics of local appearance and position
- Hidden Markov Model (HMM)	Higher order statistics with HMM
- Information-Theoretical Approach	Kullback relative information

1) Knowledge-based methods: This approach uses the human knowledge of a face and its associated features to localize a face in the given image. An example is to use the rules that capture the relationships between facial features [6].
2) Feature invariant approaches: This approach aims to locate constant features which are not affected by factors such as pose or orientation [6].
3) Template matching methods: Templates or general patterns of faces are used to find the correlation between the template (stored patterns) and the input image [6].
4) Appearance based methods: Instead of a general model/pattern in case of template matching methods, this method aims to 'learn' the model from training images. An example would be the use of neural networks [6].

It can be noted that similar techniques are used for lip segmentation; some of which are mentioned further**.** Several reviews and surveys for face detection techniques have been published [7-13].

One of the most popular techniques in face detection today is the rapid object detection by Viola-Jones, which uses features reminiscent of Haar wavelets in gray scale intensities. It uses simplistic rectangular features to calculate difference in intensities [14]. Conversely, work done by Robert Hursig involves color information of the skin [15]. A boundary threshold is applied to the shifted hue color space as a skin detector. In our face detection algorithm, the Viola-Jones algorithm and the algorithm applying color information was evaluated individually and eventually a new algorithm was developed to integrate both techniques, in order to eliminate the shortcoming of each single method.

1.2.2. Lip Localization

To look into the previous work done in the topic of lip localization, the first topic is to explore the features used in lip localization. Features used in lip segmentation can broadly be divided into two main categories, appearance based features and shape-based features [16].

Appearance-based features can generally be obtained after face detection. Several techniques are employed to find facial features, such as eyes and nostrils, based on their relative position on the face. An example of eye localization algorithm can be found in [17]. Another example of appearance-based feature includes pixel color information or intensity value. Color information provides benefits in either extracting feature (such as lips) or suppressing the undesirable ones. The appearance-based features are typically extracted from region of interest (ROI) using image transforms, such as transformation to different color space component, where pixel values of the face/mouth ROI are used. Another powerful appearance-based feature is the Haar-like feature proposed by Viola-Jones which we experiment with in this thesis.

Shape-based features, which are generally divided into geometric, parametric and statistical models of the lip shape, are extracted using techniques such as snakes [18], template models [19-20] and active shape and appearance model [21-22]. This feature assumes that most of the information is contained in contours or shape of speaker's lips. Geometric features, such as height, width, perimeter of mouth, can be readily extracted from the ROI. Alternatively, model- based features are obtained in conjunction with parametric or statistical feature extraction algorithm. The following paragraph, followed closely with the section in [16], describes the lip localization techniques further.

A) Region based Approach

Region based segmentation methods can be divided into three main categories. The Deterministic approach is based on color distribution modeling and thresholding operations. The Classification approach (supervised or non-supervised) considers lip segmentation as pixel class problems, generally between skin pixel and lip pixel. Lastly, the Statistical method is based on mouth shapes and appearance. As all these methods are region-based, accurate lip segmentation around the lip contours is not always achieved.

<u>Deterministic Approach</u>

In this method, no prior knowledge and no prior models about the statistics of the lip and skin color distribution are used. The lip segmentation is performed with a thresholding step on luminance or particular chromatic component. Automatic computation of the robust thresholds in various lighting condition is the main challenge and limitation of this method [22]. One way to determine the threshold is shown in [23]. With a sample image data set, the histogram threshold value for each chrominance component varies from the lowest value to the highest value. For each threshold value, the skin and lip segmentation errors are calculated and plotted. The plots identify the threshold for the minimum error.

<u>Classification Approach</u>

With face detection as preprocessing step, the lip segmentation can also be viewed as a classification problem with two classes: the lip class and the skin class. Using different attributes characterizing each class, the classification method used in face detection between the skin and non-skin class can also be applied for lip segmentation. The most frequently used methods include statistical techniques (Estimation theory, Bayesian decision theory and Markov random field), neural networks, support vector machine and fuzzy C-mean. They can be classified into supervised and unsupervised approach. Supervised methods utilize prior knowledge about the different classes. It involves the construction of the training data set that covers wide range of cases and environmental conditions. Then, the data set is used to train a classifier to distinguish lip pixels from skin pixels. By using predictive validation technique, [24] built a Gaussian Mixture Model with normalized RGB color value; then, Bayes classifier performs the segmentation. Viola-Jones algorithm falls into the supervised classification approach. Unsupervised methods require no training stage. As a result, no prior knowledge about the classes is taken into account. The most common unsupervised classification approach is K-mean clustering and fuzzy K-mean clustering. In terms of fuzzy clustering, [25] proposed a modified fuzzy clustering based algorithm which utilizes the lip shape information – fuzzy c-means with shape function (FCMS).

Statistical Method

Another supervised technique is the Statistical Shape Models in which the training set is compiled to describe the lip shape or appearance variation axes instead of the lip color distributions. In [20], Cootes and Taylor introduced the active shape models. A shape model is learned from a training set of annotated images. The principal component analysis (PCA) generates a small set of parameters to drive the model. By minimizing a cost function of the feature, the parameters are varied to track the lip contour. This method is known as Active Shape Model (ASM) [21]. Active Appearance Model (AAM) was introduced by Cootes and Taylor to add grey-level appearance information in the training. Paper [21] is an example of Active Shape Model and Active Appearance Model. Gaussian Mixture Models are used to train the active shape model; then, the appearance feature is used to find the parameter to fit the unknown image.

B) Contour Based Approach

The algorithms in the contour-based approach are based on deformable models. An appropriate mathematical model of the contour (a spline or a parametric curve) is chosen and placed in the image space. The model energy terms, internal term for curve geometric properties and external term from image data, are then minimized to lock the initial contour to the object of interest.

Active Contour and Snakes

Active Contour doesn’t contain any priori information about the shape of the object. There are model points that define the active contour which are modified one by one to the edges. The Active Contours, known as “snake”, was first introduced by Kass [17]. An application of snake and a gradient criterion to detect lips is found in [26].

Parametric Model

A parametric model, also known as deformable template, is similar to active contour in which an energy-minimizing algorithm is applied to deform and fit to the object contour. However, an explicit assumption about the object shape is utilized. The parametric model addresses the issue of active contour producing non-realistic lip shapes. Figure 1.4 below is the definition of the geometric lip model used in [27]. The parameters are then fitted to a skin/lip probability map by maximizing the joint probability.

$$y_1 = \frac{-h_1}{\left(w - x_{off}\right)^2}\left(|x - sy_1| - x_{off}\right)^2 + h_1$$

$$y_2 = h_2\left(\left(\frac{x - sy_2}{w}\right)^2\right)^{1+\delta^2} - h_2$$

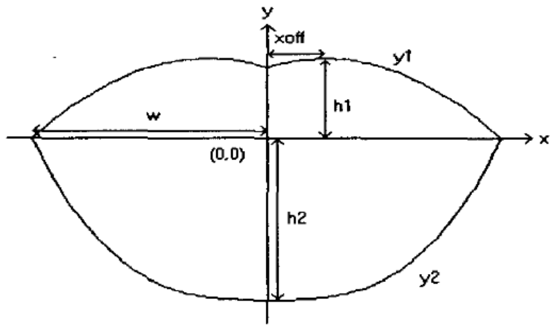


Figure 1.4: Mathematical Model and its Graphic Representation [27]

1.3. Organization of Thesis

This thesis integrates different approaches of face detection and feature extraction. Chapter 2 deals with preprocessing techniques where we compare the three illumination compensation methods which are the Yavg, minimum Y interpolation and the Average Gray. Chapter 2 also includes face detection algorithms using Bayes classifier and template matching and finally a concatenation with Viola-Jones algorithm to improve accuracy. Chapter 3 discusses the lip enhancement method used for Hybrid Gradients which was developed as a lip segmentation technique. Chapter 4 summarizes the results of face and lip detection and recommends future improvements.

2. FACE DETECTION

2.1. Illumination Compensation

Before face detection or lip segmentation, some image processing steps can be performed to enhance the desired visual features while reducing the effect of the background noise, such as the varying lighting conditions. Previous face detection algorithms were developed based on constrained images, for images taken under controlled lighting conditions where the features of the subject were clearly visible. However, it is not the case for the in-car environment. Images captured in the moving car, are constantly subjected to changes in the light from the background. Also, since the light source is from one side of the subject, invariably the other half of the subject face is in shadow. This makes detection of the face difficult as the skin color range become very large. To combat these problems, three Illumination compensation algorithms in the literature are discussed and examined using our data set. We implement the Yavg method [28], the minimum Y interpolation method [29], and the Average gray method [30].Source code is available in the Appendix. The objective of our experimentation is to determine their effectiveness for our chosen database.

Color constancy is the ability to measure color of objects without the influence of the color of the light source. In the context of face segmentation, lighting can have significant impact on the result of segmentation, especially when color information is used for detection. Several illumination compensation techniques were proposed in the literature to address the issue, such as the adaptive Gamma correction method used for color image compensation. Paper [28] introduces a light compensation technique using the average luminance value (Y in YCbCr Color Space, further information on color space will be discussed in the next section) in the image. According to the value of average luminance value (Yavg), the R and G components in RGB color space are adjusted based on the following equations (2.1). We apply this method to one of the images extracted from our database. The image contains a prominent effect of light which prevents the face detection algorithm to segment the face. The result is shown in Figure 2.1.

$$R_{new} = (R_{old})^{\tau} \qquad G_{new} = (G_{old})^{\tau} \qquad \text{in which, } \tau = \begin{cases} 1.4 & \text{Yavg} < 64 \\ 0.6 & \text{Yavg} > 192 \\ 1 & \text{otherwise} \end{cases} \qquad (2\text{-}1)$$

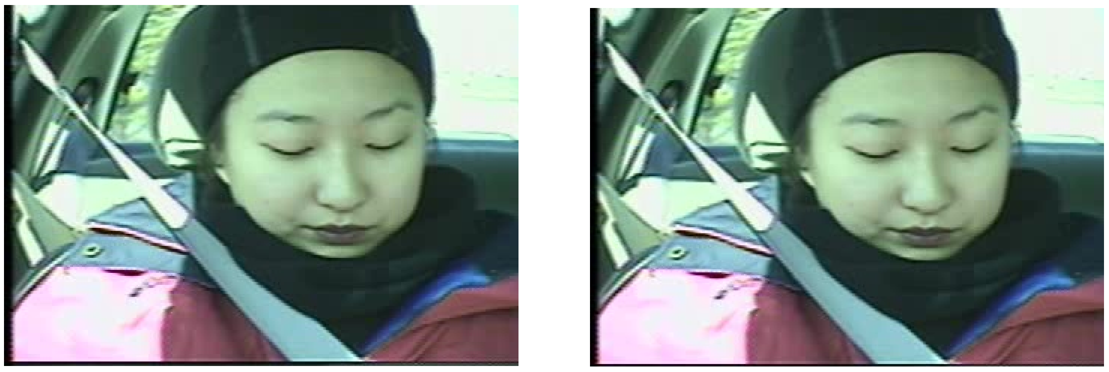

(a) (b)

Figure 2.1: (a) original image (b) illumination compensated image by Yavg

As can be seen, this method does not produce any difference in the result. The threshold is selected on the basis of only the average Y value of the image. The underexposed or overexposed images in the test data set which required illumination compensation, also had Yavg values which fell inside the range of 64 to 192. Therefore, this method is not suitable for our dataset.

The central idea behind the minimum Y interpolation method is to use minimum Y values of subsets of the image and create a new image. This method tries to equalize the Y component throughout the image. Figure 2.2 below provides a flow chart for the method. Masking involves dividing the entire image (MxN) into smaller 20 x 20 pixel sub images. The minimum Y value of each sub image was calculated. Using all the minimum values of the sub image, a smaller version of the image (M/20 x N/20) was constructed. Bilinear interpolation is then used to expand the image to its original size. Next, image subtraction is performed to find the difference image between the Y component of the original image and the expanded interpolated image. Finally, the subtracted image is histogram equalized, which forms the new Y component of original image [29].

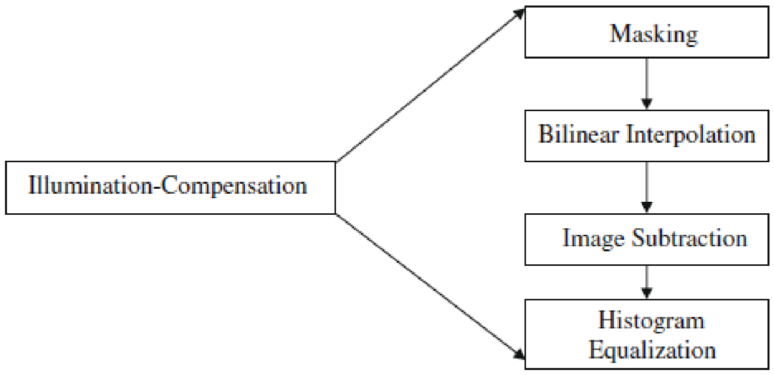
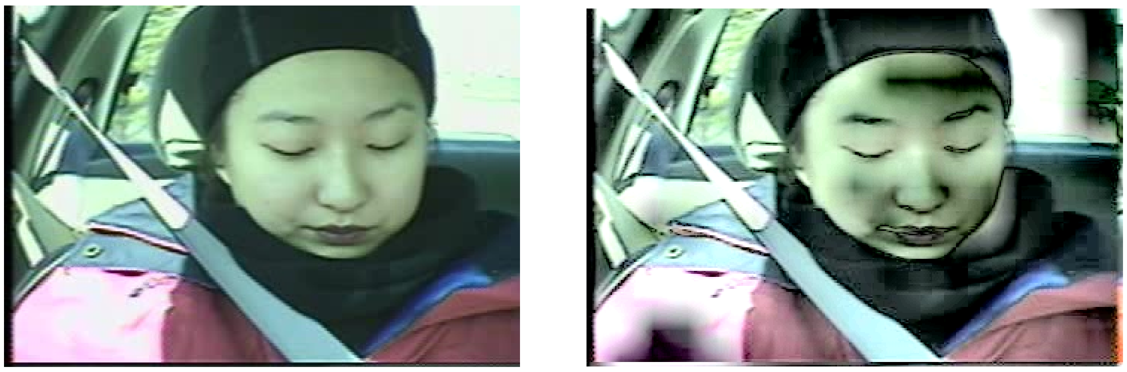


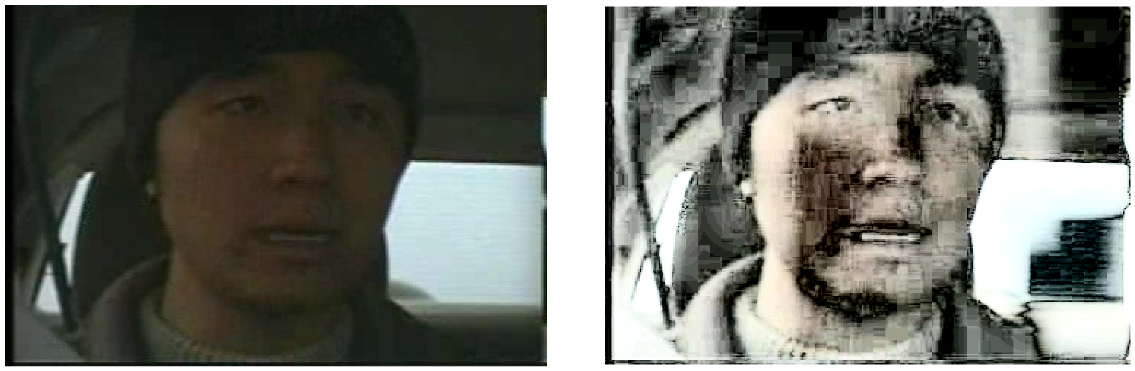
Figure 2.2: Illumination compensation method [29]

The minimum Y interpolation method is applied to our dataset. Figure 2.3 shows the results for one sample image. It can be seen that this illumination compensation technique significantly degenerates the image, and do not work well for our dataset as a large part of the skin region is lost. In certain cases though, it is observed that very dark pictures are lightened to a great extend as shown in Figure 2.4. Although still not beneficial for our face detection as the image is still too noisy, this illumination compensation technique seems to have potential to work with underexposed images.



(a) **(b)**

Figure 2.3: (a) Original Image (b) Illumination compensated image by minimum Y interpolation method



(a) **(b)**

Figure 2.4: Underexposed Original Image (b) Lightened Illumination compensated image by minimum Y interpolation method

Our last examined method deals with images in the RGB color space. The principle of this method assumes that the average reflectance of the surface in the image is grey. Any shift from the gray of the measured average corresponds to the color of the illuminant. Therefore, we aim to modify the value of the components of all pixels to achieve the average grey [30].

$$R_{mean} = Average\ Red\ value\ of\ all\ the\ pixels\ in\ the\ image$$

$$G_{mean} = Average\ Green\ value\ of\ all\ the\ pixels\ in\ the\ image$$

$$B_{mean} = Average\ Blue\ value\ of\ all\ the\ pixels\ in\ the\ image$$

$$AverageGray = \frac{R_{mean} + G_{mean} + B_{mean}}{3}$$

$$R_{scalevalue} = \frac{AverageGray}{R_{mean}};$$

$$G_{scalevalue} = \frac{AverageGray}{G_{mean}}; \qquad (2\text{-}2)$$

$$B_{scalevalue} = \frac{AverageGray}{B_{mean}}$$

$$NewR = R_{scalevalue} * R; \quad NewG = G_{scalevalue} * G; \quad NewB = B_{scalevalue} * B$$

Figure 2.5 (a) and (b) shows the effects of illumination correction on of the test images. It is observed that the original image has a paler yellow. However, after the illumination correction algorithm, the lighting effect on the face is eliminated. We verify the effectiveness of this compensation method by applying the skin detection technique on the illuminated corrected image. The difference between the detected results is shown in the Figure 2.5 (c) and (d). It is apparent that without illumination compensation face detection would have completely failed in this case.

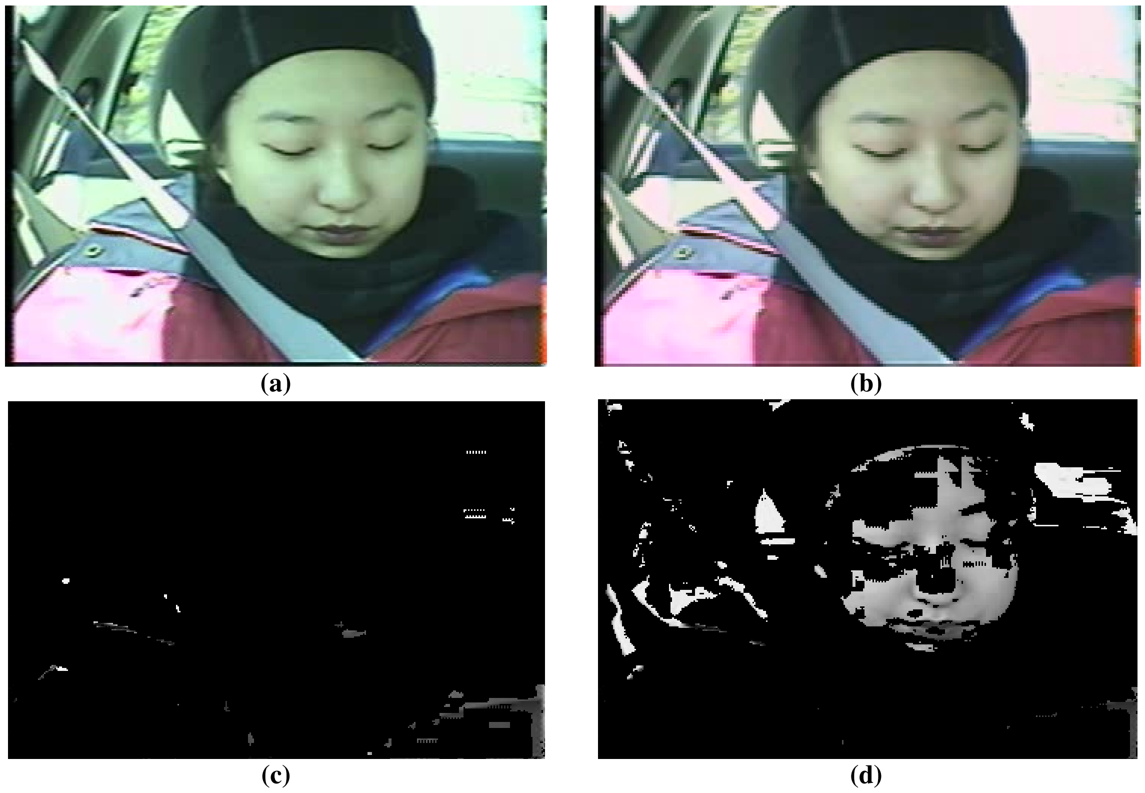


Figure 2.5: (a) Original Image (b) Illumination compensated image by Average Gray method (c) Application of Bayes classifier on (a) (d) Application of Bayes classifier on (b)

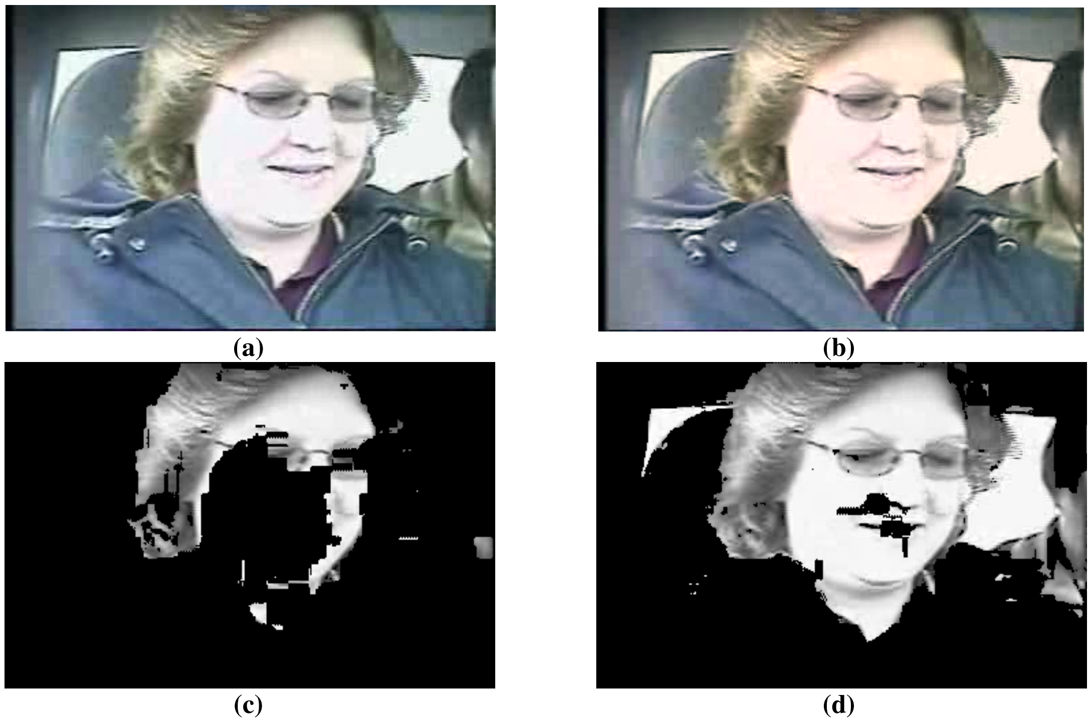


Figure 2.6: (a) Original Over-exposed Image (b) Illumination compensated image by Average Gray method (c) Application of Bayes classifier on (a) (d) Application of Bayes classifier on (b)

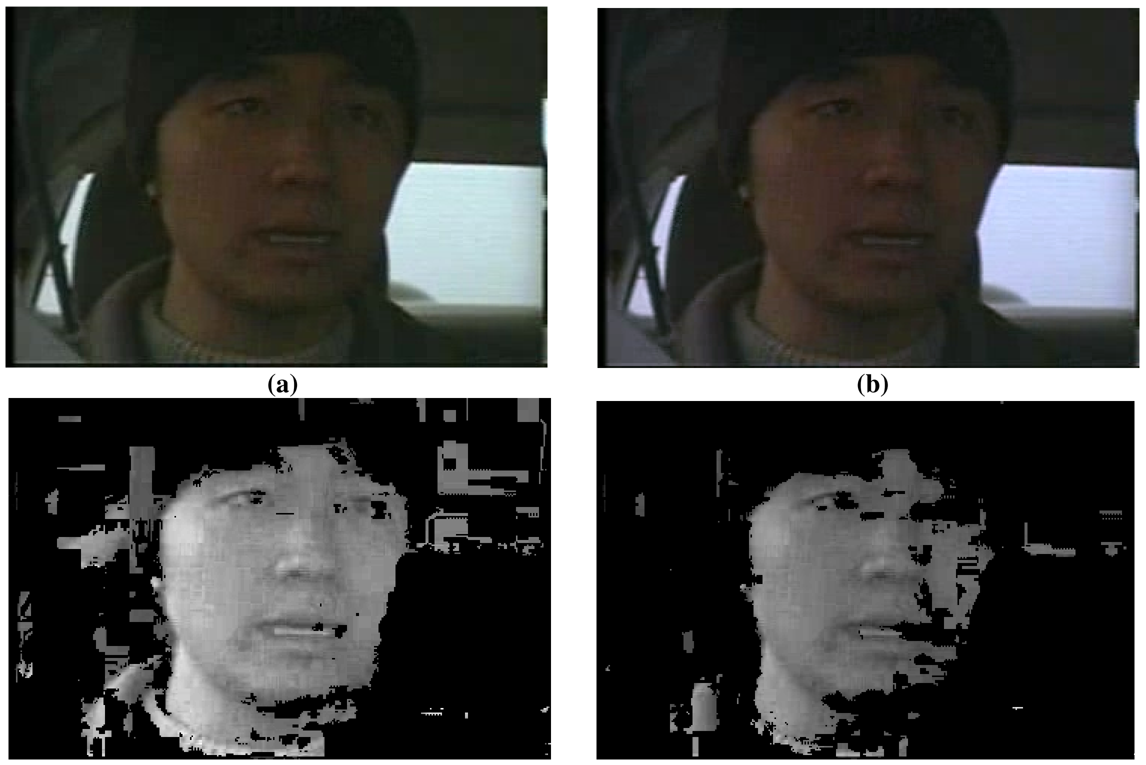


Figure 2.7: (a) Original Under-exposed Image (b) Illumination compensated image by Average Gray method (c) Application of Bayes classifier on (a) (d) Application of Bayes classifier on (b)

In Figure 2.6 and Figure 2.7 we show an example of over- and under-exposed images. In both images, it can be noticed that the face region comes closer to skin color range. The over-exposed image, where a large area of the face was not classified as skin in the original image, was remedied by illumination compensation with the trade off of extra noise. Conversely, in the under-exposed image, fewer background pixels were wrongly classified as skin. Thus it can be concluded from our experimentation, the last method (using the concept of average gray) gave the most satisfactory results for our data and will be used in future algorithms. Therefore, from this section onward all images under consideration have been illumination compensated by the average gray method.

2.2. Face Detection

Lips comprise a very small part which is volatile and constantly in motion for a speaker. Therefore, to limit the area in which lips need to be searched, it is a popular practice to perform face detection first. It then becomes important to have a robust face detection algorithm in order to correctly locate the lips. Face Detection is defined as 'Given an arbitrary image, the goal of face detection is to determine whether or not there are any faces in the image, and if present, return the image location and extent of each face [7]. There are several challenges associated with face detection, some of which were faced during the process of this thesis; namely, pose of the face which indicates if the face to be detected is frontal, profile, tilted at an angle etc. Another hindrance is when part of the face is occluded as for example through beards, sunglasses or overlapped faces which does not leave the entire face exposed. Another important criterion is the imaging condition. In our data set the lighting and mechanical vibrations included by the moving car add significant noise to the images.

In this thesis, we first analyze the color spaces in order to select one with maximum potential. Using the color space selected we train a Bayes classifier to detect skin pixels in the image. The images which could not be classified as a face is further passed through template matching.

We independently summarize and evaluate the implementation of the Viola-Jones algorithm and compare the results. The final detection algorithm concatenates all three techniques to give better accuracy.

In order to develop a robust face detection algorithm, there are several components in the algorithm. The first step is skin classification, which helps to locate the general face location. Section 2.2.1 discusses the color spaces which can be used for skin detection. Section 2.2.2 selects a color space and trains a Bayes classifier for the purpose of skin classification. Then, Section 2.2.3 details face localization methods such as morphological operations and template matching. The purpose of morphological operations and template matching is to extract the exact face location. Section 2.2.5 records the results of the detection algorithm. In Section 2.2.5, we introduce Viola-Jones and compare results with the developed face detection algorithm. Lastly, in Section 2.2.6, our final detection algorithm is developed by combining the two algorithms.

2.2.1. Color Space Analysis

Five hundred images from AVICAR database were selected as training data where care was taken such that equal numbers of images were selected to represent different skin color. Each image contained 4 sub-images representing the four camera angles placed in the car as shown in Figure 2.8. The skin region for each image was manually selected and stored as a mask as displayed in Figure 2.9.

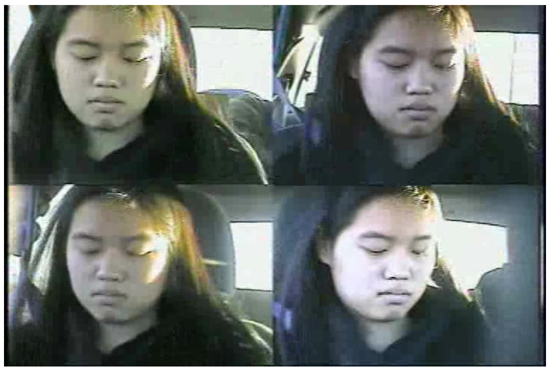


Figure 2.8: Original Image of AVICAR database

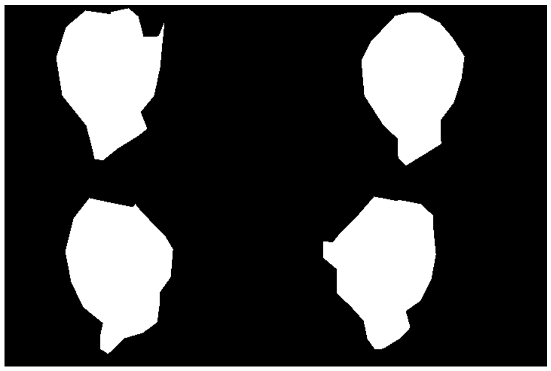


Figure 2.9: Mask of Image in Figure 2.8

The original image and its corresponding mask were used to determine the color space best representing the face. Ideally, the skin and the non-skin cluster should be separate with a clear threshold between them and each cluster individually should have a very small variance. That is, we require a small intra cluster variance and a large inter cluster variance between skin and non-skin pixels in their feature space.

In the following section we plot the histograms of skin pixels vs. the non-skin pixels in several color spaces to evaluate which color space can be used as a classifier. There are two conditions we require from the histograms of a good color space. It should be possible to draw a boundary between the skin and the non-skin cluster and both clusters should have a low intra-class variance. As it was observed that illumination compensation on images improved the classification result, it may be noteworthy to compare if histograms of illumination compensated images differ from the original image histogram and result in better classifiers. Therefore, for each color space we plot histograms of original and illumination compensated images.

RGB Color Space

Commonly viewed images are in the RGB color space which is represented as a 3D cube (Figure 2.10). The origin forms the black color, which represents the absence of all color. The three axes of the cube form the red, green and blue color components. The maximum value of all components combined, forms a pure white. The histogram of skin vs. that of background pixels in the RGB color space is shown in Figure 2.11.

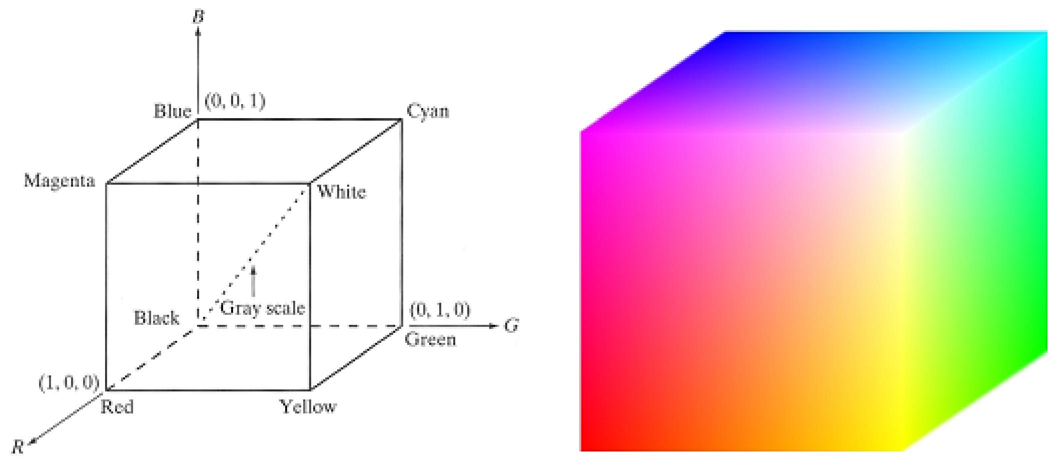


Figure 2.10: The RGB Color Cube [31]

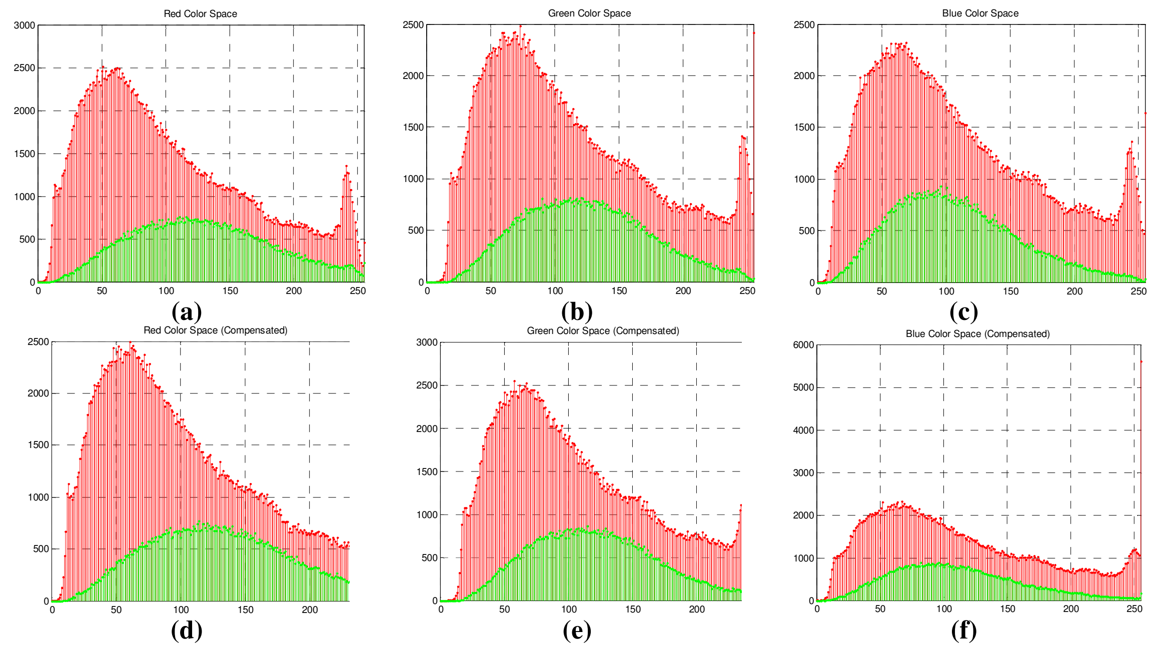


Figure 2.11: Skin v/s non skin histogram (a) Red component (b) Green component (c) Blue component (d) Illumination compensated Red component (e) Illumination compensated Green component (f)Illumination compensated Blue component. In the figure Red signifies non skin pixels and green signifies skin pixels

It is obvious that the RGB color space does not satisfy either of those requirements. There is no clear boundary that can be found between the skin and the non-skin pixels. Therefore, RGB color space is not suitable for the purpose of face detection. Applying illumination compensation on the training images before plotting the histograms does not make a difference in the eligibility of the RGB color space to train a classifier.

<u>HSV Color Space</u>

Although the color spaces, HSV (Hue Saturation Value), HSI (Hue Saturation Intensity) and HSL (Hue Saturation Lightness) are transformed from the RGB color cube differently[32], the chrominance component of all three color spaces (Hue) describes similar information while it is separated from luminance channels (SV, SI and SL channels). The transformation equation is shown below:

$$H = arccos \frac{\frac{1}{2}\left((R-G)+(R-B)\right)}{\sqrt{((R-G)^2+(R-B)(G-B))}} \tag{2-3}$$

Hue component lies on the top circle of the HVS cone or the 'color wheel' shown in Figure 2.10. Therefore, in angular measurement Hue ranges from 0° to 360° which means it experiences the wrap around effect where the two extreme angles, 0° and 360°, are of the same hue value.

This effect is noticed during evaluation of the hue component for skin. As skin is in the range of red, the hue values are either extremely high (>0.8) or extremely low (<0.1), therefore we introduce shifted hue, which shifts the cluster by 0.2 units to the right. The resulting histogram is for the shifted Hue component.

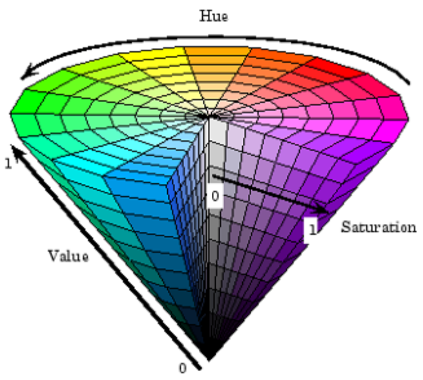


Figure 2.12: The HSV Color Model [33]

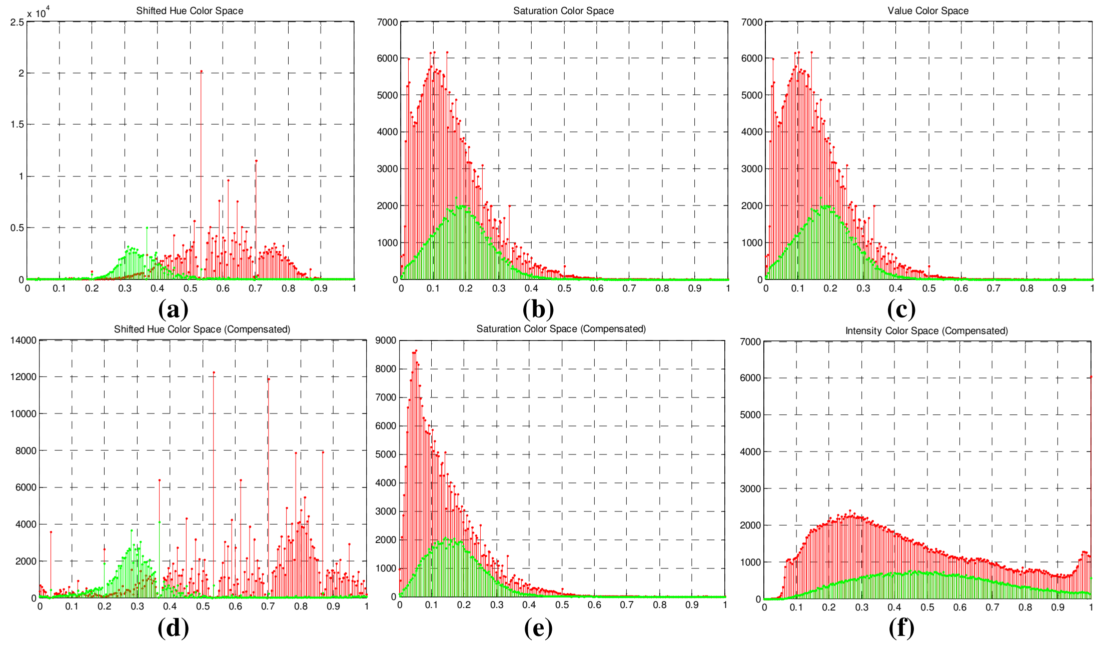


Figure 2.13: Skin v/s non skin histogram (a) shifted-Hue component (b) Saturation component (c) Intensity component (d) Illumination compensated shifted-Hue component (e) Illumination compensated Saturation component (f) Illumination compensated Intensity component. In the figure Red signifies non skin pixels and green signifies skin pixels

In the Saturation and Intensity component the skin and non-skin pixels completely overlap each other. No definite boundary can be drawn to distinguish between the two. . It can be confirmed by the histograms in Figure 2.13 that S and I channel should not be considered, whereas for the shifted-Hue component a definite boundary for the two classes can be constructed as shifted-hue is concentrated between 0 to 0.25.

It is noticed in Figure 2.13 (d) that after compensation by the Average Gray method the overlap between the skin and non-skin cluster is reduced in hue space. At the same time it also seems that the variance on the skin cluster is reduced. Training a classifier with the illumination compensated shifted-hue space might lead to better results than using the original images.

<u>Normalized RGB</u>

Figure 2.14 represents the normalized red, green and blue components

$$r = \frac{R}{R+G+B}; g = \frac{G}{R+G+B}; b = \frac{B}{R+G+B} \tag{2-4}$$

As *r+g+b=1,* the third component is redundant in the mathematical sense and can be eliminated. Also, the dependence of brightness seems to be reduced in the normalized r and g components [34-35]. Also it is observed from Figure 2.14 that although the skin and non-skin pixels overlap, they seem to have a very small intra class variance and do not spread out too much. All three components of this color space have the potential in skin classification.

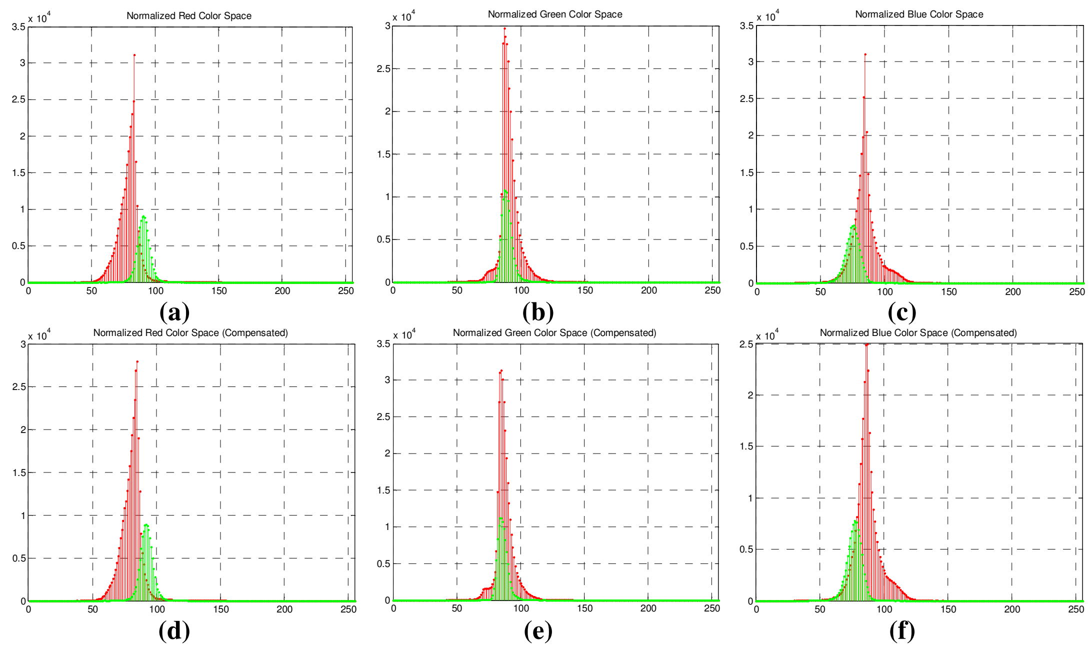


Figure 2.14: Skin v/s non skin histogram (a) norm red component (b) norm green component (c) norm blue component (d) Illumination compensated norm red component (e) Illumination compensated norm green component (f) Illumination compensated norm blue component. In the figure Red signifies non skin pixels and green signifies skin pixels

Histograms of images with and without illumination compensation seemed to have negligible difference.

YCbCr Color Space

In Figure 2.15, YCrCb can be transformed from the RGB color space according to the following equation.

$$\begin{aligned} Y &= 0.299 * R + 0.587 * G + 0.114 * B \\ Cr &= R - Y \\ Cb &= B - Y \end{aligned} \tag{2-5}$$

Y represents the luminance component whereas Cr and Cb represent chrominance. Due to the separation of luminance and chrominance components, Cb and Cr are independent of illumination effect. Cr and Cb have a small variation and the two clusters don’t completely overlap and can be considered for skin classification. Y component on the other hand is rejected as both clusters completely overlap.

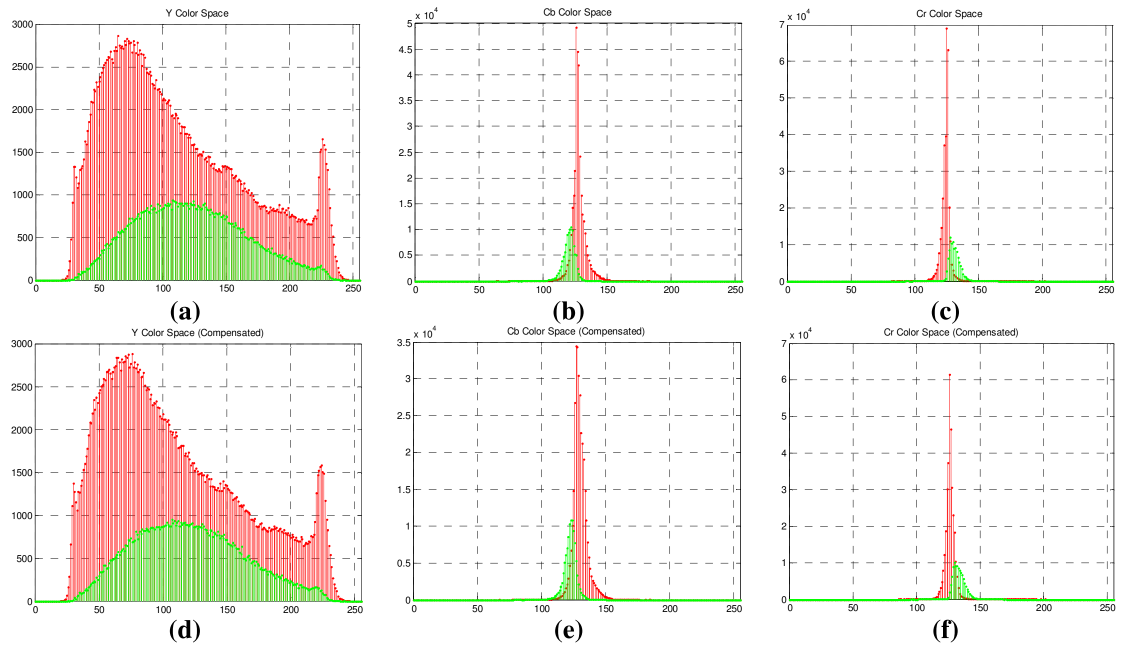


Figure 2.15: Skin v/s non skin histogram (a) Y component (b) Cb component (c) Cr component (d) Illumination compensated Y component (e) Illumination compensated Cb component (f) Illumination compensated Cr component. In the figure Red signifies non skin pixels and green signifies skin pixels

The Cb component of the YCbCr color space shows improvement due to illumination compensation. The skin cluster is ‘taller and skinnier’ which means the class variance becomes smaller.

Other Color Conversions

We also implement popular color spaces suggested in paper [16].

$$E = 0.5\,(R - G)$$

$$RG\,ratio = R/G \qquad (2\text{-}6)$$

$$RGB\,combination = \frac{R}{G} + \frac{G}{B} + \frac{B}{R}$$

These color components did not have good results for our dataset as defining a boundary between the classes was not possible. Histograms of the illuminated compensated images seemed to have similar results.

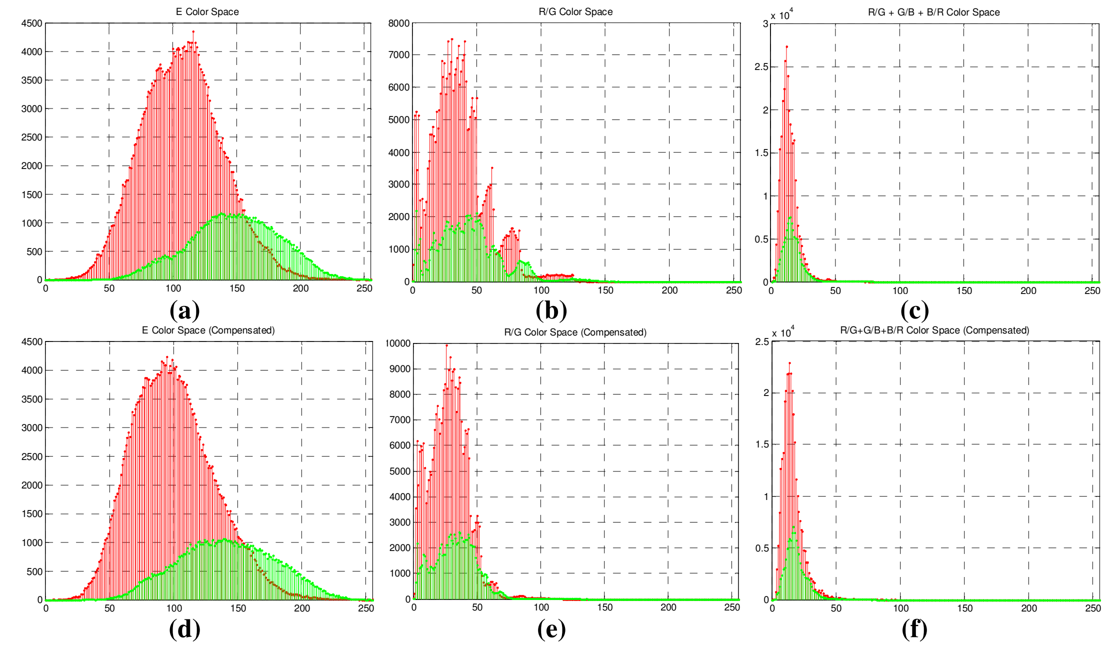


Figure 2.16: Skin v/s non skin histogram (a) E = 0.5(R-G) (b) R/G component (c) R/G + G/B + R/B (d) Illumination compensated E = 0.5(R-G) (e) Illumination compensated R/G component (f) Illumination compensated R/G + G/B + R/B. In the figure Red signifies non skin pixels and green signifies skin pixels

Another popular color space is CIE-Luv[35]. CIE-Luv is the modification of the CIE-XYZ color space which also aimed to separate the brightness and chromaticity of color. The transformation from RGB to CIE-Luv is a set of complicated equations and an inbuilt Matlab function was used for the conversion. The results of skin v/s non-skin class for the components of the CIE-Luv color space is shown in Figure 2.17. Although it did not yield satisfactory results for our dataset, based on the histograms it seems it could prove to have interesting results. Illumination compensation also did not improve the results. CIE-Luv color space is also computationally more expensive compared to other color spaces due to its transformation.

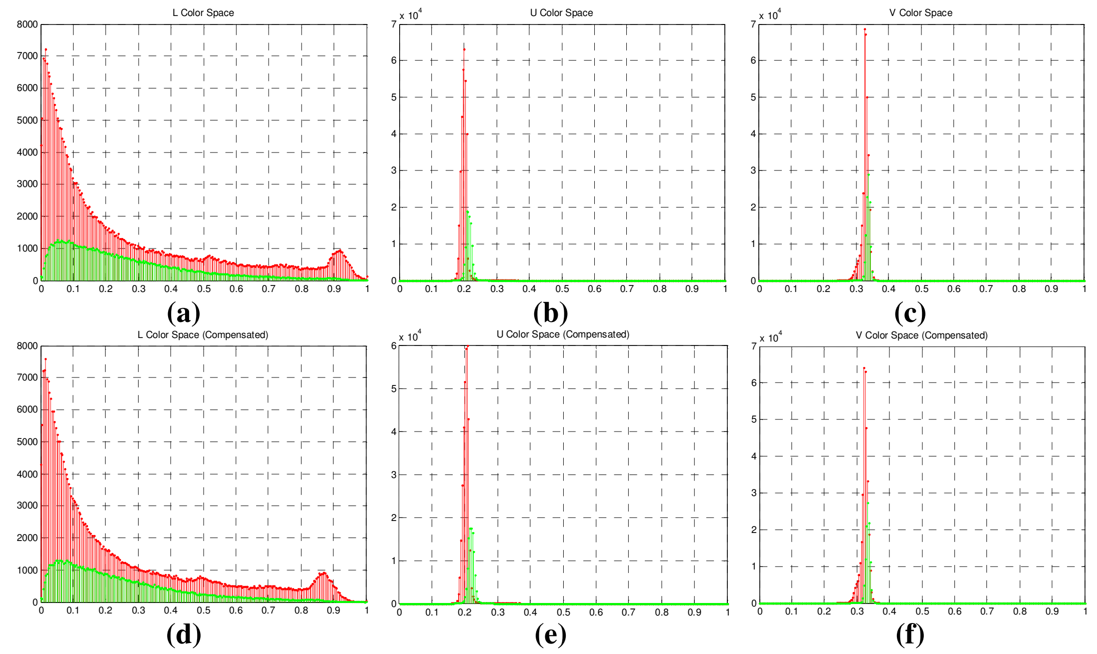
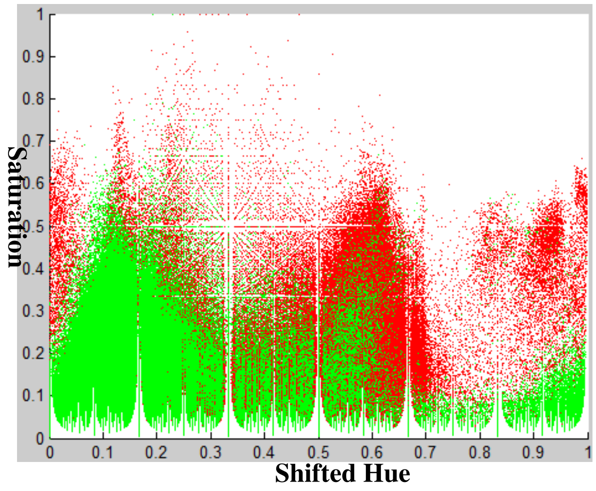
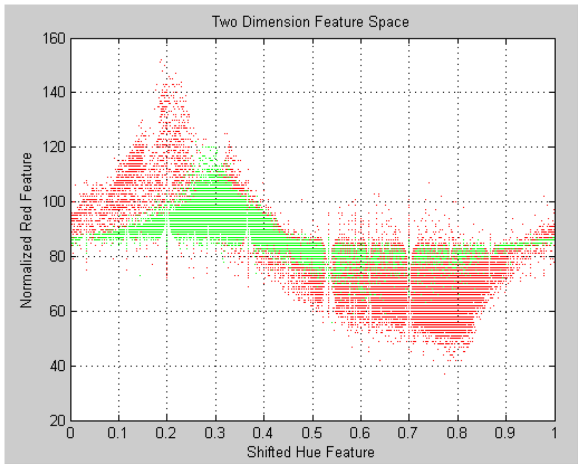


Figure 2.17: LUV skin v/s non skin histogram (a) L component (b) u component (c) v component (d) Illumination compensated L component (e) Illumination compensated u component (f) Illumination compensated v component. In the figure Red signifies non skin pixels and green signifies skin pixels

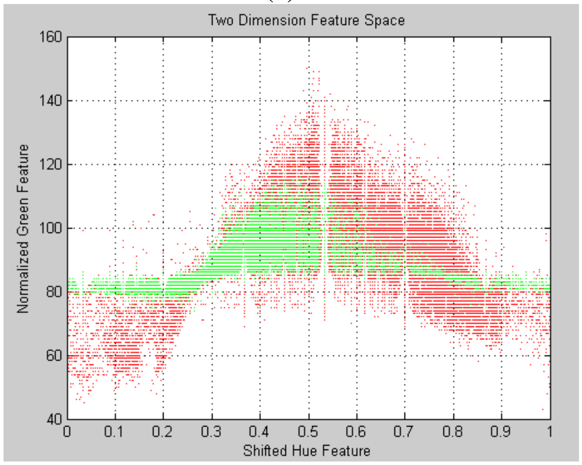
Through experimentation, we observed that 1D color spaces did not yield satisfactory results. Using only one color component as a classifier in our data set left too much noise detected as skin. Therefore, we tried combinations of color components in 2D as the classifier conditions get more restricted and the noise can be partially eliminated. Figure 2.18 depicts a few examples of the 2D combinations implemented. Of all the combinations tried Shifted Hue vs. normalized green gave the best results for a Bayes classifier.



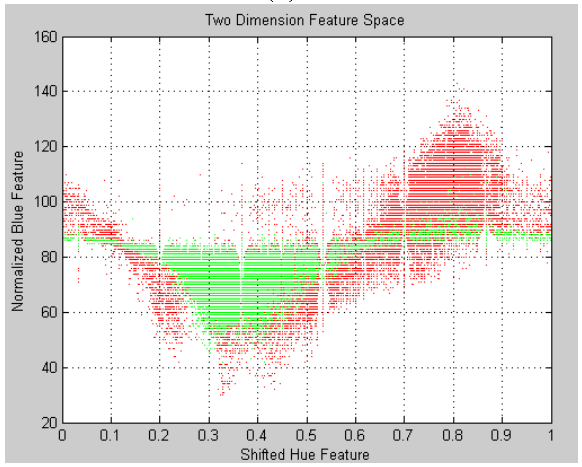
(a)



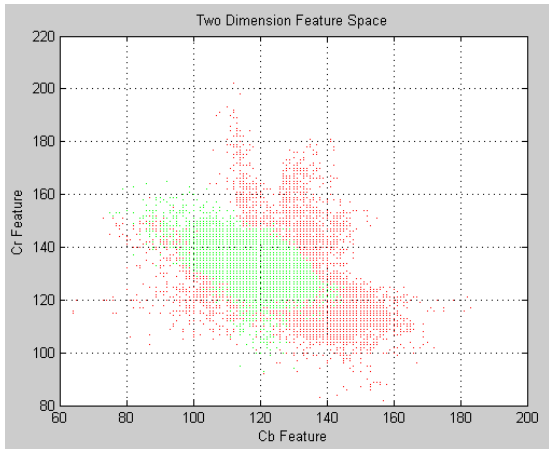
(b)



(c)



(d)



(e)

Figure 2.18: (a) 2D histogram of Shifted Hue on the x-axis and Saturation on the y axis. (b) 2D histogram of Shifted Hue on the x-axis and normalized red on the y axis. (c) 2D histogram of Shifted Hue on the x-axis and normalized green on the y axis. (d) 2D histogram of Shifted Hue on the x-axis and normalized blue on the y axis. (e) 2D histogram of Cb on the x-axis and Cr on the y axis. In the figure Red signifies non skin pixels and green signifies skin pixels

Once the color component of the classifier is decided, we define and train the classifier. The next section describes the principle of the classifier, the training procedure and the parameters obtained.

2.2.2. Skin Classification

The Bayesian Classification theory is a fundamental statistical approach in pattern classification [36]. The decision is based on probabilistic terms assuming all relevant probabilities were known. In the case of skin detection two cases exist for pixel classification. The pixel can be classified as either a skin pixel or a background/ non-skin pixel.

Let Skin be the class that the pixel belongs to skin and non-skin the class that the pixel belongs to non-skin. As we have only two possibilities $P(skin) + P(nonskin) = 1$ where P(skin) and P(non-skin) are known as prior probabilities, representing the probability that the pixel is skin and the probability that the pixel is non-skin. . We use the training data collected for color space analysis to calculate these prior probabilities. The probability of skin pixels is calculated by summing the mask pixels in all 500 images and dividing it by the number of all the pixels present in to 500 images.

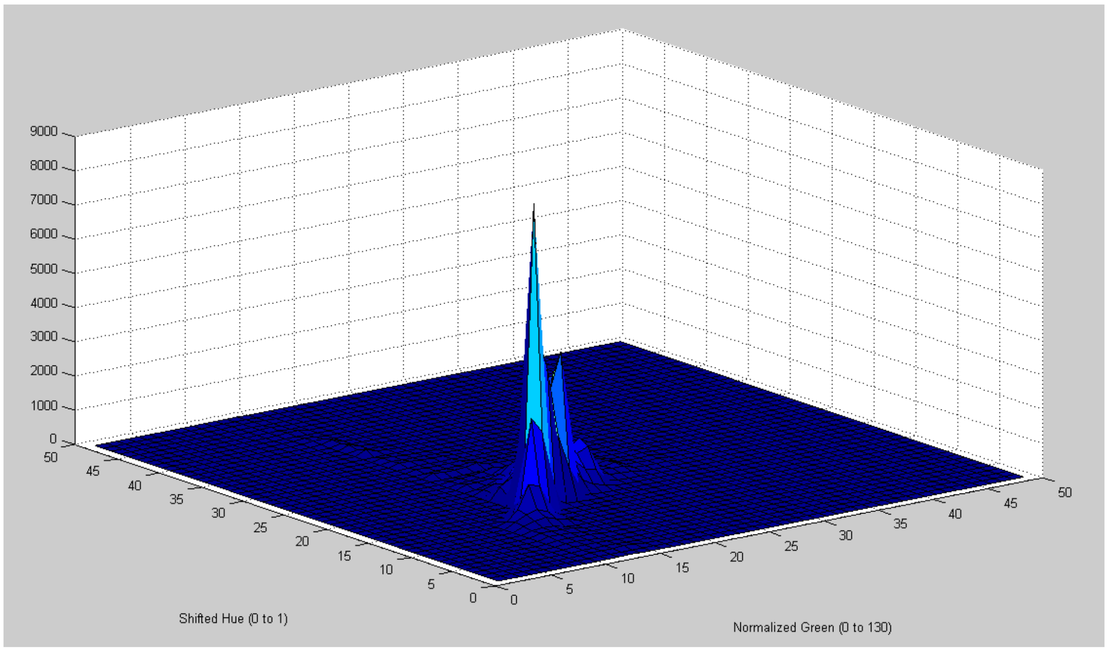


Figure 2.19: Shifted-Hue and normalized green scatter plot projected in 3D

As mentioned in the previous section, the approach here is one based on 2-D cluster (shifted-hue vs. normalized green) of the training data. The feature space used here is hence a 2D feature space, containing a feature vector **c** = [c_1 c_2], in which c_1 is the pixel's shifted-hue value whereas c_2 is the pixel's normalized green value. We assume the 2D cluster of skin pixels to be Gaussian based on the general shape observed in Figure 2.19.

According to Bayes rule, a pixel is classified as skin *if* $P(\boldsymbol{c}/skin)\ P(skin) >$ $P(\boldsymbol{c}/non\text{-}skin)\ P(nonskin)$ *otherwise the pixels is classified as non-skin.*

The conditional probabilities $P(\boldsymbol{c}/skin)$ and $P(\boldsymbol{c}/non\text{-}skin)$ also known as likelihood of skin/non-skin indicate that. If under the condition of equal prior probabilities, conditional probabilities become the deciding factor.

In the implementation, the Bayes Classifier (minimum-error-rate) can be represented by the discriminant function $g_i(\boldsymbol{c})$, i = skin or non-skin, where feature vector **c** gets assigned to class skin if $g_{skin}(\boldsymbol{c}) > g_{non-skin}(\boldsymbol{c})$

where

$$g_{skin}(\boldsymbol{c}) = \ln P\left(\boldsymbol{c}/_{skin}\right) + \ln P(skin) \tag{2-7}$$

The $P\left(\boldsymbol{c}/_{skin}\right)$ can be modeled as multivariate normal – the basis for the normal distribution is due to its analytical tractability and as an appropriate model for the continuous-valued, randomly corrupted case. The $P\left(\boldsymbol{c}/_{skin}\right)$ is modeled according to the general multivariate normal densities in d dimensions as shown below, in which **c** is the dx1 feature vector, μ is the dx1 mean vector whereas Σ is the d x d covariance matrix. |Σ| and Σ^{-1} are its determinant and inverse, respectively. The superscript t represents the transpose.

$$p(\boldsymbol{c}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\,[-\frac{1}{2}(\boldsymbol{c}-\boldsymbol{\mu})^t \Sigma^{-1}(\boldsymbol{c}-\boldsymbol{\mu})]$$

in which (2-8)

$$\boldsymbol{\mu} = \int \boldsymbol{c}p(\boldsymbol{c})\,dc \quad and \quad \Sigma = \int (\boldsymbol{c}-\boldsymbol{\mu})(\boldsymbol{c}-\boldsymbol{\mu})^t\,p(\boldsymbol{c})dc$$

The discriminate function can be modified to

$$g_{skin}(\boldsymbol{c}) = -\frac{1}{2}(\boldsymbol{c}-\boldsymbol{\mu_{skin}})^t \Sigma_{skin}^{-1}(\boldsymbol{c}-\boldsymbol{\mu_{skin}}) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_{skin}| + \ln P(skin) \tag{2-9}$$

In the general multivariate normal case, the covariance matrices for each class are different and the term $\frac{d}{2}\ln 2\pi$ can be dropped. The resulting equation is

$$g_{skin}(c) = c^t W_{skin} c + w_{skin}^t c + \omega_{skin\,0}$$

where,

$$W_{skin} = -\frac{1}{2}\Sigma_{skin}^{-1} \tag{2-10}$$

$$w_{skin} = \Sigma_{skin}^{-1}\mu_{skin}$$

$$\omega_{skin\,0} = -\frac{1}{2}\mu_{skin}^t \Sigma_{skin}^{-1}\mu_{skin} - \frac{1}{2}\ln|\Sigma_{skin}| + \ln P(skin)$$

To calculate the parameters required to train the classifier, we use the 500 images used previously for the color space analysis training. We first calculate the mean $\boldsymbol{\mu}$ and the covariance matrix Σ of all the pixels values (normalized green and shifted hue) for the skin and non-skin class. This was achieved by using inbuilt Matlab functions. The three key values $\mathbf{W}_{\text{skin}}$, $\mathbf{w}_{\text{skin}}$ and $\mathbf{w}_{\text{skino}}$ for $g_{skin}(\boldsymbol{c})$ and $g_{nonskin}(\boldsymbol{c})$ can be obtained from the mean vector and covariance matrix according to equation (2-10). The probability P(skin) is calculated based on the number of skin pixel in the training set over the total number of pixels used. The values of the parameters obtained were:

$$\boldsymbol{W}_{skin} = \begin{bmatrix} -33.6276 & -0.2695 \\ -0.2695 & -0.0119 \end{bmatrix}$$

$$\boldsymbol{W}_{non-skin} = \begin{bmatrix} -26.8821 & 0.0581 \\ 0.0581 & -0.0273 \end{bmatrix}$$

$$\boldsymbol{w}_{skin} = \begin{bmatrix} 77.6957 \\ 2.3478 \end{bmatrix}$$

$$\boldsymbol{w}_{non-skin} = \begin{bmatrix} -1.9119 \\ 4.8943 \end{bmatrix} \tag{2-11}$$

$$\omega_{skin\,0} = -123.8033$$

$$\omega_{non-skin\,0} = -215.5282$$

$$P(skin) = 0.242734$$

$$P(non-skin) = 1 - P(skin) = 0.757266$$

With those parameters, the skin and non-skin discriminate function values can be calculated for each pixel under test (a feature vector c). The feature vector **c** gets assigned to the skin class if $g_{skin}(c) > g_{non-skin}(c)$; otherwise, it gets assigned to the non-skin class.

The input image (I) is first read and passed through the illumination compensation algorithm discussed in the previous section. The output (NewI as shown in Figure

2.20(a)) is passed through the Bayes classifier where each pixel is either classified as skin or as background pixel (SkinImg, Figure 2.20(b)).

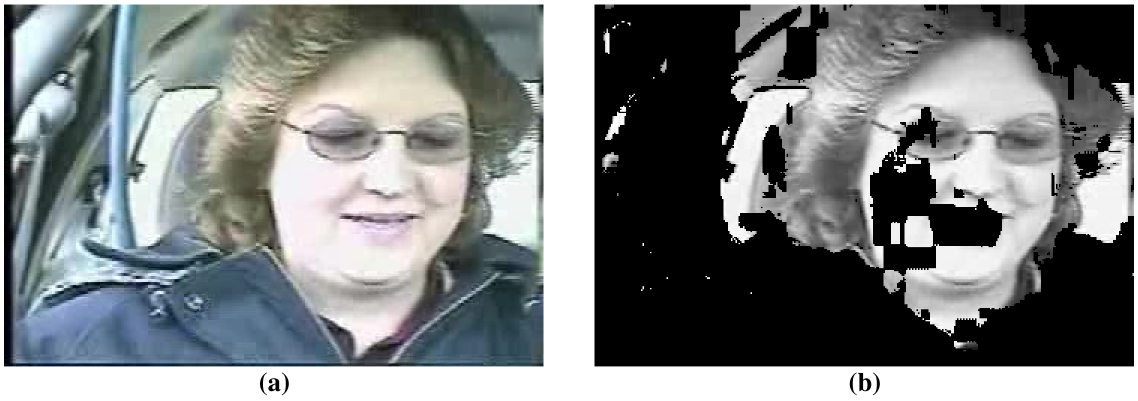


Figure 2.20: a) Illumination Compensated image (NewI) and b) Skin detected image after Bayes classifier (SkinImg)

2.2.3. Face Localization

Using the 2D color space and the Bayes classifier we obtain the pixels classified as skin. In most cases, not all pixels belonging to the face are correctly classified. Similarly, background pixels are mis-classified as skin.

The next stage of face detection involves selecting the pixels which comprise the face and eliminating the rest. Here, we first perform blob analysis, in which we consider clusters of pixels joined together and determine if they match face dimension. We continue with morphological operations which are responsible for completing the holes that exist within the face or separating the face cluster from the surrounding background noise. Lastly, for the images which are extremely noisy and the face and cluster cannot be separated, we perform face template matching.

A) Blob Analysis and Golden Ratio

On the skin detected images, we perform the 'Blob analysis' function in Matlab. A blob is referred to as a cluster of all the pixels connected to each other in a binary image. Using the Blob analysis function, Matlab draws a smallest rectangle possible which encompasses all the connected pixels.

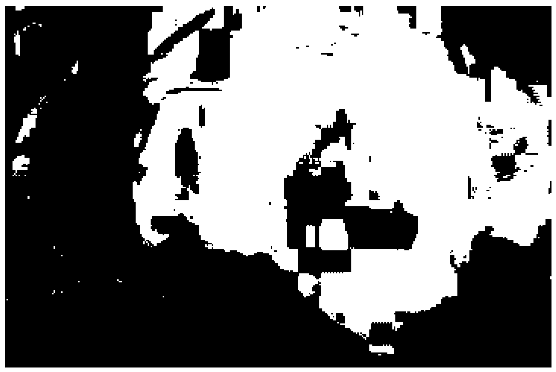


Figure 2.21: Binary image of SkinImg used to perform blob analysis

We need to distinguish if the blob or the connected components form a face or not. The key assumption in our dataset is that each image has one face and in fact all images have a face. Therefore, we limit all our discussions to the largest connected components or the cluster with the maximum number of pixels in the bounding box.

It was observed, that in certain cases the largest bounding box consists of a perfect face. In order to isolate these regions we use dimension of the faces as a restriction. We restrict the ratio of the height to the width of the detected bounding box to be close to the golden ratio (1.61803). To include the variations of the face sizes and also to consider the fact that the neck region is very likely to be part of the connected components if exposed, we introduce a tolerance. Through experimentation we set the threshold as

$$1.2 < {}^{Height}/_{Width} < 1.78$$

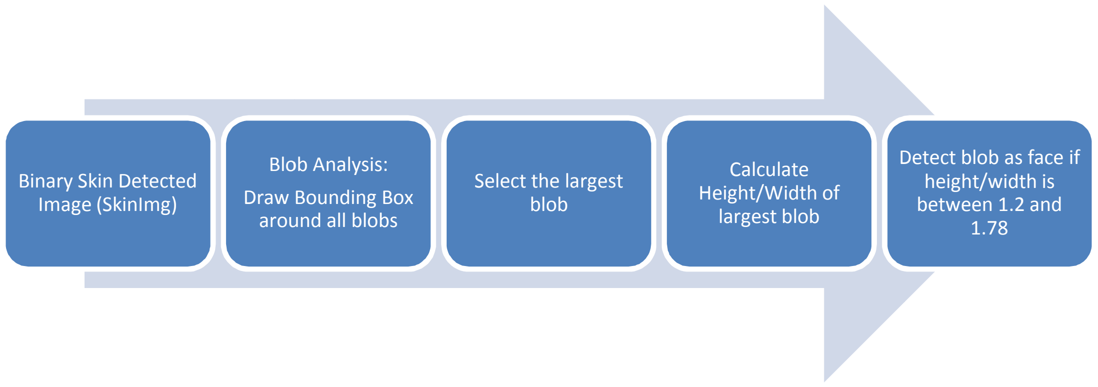

Binary Skin Detected Image (SkinImg)
Blob Analysis:
Draw Bounding Box around all blobs
Select the largest blob
Calculate Height/Width of largest blob
Detect blob as face if height/width is between 1.2 and 1.78

Figure 2.22: Flowchart representing the blob analysis and Golden Ratio

Figure 2.22 summarizes the first stage of the face localization algorithm. The input to the system is an illumination compensated image, which we pass through the Bayes classifier for skin detection. All pixels detected as skin as set as binary 1 (white), whereas all non-skin pixels are set as binary 0 (black). We then draw a bounding box around all the clusters of pixels and select the largest cluster. The height to width ratio of the largest bounding box is measured and if it falls in the range of the threshold equation that cluster is classified as a face. Figure 2.23(b) represents a correctly classified cluster as face.

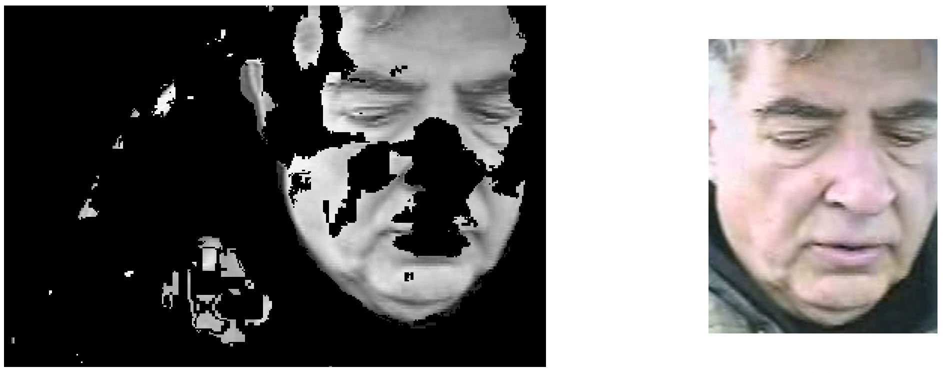


Figure 2.23: a) Largest blob of skin detected image b) Cropped image of the largest blob

For the purpose of experimentation of our face detection algorithm we create a set of 181 test images from the AVICAR database. Each image consists of 1 face. The test set contains subjects of all ethnicities present in the AVICAR database. We have also included underexposed and overexposed images to test our algorithm on severe lightening conditions.

We run the first stage of the face detection algorithm as described by Figure 2.22 on the 181 test images and tabulate the results in Table 2-1.

Table 2-1: Results of implementing the Bayes Classifier on 181 test images

Face Detection Technique	Color Space	Total Faces	Faces that pass the Golden Ratio	Faces that miss the Golden Ratio	Percentage Detected	Percentage Failed
Bayes Classifier + Golden Ratio	Hue and nG	181	71	110	39.22%	60.773%

Seventy-one images of the data set contained the largest bounding box which fell in the threshold range of the golden ratio and were segmented as faces. All 71 images were correctly classified without the presence of excessive noise. It is obvious that the percent of images which pass the restriction of the golden ratio is very low; nonetheless, it eliminates a large chunk of images which do not require any further modifications.

B) Morphological Operation and Constraints (Binary Filtering)

For the 110 images which do not meet the golden ratio criteria, we need further modifications in order to localize the face.

The largest blobs in images which do not match the human face dimensions may be for the reasons that, either the face is incomplete, that is the face is divided into disjointed two clusters and do not fall into one bounding box. Another reason is the fact that several non-skin pixels which were falsely classified as skin are part of the largest cluster and are distorting the dimensions of the blob.

We perform morphological operations in order to address these challenges, separate the noise from the face cluster and complete the face if necessary.

The basic two processes of morphological operations are:

a) Erosion: It is the shrinking or thinning of a binary image controlled by the structuring element.
b) Dilation: It is the growing of thickening of a binary image controlled by the structuring element.

A structuring element is a shape, used to probe or interact with a given image, with the purpose of drawing conclusions on how this shape fits or misses the shapes in the image. Through experimentation, it was found that the best combination of morphological operations is depicted in Figure 2.24. On the images which failed the golden ratio criteria, the first operation performed was closing which is basically dilation followed by erosion. Closing is performed to fill all the gaps in the face region so as to ensure that the face in considered as 1 blob and does not break away, followed by performing erosion to eliminate the small blobs surrounding the face and lastly performing opening,   Opening is also erosion followed by dilation by the same structuring element. Opening is performed to eliminate the small objects which are noise in the background. Also it will break away thin connections between objects which is important as it will help in preserve the dimension of the face in order to separate the face blob from the connected background blobs. The typical example is depicted in Figure 2.25.

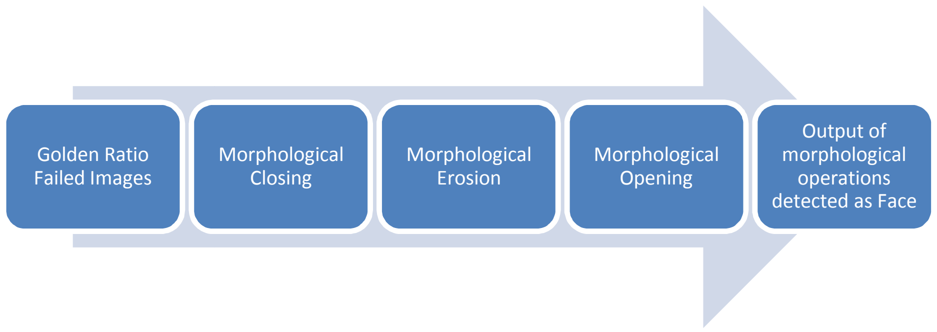

Golden Ratio Failed Images
Morphological Closing
Morphological Erosion
Morphological Opening
Output of morphological operations detected as Face

Figure 2.24: Flowchart representing the morphological operation

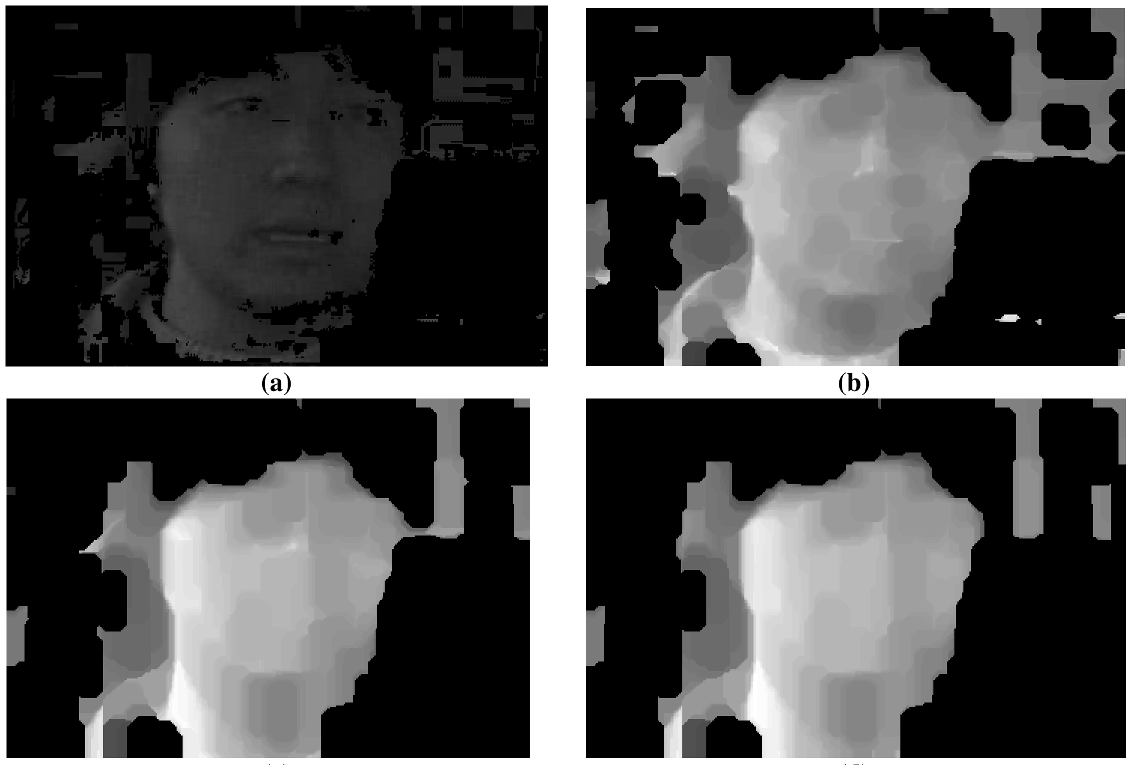


Figure 2.25: (a) Image after application of Bayes Classifier (skin detected image) (b) Performing closing on image (a) (c) Performing Erosion on image (b) (d) Performing Opening on Image (c)

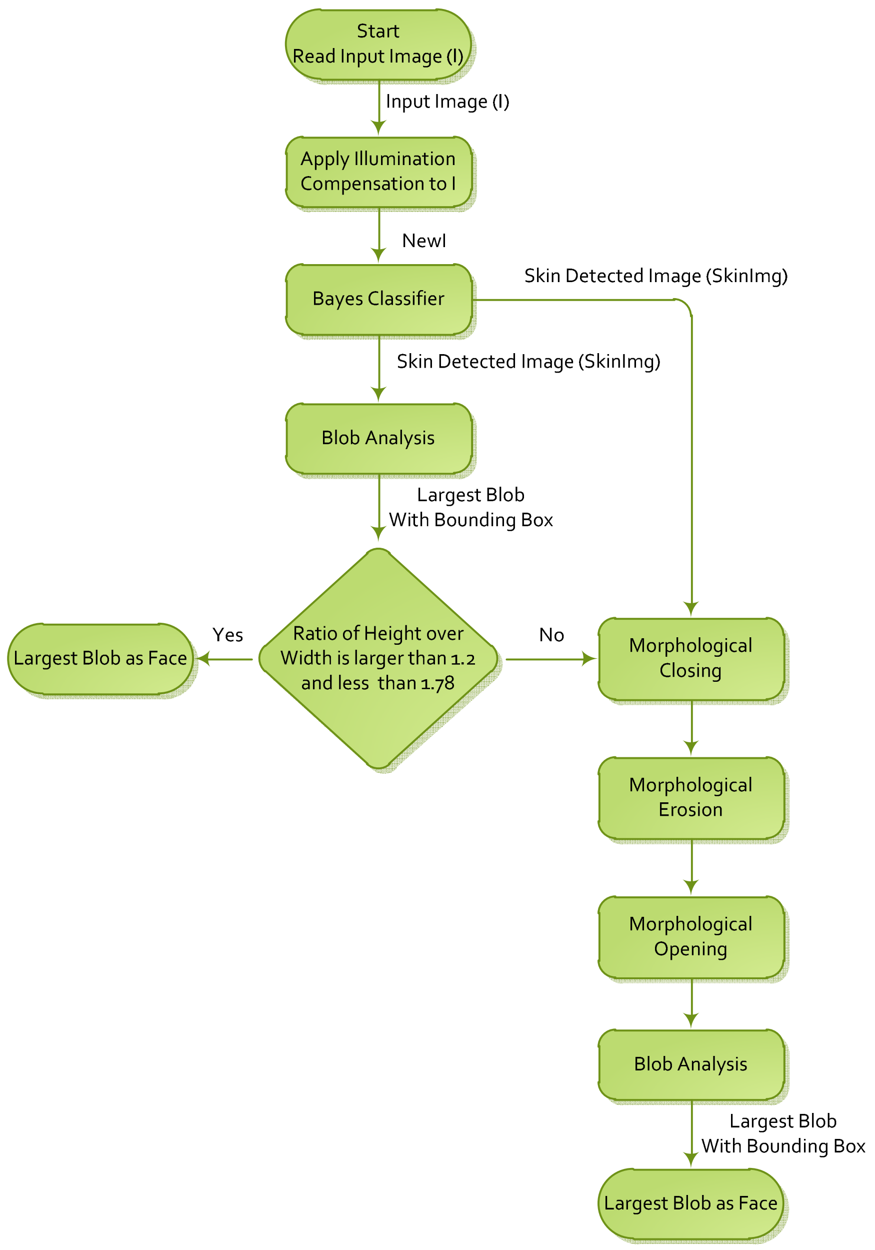

Start
Read Input Image (I)
Input Image (I)
Apply Illumination Compensation to I
NewI
Bayes Classifier
Skin Detected Image (SkinImg)
Skin Detected Image (SkinImg)
Blob Analysis
Largest Blob With Bounding Box
Ratio of Height over Width is larger than 1.2 and less than 1.78
Yes
Largest Blob as Face
No
Morphological Closing
Morphological Erosion
Morphological Opening
Blob Analysis
Largest Blob With Bounding Box
Largest Blob as Face

Figure 2.26: Flowchart representing face detection algorithm using Bayes Classifier and Morphological Operations

Flowchart represented in Figure 2.26 depicts the first two stages of the face detection algorithm. We start with the input image which we correct for illumination. After passing through the Bayes classifier we select the largest cluster of pixels. If the cluster falls in the golden ratio range we segment the cluster as a face. If the cluster fails the golden ratio threshold, the skin classified image is passed through a series of morphological operation. We once again select the largest blob and segment the blob as a face.

Table 2-2 contains the results of passing the 181 test images through the first two stages of the face localization algorithm as represented in Figure 2.26.

Table 2-2: Results of face detection using Bayes classifier with Morphological operations on 181 test images

Face Detection Technique	**Color Space**	**Total Faces**	**Faces correctly classified**	**Faces incorrectly classified**	**Percentage Detected**	**Percentage Failed**
Bayes Classifier + Golden Ratio + Morphological Operations	Hue and nG + Gray Scale	181	119	62	65.745%	34.254%

Of the 181 images, seventy-one images passed the golden ratio in the first stage, and 110 images were passed through the morphological operations after which 48 images were correctly segmented as faces.

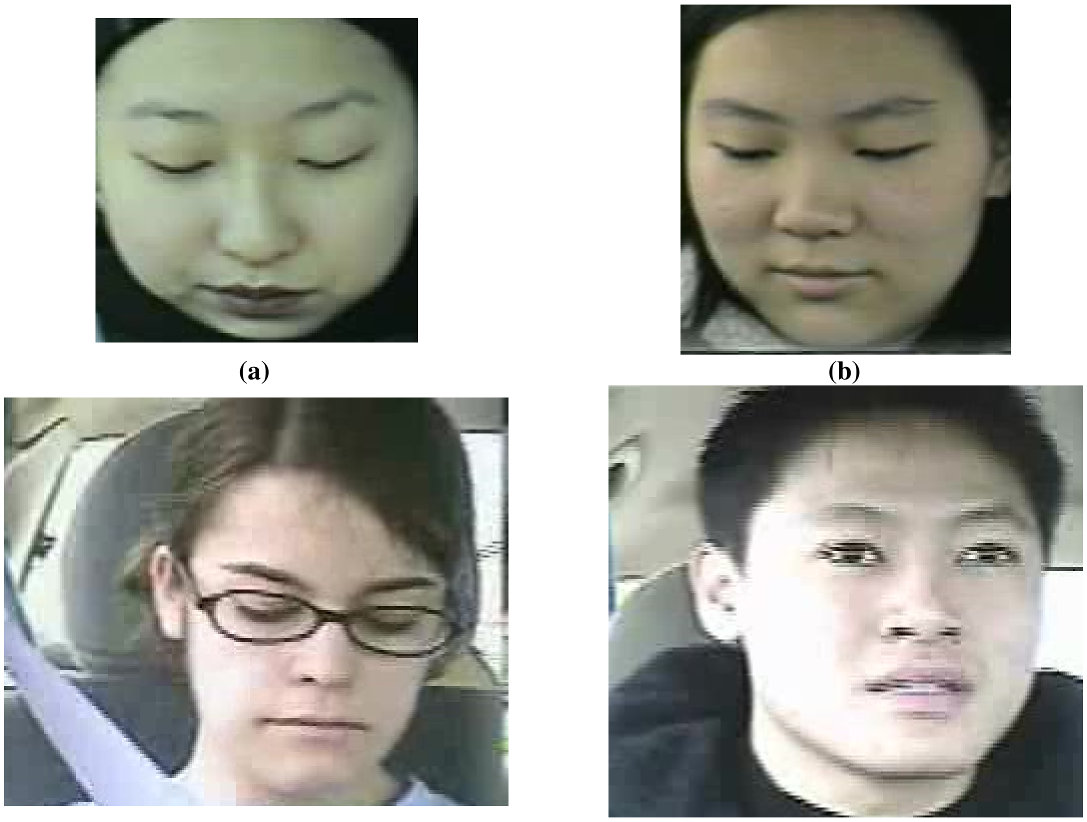


Figure 2.27: (a) and (b) Correctly classified as faces after Bayes classifier and Morphological Operations (c) and (d) Incorrectly classified faces with excessive background noise

Figure 2.27 displays the images which were correctly classified as faces after Bayes classifier and morphological operations and the images which do not pass the golden ratio criteria due to excessive noise are shown in Figure

The result solely after the Bayes classifier and morphological operation is fairly poor with only 65% detection rate. It was observed that the 62 images which were falsely segmented as faces contained a large number of background pixels incorrectly classified as skin. Due to the large amount of noise present in the image, only skin color and morphological operations were not enough to detect faces.

In the last stage of the face localization method, we focus our attention to template matching. Template matching does not use the color information of the image, but instead the intensity component which highlights the features of the face. We aim to remove the shortcomings of the skin detector due to difficult background conditions by implementing the template matching algorithm.

C) Template Matching

The concept of template matching is fairly simple. We create a sample face known as a template using training data. This template is then passed over the test data and any region which resembles the template the most is selected as the face region.

Creating the Template:

It was noticed that due to difference in position of the cameras, one template might not work for all test images, for example in cases with a turned face with some parts occluded. We created templates for all four camera angles but the run time of the algorithm significantly increased. Therefore, we restricted to two templates which gave the same accuracy as four templates.

For the training of the template, we take 315 images from camera angles 3 and 4 each and manually select the face region. The selected region is cropped and placed on a black background. The sample face used for the training is depicted in the Figure 2.29.

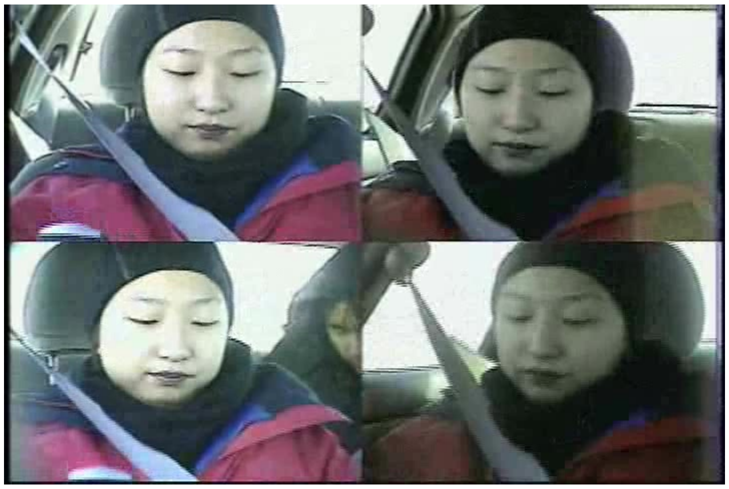


Figure 2.28: Original Image of AVICAR database representing the four camera angles. The bottom two images represent camera 3 and 4.

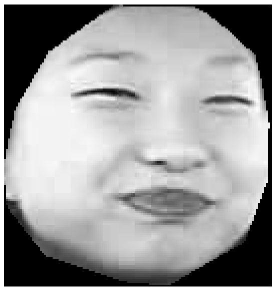


Figure 2.29: Sample image for creating the template

To create the face template, each image is first resized to the size 200 x 150 and then normalized through the following equation to reduce the error due to lighting conditions and backgrounds [37]:

$$\frac{\qquad}{\qquad} \qquad (2\text{-}12)$$

where the is the mean of the image and is the standard deviation of the image. and are variables, which are chosen as values as close to the mean and standard deviations as most images. By experimentation, the values were chosen as

.

The mean of all normalized images forms the face template. The two face templates used for template matching are indicated in Figure 2.30.

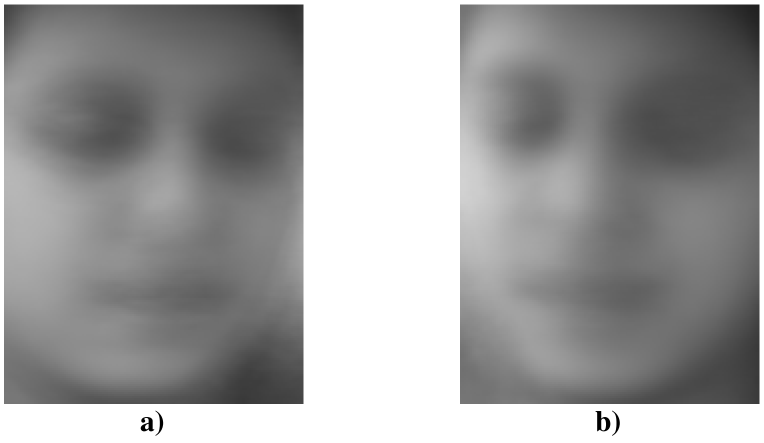


Figure 2.30: Resulting template for camera a) 3 and b) 4

Applying the Template:

We use a Matlab function, Normxcorr2, to run the templates over the test image. This function computes the normalized cross correlation of the test image and the template, where the result is the correlation coefficient with values ranging from 1 to -1.
Before the application of the function, we resize all the test images so that the ratio of their sides is 4:3, in proportion to the size of the template which is 200 x150. To compensate for difference in face sizes in comparison to the template, we run the function at different image sizes. We resize the test image using the following equation

(2-13)

The variable   is increased from 1 to 15 in steps of 1. We run both the templates individually on the test image and determine the template and its size with the largest correlation coefficient. A bounding box of that size is drawn around the detected region. Figure 2.31 represents correctly classified faces and incompletely detected faces.

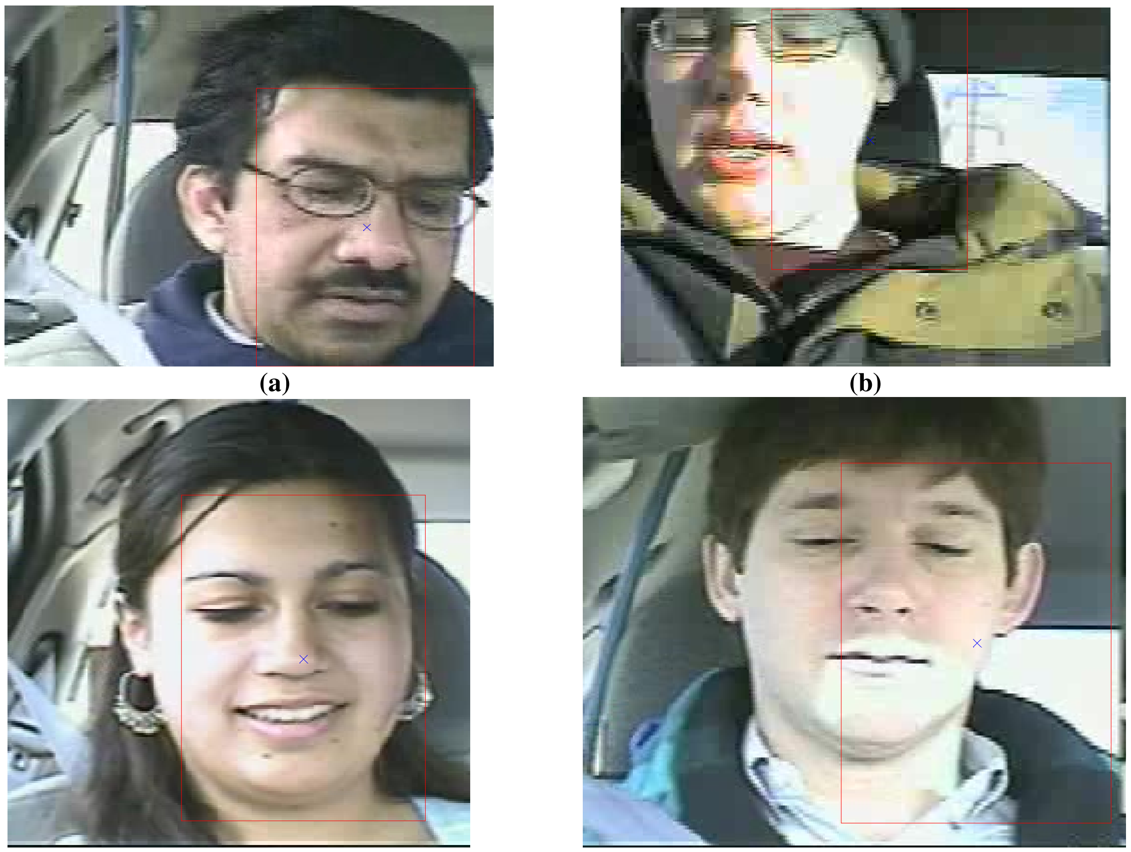

(a) (b) (c) (d)

Figure 2.31: (a) and (c) Correctly detected face through template matching and (b) and (d) Incomplete face detected through template matching

To evaluate the accuracy of the template matching algorithm alone without the first two stages of skin classification and morphological operations, we pass the illumination corrected 181 images through the template matching algorithm.

Table 2-3: Results of face detection using Template Matching on 181 test images

Face Detection Technique	**Color Space**	**Detected Faces**	**Incomplete Faces**	**Missed Faces**	**Mis-classified Faces**	**Percentage Detected**	**Percentage Failed**
Template Matching	Gray Space	135	36	0	10	74.585%	25.414%

Of the 181 test images 135 images were correctly classified without excessive background noise. The 36 incomplete faces refer to any image where only a part of the face was detected, or where any part of the lips was missed. Mis-classified faces refer to cases where none of the important features, such as the eyes, nose, and mouth region was detected and large part of the face was missed.

2.3. Face Detection Results

The results of face detection by template matching are also much lower than desired. Therefore, in order to increase the accuracy we concatenate Bayes classifier with the template matching technique.

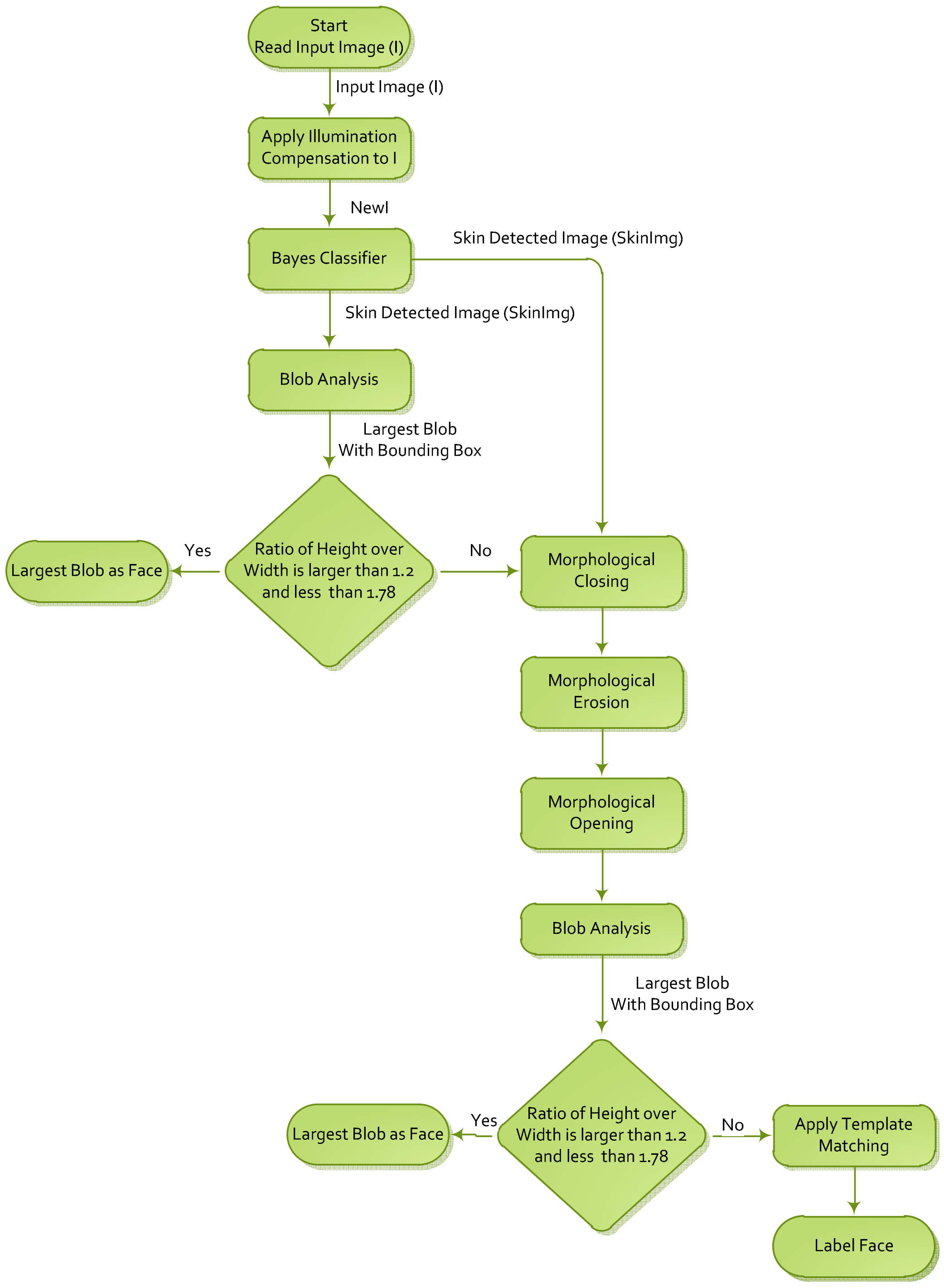

Start
Read Input Image (I)
Input Image (I)
Apply Illumination Compensation to I
NewI
Bayes Classifier
Skin Detected Image (SkinImg)
Skin Detected Image (SkinImg)
Blob Analysis
Largest Blob With Bounding Box
Ratio of Height over Width is larger than 1.2 and less than 1.78
Yes
Largest Blob as Face
No
Morphological Closing
Morphological Erosion
Morphological Opening
Blob Analysis
Largest Blob With Bounding Box
Ratio of Height over Width is larger than 1.2 and less than 1.78
Yes
Largest Blob as Face
No
Apply Template Matching
Label Face

Figure 2.32: Flowchart representing face detection algorithm using Bayes Classifier, Morphological Operations and Template Matching

In Figure 2.32 the flowchart represents the concatenation of Bayes classifier with template matching. At the end of the second stage of face localization we obtain images after performing morphological operations. These images are once again checked for the golden ratio criteria, only the images which don't fall within the threshold the second time also, are passed through the template matching algorithm.

Table 2-4: Results of face detection using Bayes Classifier followed by the Template Matching on 181 test images

Face Detection Technique	Color Space	Detected Faces	Incomplete Faces	Face with excessive noise	Mis-classified Faces	Percentage Detected	Percentage Failed
Bayes Classifier +Morphological Operations + Template Matching	Hue and Ng + Gray Space	160	5	16	0	88.39%	11.60%

The result of the concatenated algorithm for 181 test images is tabulated in Table 2-4. Of the 181 images, seventy-one images were correctly classified with the Bayes classifier and the golden ratio. After performing morphological operations, 48 more images passed the golden ratio threshold. Of the remaining 62 images on which template matching was performed, 41 images were accurately detected.

2.4. Viola-Jones Algorithm

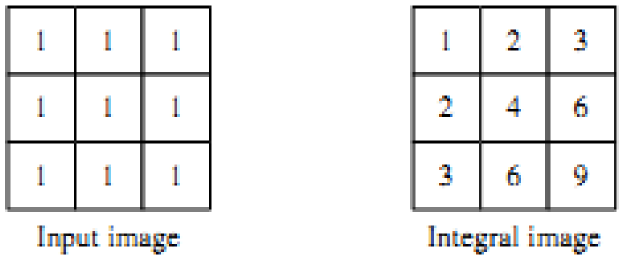
A) Viola-Jones Description

With the first approach, 88.4% detection rate was achieved. In order to further improve the detection rate, another famous face detection algorithm is examined, Viola-Jones algorithm [14]. Viola-Jones algorithm is currently one of the most robust face detection techniques in implementation. We aim to compare our concatenated face detection algorithm with the Viola-Jones technique. This method is essentially a rapid object detection algorithm. Contrary to popular methods of object detection using color information, [14] uses monochromatic information to compute ‘Haar like’ features.

The basic principle of Viola-Jones algorithm is to scan a sub-window (detector) capable of detecting faces across a given input image. This detector is rescaled and run through the input image many times, each time with a different size. The detector constructed contains some simple rectangular features reminiscent of Haar wavelets.

Integral Image:

The first step was to convert the input image into an integral image. Each pixel is made equal to the entire sum of all pixels above and to the left of the cornered pixel.



1	1	1
1	1	1
1	1	1

Input image

1	2	3
2	4	6
3	6	9

Integral image

Figure 2.33: Input Image and the Integral Image proposed by Viola-Jones [14]

This allows for the calculation of any given rectangular area within the desired region using only four values.

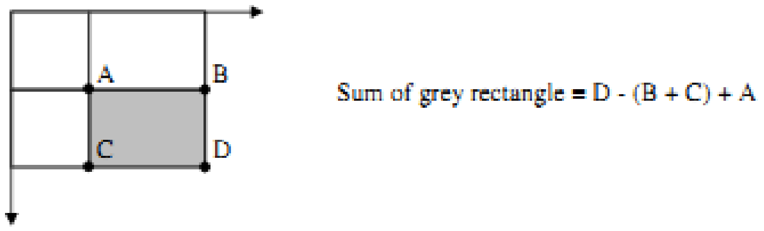


Figure 2.34: Example to show the calculation via the integral image [14]

Feature Discussion:

This object detection procedure classifies images based on the value of simple features. The value of a two-rectangular feature is the difference between the sum of the pixels within the two rectangular regions as in Type 1 and Type 2. A three rectangular feature computes the sum within two outside rectangles subtracted from the sum in the center rectangle as shown in Type 3 and Type 4. Finally, as represented by Type 5, a four rectangle feature computes the difference between diagonal pairs of rectangles.

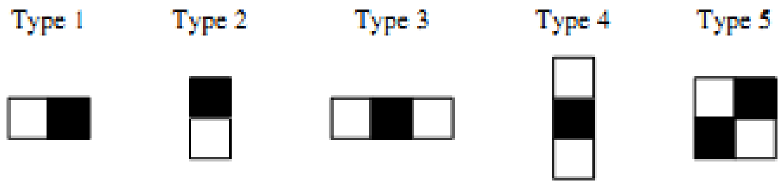


Figure 2.35: The types of 'Haar like' features used in the training the Viola-Jones classifier [38]

The concept of integral image makes the calculation of all features significantly faster. Implementation of the Viola-Jones algorithm for face detection has resulted in determining that the detector size of around 20 x 20 pixels gives satisfactory results. Therefore, allowing for all possible sizes and positions of the features approximately 180,000 different features can be constructed. The objective now is to construct a mesh of features capable of detecting faces.

Modified AdaBoost Algorithm:

From the 180,000 features present in the detector, some features give consistently high values when over a face than the others. Viola-Jones uses a modified version of the AdaBoost algorithm to find these features.

AdaBoost algorithm is based on constructing a strong classifier through a weighted combination of weak classifiers.

A weak classifier is mathematically represented as:

$$h(x,f,p,\theta) = \begin{cases} 1 & if\ pf(x) > p\theta \\ 0 & otherwise \end{cases} \tag{2-14}$$

Where x is the 20 x 20 pixel sub-window, f is the applied feature, p the polarity and θ the threshold that decided whether x should be classified as a positive (a face) or a negative (a non-face).

As only a small number of features are expected to be potential classifiers, the algorithm is modified to select only the best features. The method to select the best feature, polarity and threshold used by Viola-Jones is of brute force. Which means that to determine each weak classifier, each feature is evaluated on all training examples in order to find the best performing feature. This is expected to be the most time consuming procedure of the training.

The best performing feature is chosen based on the weighted error it produces. As seen in part 4 the weight of the correctly classified example is decreased and the weight of the misclassified example is kept constant. Therefore, it is more expensive for the second feature to misclassify an example also misclassified by the first feature, than an example classified correctly.

```
- Given examples images $(x_1,y_1),\dots,(x_n,y_n)$ where $y_1=0,1$ for negative and positive examples.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_1=0,1$, where $m$ and $l$ are the numbers of positive and negative
  examples.
- For $t=1,\dots,T$:
1) Normalize the weights, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
2) Select the best weak classifier with respect to the weighted error:
        $\varepsilon_t = \min_{f,p,\theta} \sum_i w_i |h(x_i, f, p, \theta) - y_i|$
3) Define $h_t(x) = h(x, f_t, p_t, \theta_t)$ where $f_t$, $p_t$ and $\theta_t$ are the minimizers of $\varepsilon_t$.
4) Update the weights:
        $w_{t+1,i} = w_{t,i}\beta^{1-e_i}$
   where $e_i = 0$ if example $x_i$ is classified correctly and $e_i = 1$ otherwise, and $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$
- The final strong classifier is:
        $C(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$
  where $\alpha_t = \log \frac{1}{\beta_t}$
```

Figure 2.36 shows the features which are generally chosen as the first and second by the modified AdaBoost learning algorithm. The first feature takes advantage of the difference in intensity between the region of the eye and the region of the upper cheek. The second feature calculates the difference between the two eyes and the center bridge of the nose.

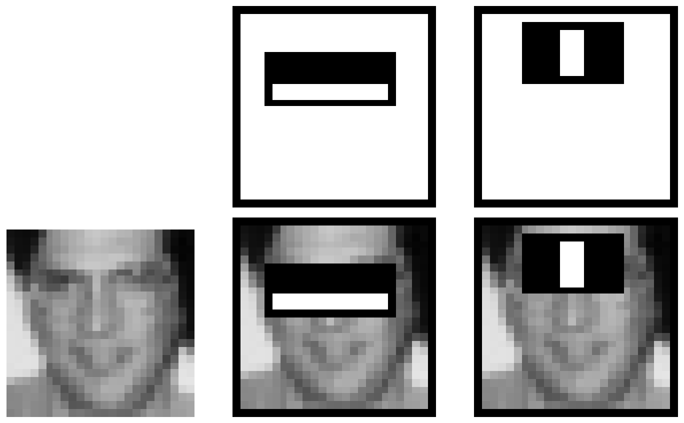


Figure 2.36: The most common first and second feature obtained by the modified AdaBoost algorithm for the Viola-Jones classifier [38]

Cascaded Classifier:

Viola-Jones utilized cascaded classifier in order to approach the face detection problem from another point of view. Instead of finding face, the algorithm should discard non-faces. Therefore, the idea of the detector using only one strong classifier seemed inefficient.

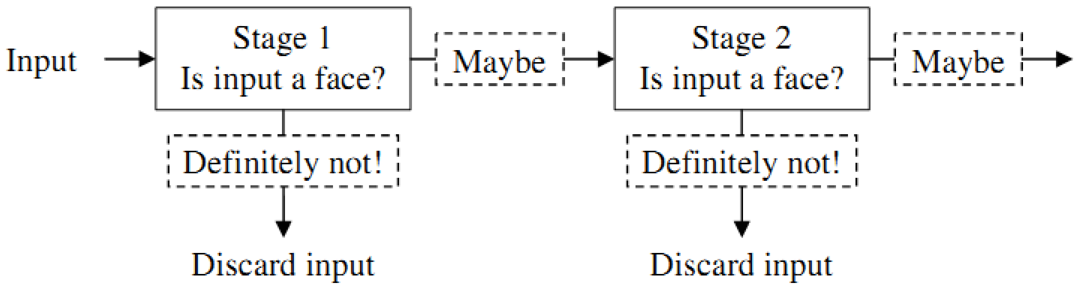


Figure 2.37: The cascaded classifier [38]

The cascaded classifier is composed of stages, where each stage contains a strong classifier. Each stage determines if the given sub-window is definitely not a face or maybe a face. The sub-window is immediately discarded when classified as a non-face, but if the sub-window is classified as maybe face, it is passed on to the next stage in the cascade.

Implementation of Viola-Jones Algorithm for Face Detection:

The Viola-Jones algorithm was obtained from OpenCV (Open Source Computer Vision). OpenCV is a library on Computer Vision created by Intel. The algorithms from OpenCV can be accessed using Visual Studio C++.

OpenCV already contains an implemented version of the Viola-Jones algorithm with a trained classifier which can be directly implemented. We download this classifier from openCV and extract the xml file through Visual Studio C++ which is explained later. This xml file can be implemented as a classifier on the images through Matlab (attached Matlab algorithm in Appendix). The parameters of the classifier were not altered from [39] except the window size was restricted to 20 X 20. Training of the classifier is a lengthy and time consuming process which takes anywhere between 5 days to 2 weeks.

OpenCV also provides a way to train and create your own classifier using Haar-like features in HaarTraining. The result of the training is an xml file that contains the definitions of the classifier. The details of training a classifier are discussed later.

Table 2-5: Results of implementing the in-built Viola-Jones Classifier on 181 test images

Face Detection Technique	Color Space	Detected Faces	Incomplete Faces	Missed Faces	Mis-classified Faces	Percentage Detected	Percentage Failed
Viola-Jones	Gray Space	164	5	12	1	90.6077%	9.3923%

The 181 test images were passed through the in-built gray scale classifier in order to compare the results with the concatenated face detection algorithm of Bayes classifier and template matching. For the 5 incomplete faces, it was due to the fact that not the entire lip region was included. With 90% detection rate, the Viola-Jones algorithm results in slightly better accuracy than the Bayes classifier and template matching algorithm which achieve an accuracy of 88%.

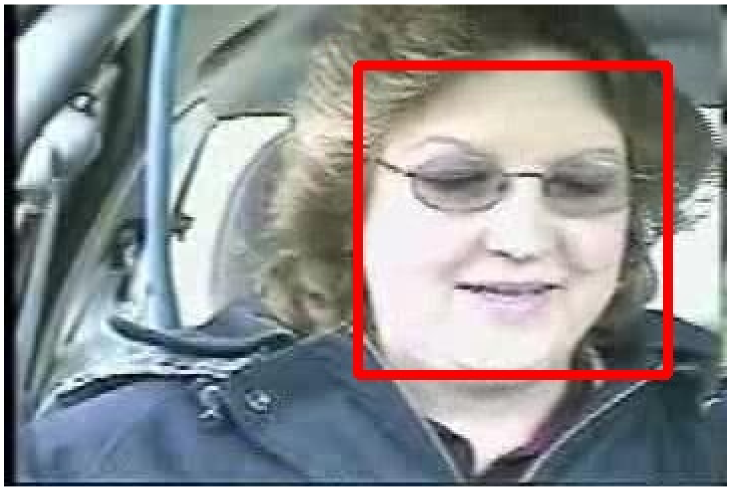


Figure 2.38: Correctly classified face by the Viola-Jones Algorithm

As noticed, the accuracy of the Viola-Jones algorithm is around 90%. Although by itself the performance is acceptable, this algorithm completely misses 12 out of the 181 images. In order to further improve the detection rate, we examined the possibility of using color information as opposed to Viola-Jones monochromatic gray scale intensities as the training data.

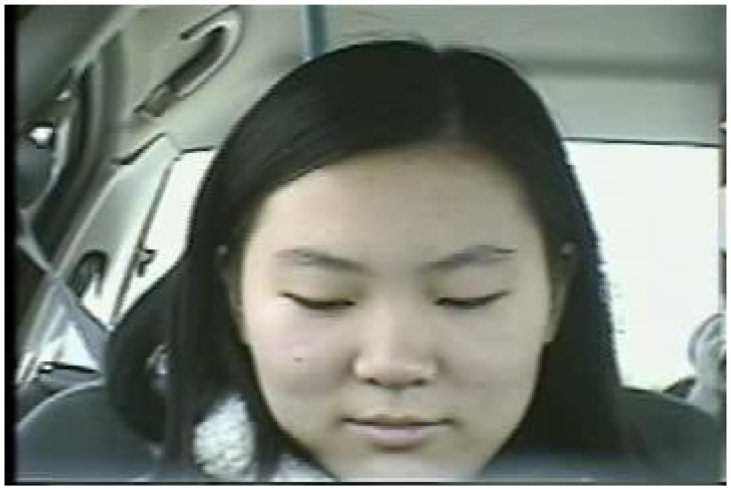


Figure 2.39: Example of an image missed by the Viola-Jones Algorithm

<u>Viola-Jones Algorithm (Modification)</u>

The rapid object detection algorithm proposed by Viola-Jones uses the gray scale images for its positive and negative image training. The basic principle of the method requires at least the positive training data set to have distinct features in order to train the 'Haar like' wavelets. This makes an interesting study, as to if there is a color space better suited than the gray space for the implementation of Viola-Jones.

In this experiment, we train our own classifier in OpenCV and compare it to the inbuilt classifier obtained from OpenCV. For training the classifier we used 7096 positive images and 3138 negative images as represented in the figure.

Training a Classifier [39]:

The training of a classifier is divided into 4 parts.

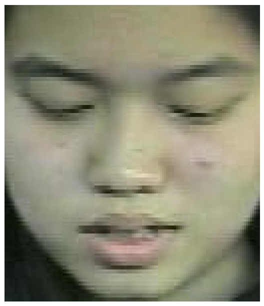
1) Data Preparation: The first step to training the classifier is the preparation of the positive and negative data. The positive data consists of only the object desired to be detected. In this case the positive data is face. It has been determined that to achieve desired results at least 5000 positive images are required. The negative data should consist of several background images as varied as possible without including any faces. The first step is complete when all the data is acquired and placed into the respective folders.
2) Creating Sample Images and Generating .vec: This step is performed only on the positive data set. The createsample.exe is used to create the .vec file for the faces, where the size of every positive image is defined.
3) Training the classifier: The executable available in OpenCV called Haartraining.exe is used for the next step which is the actual training of the classifier using the negative data and the positive samples created in step 2. Depending on the number of

negatives and positives collected, the time of the training varies. This may take for several hours to several days. It is a lengthy and time consuming process.

4) Testing: Once the training of the cascaded classifier, the last step is to run the classifier on the test images.

For the sake of our face detection algorithm, we have used the in-built pre trained classifier in the gray scale. We have also trained classier in different color spaces, whose results are summarized and analyzed in the results section.

Below is the tabulated result of the Viola-Jones face detection algorithm. From the results it could be observed that, the boundary across the detected faces did not include any background noise and perfectly encompassed the face region. A very small percentage of faces were incomplete, and only 1 image had a misclassification.


(a)

(b)

Figure 2.40: a) Sample positive image and b) sample negative image for the training of the Viola-Jones classifier

Table 2-6: Results of face detection of 181 images by trained Viola-Jones Classifier

Color space	Detected	Incomplete Faces	Missed	False Detection	Percent Faces Detected
Gray inbuilt	164	5	12	1	90.6077%
Red	168	12	1	7	92.8177%
Gray	162	17	2	12	89.50%
Green	167	11	3	12	92.265%
Saturation	151	5	25	33	83.425%
Green (Illumination Compensated)	170	10	1	16	93.92%

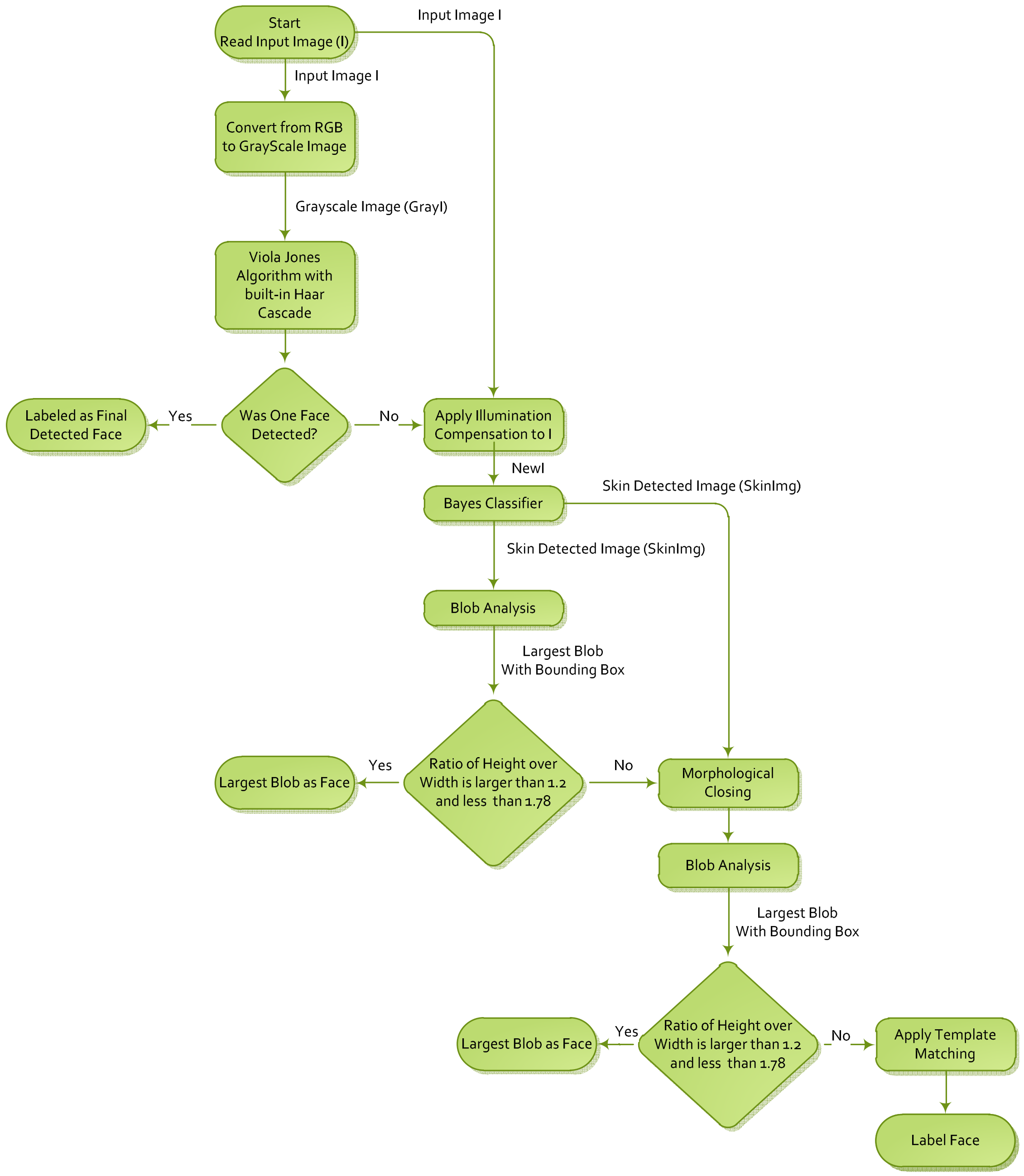
Our objective was to compare results between the classifiers trained in different color spaces and to observe if a color space other than gray scale can improve the detection rate.

The inbuilt gray scale classifier in OpenCV used positive and negative images to train which exceed the number of images we use in our training by a great number. In order to have a fair comparison between all color spaces, we first train the classifier using our training images in gray scale.

In the color spaces without illumination compensation it can be observed that the classifier trained in the red color space gives the best performance for true positives. It is also interesting to note that the red color space classifier had the least amount of false detection rate. The performance of the green color space trained classifier was very similar to that of the red which outperformed the gray scale classifier. As noticed red and green classifiers outperformed even the in-built classifier from OpenCV for correct face detection but performed more false detections and incomplete faces, which might indicate that a larger training data set might improve the performance.

Lastly, we trained a classifier in the green color space with all the positive and negative images first illumination compensated. It is apparent that the detection rate for illumination compensated green outperforms all other classifiers. Future work can investigate different color spaces with and without illumination compensation to achieve the highest detection rates.

2.5. Final Face Detection Algorithm



Start
Read Input Image (I)
Input Image I
Input Image I
Convert from RGB to GrayScale Image
Grayscale Image (GrayI)
Viola Jones Algorithm with built-in Haar Cascade
Was One Face Detected?
Yes
Labeled as Final Detected Face
No
Apply Illumination Compensation to I
NewI
Bayes Classifier
Skin Detected Image (SkinImg)
Skin Detected Image (SkinImg)
Blob Analysis
Largest Blob With Bounding Box
Ratio of Height over Width is larger than 1.2 and less than 1.78
Yes
Largest Blob as Face
No
Morphological Closing
Blob Analysis
Largest Blob With Bounding Box
Ratio of Height over Width is larger than 1.2 and less than 1.78
Yes
Largest Blob as Face
No
Apply Template Matching
Label Face

Figure 2.41: Flowchart for the final face detection algorithm

The concatenation of Bayes Classifier with the template matching algorithm, although yielding respectable results, is slightly lower than the Viola-Jones technique previously discussed. Therefore, each method on its own is not capable of providing desired face detection results. Hence, we first perform the Viola-Jones technique on a set of Test Images. This technique contained very 'few incomplete face' images, and more 'missed face' images. The images which were completely missed by the Viola-Jones technique are then passed through the Bayes classifier and if need be the morphological operations. Any image, which finally fails the golden ratio, is then passed through the template matching algorithm.

It is essential to note that we use the in-built Viola-Jones classifier even though the red color space trained classifier gave a better face detection result. Due to its large training set of the in- built classifier, it results in fewer false detected images and more missed images, which in the case of red is more false detected and almost none missed which means it would be impossible to concatenate further with any other technique.

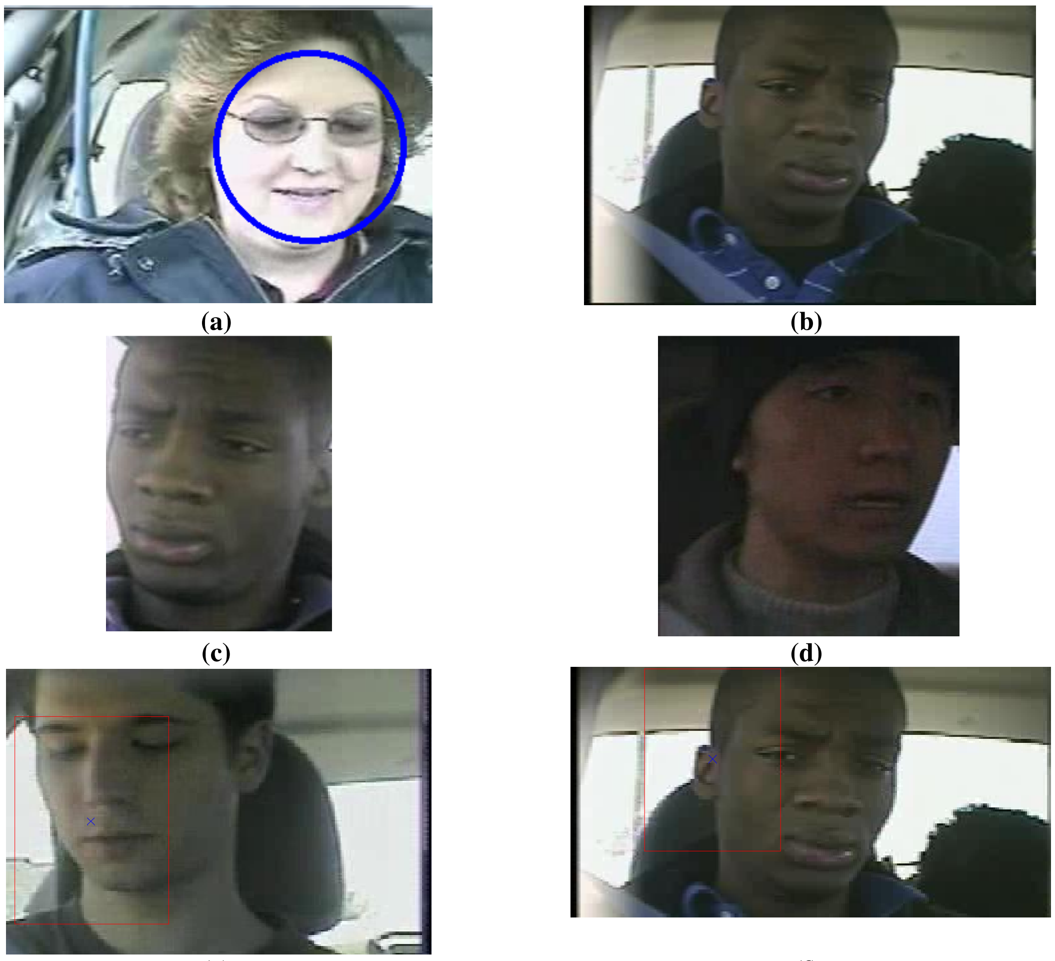


Figure 2.42: a) Face detected by the in-built Viola-Jones algorithm b) Face missed by the in-built Viola-Jones algorithm c) Face which was missed by Viola-Jones is correctly detected my Bayes Classifier d) Face missed by Viola-ones algorithm and Bayes Classifier e) Face Missed by Viola-Jones algorithm and Bayes classifier and correctly detected by Template matching f) Face missed by all methods.

Final Face Detection Result:

Step 1: Inbuilt Viola Jones: Detected: 164/181 (90.6077%)

Missed: 12/181 (6.628%) Incomplete: 5/181 (2.762%)

Step 2: Detection using Bayes Classifier: Detected: 6/12 (50%)

Missed: 6/12(50%)

Step 3: Template Matching: Detected: 2/6 (33.33%) Incomplete: 4/6 (66.66%)

Overall Face Detection Result:

Faces Detected: 172/181 (95.027%) **Faces Missed: 9/181 (4.972%)**

3. LIP LOCALIZATION

The stage after successful face detection is mouth localization. Several techniques have been under survey and experimentation.

In this thesis we focus on a lip enhancement technique which defines the contours of our lips. We further isolate the lip contour using multiple restrictions.

In Section 3.1 we perform color space analysis to evaluate if any space can be used to train a classifier. In Section 3.2 we discuss different lip enhancement techniques and define our lip localization algorithm. In Sections 3.3 and 3.4 we perform operations to isolate the lips. Lastly, in section 3.5 we discuss the results and scope of improvement.

3.1. Color Space Analysis

Several color spaces have been discussed for the problem of face detection such as RGB, normalized RGB, HSV, YCbCr as shown in section 2.2.1. Color space for lip segmentation, on the other hand, poses a greater challenge, due to the small variation between the color of the skin and the color of the lips.

Similar to the process in face detection, we collect training data from about 1500 images including all skin color and lightening conditions available in the data set. The background image in this instance consists of only the face, as the primary objective is to determine a boundary between lip pixels and face pixels. The lip region is manually cut out from each face as shown in the Figure 3.2.

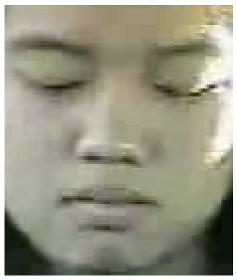


Figure 3.1: Face or background image

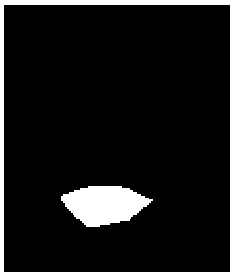
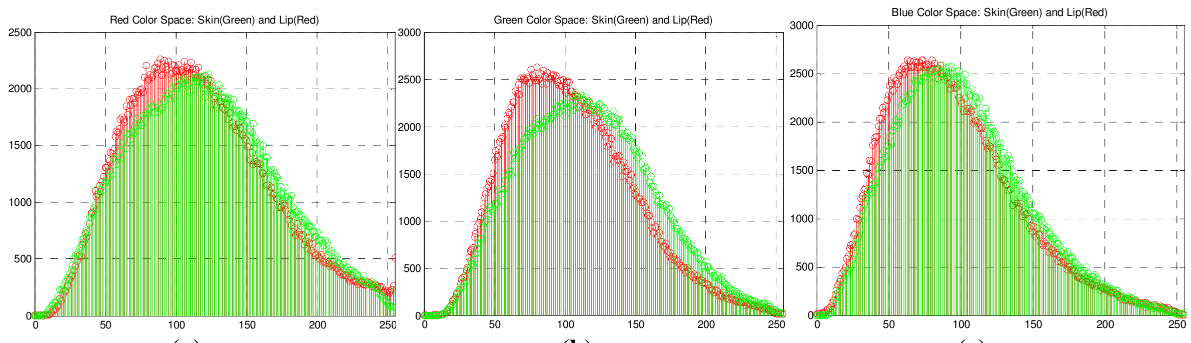


Figure 3.2: Mask of a lip from Figure 3.1

We plot histograms of face pixels vs. the lip pixels in several popular color spaces to evaluate their potential for lip segmentation.

<u>RGB Color Space</u>

The color distributions between skin and lip pixels samples overlap each other along with low inter-class variances for each R, G and B component. As a result, RGB color space is not suitable for direct lip segmentation as it demonstrates a strong correlation between light and color information.



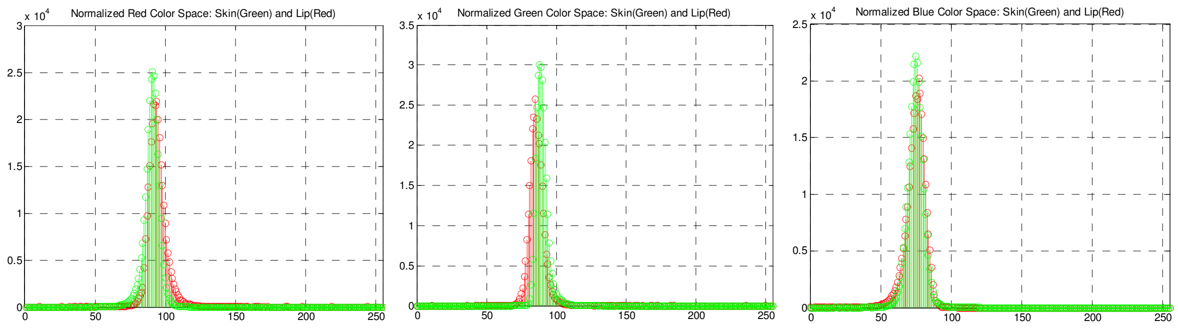
(a) **(b)** **(c)**

Figure 3.3: Lip v/s face histogram (a) Red component (b) Green component (c) Blue component. In the figure Red signifies face pixels and green signifies lip pixels

<u>Normalized RGB</u>

In the case of normalized RGB, although the intra class cluster deviation is much smaller than RGB, the two color spaces almost completely overlap. Variations have also been experimented in papers which investigate the result of histogram thresholding using maximized intensity normalization, in which instead of dividing by the sum of the RGB component at each pixel location, the denominator is the maximum value of R+G+B over the entire image, or max(R+G+B). The paper concludes that this transformation of maximum intensity thresholding achieves better results for both skin and lip segmentation. Another variation is mentioned in [40] in which a weight of 1/3 is applied to the denominator; instead of R+G+B as denominator, it becomes

$$\frac{1}{3}(R+G+B)$$

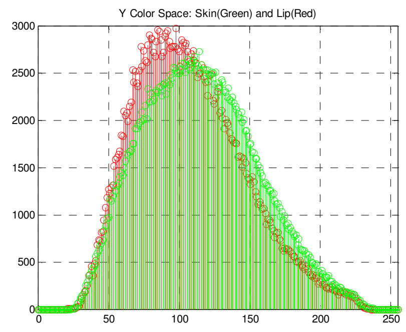


(a) (b) (c)

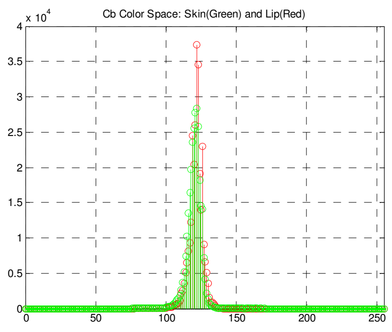
Figure 3.4: Lip v/s face histogram (a) normalized Red component (b) normalized Green component (c) normalized Blue component. In the figure Red signifies face pixels and green signifies lip pixels

<u>YCbCr Color Space</u>

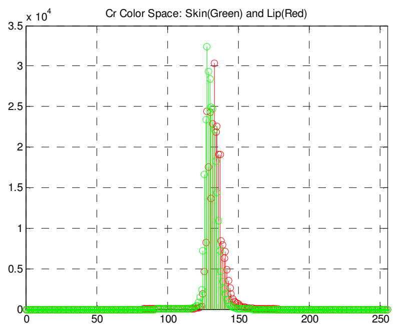
The Y component in the YCbCr color space does not satisfy the criteria of low intra class variance and the two clusters completely overlap. Cb and Cr, on the other hand, do not directly seem to have a boundary that can be defined between the two clusters, it is considered that lip region have high Cr and low Cb values [41]. This can be used to enhance the lips in an image, which is later discussed.



(a)



(b)



(c)

Figure 3.5: Lip v/s face histogram (a) Y component (b) Cb component (c) Cr component. In the figure Red signifies face pixels and green signifies lip pixels

Other Color Transformations:

The other popular color spaces experimented include Hue, which proves to be a valuable component in face detection but does not have the same results in lip segmentation. The U and V space have been used to model curves for lip detection but do not work well with Bayes classifier. The RGB combination techniques do not prove useful by the histograms whereas the ‘a’ space might have potential.

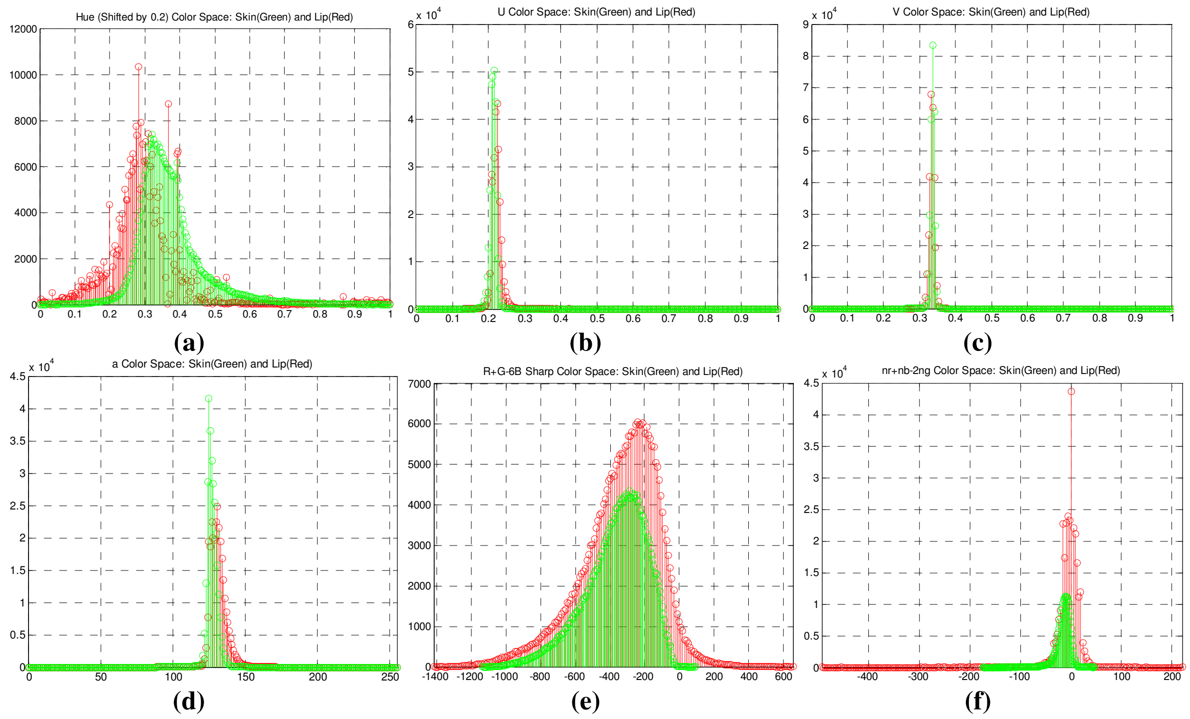
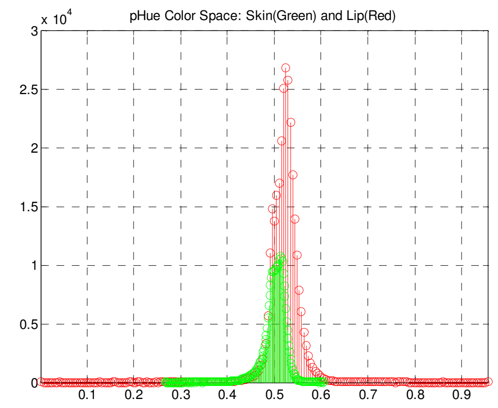


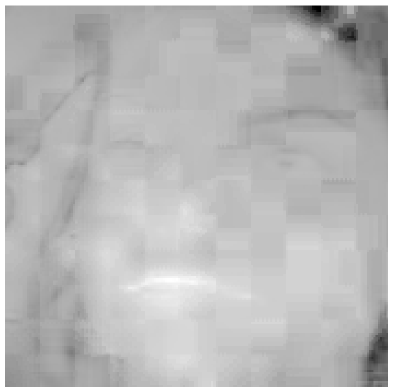
Figure 3.6: Lip v/s face histogram (a) Shifted Hue component (b) U component (c) V component (d) a component (e) R+G-6B combination (f) nr+nb-2ng combination. In the figure Red signifies face pixels and green signifies lip pixels

Based on our experiment, appearance-based classification approach, such as Bayes Classifier using various color spaces, to classify the lip and face did not yield satisfactory results; therefore, we experiment with these color spaces to enhance the lips against the skin.

Commonly used equations for either lip color or lip boundary enhancement include Pseudo Hue given by the equation [42]

$$Pseudo\ Hue\ (\widehat{H}) = \frac{R}{R+G} \tag{2-14}$$



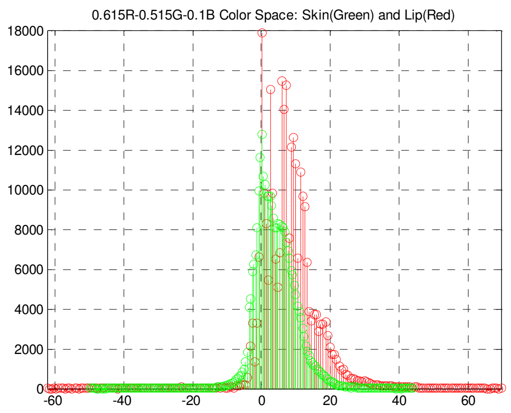


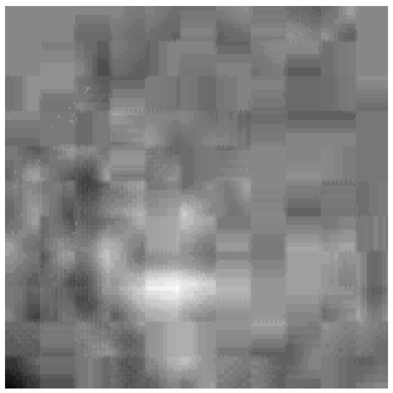
(a) **(b)**

Figure 3.7: (a) Lip v/s face histogram for pHue. In the figure Red signifies face pixels and green signifies lip pixels (b) Image displayed in pHue color space for lip enhancement.

The lip region is represented by pixels of much higher intensity values than the rest of the face. Although, lip region represented higher intensity values, determining a threshold was difficult.

A combination of RGB color spaces represented by the equation $0.5774 * R - 0.7887 * G + 0.2113 * B$ is used to enhance the outer boundary of the upper lip [16].





(a) **(b)**

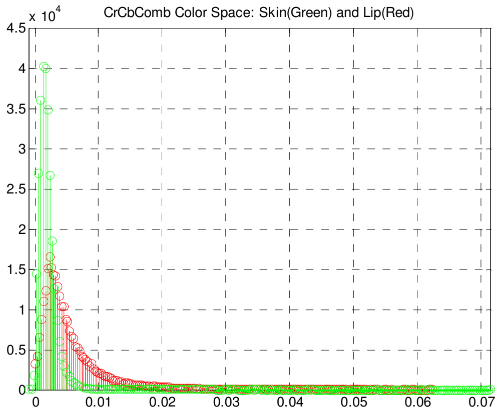
Figure 3.8: (a) Lip v/s face histogram for RGB combination. In the figure Red signifies face pixels and green signifies lip pixels (b) Image displayed in RGB combination color space for lip enhancement.

The RGB combination technique yielded poor results as, the lip region was generally not defined, and a very large area around the mouth contained higher intensity values.

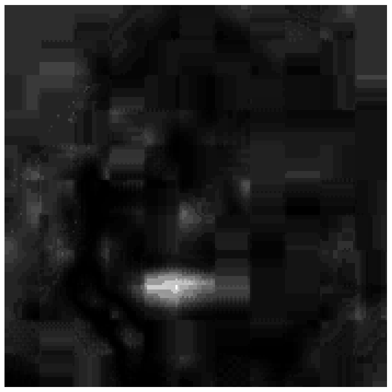
In paper [44], they have followed the principle that lip regions have high Cr and low Cb values and have used the formulas below which aims at brightening pixels with high Cr and low Cb.

$$LipMap = C_r^2\left(C_r^2 - \eta\frac{C_r}{C_b}\right)^2 \quad (3\text{-}1)$$

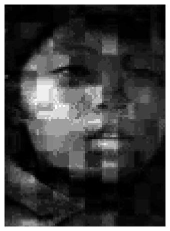
$$\text{where } \eta = 0.95\,\frac{\frac{1}{m}\sum C_r(x,y)^2}{\frac{1}{m}\sum C_r(x,y)/C_b(x,y)}$$



(a)



(b)



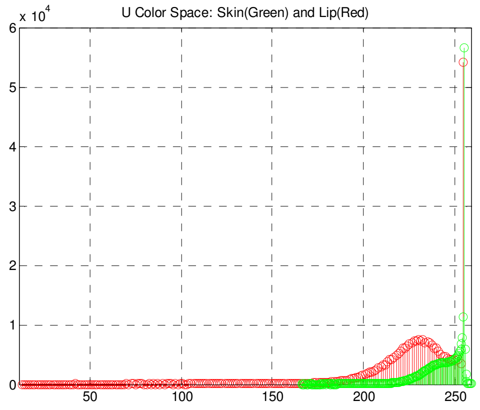
(c)

Figure 3.9: (a) Lip v/s face histogram for CrCb combination. In the figure Red signifies face pixels and green signifies lip pixels (b) Image displayed in CrCb combination color space for lip enhancement (c) Image where CrCb combination fails for lip enhancement.

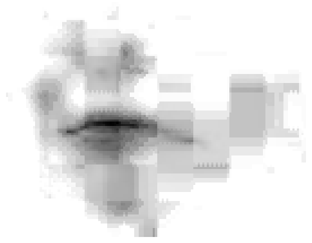
The result of this enhancement technique is impressive as the lip region is significantly brighter than rest of the face image, but the drawback is that it is not consistent with all the test images and variations in lighting conditions severely affect the results.

The Û Component, a simplification of the LUX color space, has been used in paper [45] and [46] for lip segmentation. The equation to calculate the Û component is shown below.

$$\hat{U} = \begin{cases} 256 \times \frac{G}{R} & if\ R > G \\ 255 & otherwise \end{cases} \tag{3-2}$$



(a)



(b)

Figure 3.10: (a) Lip v/s face histogram for $\hat{U}$ componnt. In the figure Red signifies face pixels and green signifies lip pixels (b) Image displayed in $\hat{U}$ component color space for lip enhancement.

One of the most promising results for lip boundary enhancement was observed in the color-based gradients mentioned in the paper [46] and [47] discussed in section 3.2.

3.2. Lip enhancement Techniques

Often times, only color space transformation is not sufficient for lip extraction. When the algorithm tries to extract the lip contour, the gradient information between lips and the other face features can be very useful and can be first used to enhance the lip boundary. There are several techniques used to provide the strong gradient – high variation values between skin and lip regions.

A) Intensity-based Gradient

Intensity-based gradient is the most extensively used gradient, which aims to detect the illumination changes between the skin and lips. If the illumination changes between the skin and lips are enhanced, the lip boundary becomes more obvious to the detection algorithm. For example, if the light source is from above, the upper boundary of the upper lip is bright whereas the upper lip itself is dark, similarly the lower lip is bright but the lower lip boundary is in the dark. Lip boundary enhancement can hence be achieved by utilizing the intensity-based gradient caused by the light source effect on the upper and lower lip. The gradient consists to horizontal and vertical components, but the vertical components are more frequently used. One example is the use of gradient-based Canny edge detector for the mouth corner detection in [48]. The drawbacks consist of the shadow effect generally due to the lower lip which results in false contours. Also, the lower lip contour has a weaker gradient and is more difficult to extract. In cases with the open mouth, the oral cavity, teeth, tongue will make a strong gradient along with the lip contour and the algorithm needs to be able to extract the right contours.

B) Color-based Gradient

Paper [47] proposes two specific color gradients for the outer lip contour extraction whereas in [49] two gradients are proposed for the inner lip contour. The gradient R_{top} in

[47] characterizes the upper boundary and R_{bottom} is used for the lower one. The equations are as follows where I is the luminance

$$R_{top}(x,y) = \nabla\left(\frac{R}{G}(x,y) - I(x,y)\right) \quad \text{and} \quad R_{bottom}(x,y) = \nabla\left(\frac{R}{G}(x,y)\right) \tag{3-3}$$

The reasons behind selecting R/G are mentioned in the Part E of Section III. Similarly, in [49], G_1 is used for the upper inner boundary of the lip and G_2 is for the lower one. In the equation $\widehat{H}$ is pseudo-hue and u is the component of the CIELuv color space.

$$\begin{gathered} G_1(x,y) = \nabla\left(R(x,y) - u(x,y) - \widehat{H}(x,y)\right) \\ G_2(x,y) = \nabla\left(I(x,y) + u(x,y) + \widehat{H}(x,y)\right) \end{gathered} \tag{3-4}$$

Another example of upper, middle and lower lip color-based gradient (Pseudo-Hue and Y component) can be found in [50].

3.3. Thresholding the Lip Gradient

It has been experimented and observed that pseudo hue is higher for lips than for skin; hence, we utilize pseudo hue in our definition of upper and lower lip contour [45]. Intensity is another important factor to be taken into account. Generally, the illumination source is above the speaker. Therefore the top frontier of the upper lip is well illuminated while the lower part of the upper lip is in shadow. For the lower lip, the bottom contour is in light whereas the central lower boundary is in shadow. Thus using, the gradient of the hybrid edges, it is possible to define the contours of the lips.

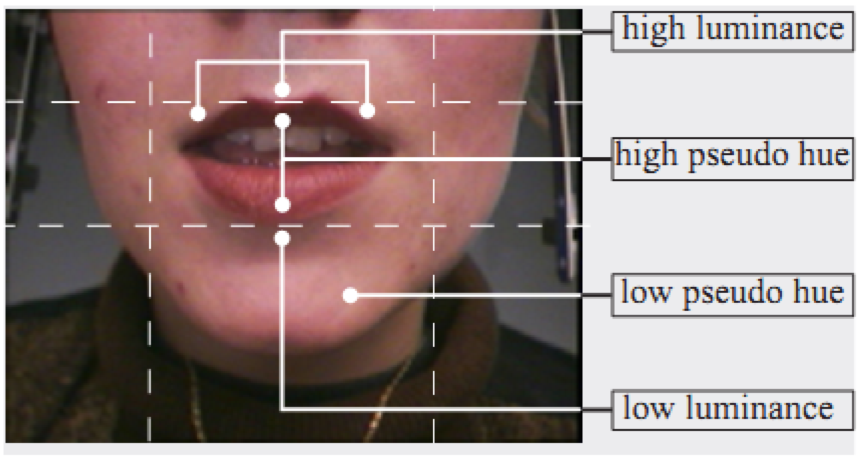


Figure 3.11: Indicate the higher and lower values of pseudo Hue and Luminance for lip boundaries [47]

$$R_{top}(x,y) = \nabla\big(h_N(x,y) - Y_N(x,y)\big)$$
$$R_{low}(x,y) = \nabla\big(h_N(x,y) + Y_N(x,y)\big) \qquad (3\text{-}5)$$

Where h_N is pseudo hue and Y is from the color space YCrCb.

Therefore, the first step in our lip detection implementation is the application of the gradient hybrid equations on the face detected images.

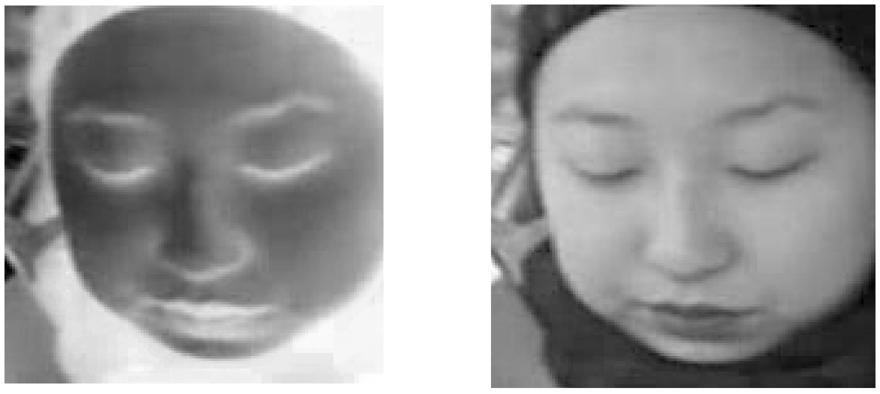


Figure 3.12: (a) Image after application of $h_N(x,y) - Y_N(x,y)$ equation (b) Image after application of $h_N(x,y) + Y_N(x,y)$

We are interested in only the vertical gradient of the hybrid equations. For a function $f(x, y)$, the gradient of f at coordinates (x,y) is defined as the two-dimensional column vector.

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}, \ where\ \frac{\partial f}{\partial y}\ forms\ the\ vertical\ gradient \qquad (3\text{-}6)$$

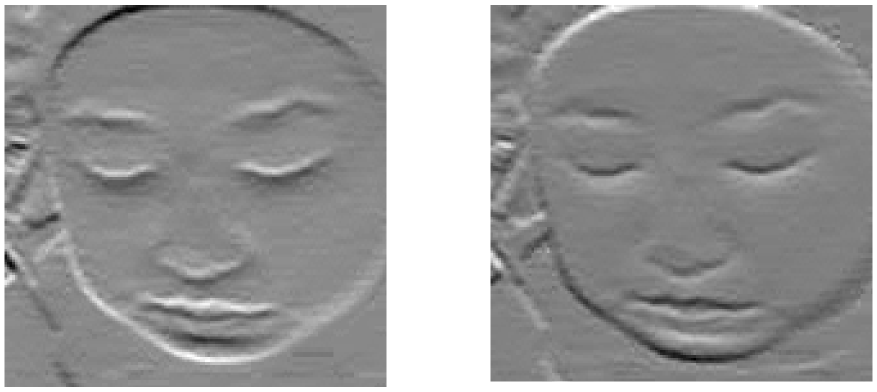

(a) **(b)**

Figure 3.13: (a) Vertical gradient of image in Figure 3.12(a) ($\boldsymbol{R_{top}(x, y)}$) and (b) Vertical gradient of image in Figure 3.12(b) ($\boldsymbol{R_{low}(x, y)}$)

As depicted by the images, using the hybrid lip contour equations, the upper and lower lip are highlighted in comparison to the surrounding area of the mouth. The next step involves thresholding to isolate the mouth contour from the rest of the skin. With optimization, the threshold is set as below.

$$\begin{aligned} &If\ R_{top}(x, y) > 0.028,\ R_{top}(x, y) = 1; \text{ otherwise, Rtop (x,y) = 0} \\ &If\ R_{low}(x, y) > 0.03,\ R_{low}(x, y) = 1; \text{ otheriwise, Rlow(x,y) = 0} \end{aligned} \qquad (3\text{-}7)$$

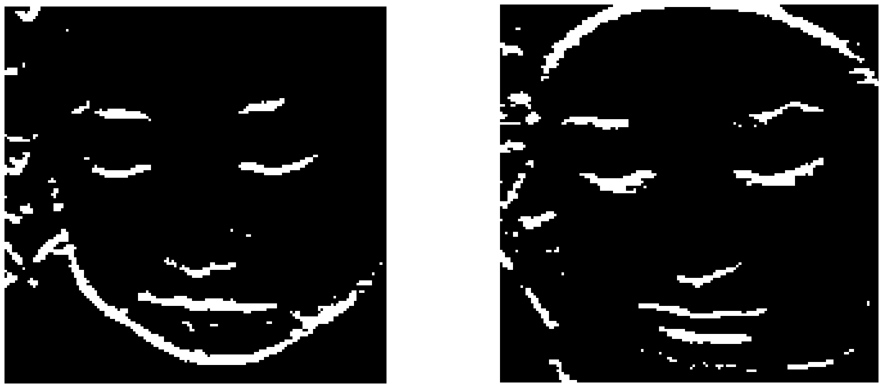

(a) **(b)**

Figure 3.14: (a) and (b) Images obtained after thresholding images in Figure 3.13(a) and (b)

3.4. Lip Localization

Once the thresholded images are obtained, we observe several connected components or blobs. We need to differentiate between the blobs belonging to lips from the ones belonging to the face. We draw a bounding box around all blobs and pass then through certain restrictions.

a) Orientation: Assuming the lips are primarily horizontal, or slightly inclined if considering a tilted face, we apply the threshold limits to the orientation. This is successful in eliminating any vertically inclined blobs. Orientation is calculated by measuring the angle between the x-axis and the major axis that has the same second moments as the origin.

$$-18° \leq Orientation \leq +18°$$

b) Blob Area: It was determined through data that the lip area is approximately 2% of the entire face. Hence to eliminate larger blobs, we apply a threshold restriction. We also apply a lower area limit to eliminate small blobs.

$$0.02 < \frac{Blob\ Area}{Total\ Area} < 0.002$$

c) Height to width: following through the assumption that the lips are primarily horizontal, we restrict the blobs which have the height greater than the width.

$$Height\ of\ the\ Bounding\ Box < Width\ of\ the\ Bounding\ Box$$

d) Location Constraints: This restriction is applicable subjected to satisfactory face detection. We estimate the position of lips in the image firstly by limiting the blobs only in the lower half of the image, and secondly by rejecting the blobs in the lower 1/8th of the image. This successfully eliminates the cases where we can mistake eye boundaries or then chin boundary as lips.

e) Overlapping Threshold: Lips are normally symmetrical along its vertical bisector. Therefore, we use the overlap threshold to eliminate random blobs which don't satisfy the overlap criteria.

To determine the percent of overlap, we first create a mirror image of the cluster in the bounding box. We then add the original and the mirror image. Now, we consider only the overlapped region between the original and mirrored cluster and divide it by the total area of the bounding box. Therefore, the higher the value of the division is, the higher the overlap percent is. The values for the thresholds were obtained through experimentation and may need to be varied for a different test set.

$$0.08 < Overlap\ Threshold < 0.7$$

Based on these restrictions, we finally obtain our vertical gradient hybrid images of lip contours. The next step is to add the two hybrid images.

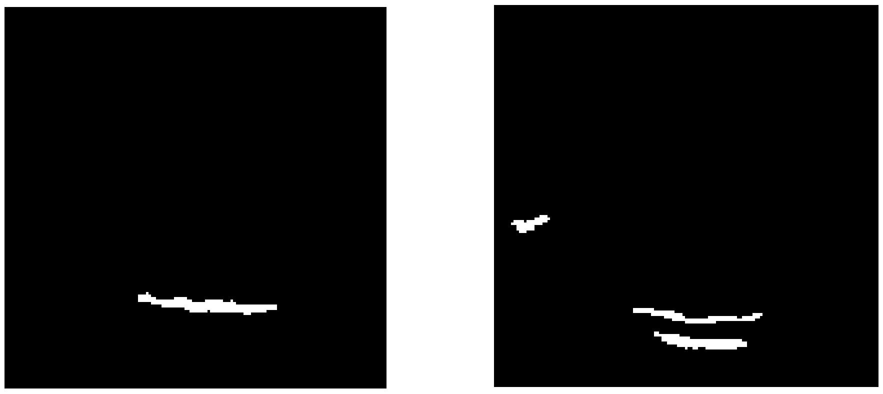

(a) (b)

Figure 3.15: (a) Upper lip boundary after restrictions. (b) Lower lip boundary after restrictions.

In the Figure 3.15, it can be observed that (a) preserved the boundary of the upper lip and the (b) preserved the contour of the lower lip along with the lower boundary of the upper lip. We perform addition on (a) and (b) in order to complete the contour of lips as shown in Figure 3.16 (a). At this stage, in order to achieve a blob comprising of the entire mouth, we can perform closing. Through experimentation it was concluded that a disk shaped structuring element gave best results.

Assuming the restrictions mentioned above were successful in eliminating blobs other than lips, the only blob present should be of the lips. In cases where small clusters passed through all restrictions, we detect the largest blob as the mouth region.

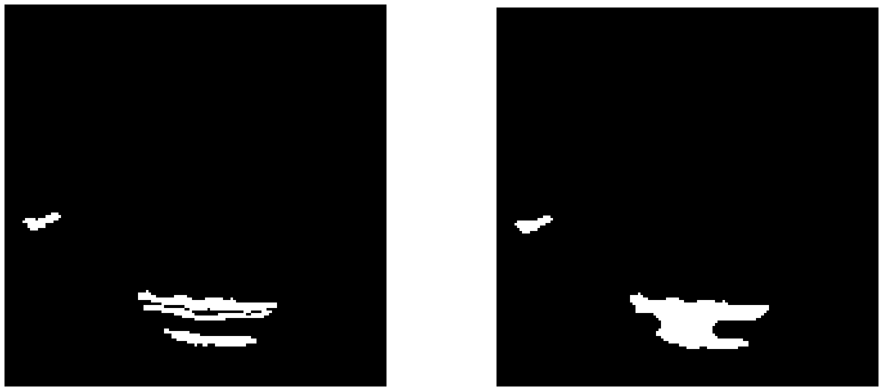

(a) (b)

Figure 3.16: (a) Addition of images (a) and (b) in Figure 42 (b) Image after performing closing.

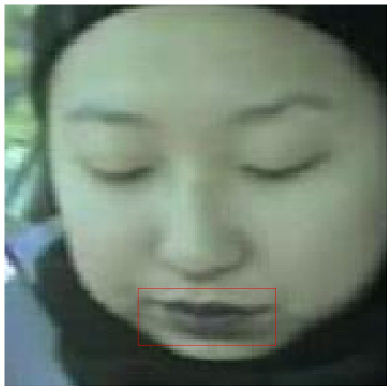


Figure 3.17: Correctly Detected Lip Region

3.5. Lip Detection Results

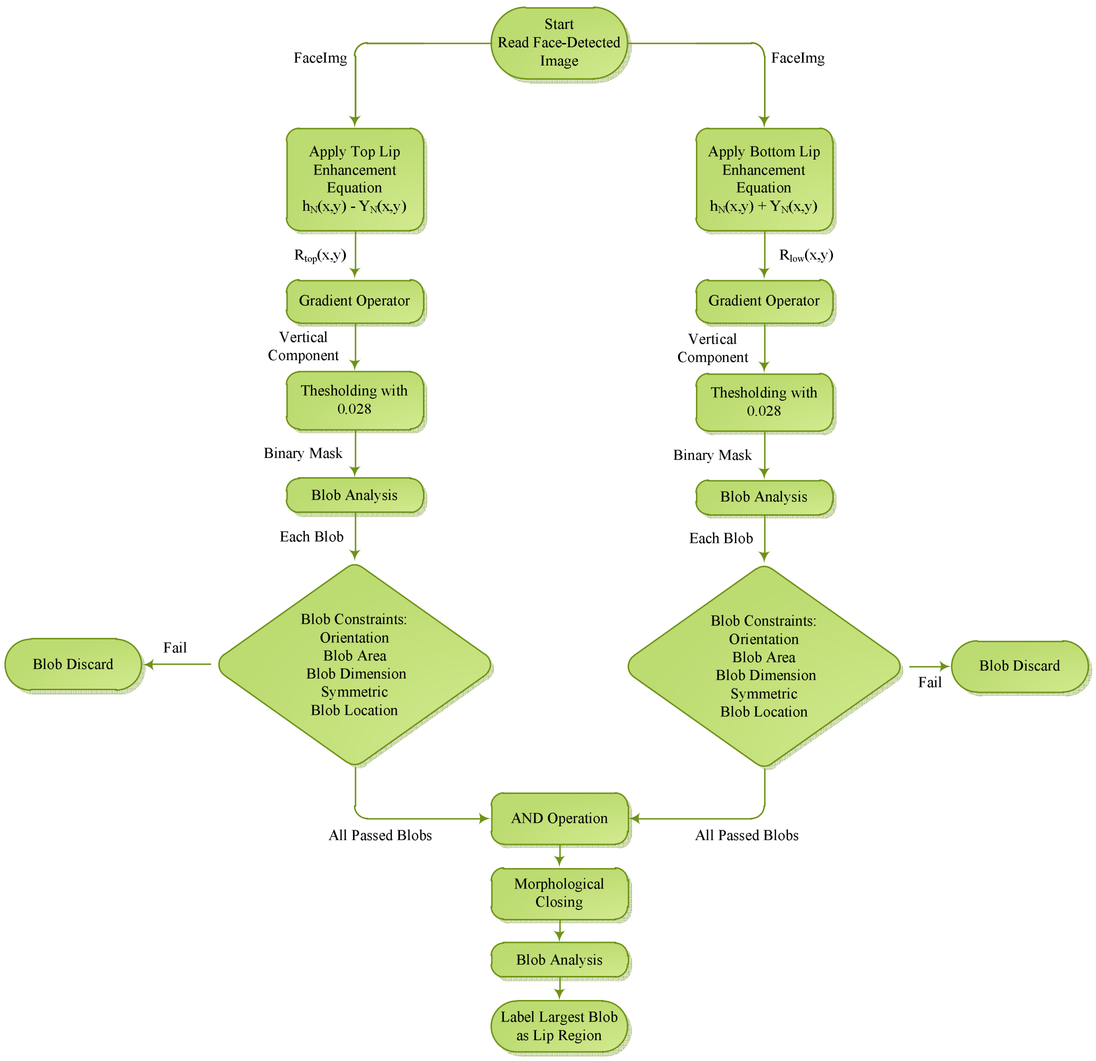


Figure 3.18: Flowchart representing the lip segmentation algorithm

Figure 3.18 summarizes the lip localization algorithm. Figure 3.19 (a) and (c) displays the correctly classified lip regions, whereas Figure 3.19 (b) and (d) shows images where the lip region detected was incomplete.

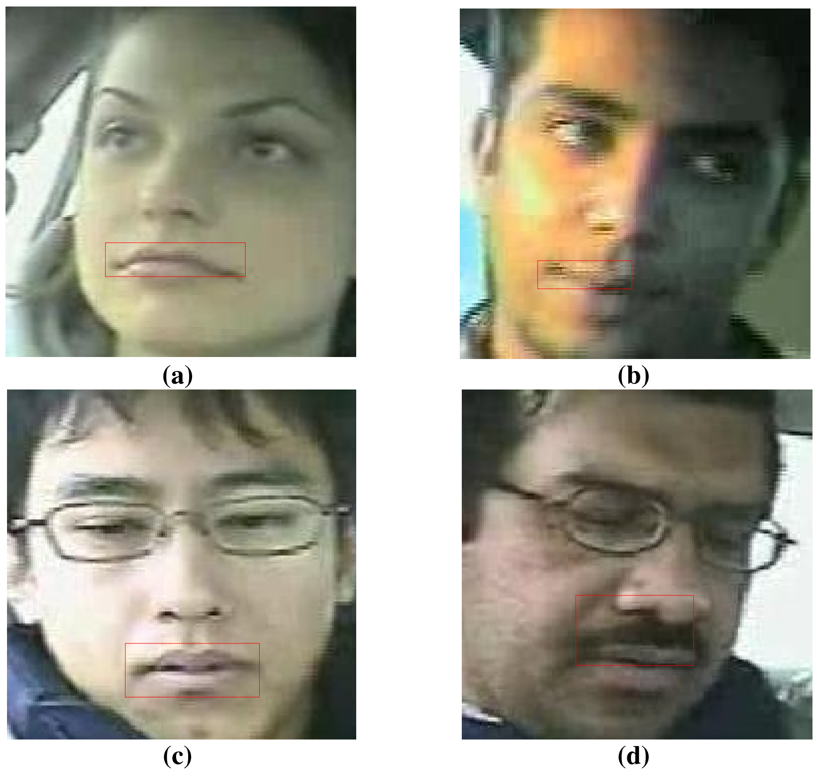

(a) (b) (c) (d)

Figure 3.19: (a) and (c) Correctly detected lips (b) and (d) Incomplete lips detected

Similar to face detection, we implement Viola-Jones in-built classifier for mouth detection on our test set of 169 images. We also train a classifier in gray scale using the training images shown in Figure 3.20.

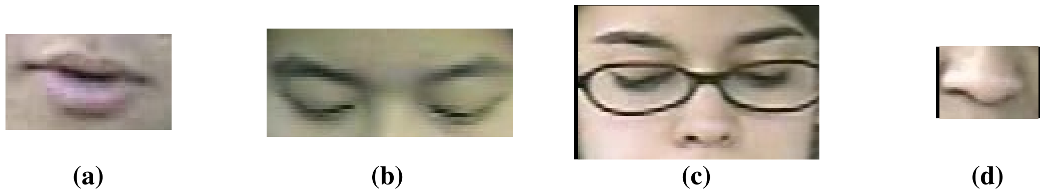

(a) (b) (c) (d)

Figure 3.20: (a) example of the images used for training of the Viola-Jones classifier for lip detection (a) positive training image (b), (c) and (d) negative training image

.

The negative images we collected consisted of parts of the face other than the mouth area. For the positive training data we collected images only around the mouth making sure to avoid the chin of the nose region. We used 7080 positive images and 6539 negative images for the training.

We recorded and compared the results of lip localization from the in-built Viola-Jones classifier and the classifier trained by us to the Gradient thresholding lip segmentation technique described in this paper.

The test set contained 169 images of faces. These face images were the successful output of the face detection algorithm. We ran this test set through the lip localization method proposed in this thesis, the in-built Viola-Jones mouth detection algorithm and the gray-scale trained Viola-Jones. The classifiers were obtained in the same way as the face detection Viola-Jones algorithm. The Gray Scale classifier we trained also had unaltered parameters except the window size was defined as 20 X 20.

The classifier we trained gave results much worse than expected. We attained detection rates of only 45% which was much lower than the other methods. The in-built classifier and our gradient thresholding lip segmentation methods results were similar, although the in-built classifier had several misclassifications. It is important to note, that the input image to the Viola-Jones classifiers (both Gray Scale Trained and In-built gray scale) were only lower half of the face detected image. This was done to limit the mis-classifications by the classifier. For input as the whole image in almost all cases the eyes were detected as mouth.

Table 3-1: Comparison of Lip Detection results of 169 Images

	Color space	**Detected**	**Incomplete Lips**	**Missed**	**False Detection**	**Percent Faces Detected**
Viola Jones	Gray Scale Trained	77	12	80	8	45.56%
Viola Jones	In built gray scale	146	1	22	35	86.39%
Gradient Thresholding	Pseudo Hue and Y	149	4	16	0	88.16%

The gradient thresholding lip localization method proposed gave accuracy of about 88% which was also higher than the Viola-Jones' mouth detection algorithm. The gradient thresholding technique works well with our dataset but may not be universal. Therefore, although a crude method, it attains impressive results and can be utilized with modification for a more robust algorithm.

4. CONCLUSION AND FUTURE WORK

4.1. System Performance

The object detection technique proposed by Viola-Jones is widely accepted as one of the most robust face detection algorithms. In the car environment of our images, with different lighting conditions and poor image quality in certain cases, the detection result was 90%, which leaves room for improvement.

The skin detection using Bayes classifier followed by template matching comes close to the detection rates of Viola-Jones. It had become apparent that a single method using only one type of information from the image may not be sufficient to correctly classify faces from the background. By means of simple concatenation of the three different detection techniques we have reached detection rates of 95%, which results in higher probability of better lip segmentation.

Table 4-1: Summary of the Face Detection Algorithms implemented with 181 face test images

Face Detection Technique	Detected Faces	Incomplete Faces	Missed Faces	Mis-classified Faces	Percentage Detected	Percentage Failed
Viola-Jones	164	5	12	1	90.6077%	9.3923%
Bayes Classifier + Golden Ratio	71	N/A	110	N/A	39.22%	60.773%
Bayes Classifier + Golden Ratio + Morphological Operations	119	N/A	62	N/A	65.745%	34.254%
Template Matching	135	36	0	10	74.585%	25.414%
Bayes Classifier +Morphological Operations +Template Matching	160	5	16	0	88.39%	11.60%
Final Face Detector	**172**	**9**	**0**	**0**	**95.027**	**4.972**

For the case of lip segmentation, we use a simplistic approach of enhancing the lip contours and isolating them based on the feature information known on the face. It also seemed to slightly surpass the in-built Viola-Jones classifier, which might indicate extracting color information from the image for lip localization might be essential along with other features.

4.2. System Limitation and Suggestions

4.2.1. Face Detection

One of the biggest drawbacks that can be noted in the concatenated face detection technique is that only missed faces are considered for the next stage. In the case of incomplete faces that were detected, they do not get rectified. There is no straightforward way to avoid such a situation. One possibility for future research is to experiment with different concatenation possibilities.

4.2.2. Modification Viola-Jones Classifier

Through the test results of red and green color space trained classifier it could be observed that color spaces other than gray could also be experimented with. As the principle of Viola-Jones is to use Haar like features to calculate the values across the face, a color space which gives a higher contrast to the face image is desired. We have experimented with only a few of them in this thesis. Color spaces which were good for skin detection through a classifier seem to give terrible results, as all features are lost and the entire region combines to form one color blob. The color space analysis performed in

2.2.1 can be an indicator as to which color spaces might not work well and which should be considered.

Other factors, that can be considered, are the change in the size of detector window for different color spaces and the implementation of illumination compensation.

4.2.3. Lip Localization

Our lip localization technique works well with our dataset, but with the inclusion of hard thresholds on the gradient of the hybrid equation, these values may not be applicable universally. One of the methods to eliminate hard thresholding is to evaluate that generally in what range of intensities for that image does the lip region fall. For example, if the lips area falls in the top 10% of the intensities, the threshold need not be hard for all images but is constantly modified per image.

Also the last step of lip localization assumed the largest blob as lips, which is a rudimentary technique and leaves a large area for error. Hence, integrating it with a technique which uses lip shape information such as parametric models might improve accuracy and may allow this technique to be extended to other data sets.

REFERENCES

[1]. Lee, B., Hasegawa-Johnson, M., Goudeseune, C., Kamdar, S., Borys, S., Liu, M., et al. “AVICAR: Audio-visual speech corpus in a car environment.” Paper presented at the Conf. Spoken Language, 2004

[2]. Petajan, Eric D. "Automatic Lipreading to Enhance Speech Recognition." in Proc IEEE Conf Computer Vision and Pattern Recognition,1985, pp. 40-47.

[3]. Aleksic, Petar S., and Katsaggelos, Aggelos K., "Lip Feature Extraction and Feature Evaluation in the Context of Speech and Speaker Recognition." Visual Speech Recognition: Lip Segmentation and Mapping. Hershey PA, USA: IGI Global, 2009. 39-69

[4]. Pham, Thanh Trung; Song, Min Gyu; Kim, Jin Young; Na, Seung You; Hwang, Sung Taek; , "A Robust Lip Center Detection in Cell Phone Environment," Signal Processing and Information Technology, 2008. ISSPIT 2008. IEEE International Symposium on , vol., no., pp.390-395, 16-19 Dec. 2008

[5]. Free-Video-to-JPG-Convertor. Jan 2011. Available Online: http://download.cnet.com/Free-Video-to-JPG-Converter/3000-2194_4-10764760.html

[6]. Ming-Hsuan Yang; Kriegman, D.J.; Ahuja, N.; , "Detecting faces in images: a survey," Pattern Analysis and Machine Intelligence, IEEE Transactions on , vol.24, no.1, pp.34-58, Jan 2002

[7]. Yang, Ming-Hsuan; Kriegman, D.J.; Ahuja, N.; , "Detecting faces in images: a survey," Pattern Analysis and Machine Intelligence, IEEE Transactions on , vol.24, no.1, pp.34-58, Jan 2002

[8]. Hjelmas, E. and Low, B. K.. Face detection: A Survey. Computer. Vis. Image Understand. 83, 236–274, 2001

[9]. Hsu, Rein-Lien; Abdel-Mottaleb, M.; Jain, A.K.; , "Face detection in color images," Pattern Analysis and Machine Intelligence, IEEE Transactions on , vol.24, no.5, pp.696-706, May 2002

[10]. Zhao, W, Chellappa, R, Rosenfeld, A, and Phillips, P. J., ªFace Recognition: A Literature Survey,º UMD CfAR Technical Report CAR-TR-948, 2000.

[11]. Ismail, N.,"Review of Existing Algorithms for Face Detection and Recognition," 8th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, December 14-16,2009,pp.30-39.

[12]. Zhang, X., Gao, Y., “Face recognition across pose: A review”, Pattern Recognition 42 (2009) 2876 -2896.

[13]. Wang, H., Wang, Y., Cao, Y.: “Video-based face recognition: A survey”. World Academy of Science, Engineering and Technology 60 (2009)

[14]. Paul Viola, Michael J. Jones, Robust Real-Time Face Detection, International Journal of Computer Vision 57(2), 2004.

[15]. R. Hursig, “Robust Unconstrained Face Detection and lip localization algorithm using Gabor Filters,” Master’s Thesis, California Polytechnic State University, 2009.

[16]. Liew, Alan W., and Wang, Shilin. "Lip Modelling and Segmentation." Visual Speech Recognition: Lip Segmentation and Mapping. Hershey PA, USA: IGI Global, 2009. 70-127

[17]. Kass, M., Witkin, A., and Terzopoulos, D., “Snakes: Active contour models”, International Journal of Computer Vision, pp.321–331, Oct. 1987.

[18]. Eveno, N.; Caplier, A.; Coulon, P.-Y.; , "A parametric model for realistic lip segmentation," Control, Automation, Robotics and Vision, 2002. ICARCV 2002. 7th International Conference on , vol.3, no., pp. 1426- 1431 vol.3, 2-5 Dec. 2002

[19]. Bouvier, C.; Coulon, P.-Y.; Maldague, X.; , "Unsupervised Lips Segmentation Based on ROI Optimisation and Parametric Model," Image Processing, 2007. ICIP 2007. IEEE International Conference on , vol.4, no., pp.IV-301-IV-304, Sept. 16 2007-Oct. 19 2007

[20]. Cootes, T. F.. “Statistical models of appearance for computer vision”. Online technical report available from http://www.isbe.man.ac.uk/˜bim/refs.html, Sept. 2001.

[21]. Gacon, P.; Coulon, P.-Y.; Bailly, G.; , "Statistical active model for mouth components segmentation," Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on , vol.2, no., pp. ii/1021- ii/1024 Vol. 2, 18-23 March 2005

[22]. Zhang, X.; Mersereau, R.M.; , "Lip feature extraction towards an automatic speechreading system ," Image Processing, 2000. Proceedings. 2000 International Conference on , vol.3, no., pp.226-229 vol.3, 2000

[23]. Dargham, J.A.; Chekima, A.; , "Lips Detection in the Normalised RGB Colour Scheme," Information and Communication Technologies, 2006. ICTTA '06. 2nd , vol.1, no., pp.1546-1551

[24]. Sadeghi, M.; Kittler, J.; Messer, K.; , "Modelling and segmentation of lip area in face images," Vision, Image and Signal Processing, IEE Proceedings - , vol.149, no.3, pp. 179- 184, Jun 2002

[25]. Leung, Shu-Hung; Wang, Shi-Lin; Lau, Wing-Hong; , "Lip image segmentation using fuzzy clustering incorporating an elliptic shape function," Image Processing, IEEE Transactions on , vol.13, no.1, pp.51-62, Jan. 2004

[26]. Delmas, P., Eveno, N., and Lievin, M., “Towards Robust Lip Tracking”, International Conference on Pattern Recognition (ICPR’02), Quebec City, Canada, August 2002

[27]. Liew, A.W.C.; Leung, S.H.; Lau, W.H.; , "Lip contour extraction using a deformable model," Image Processing, 2000. Proceedings. 2000 International Conference on , vol.2, no., pp.255-258 vol.2, 10-13 Sept. 2000

[28]. Aravabhumi, V.R.; Chenna, R.R.; Reddy, K.U.; , "Robust method to identify the speaker using lip motion features," Mechanical and Electrical Technology (ICMET), 2010 2nd International Conference on, vol., no., pp.125-129, 10-12 Sept. 2010

[29]. Jae-Ung Yun; Hyung-Jin Lee; Paul, A.K.; Joong-Hwan Baek; , "Robust Face Detection for Video Summary Using Illumination-Compensation and Morphological Processing," Natural Computation, 2007. ICNC 2007. Third International Conference on , vol.2, no., pp.710-714, 24-27 Aug. 2007

[30]. mcvai-tracking: Face detection, recognition and tracking. 2011. http://code.google.com/p/mcvai-tracking/

[31]. R.C. Gonzalez, R.E. Woods, Digital Image Processing Third Edition.. Upper

[32]. Saddle River, NJ: Pearson Prentice Hall, 2008, pp. 402-3,410, 633-5.

[33]. Ford, A., Roberts A. “Colour Space Conversions” (Tech. Rep.). 1998. Available Online: http://www.poynton.com/PDFs/coloureq.pdf

[34]. The MathWorks, Matlab Image Processing Toolbox, <http://www.mathworks.com/access/helpdesk/help/toolbox/images/index.html?/access/helpdesk/help/toolbox/images/index.html>, July 3, 2009.

[35]. Kakumanu, P., Makrogiannis S., Bourbakis, N. “Skin-color modeling and detection methods”, Pattern Recognition 40, pp 1106-1122, 2007. Available at www.sciencedirect.com.

[36]. Duda, Richard O., Peter E. Hart, and David G. Stork. "Bayesian Decision Theory." Pattern Classification. Canada: John Wiley, 2001. 20-83

[37]. Serrano, Santiago. "Eigen Face Matlab Code." Drexel University. N.p., n.d. Web. Mar. 2011. <http://www.pages.drexel.edu/~sis26/Eigencode.htm>.

[38]. Jensen, O. “Implmenting the Viola-Jones Face Detection Algorithm,” Technical University of Denmark, 2008.

[39]. Barnes, David. "OpenCV HaarTraining." Quotient Robotics. N.p., 8 Apr. 2008. Web. Feb. 2011. <http://www.quotientrobotics.com/2010/04/opencv-haartraining-object-detection.html>.

[40]. Sadeghi, M.; Kittler, J.; Messer, K.; , "Modelling and segmentation of lip area in face images," Vision, Image and Signal Processing, IEE Proceedings - , vol.149, no.3, pp. 179- 184, Jun 2002

[41]. Chin, Siew Wen; Seng, Kah Phooi; Ang, Li-Minn; Lim, King Hann; , "Improved Watershed Lips Detection and Modified H8 Tracking System Based on Lyapunov Stability Theory," Intelligent Human-Machine Systems and Cybernetics, 2009. IHMSC '09. International Conference on , vol.2, no., pp.355-358, 26-27 Aug. 2009

[42]. Hulbert, A., Poggio, T., "Synthesizing a Colour Algorithm From Examples", Science, Vol 239, pp 482-485, 1998

[43]. Kakumanu, P., Makrogiannis, S., and Bourbakis, N.. "A survey of skin-color modeling and detection methods". Pattern Recogn., 40(3):1106-1122, 2007.

[44]. Chin, Siew Wen; Seng, Kah Phooi; Ang, Li-Minn; Lim, King Hann; , "Improved Watershed Lips Detection and Modified H8 Tracking System Based on Lyapunov Stability Theory," Intelligent Human-Machine Systems and Cybernetics, 2009. IHMSC '09. International Conference on , vol.2, no., pp.355-358, 26-27 Aug. 2009

[45]. Kim, Young-Un; Kang, Sun-Kyung; Jung, Sung-Tae; , "Design and implementation of a lip reading system in smart phone environment," Information Reuse & Integration, 2009. IRI '09. IEEE International Conference on , vol., no., pp.101-104, 10-12 Aug. 2009

[46]. Bruce, J.; Balch, T.; Veloso, M.; , "Fast and inexpensive color image segmentation for interactive robots," Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on , vol.3, no., pp.2061-2066 vol.3, 2000

[47]. Eveno, N.; Caplier, A.; Coulon, P.-Y.; , "Accurate and quasi-automatic lip tracking," Circuits and Systems for Video Technology, IEEE Transactions on , vol.14, no.5, pp. 706- 715, May 2004

[48]. Ouyang, H. and Lee, T., "A new lip feature representation method for video-based bimodal authentication," in MMUI '05: Proceedings of the 2005 NICTA-HCSNet Multimodal User Interaction Workshop. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2006, pp. 33–37.

[49]. Stillittano, S. and Caplier,, A. "Inner lip segmentation by combining active contours and parametric models," in Proceedings of the 3rd International Conference on Computer Vision Theory and Applications (VISAPP '08), Madeira, Portugal, January 2008

[50]. Eveno, N.; Caplier, A.; Coulon, P.-Y.; , "A parametric model for realistic lip segmentation," Control, Automation, Robotics and Vision, 2002. ICARCV 2002. 7th International Conference on , vol.3, no., pp. 1426- 1431 vol.3, 2-5 Dec. 2002

APPENDIX A: PROJECT ALGORITHM

A.1: Color Space Anlaysis (1D and 2D)

```
function y = MakeMask (directory)
%-----------------------------------------------------------------------
-
%PURPOSE:   This function is used to create masks (binary images) for
%           face or lip in color space analysis or trainning
%INPUT:     The directory where the original JPEG images are (for
example,
%           image\)
%OUTPUT:    Masks BMP images into the folder \masks
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-

    files = dir ( [directory '*.jpg'] );
    mkdir('masks');

    for n = 1:1:length(files)
        disp(['Images:' int2str(n) '/' int2str(length(files))])
        img = imread ( [directory files(n).name] );

        figure(1), imshow(img);
        FaceNum = input('How many faces are there?');

        masktot = logical(zeros(size(img(:,:,1))));
        for m = 1:1:FaceNum
            mask = roipoly (img);
            img(mask)=0;
            masktot = masktot | mask;
        end


        for c = length(files(n).name):-1:1
            char = files(n).name(c);
            files(n).name(c) = [];

            if char == '.'
                break;
            end
        end

        imwrite (masktot,[directory 'masks/mask_' files(n).name
'.bmp'],...

'BMP');
    end
    y = 1;
end
```

```
%-----------------------------------------------------------------------
-
%PURPOSE:   This code performs the color space analysis, plotting the
%           histogram between skin and nonskin
%REQUIREMENT: The images are stored in the folder 'images' and the
%           associated masks are created and stored in the subfolder
%           'masks'
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-
clear all
filename = dir ('images\*.jpg');

% This loop takes away the file extension of the filename.
for n = 1:1:length(filename)
    for c = length(filename(n).name):-1:1
        char = filename(n).name(c);
        filename(n).name(c) = [];

        if char == '.'
            break;
        end
    end
end

SkinArray = [];
NonSkinArray = [];

% This loop create the overall histogram arrays
for n = 1:1:length(filename)

    disp (['Image: ' num2str(n) '/' num2str(length(filename))]);
    imgrgb = imread (['images\' filename(n).name '.jpg']);

    imgmsk = imread (['images\masks\mask_' filename(n).name '.bmp']);

    % Shifted Hue Space
    SkinArrayH = FindHueHalfArray(imgrgb,imgmsk);
    NonSkinArrayH = FindHueHalfArray (imgrgb,~imgmsk);
    SkinArray = [SkinArray SkinArrayH'];
    NonSkinArray = [NonSkinArray NonSkinArrayH'];

end

doubleHist(SkinArray,NonSkinArray,'Shifted Hue',0,1);
```

```
function ResultArray = FindHueHalfArray (img1, img1msk)
%-----------------------------------------------------------------------
-
%PURPOSE:   This function finds the shifted hue color space
%INPUT:     The image and the associated mask (skin vs. nonskin)
%OUTPUT:    A array containing the shifted hue values for each pixels
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
% Resize the image
imga = imresize(img1,0.05);
imgb = imresize(img1msk,0.05);

% Find the Hue Space Array
hsvdata = rgb2hsv(imga);
imgahue = hsvdata (:,:,1);
hueofface = imgahue(imgb);

 % Shift to the right by 0.2.
        imghueshift = hueofface + 0.2;
        imghueshift (find(imghueshift > 1)) = ...
                            imghueshift (find(imghueshift > 1)) - 1;

ResultArray = imghueshift;

end

function y = doubleHist(img1, img2, colspa, min, max)
%-----------------------------------------------------------------------
%PURPOSE:   This function plot two histograms into the same plot.
%INPUT:     Image #1 (Skin) and Image #2 (Non-Skin)
%           colspa: the name for the color space
%           min:    minimum x-axis value
%           max:    maximum x-axis value
%OUTPUT:    A figure with both histograms
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-

[c1,x1] = imhist(img1);
[c2,x2] = imhist(img2);

figure(1)
stem (x2, c2, 'r','MarkerSize',1);
hold on;
stem (x1, c1, 'g','MarkerSize',1);
hold off;
xlim([min max])
title(colspa)
grid

y=1;

end
```

```
%-----------------------------------------------------------------------
-
%PURPOSE:    This code performs the color space analysis, plotting the
%            2D scatter plot between skin and nonskin
%REQUIREMENT: The images are stored in the folder 'images' and the
%            associated masks are created and stored in the subfolder
%            'masks'
%AUTHOR:     Benafsh Husain, Cal Poly San Luis Obispo
%DATE:       August 1, 2011
%-----------------------------------------------------------------------
-
clear all
filename = dir ('images\*.jpg');

% This loop takes away the file extension of the filename.
for n = 1:1:length(filename)
    for c = length(filename(n).name):-1:1
        char = filename(n).name(c);
        filename(n).name(c) = [];

        if char == '.'
            break;
        end
    end
end

ArraySkinCoSp1 = [];
ArrayNonSkinCoSp1 = [];
ArraySkinCoSp2 = [];
ArrayNonSkinCoSp2 =[];

% This loop create the overall histogram arrays
for n = 1:1:length(filename)

    disp (['Image: ' num2str(n) '/' num2str(length(filename))]);
    imgrgb = imread (['images\' filename(n).name '.jpg']);

    imgmsk = imread (['images\masks\mask_' filename(n).name '.bmp']);

    % Shifted Hue Space
    SkinArrayH = FindHueHalfArray(imgrgb,imgmsk);
    NonSkinArrayH = FindHueHalfArray(imgrgb,~imgmsk);
    ArraySkinCoSp1 = [ArraySkinCoSp1 SkinArrayH'];
    ArrayNonSkinCoSp1 = [ArrayNonSkinCoSp1 NonSkinArrayH'];

    % Normalized Green Space
    SkinArrayNG = FindNormGArray(imgrgb,imgmsk);
    NonSkinArrayNG = FindNormGArray(imgrgb,~imgmsk);
    ArraySkinCoSp2 = [ArraySkinCoSp2 SkinArrayNG'];
    ArrayNonSkinCoSp2 = [ArrayNonSkinCoSp2 NonSkinArrayNG'];

end

SkinArray = [ArraySkinCoSp1; double(ArraySkinCoSp2)];
NonSkinArray = [ArrayNonSkinCoSp1; double(ArrayNonSkinCoSp2)];
```

```
% Plot the sample
figure(1)
plot(ArrayNonSkinCoSp1,double(ArrayNonSkinCoSp2),'.r','MarkerSize',4)
hold on;
plot(ArraySkinCoSp1,double(ArraySkinCoSp2),'.g','MarkerSize',4)
hold off;
title(['Two Dimension Feature Space']);
xlabel ('Shifted Hue Feature');
ylabel ('Normalized Green Feature');
grid
```

```
function ResultArray = FindNormGArray (img1, img1msk)
%----------------------------------------------------------------------
-
%PURPOSE:   This function finds the normalized green color space
%INPUT:     The image and the associated mask (skin vs. nonskin)
%OUTPUT:    A array containing the normalized green values for each
pixels
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------
imgnrgb = Createnrgb(img1);

imga = imresize(imgnrgb,0.05);
imgb = imresize(img1msk,0.05);

% Find the Normalized Green Space Array
imgnormg = imga(:,:,2);
imgng = imgnormg(imgb);

ResultArray = imgng;

end

function imgnrgb = Createnrgb (img)
%----------------------------------------------------------------------
-
%PURPOSE:   This function takes in an RGB image and create a normalized
%           RGB image as the output.
%INPUT:     RGB Image
%OUTPUT:    Normalized RGB Image
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------
% Red Space
imgr = img(:,:,1);

% Green Space
imgg = img(:,:,2);

% Blue Space
imgb = img(:,:,3);

% To normalized rgb space.
rgbsum = double(imgr) + double(imgg) + double(imgb);
rgbsum(rgbsum == 0) = 1;

imgnr = uint8 (double(imgr)./ rgbsum * 255);
imgng = uint8 (double(imgg)./ rgbsum * 255);
imgnb = uint8 (double(imgb)./ rgbsum * 255);

% Construct the final color image
imgnrgb = cat(3, imgnr, imgng, imgnb);

end
```

A.2: Bayes Classifier Training Stage

```
%----------------------------------------------------------------------
-
%PURPOSE:     This is the m-file which performs MLE for two feature
space
%             case (skin classification).
%PROCEDURE:   It takes the orignal image and its masked file, the
%             parameters for the Gaussian Distribution will then be
%             generated based on all the testing sample.
%REQUIREMENT: The images are stored in the folder 'images' and the
%           associated masks are created and stored in the subfolder
%           'masks'
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------

clear all
filename = dir ('images\*.jpg');

% This loop takes away the file extension of the filename.
for n = 1:1:length(filename)
    for c = length(filename(n).name):-1:1
        char = filename(n).name(c);
        filename(n).name(c) = [];

        if char == '.'
            break;
        end
    end
end

HueArraySkin = [];
HueArrayNonSkin = [];
NormGArraySkin = [];
NormGArrayNonSkin =[];

% This loop create the overall histogram arrays
for n = 1:1:length(filename)

    disp (['image' num2str(n) '/' num2str(length(filename))]);
    imgrgb = imread (['images\' filename(n).name '.jpg']);
%   imghsv = rgb2hsv (imgrgb);
%   imgnrgb = Createnrgb (imgrgb);

    imgmsk = imread (['images\masks\mask_' filename(n).name '.bmp']);

    % Hue Space
    SkinArrayH = FindHueHalfArray(imgrgb,imgmsk);
    NonSkinArrayH = FindHueHalfArray(imgrgb,~imgmsk);
    HueArraySkin = [HueArraySkin SkinArrayH'];
    HueArrayNonSkin = [HueArrayNonSkin NonSkinArrayH'];

    % Normalized Green Space
    SkinArrayNG = FindNormGArray(imgrgb,imgmsk);
    NonSkinArrayNG = FindNormGArray(imgrgb,~imgmsk);
    NormGArraySkin = [NormGArraySkin SkinArrayNG'];
    NormGArrayNonSkin = [NormGArrayNonSkin NonSkinArrayNG'];

end

SkinArray = [HueArraySkin; double(NormGArraySkin)];
NonSkinArray = [HueArrayNonSkin; double(NormGArrayNonSkin)];

[ParaWi Paraw_i Parawi0] = TwoDMLEstimation(NonSkinArray',SkinArray');

SumPix = numel(HueArraySkin)+ numel(HueArrayNonSkin)+...

numel(NormGArraySkin)+numel(NormGArrayNonSkin);

SkinProb = double(numel(HueArraySkin)+ numel(NormGArraySkin))...

/double(SumPix);
NonSkinProb = 1 - SkinProb;
```

```
function [ParaWi Paraw_i Parawi0] = TwoDMLEstimation (r1, r2)

%----------------------------------------------------------------------
-
%PURPOSE:   This function helps to locate the decision boundary for
%           the Two Feature case.
%INPUT:     r1: NonSkin, r2: Skin (D X 2 matrix contains both feature
%           training data
%OUTPUT:    Parameters for the discriminating function
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------
-

% Plot the sample
plot(r1(:,1),r1(:,2),'.g')
hold on;
plot(r2(:,1),r2(:,2),'.r')
hold off;
title(['Two Dimension Feature Space']);
xlabel ('Hue Feature');
ylabel ('Normalized Green Feature');

% For the means and variance using ML estimation
sum1 = [0;0];
sum2 = [0;0];
varsum1 = [0 0;0 0];
varsum2 = [0 0;0 0];

DataLengthr1 = length(r1);
DataLengthr2 = length(r2);

r1 = double(r1);
r2 = double(r2);

for n=1:1:DataLengthr1
    sum1 = sum1 + r1(n,:)';
end
% sum1 = sum(r1(n,:)');

for n=1:1:DataLengthr2
    sum2 = sum2 + r2(n,:)';
end
% sum2 = sum(r2(n,:)');

mean1 = sum1 / DataLengthr1;
mean2 = sum2 / DataLengthr2;

for n=1:1:DataLengthr1
    varsum1 = varsum1 + (r1(n,:)' - mean1) * (r1(n,:)' - mean1)';
end
% varsum1 = sum((r1(n,:)' - mean1) * (r1(n,:)' - mean1)');

% Convariance Matrix
varb1 = varsum1 / DataLengthr1;
```

```
% varub1 = varsum1 / (DataLengthr1-1);

for n=1:1:DataLengthr2
    varsum2 = varsum2 + (r2(n,:)' - mean2) * (r2(n,:)' - mean2)';
end
% varsum2 = sum((r2(n,:)' - mean2) * (r2(n,:)' - mean2)');

varb2 = varsum2 / DataLengthr2;
% varub2 = varsum2 / (DataLengthr2-1);

% Calculating the parameters for the discriminant functions
W1 = (-1/2) * (inv(varb1));
w_1 = (inv(varb1)) * mean1;
w10=(-1/2) * (mean1)' * (inv(varb1)) * (mean1) - (1/2) * log
(det(varb1));

W2 = (-1/2) * (inv(varb2));
w_2 = (inv(varb2)) * mean2;
w20=(-1/2) * (mean2)' * (inv(varb2)) * (mean2) - (1/2) * log
(det(varb2));

ParaWi = [W1 W2];
Paraw_i = [w_1 w_2];
Parawi0 = [w10 w20];

end
```

A.3: Face Detection

```
%-----------------------------------------------------------------------
-
%PURPOSE:       This program outputs the face bounding box
%REQUIREMENT:   The candidate BMP images are stored in the folder
%               'FinalTestSetOld'
%OUTPUT:    Results in different detection stage are sorted into
different
%           folders
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-

clear all
mkdir('Gold_rat_fail');
mkdir('Gold_rat_pass');
mkdir('ResComp');
mkdir('ResComp1');
mkdir('Gold_rat_fail_1');
mkdir('Gold_rat_pass_1');
Count = 0;

filename = dir ('FinalTestSetOld\*.bmp');

for n = 1: 1: length(filename)

    disp(['Images: ' int2str(n) '/' int2str(length(filename))])

    inImage = imread (['FinalTestSetOld\' filename(n).name]);

    newI = ColorComp(inImage);

    % 2D SPM Method using Hue and Normalized Green
    % Parameters are calculated using FindTwoDPara.m and
TwoDMLEstimation.m
    ParaWi = [-33.6276,-0.2695,-26.8821,0.0581;-0.2695,-
0.0119,0.0581,...
                                                                   -
0.0273;];
    Paraw_i = [76.1414,-0.9345;2.3888,4.8455;];
    Parawi0 = [-123.8033,-215.5282;];
    SkinProb = 0.242734075770451;

    % Skin Classification
    maskbw = ApplyTwoDClasf(newI,ParaWi,Paraw_i,Parawi0,SkinProb);

    imgray = rgb2gray(newI);
    imgray(~maskbw) = 0;

    imwrite(imgray, ['ResComp\' filename(n).name]);
    % Blob Analysis (Golden Ratio, Passed image to 'Gold_rat_pass'
folder,
    % Failed images to 'Gold_rat_fail' folder.
    imopmsk = (imgray > 0);
    [L num]= bwlabel(imopmsk);

    s  = regionprops(L, 'all');
    bdbox = cat(1, s.BoundingBox);

    imarea =  cat(1,s.Area);
    maxarea = find(imarea == max(imarea));

    ratio = bdbox(maxarea,4)/bdbox(maxarea,3);

    if (ratio > 1.2) && (ratio < 1.78)
        cropim = imcrop(newI,bdbox(maxarea,:));

        if (~isempty(cropim))
            imwrite(cropim, ['Gold_rat_pass\' filename(n).name]);
        else
            temp = zeros(size(imgray));
            imwrite(temp, ['Gold_rat_pass\' filename(n).name]);
        end

    else
        imopmsk = (imgray > 0);
        [L num]= bwlabel(imopmsk);

        s  = regionprops(L, 'all');
        bdbox = cat(1, s.BoundingBox);

        if (~isempty(bdbox))
            imarea =  cat(1,s.Area);
            maxarea = find(imarea == max(imarea));
            cropim = imcrop(newI,bdbox(maxarea,:));
            imwrite(cropim, ['Gold_rat_fail\' filename(n).name]);
        else
            temp = zeros(size(imgray));
            imwrite(temp, ['Gold_rat_fail\' filename(n).name]);
        end
    end

end

% Process the images in 'Gold_rat_fail' folder that failed the golden
% ratio test.

filefail = dir ('Gold_rat_fail\*.bmp');

for m = 1: 1: length(filefail)

    disp(['Images: ' int2str(m) '/' int2str(length(filefail))])

    inImg = imread (['ResComp\' filefail(m).name]);
    % Morphological Operation for face detection
    se = strel('disk',10);
    imgcl = imclose(inImg,se);

    se1 = strel('line',10,90);
    imgero = imerode(imgcl,se1);

     se2 = strel('line',10,90);
    imgop = imopen(imgero,se2);

    imwrite(imgop, ['ResComp1\' filefail(m).name]);

    imopmsk1 = (imgop > 0);
    [L1 num1]= bwlabel(imopmsk1);

    s1  = regionprops(L1, 'all');
    bdbox1 = cat(1, s1.BoundingBox);

    imarea1 =  cat(1,s1.Area);
    maxarea1 = find(imarea1 == max(imarea1));

    ratio1 = bdbox1(maxarea1,4)/bdbox1(maxarea1,3);

    inImage = imread (['FinalTestSetOld\' filefail(m).name]);

    % Another golden ratio test (after morphological operation)
    if (ratio1 > 1.0) && (ratio1 < 1.78)
        cropim = imcrop(inImage,bdbox1(maxarea1,:));

        if (~isempty(cropim))
            imwrite(cropim, ['Gold_rat_pass_1\' filefail(m).name]);
        else
            temp = zeros(size(imgray));
            imwrite(temp, ['Gold_rat_pass_1\' filefail(m).name]);
        end

    else
        cropim = imcrop(inImage,bdbox1(maxarea1,:));

        if (~isempty(cropim))
            imwrite(cropim, ['Gold_rat_fail_1\' filefail(m).name]);
        else
            temp = zeros(size(imgray));
            imwrite(temp, ['Gold_rat_fail_1\' filefail(m).name]);
        end
    end

end

% The failed images in 'Gold_rat_fail_1' folder are passed through
% Template Matching Algorithm
imgfilename = dir ('Gold_rat_fail_1\*.jpg');
```

```
for m =1: 1: length(imgfilename)

    disp(['Image: ' int2str(m) '/' int2str(length(imgfilename))]);
    inImage = imread(['Gold_rat_fail_1\' imgfilename(m).name]);

    newI = ColorComp(inImage);

    % Apply the template (the templates are in 'Template' folder)
    % The template is the mean image from the following program:
    % Face recognition by Santiago Serrano
    % http://www.pages.drexel.edu/~sis26/Eigencode.htm
    filename = dir ('Template\*.bmp');
    Result = struct('CCMax',{},'CenX',{},'CenY',{},'Rec',{});

    for n =1: 1: length(filename)

        disp([' Template: ' int2str(n) '/' int2str(length(filename))]);

        Temp =imread(['Template\' filename(n).name]);
        img1 = imresize(newI,.5);
        img2 = rgb2gray(newI);

        Result(n) = MatchTemp(img2,Temp);

    end

    CCMax = cat(1,Result.CCMax);
    SelCCMax = find(CCMax == max(CCMax(:)));
    imshow(inImage),hold on,
    plot(Result(SelCCMax).CenY,Result(SelCCMax).CenX,'x');
    rectangle('Position',Result(SelCCMax).Rec,'EdgeColor','r');
    hold off

    pause(1)

end
```

```
function out = ColorComp(inImage)
%-----------------------------------------------------------------------
-
%PURPOSE:   This function applies illumination compensation.
%INPUT:     RGB image
%OUTPUT:    Illumination compensated Image
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-

        r=inImage(:,:,1);
        g=inImage(:,:,2);
        b=inImage(:,:,3);

        avgR = mean(mean(r));
        avgG = mean(mean(g));
        avgB = mean(mean(b));
        avgRGB = [avgR avgG avgB];
        grayValue = (avgR + avgG + avgB)/3;
        scaleValue = grayValue./avgRGB;

        out = cat(3,round(scaleValue(1)*r),round(scaleValue(2)*g),...
            round(scaleValue(3)*b));

 end

function imgout = ApplyTwoDClasf(img,ParaWi,Paraw_i,Parawi0,SkinProb)
%-----------------------------------------------------------------------
-
%PURPOSE:   This function apply the Bayes classifier to the image.
%INPUT:     RGB image and the Parameters of discrimiant function
%OUTPUT:    Classified Image
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-

    % 1: Skin Pixel and 2: NonSkin Pixel
    W1 = ParaWi(1:2,1:2);
    W2 = ParaWi(1:2,3:4);
    w1 = Paraw_i(:,1);
    w2 = Paraw_i(:,2);
    w10 = Parawi0(1);
    w20 = Parawi0(2);
    NonSkinProb = 1-SkinProb;

    imghsv = rgb2hsv(img);
    imghue = imghsv(:,:,1);

    %Shift to the right by 0.2.
    imghueshift = imghue + 0.2;
    imghueshift(find(imghueshift > 1)) = ...
                                imghueshift(find(imghueshift > 1)) - 1;
    imgnrgb = Createnrgb(img);
    imgng = double(imgnrgb(:,:,2));

    Siz = size(imgng);

    gx1 = zeros(Siz);
    gx2 = zeros(Siz);
    for m = 1:1:Siz(1)
        for z = 1:1:Siz(2)
            x = [imghueshift(m,z);imgng(m,z)];
%             x = [imghue(m,z); imgng(m,z)];
            gx1(m,z) = x'*W1*x + w1'*x + w10 + log(NonSkinProb);
            gx2(m,z) = x'*W2*x + w2'*x + w20 + log(SkinProb);
        end
    end

    % Compute the discrimiant function

    imgout = (gx1 <= gx2);

end

function ResultSt = MatchTemp(img,AvgFaceT)

%----------------------------------------------------------------------
-
%PURPOSE:   This function applies template matching algorithm
%INPUT:     RGB image and the template
%OUTPUT:    The bounding box for the face is shown
%REQUIREMENT:  The face template was created and as the input
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------
-

    imgratio = 3/4;

    if (ceil(size(img,1)*3/4)>= size(img,2))
        LengRS = size(img,2);
        WidRS = ceil(size(img,2)*1/imgratio);
    else
        WidRS = size(img,1);
        LengRS = ceil(size(img,1)*imgratio);
    end

    AvgFaceRS = imresize(AvgFaceT,[WidRS LengRS]);

    Result = struct('CCMax',{},'CenX',{},'CenY',{},'Rec',{});

    for i = 1:1:15

        imtemp = imresize(AvgFaceRS,0.95-0.015*i);

        CC = normxcorr2(imtemp,img);
        crop = imcrop(CC,[ceil(size(imtemp,2)/2)
ceil(size(imtemp,1)/2)...
                                            size(img,2)-1 size(img,1)-
1]);

        crop1 = imcrop(crop,[floor(size(imtemp,2)/2) ...
                floor(size(imtemp,1)/2) size(img,2)-size(imtemp,2)-1
...
                size(img,1)-size(imtemp,1)-1]);
        [Mx My] = find(crop1 == max(max(crop1)));

        MxFinal = Mx + floor(size(imtemp,1)/2)+1;
        MyFinal = My + floor(size(imtemp,2)/2)+1;

        Result(i).CCMax = max(crop1(:));
        Result(i).CenX = MxFinal;
        Result(i).CenY = MyFinal;
        Result(i).Rec = [MyFinal-floor(size(imtemp,2)/2) ...
        MxFinal-floor(size(imtemp,1)/2),size(imtemp,2)-1
size(imtemp,1)-1];

    end

    CCMax = cat(1,Result.CCMax);
    SelCCMax = find(CCMax == max(CCMax(:)));
    ResultSt = Result(SelCCMax);

end
```

A.4: Lip Detection

```
%-----------------------------------------------------------------------
-
%PURPOSE:       This program outputs the lip bounding box
%REQUIREMENT:   The candidate JPEG images are stored in the folder
%               'lipdettest'
%OUTPUT:    The image with lip bounding box outlined and images from
%           different lip detection stage are also sorted into
different
%           folders.
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%-----------------------------------------------------------------------
-

clear all
close all
clc

filename = dir ('lipdettest\*.jpg');
mkdir('BW');
mkdir('CloseBW');
mkdir('CropRe');

for n = 1: 1: length(filename)

    disp(['Images: ' int2str(n) '/' int2str(length(filename))])

    inImage = imread (['lipdettest\' filename(n).name]);

    [result1 gXt gYt] = gradHnL(inImage);
    [result2 gXb gYb]= gradHnLlow(inImage);

    gradtht = im2bw(gYt,0.028);
    gradthb = im2bw(gYb,0.03);

    se = strel('line',1,90);
    bw = imopen(gradtht,se);
    bw1 = imopen(gradthb,se);

    outLipsMaskdb= restrMask(bw1);
    outLipsMaskdt= restrMask(bw);

    outLipsMaskd = outLipsMaskdb | outLipsMaskdt;

    imwrite(outLipsMaskd, ['BW\BW_' filename(n).name],'JPEG');

    se = strel('disk',4);
    closeBW = imclose(outLipsMaskd,se);
    imwrite(closeBW, ['CloseBW\CloseBW_' filename(n).name],'JPEG');

    L = bwlabel(closeBW);
    s  = regionprops(L, 'all');
    bdbox = cat(1, s.BoundingBox);
    imarea = cat(1,s.Area);
    maxarea = find(imarea == max(imarea));

    figure(1), imshow(inImage);
    hold on

    if (~isempty(bdbox))
        crop = imcrop(inImage,bdbox(maxarea,:));
        imwrite(crop,['CropRe\crop_' filename(n).name],'JPEG');
        rectangle('Position',bdbox(maxarea,:),'EdgeColor','r');
        hold off
    else
        imwrite(inImage,['CropRe\crop_' filename(n).name],'JPEG');
    end

    pause();

end



function [outimg gX gY] = gradHnL(inimg)

%-----------------------------------------------------------------------
-
%PURPOSE:    This program creates the gradient after the algorithm
%            emphasizing upper lips
%INPUT:      RGB Image
%OUTPUT:     The gradient after the algorithm emphasizing upper lips
%AUTHOR:     Benafsh Husain, Cal Poly San Luis Obispo
%DATE:       August 1, 2011
%-----------------------------------------------------------------------
-

    R = inimg(:,:,1);
    G = inimg(:,:,2);

    YCRCB = rgb2ycbcr(inimg);
    Y = double(YCRCB(:,:,1));

    pH = double(R)./(double(R)+double(G));

    pH(isnan(pH))=0;

    Ynorm = Y./ max(max(Y));

    outimg = pH - double(Ynorm);

    [gX gY] = gradient(outimg);

end
```

```
function [outimg gX gY]= gradHnLlow(inimg)

%----------------------------------------------------------------------
-
%PURPOSE:   This program creates the gradient after the algorithm
%           emphasizing lower lips
%INPUT:     RGB Image
%OUTPUT:    The gradient after the algorithm emphasizing lower lips
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------
-

    R = inimg(:,:,1);
    G = inimg(:,:,2);

    YCRCB = rgb2ycbcr(inimg);
    Y = double(YCRCB(:,:,1));

    pH = double(R)./(double(R)+double(G));

    pH(isnan(pH))=0;

    Ynorm = Y./ max(max(Y));

    outimg = pH + double(Ynorm);

    [gX gY] = gradient(outimg);

end

function outMask = restrMask(bw)

%----------------------------------------------------------------------
-
%PURPOSE:   This program provides the constraints in selecting lips
%INPUT:     The binary image containing lip
%OUTPUT:    The binary image after the impossible lip candidate is
%           eliminated based on the constraints
%AUTHOR:    Benafsh Husain, Cal Poly San Luis Obispo
%DATE:      August 1, 2011
%----------------------------------------------------------------------
-

    L = bwlabel(bw);

    s  = regionprops(L, 'all');
    bdbox = cat(1, s.BoundingBox);
    orent = cat(1, s.Orientation);
    imarea = cat(1, s.Area);

    orthl = -18;
    orthr = 18;
    Siz = size(bw);
    outLipsMaskd = zeros(Siz);
    TotArea = double(Siz(1)*Siz(2));

    for k=1:1:size(bdbox,1)

        bwcrop = imcrop(bw,bdbox(k,:));
        bwcropMir = fliplr(bwcrop);
        overlap = bwcrop & bwcropMir;
        percent =
sum(sum(double(overlap)))/(size(bwcrop,1)*size(bwcrop,2));

        if (orent(k)>= orthl) && (orent(k)<= orthr)  && ...
               (imarea(k)/TotArea < 0.02) && (bdbox(k,4)< bdbox(k,3))
&&...
                 (imarea(k)/TotArea > 0.002)&& (bdbox(k,2)>
Siz(1)*1/2)&&...
                 (percent > 0.08) && (percent < 0.7) && ...
                 (bdbox(k,2)< (Siz(1)-Siz(1)/8))

             for a = round(bdbox(k,1)):1:(round(bdbox(k,1))+bdbox(k,3))
                for b =
round(bdbox(k,2)):1:(round(bdbox(k,2))+bdbox(k,4))
                    if (b <= Siz(1))&& (a <= Siz(2))
                        outLipsMaskd(b,a) = bw(b,a);
                    end
                end
            end
        end

    end

    outMask = (outLipsMaskd > 0);

end
```