ROBUST UNCONSTRAINED FACE DETECTION AND LIP LOCALIZATION ALGORITHM USING GABOR FILTERS

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

Robert Edward Hursig

July 2009

© 2009
Robert Edward Hursig
ALL RIGHTS RESERVED

COMMITTEE MEMBERSHIP

TITLE: Robust Unconstrained Face Detection and Lip Localization using Gabor Filters

AUTHOR: Robert Edward Hursig

DATE SUBMITTED: July 2009

COMMITTEE CHAIR: Dr. Xiaozheng (Jane) Zhang

Professor of Electrical Engineering, Advisor

COMMITTEE MEMBER: Dr. John Saghri

Professor of Electrical Engineering

COMMITTEE MEMBER: Dr. Xiao-Hua (Helen) Yu

Professor of Electrical Engineering

ABSTRACT

Robust Unconstrained Face Detection and Lip Localization using Gabor Filters

Robert Edward Hursig

Automatic speech recognition (ASR) is a well-researched field of study aimed at augmenting the man-machine interface through interpretation of the spoken word. From in-car voice recognition systems to automated telephone directories, automatic speech recognition technology is becoming increasingly abundant in today's technological world. Nonetheless, traditional audio-only ASR system performance degrades when employed in noisy environments such as moving vehicles. To improve system performance under these conditions, visual speech information can be incorporated into the ASR system, yielding what is known as audio-video speech recognition (AVASR). A majority of AVASR research focuses on lip parameters extraction within controlled environments, but these scenarios fail to meet the demanding requirements of most real-world applications. Within the visual unconstrained environment, AVASR systems must compete with constantly changing lighting conditions and background clutter as well as subject movement in three dimensions.

This work proposes a robust still image lip localization algorithm capable of operating in an unconstrained visual environment, serving as a visual front end to AVASR systems. A novel Bhattacharyya-based face detection algorithm is used to compare candidate regions of interest with a unique illumination-dependent face model probability distribution function approximation. Following face detection, a lip-specific

Gabor filter-based feature space is utilized to extract facial features and localize lips within the frame. Results indicate a 75% lip localization overall success rate despite the demands of the visually noisy environment.

Keywords: Lip Localization, Face Detection, Feature Extraction, Gabor, Bhattacharyya, Epanechnikov, HSV, Hysteresis Thresholding

ACKNOWLEDGMENTS

Completion of this thesis represents the culmination of my academic career, the crowning achievement upon half a decade's worth of higher education, dedication, and life-changing experiences. More important than this achievement are the countless role models who have shaped my life more than they will ever know. I devote this work to all of you.

To my parents, Marla and Dave, and my brother, Allan, you have always been my rock, my inspiration, and my number one fan. You instilled in me commitment, respect, and, most importantly, humility, taking genuine interest in everything I do and lending support whenever needed. Words cannot describe how much I appreciate the active and abundant role you all have played in my life. Mom, I can only hope to do just as much for just as many people as you do. Between the exams, the lab reports, and the homework sets, you were constantly there to remind me to take a step back, breathe, and appreciate life. There was nothing in life that your care packages or phone calls couldn't remedy. Dad, I take all the best of my abilities from you. You have fostered my love of engineering, my inquisitiveness, and even my sense of humor, voicing support for all I do. While you would be the last to admit it, I take after you in nearly every way and I wouldn't have it any different. Allan, you have always been and will always be my best friend in life. You constantly remind me of what is truly important, providing me with an inside joke and a Halo break or two should I happen to lose track. I could not be prouder of the man you're becoming.

To Cal Poly's Electrical Engineering faculty, you have challenged my abilities, expanded my knowledge, and opened my eyes to the endless possibilities our field offers. For instilling in me a wealth of information and a passion for what is and what is yet to be, I must express my heartfelt gratitude. 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 Xiao-Hua (Helen) Yu for serving on my thesis committee and for fostering my interest in image processing and pattern recognition. I must also thank Brandon Crow for his admirable contribution to this project. The groundwork he established for my own work and the accomplishments he achieved in his thesis are more than impressive.

To all my friends and family, spread all across the nation, I would not be where I am today without the abundance of good times and life experiences that I have shared with you all. I am blessed to have you in my life and I wish you the best of luck wherever our individual passions take us.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Background

Automatic speech recognition is a well-researched field of study defined as the translation of spoken words into electro-mechanical commands. Automatic speech recognition (ASR) seeks to improve upon the ease and efficiency by which humans and machines interact with one another through the spoken word. Examples of ASR and the development of the man-machine interface include automated telephone directories, voice-activated cell phone commands, and in-car music and cell phone control systems. Nonetheless, traditional audio-only ASR systems suffer from severe performance degradation when implemented in noisy and uncontrolled environments.

Traditional ASR methods which utilize only audio information to interpret speech suffer from sensitivity to background noise, user speech variation, and limited word dictionaries. In fact, even in "clean," controlled speaking environments, state of the art ASR systems still underperform human speech perception by over an order of magnitude [23]. It has been determined that human speech perception is "bimodal" in nature, referring to the face that humans utilize both audio and visual information to analyze what was heard. The former information space is more heavily valued by

humans when in noisy environments, having specific importance to the hearing impaired. Additionally, audio and visual information are complementary in nature.

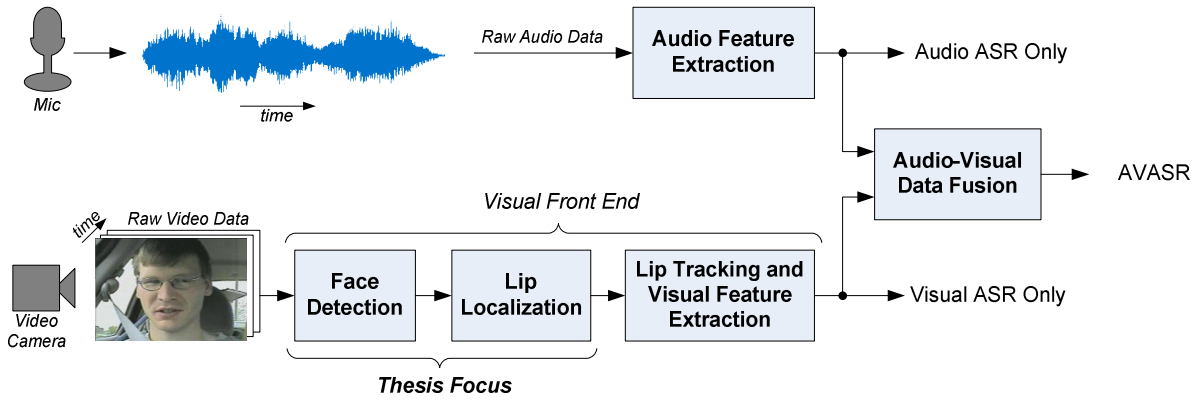


Figure 1.1: General Audio-Video Automatic Speech Recognition Diagram

Benefitting from this visual speech component, many audio-visual automatic speech recognition (AVASR) systems have been developed in an attempt to improve the accuracy of speech recognition. Combining audio and visual data into the speech recognition process, Figure 1.1 contains the general AVASR procedure. As seen, audio and visual feature extraction are processed independently over time, combining the different speech components in what is called "data fusion." The visual front end components of AVASR are composed of two highly researched components: face and lip localization and tracking. While lip parameter extraction is more highly researched, being able to locate and track the face and lips within any given video frame poses a significant logistical undertaking. For this reason, the focus of the following work will concentrate on a robust lip localization algorithm based on still images alone, realizing that the neglected time component of AVASR provides beneficial information that would aid the localization process.

To minimize the requirements demanded of such systems, a majority of documented AVASR systems focus on visual speech information within controlled lip

environments [6][16][26]. Controlled AVASR environments benefit from ample video resolution, generally monotone backgrounds for simple face detection, and optimal lighting and camera angles. However, real-world applications of AVASR inherently involve mitigating factors which degrade system performance relative to controlled cases. Audio-visual automatic speech recognition systems that operate within this so-called unconstrained environment must be able to compensate for subjects that move in three dimensions amid a possibly cluttered background and constantly changing lighting conditions. Several unconstrained AVASR techniques have been employed and well-documented but studies show that use of these systems as a pervasive user-interface has not yet been obtained [23].

Generally, the in-car audio-visual environment can be considered as a worst-case scenario for AVASR. Background noise and mechanical vibrations from traveling vehicles severely decrease operational signal-to-noise ratios for audio processing. Several products such as Ford Motor Company's Sync® and BMW's high-end Voice Command System use strictly audio information to recognize user requests. Nonetheless, these systems notably suffer from user voice dependence and background noise such as open windows or ambient noise from highway speeds. Likewise, the car visual environment is also challenging, imposing rapidly changing lighting conditions, moving faces within the vehicle, and constantly changing background clutter.

1.2 Previous Work Done in This Area

The system proposed in this paper benefits from previous work by Brandon Crow which developed a unique, fully functioning face and lip detection, localization, and tracking algorithm within the unconstrained car environment [4]. Crow proposed a

shifted version of the HSV color space as the feature space of choice for face and lip detection and localization. Furthermore, the face detection algorithm utilized three joint shifted hue and saturation join probability density functions extracted directly from the training set as a basis for a Bhattacharyya-based algorithm. Candidate and model probability density functions were approximated via histogram approximation applied to the respective, kernel-weighted region of interest. In this system each of five fixed regions of interest were compared with the three face models and the region which provided the highest response for any model was selected as a facial candidate. Facial feature localization occurred via heuristic methods applied to saturation and illumination spaces. The Bhattacharyya coefficient utilized in face detection was then employed by a mean-shift tracking algorithm to track the face and lips throughout a video sequence.

This work's face detection and lip localization success rates approached 75% and 65%, respectively, with the fixed-region face detection algorithm suffering from partial occlusions of the face candidate regions. The tracking algorithm managed impressive results tracking larger facial regions of interest while failing to accurately track smaller lip regions from frame to frame. It is the purpose of this work to improve upon Crow's contributions to the AVASR system as a whole.

1.3 Thesis Organization

As previously mentioned, the goal of this work will be to develop a robust still image lip localization algorithm. This algorithm was designed as a sub-component to the larger AVASR system as a whole. As such, the algorithm operates under the assumption that it is to serve as a front end or periodic update to lip parameter extraction and tracking which is implemented downstream. The following document details a robust skin and

face detection algorithm in conjunction with a Gabor filter-based lip localization algorithm founded within the unconstrained car environment. Building upon previous work, the in-car environment will similarly be used as this work's performance standard.

Chapter 2 will detail the skin classification, face localization, and face detection algorithms. The findings from extensive training sets will be used to select feature spaces and models for each along with the results from comprehensive testing. Chapter 3 will explain the Gabor filter and describe the Gabor-filter-based facial feature extraction and lip localization processes. Once again, extensive training and test sets will be used to justify design decisions and performance, respectively. Lastly, Chapter 4 will summarize the performance of the lip localization algorithm as a whole as well as a recommendations for future improvement and direction.

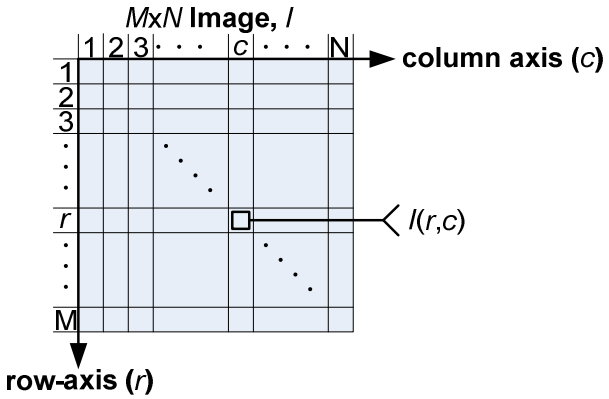


Figure 1.2: Image Coordinate Convention

The Matlab environment was used exclusively for this algorithm's development and testing. Throughout this document note that Appendices A and B contain complete records of all Matlab code used to construct the final, fully functioning algorithm. The image coordinate conventions used throughout this document are detailed in Figure 1.2, noting that positive row, or vertical, axis extends vertically downwards from the image's

(spatial) top-left corner. Image coordinates will be referred to as [*r*,*c*] while image pixel values will be denoted as $I(r,c)$, where I is the parent image and r and c are the row and column coordinates, respectively within the image. Per the Matlab environment, also note that the image's indexing starts at one. All references made to an images dimensions will be denoted as *height*-by-*width*, or *M*-by-*N* in the case of Figure 1.2. All pixel intensity values are assumed to use double arithmetic and are defined over the range [0,1].

CHAPTER 2
FACE DETECTION

Accurate face detection plays a critical role in successful lip localization, the ultimate goal of this work. The relatively small size and constantly changing shape of lips does not realistically allow for feasible direct lip detection. Coupled with the difficulties introduced by an unconstrained operational environment, a robust, computationally efficient face detection algorithm is desirable to precede lip localization itself. The human face is relatively simple to detect within an image, containing several facial features—eyes, nose, and mouth—along with a distinguishable spectral content. Nonetheless, the human face is one of the most variable and common objects that humans interact with on a daily basis. As indicated, many dependable facial recognition methods exist under controlled conditions including monotone backgrounds, optimal lighting and camera angle, and ample resolution and processing power [6][26][16]. However, real-world facial recognition applications require more versatile, noise-resistant methodologies. The following sections offer means by which skin is first classified in an appropriate color space and then subsequently classified as a face or non-face. Facial feature extraction and lip localization is discussed in Chapter 3.

Throughout this work, training and test datasets will be drawn from the AVICAR database [17]. This database contains audio-visual recordings of 50 male and 50 female

participants with varying ethnicities and constantly changing lighting conditions within a moving automobile. Datasets were created by extracting frames, or still images, from the database video files, which utilizes a wavelet-based, lossy audio-video interlaced (AVI) encoding scheme. Video and image resolution for this database is 240-by-360 pixels, height-by-width.

Section 2.1 to follow will first define the original image RGB color space and the HSV color space, justify the selection of a shifted HSV color space for skin and face detection. Section 2.2 will detail the construction of a robust skin classifier within this color space while subsequent isolation of face candidates via filtering and binary clustering operations are detailed in Section 2.3 and 2.4, respectively. Section 2.5 proposes a unique, illumination-dependent face model probability density function approximation derived through an extensive test set that will serve as the basis for face detection. Lastly, Section 2.6 will describe the Bhattacharyya-based face detection algorithm itself and results of the larger face detection algorithm as a whole as applied to a database test set.

2.1 Color Space Overview

In order to efficiently detect skin and faces within an image, the respective classifier must be developed within an appropriate color space. Proper color space selection has the effect of simplifying the classification complexity and dimensionality while improving inter-class separation. Extensive research has attempted to determine the optimal color space for skin detection with mixed findings [1][4][12][19][25]. In such studies, Shin *et al*. determined that most color space conversions fail to deliver ample skin detection improvements [25], while Jones *et al.* determined NRBG was the

optimal color space [12], Ming-Hsuan *et al.* and Crow selected perceptual color spaces such as HSV [19][4], and Abdel-Mottaleb *et al.* selected TV color spaces such as YIQ [1]. Based on previous work done by Crow [4] and further experimentation the HSV color space was selected as the optimal skin and face detection color space under the discussed operating conditions. The following sections will outline the original image's RGB color space as well as the selected HSV color space to justify this decision.

2.1.1 RGB Color Space

The predominant digital image color space, RGB encodes color information as an additive combination of the three primary colors, red (R), green (G), and blue (B). This color space can be visualized as a 3D cube with three orthogonal axes, R, G, and B. Chiefly used for its simplicity and compatibility with display devices, RGB color space does not separate chrominance (color) and illumination (brightness) information. Hence, variation in an object's luminance alters each of the object's color components. Throughout this report each color component within the RGB space will be confined to the range [0,1]. Figure 2.1 below illustrates the RGB color cube with the dotted line representing grayscale values extending linearly from the origin to the point [1,1,1].

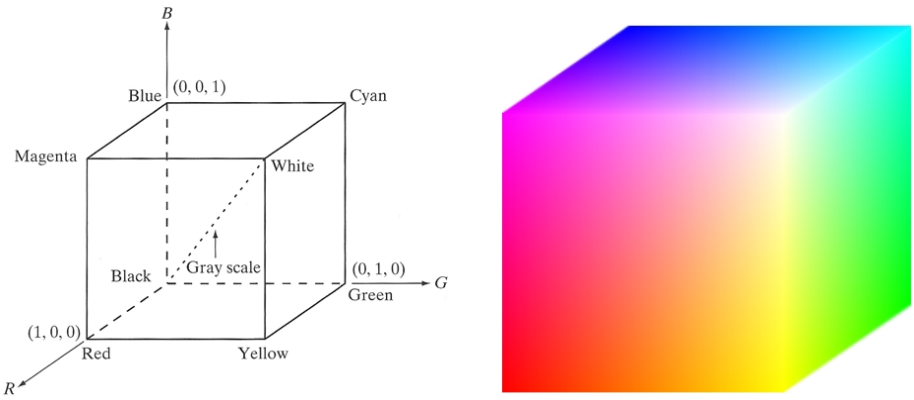


Figure 2.1: The RGB Color Cube [11]

An extension of RGB color space is the normalized RGB, or NRGB, color space which normalizes each pixel color vector to the element sum total of that pixel vector:

$$r = \frac{R}{R+G+B} \qquad g = \frac{G}{R+G+B} \qquad b = \frac{B}{R+G+B} \tag{2.1}$$

where r, g, and b are the normalized components of R, G, and B, respectively. Results of this normalization include decreased illumination dependence on chromaticity. Moreover, as the sum of all the components must sum to one, one color component can be considered redundant. This color space, in turn, provides a more illumination-invariant color space with the added dimensionality reduction resulting from the normalization. Conversely, NRGB's relative illumination-independence entirely discards useful brightness information that is preserved in other color spaces. Color-based skin detection benefits from the ability to threshold uniformly colored objects despite variations in illumination across the object's surface. As color information is contained in multiple NRGB components, this uniform color thresholding would require two at least two color components in general. As will be shown, the HSV color space provides an illumination-independent color component as well as a separate brightness information, making it ideal for simple, uniform-color thresholding. Moreover, the preservation illumination information in the HSV color space will play a critical role in the feature extraction algorithm detailed in Chapter 3.

2.1.2 HSV Color Space

Standing for hue, saturation, and value, HSV is one of the perceptual color spaces which more closely models how humans perceive color. HSV is analogous to HSI, HSL, and HSB color spaces, where I is intensity, L is lightness, and B is brightness. These

spaces differ in the definitions of the saturation and illumination components, but HSV is the only color space of these that weighs intensity values equally across all hue. Within this three-dimensional space, hue represents the color's dominant wavelength. Saturation refers to how much white light is intermixed with the pure color with 0 indicating pure white light and 1 indicating a pure spectrum color. Lastly, value refers to the achromatic intensity of the color, known as illumination, where 0 is black and 1 is white. The terms value and illumination will be used interchangeably in this document.

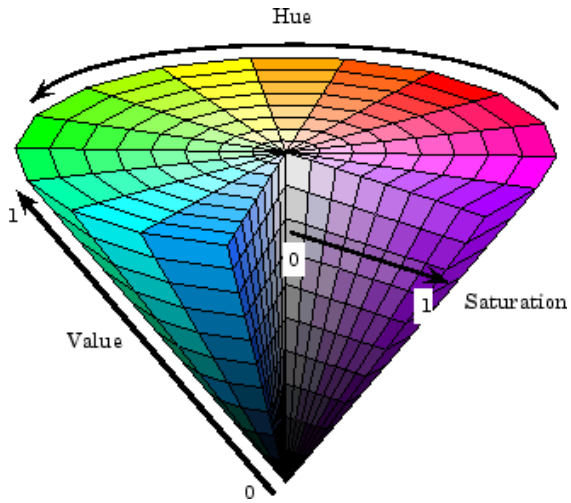


Figure 2.2: HSV Color Model [18]

Figure 2.2 illustrates the conical HSV color model as described. The hue component exists on what is called a "color wheel" and is an angular measure ranging from 0° to 360° (or 0 to 2π radians) with 0° representing a device-dependent red wavelength. Because it is an angular measure, the hue component undergoes a wrap-around effect from 0° to 360°. Saturation and intensity are magnitude measures ranging from zero to one as discussed.

Complicated by the amount of perceptive color spaces, there are several means by which one converts the RGB imagery to the HSV color space. However, as Matlab will

be used as the algorithm development and test platform, the RGB to HSV conversion implemented by Matlab's Image Processing Toolbox is defined below.

Let $r, g, b \in [0,1]$ be the red, green, and blue components of the RGB image

Let $M = \max(r, g, b)$ and $m = \min(r, g, b)$

$$H_o = \begin{cases} 0 & \text{if} \quad M = m \\ \left(\dfrac{6 + g - b}{6 \cdot (M - m)} \right) \bmod 1 & \text{if} \quad M = r \\ \dfrac{2 + b - r}{6 \cdot (M - m)} & \text{if} \quad M = g \\ \dfrac{4 + r - g}{6 \cdot (M - m)} & \text{if} \quad M = b \end{cases} \tag{2.2}$$

$$S_o = \begin{cases} 0 & \text{if} \quad M = m \\ \dfrac{M - m}{M + m} & \text{if} \quad M \le 1 - m \\ \dfrac{M - m}{2 - (M + m)} & \text{if} \quad M > 1 - m \end{cases}$$

$$V_o = M$$

where H_o, S_o, and V_o are the hue, saturation, and value components, respectively, within the standard HSV color space and *mod* is the modulo operator. Note that by this definition of the HSV space, the hue value has been re-mapped to exist over the [0,1] range instead of [0°,360°], simplifying calculations in subsequent algorithms. Hence, it can be shown that $H, S, V \in [0,1]$.

2.1.3 Shifted HSV, or sHSV, Color Space

Because of the hue's color wheel effect, certain subsets of interest within the hue domain may exist in two separate ranges. Hence, to simplify thresholding operations it is

often desirable to offset the phase of the hue space such that regions of interest incur no discontinuities. As will be developed in the skin classification section to follow, research from Crow *et al.* shows that the probability distribution of skin falls in two separate ranges at the low and high ends of the hue space [5]. Due to the distribution's closeness to the hue space discontinuity, approximately 10% of the distribution wraps around near the discontinuity at a hue value of 1. To simplify future classification, Crow proposed a shifted HSV space which offsets the standard, red-referenced hue space by a value of 0.2 (72°) [4]. In this space, skin follows a Gaussian distribution centered at a shifted hue value of 0.34 with a standard deviation of 0.11. This shifted HSV, or sHSV, color space is defined as follows:

$$H = \left(H_o + 0.2\right) \bmod 1, \quad S = S_o, \quad V = V_o \tag{2.3}$$

where H_o, S_o, and V_o refer to the standard HSV color components defined in Equation 2.2 and H, S, and V are the color components of the sHSV color space. Note that the saturation and value definitions remain the same and H, S, and V remain elements of the range [0,1]. Henceforth, this shifted HSV color space will be used in place of the standard definition throughout the final algorithm.

2.2 Skin Classification

Before reliable lip localization can occur, it is desirable to fist detect skin and the parent face. Other sources, such as Elgammal *et al.* and Crow, also promote skin detection preprocessing as an efficient means to detect faces downstream [9][4]. The following subsections will outline the training set skin classification parameters for the

AVICAR database, the theoretical Bayesian skin classifier for the database, and the final, adjusted skin classifier.

2.2.1 Bayesian Skin Classifier

Gaussian approximations of skin and non-skin class-conditional probability distributions extracted from supervised classification of the AVICAR database were shown to have the following form [4]:

$$P(h \mid Skin) \sim N(0.34, 0.11^2)$$
$$P(h \mid NonSkin) \sim N(0.55, 0.17^2) \quad \textbf{(2.4)}$$

where h is the shifted hue component of the sHSV triplet for a given pixel, *Skin* and *NonSkin* are the disjoint and completely representative classes, and $N(\mu, \sigma^2)$ represents a Gaussian distribution with mean μ and variance σ^2. Similarly, the prior probabilities for skin and non-skin classes within the same AVICAR training set were reported as

$$P(Skin) = 0.2154 \quad \textbf{(2.5)}$$
$$\text{and } P(NonSkin) = 0.7854$$

Lastly, the class posterior distributions can then be derived via Bayes' Rule as given by:

$$P(Skin \mid h) = \frac{P(h \mid Skin) \cdot P(Skin)}{P(h)} \quad \textbf{(2.6)}$$

$$P(NonSkin \mid h) = \frac{P(h \mid NonSkin) \cdot P(NonSkin)}{P(h)} \qquad \text{where } P(h) \in [0,1]$$

Bayesian classification systems have been shown to have commendable performance in conjunction with a simple implementation that often reduces to simple thresholding operations. Bayesian classification schemes classify the pixel of interest as

the class which maximizes the posterior distributions of Equation 2.6 above. Hence, the generic Bayesian decision rule for this two-class, one-dimensional case is

$$P(Skin \mid h) \underset{h \in NonSkin}{\overset{h \in Skin}{\gtrless}} P(NonSkin \mid h)$$

$$\frac{P(h \mid Skin) \cdot P(Skin)}{P(h)} \underset{h \in NonSkin}{\overset{h \in Skin}{\gtrless}} \frac{P(h \mid NonSkin) P(NonSkin)}{P(h)} \qquad \textbf{(2.7)}$$

Noting that the nonnegative $P(h)$ term is common to each side of the inequality, Duda *et al.* [8] defines the simplified, final Bayes decision rule as

$$P(h \mid Skin) \cdot P(Skin) \underset{h \in NonSkin}{\overset{h \in Skin}{\gtrless}} P(h \mid NonSkin) P(NonSkin) . \qquad \textbf{(2.8)}$$

Each side of Equation 2.8 represents un-normalized posterior distributions respective to the class in question. Figure 2.3 illustrates the un-normalized posterior distributions for the skin and non-skin classes. Here the green lines represent the zero-dimensional decision boundaries that separate the Bayesian classified skin and non-skin decision regions. Between these boundaries, from a shifted hue value of 0.052 to 0.325, the skin posterior distribution surpasses that of non-skin and will classify as a skin pixel.

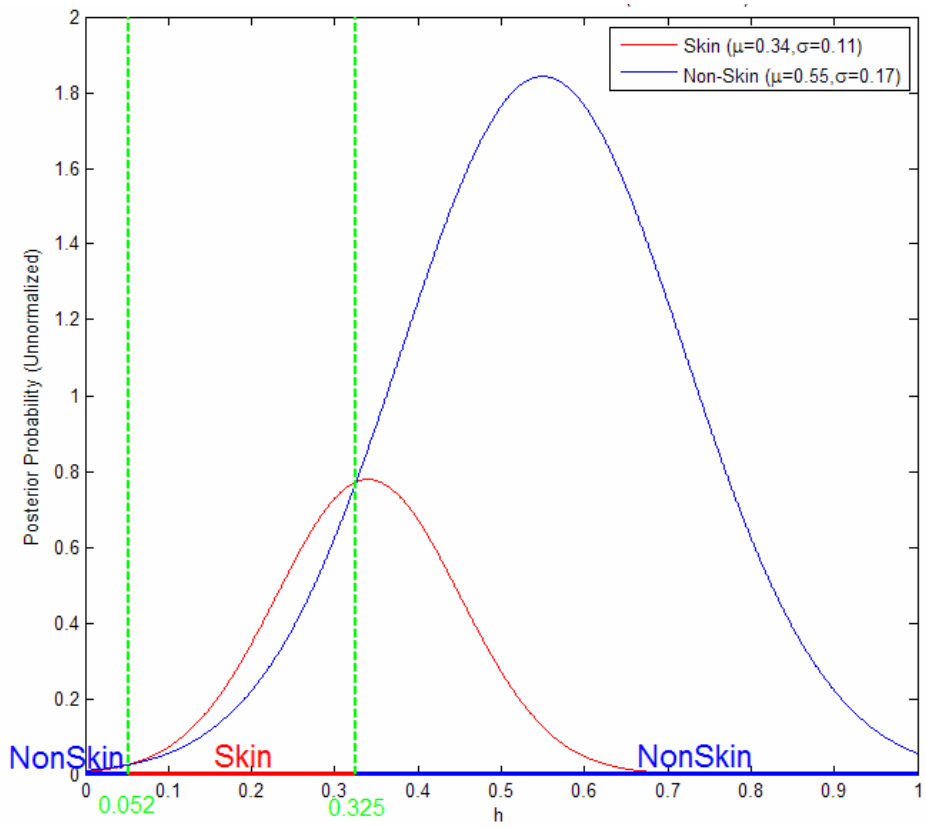


Figure 2.3: Un-Normalized Posterior Distributions for Skin and Non-Skin Classes

Letting t_{lo} and t_{hi} be the lower and upper boundaries of the classifier, respectively, the theoretical skin classifier is defined as

$$C_o(h) = \begin{cases} Skin & \text{if} \quad t_{lo} \le h \le t_{hi} \\ NonSkin & \text{otherwise} \end{cases} \tag{2.9}$$

where $t_{lo} = 0.052$ and $t_{hi} = 0.325$

As will be developed in the following subsection, this will not be the final skin classifier implemented due to practical design considerations and other limitations. Nonetheless, the skin classifier of Equation 2.9 was used as the baseline from which the final classifier was derived.

2.2.2 *Resampling and Adjusted Skin Classifier*

Building upon the theoretical Bayes classifier, the final skin classification system will add robustness and computational efficiency for subsequent face detection. To decrease false positives and increase true positives, the boundaries of the skin decision region derived from the Bayesian classifier of Equation 2.9 will be altered per experimentation. To decrease computational demands on skin classification and classification post-processing, the original input image will be resampled to reduce image dimensionality.

Figure 2.4: Resampling and Skin Classification Process Block Diagram

Figure 2.4 shows the resampling and classification process in block diagram form. Here, the MxNx3 original RGB image, I_{RGB}, is first converted to the sHSV color space. After conversion, the MxNx3 sHSV image, I, then undergoes blurring via an averaging filter across each color component separately. This process preserves low frequency color information consistent with larger objects such as skin regions and mitigates the effect of higher frequency noise. Next, the blurred image is downsampled in each spatial dimension (row and column) by a factor, F. For convenience the size of the averaging filter mask was set to F-by-F such that all pixels in the original image contribute to the downsampled image, I_d, which has the dimensionality M_dxN_dx3. A value of eight was selected for the downsampling factor, F, through testing, balancing computational speed and the effectiveness of skin classification downstream. The skin classification component of Figure 2.4 will now be discussed in depth.

Applying the theoretical Bayes classifier of Equation 2.9 has key limitations. When applied to a subset of the AVICAR database, the theoretical classifier yielded elevated partial facial skin detection, especially when applied to darker skinned individuals. Incomplete facial skin detection is especially detrimental as the face detection methodology employed in this work benefits from a cohesive (continuous) skin classification mask that minimizes background pixel contamination. Additionally, the theoretical, hue-based classifier also yielded an increased number of false positives within the images taken in low-light conditions. Three substantial alterations to the original Bayes classifier were made to increase algorithm robustness. They include modifying upper and lower shifted hue thresholds, implementation of a hysteresis threshold, and incorporation of illumination information into the classifier.

To avoid partial skin detection, the Bayes skin classifier's upper and lower thresholds, t_{hi} and t_{lo}, were independently tuned through experimentation to minimize false positives and maximize true positives. The resulting shifted hue skin classifier follows:

$$C_{sH,\mathrm{mod}}(h) = \begin{cases} Skin & 0.18 \le h \le 0.40 \\ NonSkin & \text{otherwise} \end{cases} \qquad \textbf{(2.10)}$$

where h is the shifted hue component of the pixel's sHSV triplet. It is apparent that through optimization, the upper and lower threshold's were increased. Both values were modified via experimentation to maximize classification accuracy (refer to Table 2.2). Table 2.1 compares theoretical shifted hue threshold values with those of the modified skin classifier.

Table 2.1: Theoretical and Modified Skin Classification Thresholds for Shifted Hue

Classifier	**Low Thresh., t_{lo}**	**Hi Threshold, t_{hi}**
Theoretical Bayes	0.052	0.325
Modified	0.18	0.40
Change	+0.128	+0.075

To further promote skin region continuity, a hysteresis threshold was also employed. While hard thresholding is based solely on hue value of each pixel, hysteresis thresholding uses both spatial and hue information to classify pixels. Especially when dealing with video data, flicker and lossy encoding methods will distort color and illumination information intermittently throughout the video frame. Hysteresis thresholding utilizes both hard and soft (looser) thresholds in conjunction with spatial proximity to pixels which satisfy the former. Let $C_{sH,soft}$ be defined as the shifted hue soft skin classifier which has an expanded skin decision region, per the following relation

$$C_{sH,soft}(h) = \begin{cases} LSkin & \text{if} \quad t_{lo} - t_{hys} \le h \le t_{hi} + t_{hys} \\ LNonSkin & \text{otherwise} \end{cases} \tag{2.11}$$

where h is the shifted hue component of the sHSV triplet, *LSkin* is the "soft" skin class, *LNonSkin* is the "soft" non-skin class, and t_{hys} is the hysteresis value which separates the hard and soft thresholds. The modified, hysteresis-based shifted hue skin classifier, C_{hys}, has the form

$$C_{hys}(h) = \begin{cases} Skin & \text{if} \quad C_{sH,\mathrm{mod}}(h) \in Skin \\ Skin & \text{if} \quad C_{sH,soft}(h) \in LSkin \mid \{N_8\} \cap Skin \neq \varnothing \\ NonSkin & \text{otherwise} \end{cases} \tag{2.12}$$

where h is the shifted hue component of the sHSV triplet, $C_{sH,mod}(h)$ is the modified shifted hue classifier of Equation 2.10, $\{N_8\}$ is the standard 8-pixel neighborhood set of shifted hue values about the sample pixel containing h, $A \cap B$ refers to the intersection of the sets A and B, and $\varnothing$ is the empty set. In words, hysteresis thresholding results in skin

classification if it satisfies the hard thresholding of Equation 2.10 or if the soft thresholding of Equation 2.11 is satisfied given at least one of the eight neighboring pixels satisfies Equation 2.10. For improvements in classification realized by this classifier refer to Table 2.2.

To remedy the performance drop incurred in low-light environments, achromatic intensity information contained in the value component of the sHSV triplet was added to the skin classifier. Research shows that more than 90% of skin exists above illumination values greater than 0.15 when approximated by a Gaussian distribution [4]. Through experimentation, incorporation of the illumination dimension to the classifier is shown to increase the skin detection robustness (refer to Table 2.2 for comparisons). After implementation, this lower value threshold was fine-tuned to a value of 0.2, yielding the final, skin classifier for the algorithm as

$$C(h,v) = \begin{cases} Skin & \text{if} \quad C_{hys}(h) \in Skin \quad \text{and} \quad v \geq 0.2 \\ NonSkin & \text{otherwise} \end{cases} \tag{2.13}$$

where h and v are the shifted hue and value components of the sHSV triplet, respectively, and $C_{hys}(h)$ is the hysteresis shifted hue skin classifier of Equation 2.12.

After the resampling process, the modified skin classifier of Equation 2.13 is then applied to the downsampled image, I_d. The skin classified image, BW, is binarized such that skin is represented with a logic high (binary 1) per the following equation

$$BW(r,c) = \begin{cases} 1 & \text{if} \quad C(H(r,c),V(r,c)) \in Skin \\ 0 & \text{otherwise} \end{cases} \quad \forall r,c \quad \begin{matrix} r \in 1,2,...M_d \\ c \in 1,2,...N_d \end{matrix} \tag{2.14}$$

where r is the image's row index, c is the image's column index, M_d is the downsampled image height, N_d is the downsampled image width, and $H(r,c)$ and $V(r,c)$ are the shifted hue and value sHSV components of I_d at location (r,c), respectively.

The classifier's progressive feature adjustment and incorporation was seen to drastically improve classification accuracy. Table 2.2 compares the classification accuracy of the listed classifiers applied to a manually classified subset of the AVICAR database. The set was constructed by selecting 20 male and 20 female candidates from the AVICAR database and extracting four separate images from the subject's video, comprising an 160-image training set in total. Subjects and images were selected such that the images provided a representative training set in regards to skin color (ethnicity) and provided a representative sampling of lighting conditions throughout the image. The analyzed classifiers range from the theoretical Bayes classifier to the final, multidimensional classifier with hysteresis as described. With the addition of each additional feature or adjustment, the number of images obtaining at least a 75% accuracy increases steadily. The final classifier of Equation 2.13 achieved the best performance over all accuracy metrics listed.

Table 2.2: Skin and Non-Skin Classification Accuracy over Classifier Configuration

	Classification Accuracy Percentage per Measure*		
Skin Classifier	**>75% Correct**	**<75% Correct†**	**<50% Correct†**
Theoretical Bayes, sHue Only Equation 2.9	50.6%	49.4%	23.1%
Modified Bayes, sHue Only Equation 2.10	65.0%	35.0%	15.0%
Modified Bayes with Hysteresis sHue Only, Equation 2.12	77.5%	22.5%	8.1%
Final, sHue with Hysteresis and Value Thresh., Equation 2.13	**93.8%**	**6.3%**	**0.6%**

**Defined as number of correctly classified pixels based off of manually determined ground truth divided by image pixel count. Classifiers applied to 160-image training set as described above.*

†Notice that these accuracy metrics are not disjoint (mutually exclusive). In fact, the <50% metric is a subset of the <75% metric.

Figure 2.5 and Figure 2.6 contain sample complete and incomplete skin classifications, respectively. Part (b) in each case is the original RGB image converted to the sHSV color space, where the shifted hue, saturation, and value color components are

displayed as the red, green, and blue RGB components, respectively. As seen especially within the right half of the subject's face in Figure 2.6, low-light conditions tend to distort hue information per nonlinear imaging detector limitations. Note the substantial increase in the shifted hue value (displayed as red) in Figure 2.6(b) over the right half of the subject's face. Similarly, overly bright conditions were also seen to distort the hue information and disrupt skin classification.

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

Figure 2.5: Sample Successful Skin Classification (a) Original RGB Image (b) sHSV Image Displayed as RGB (c) Skin Classified Binary Image, *BW*

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

Figure 2.6: Sample Incomplete Skin Classification (a) Original RGB Image (b) sHSV Image Displayed as RGB (c) Skin Classified Binary Image, *BW*

Nonetheless, sufficient over- and underexposure occurred in less than 5% of all images tested and the resulting 93.8% accuracy of the skin classifier remains robust within the visually noisy unconstrained environment. Despite this robustness, the complex, cluttered, and ever-changing background environment still manages to yield significant false positives within each frame. The following section discusses how each

classified image is filtered to reduce these false positives and better isolate face candidates for subsequent detection.

2.3 Filtering and Binary Clustering

The unprocessed skin-classified binary images suffer from two main undesirable effects. Impulse noise exists throughout the binary image and larger, false-positive regions tend to dominate background (non-skin) regions. As the skin-classified binary image will be used to localize the skin candidate for face detection, it is critical that this type of noise is reduced as much as possible. Moreover, reduction in the number of skin "clusters," or connected groups of similarly classified pixels, will alleviate the memory and computational requirements for subsequent binary clustering operations. Figure 2.7 illustrated the entire binary filtering and clustering process. As seen the output of this filter chain is a binary face candidate region of interest (ROI) to be used in face detection.

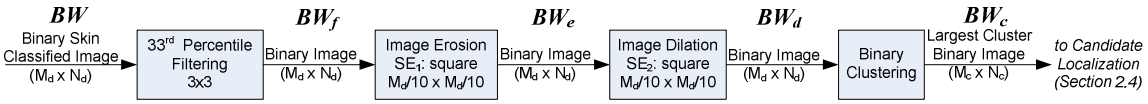


Figure 2.7: Binary Filtering and Clustering Process

The discussed single-element impulse noise, also called "salt-and-pepper" noise, manifests itself as false positives within background regions as well as false negatives within skin regions, namely within the face. Outlined in part within the green bounding boxes, Figure 2.8(b) illustrates the appearance of impulse noise within skin classified region resulting from the original image in (a). Note that the pixelization within binary classified image in this figure is a result of the resampling process detailed in Section 2.2.2. While median filtering is generally used to combat salt-and-pepper noise, this method assumes equal undesirability of each false classification. In general, false

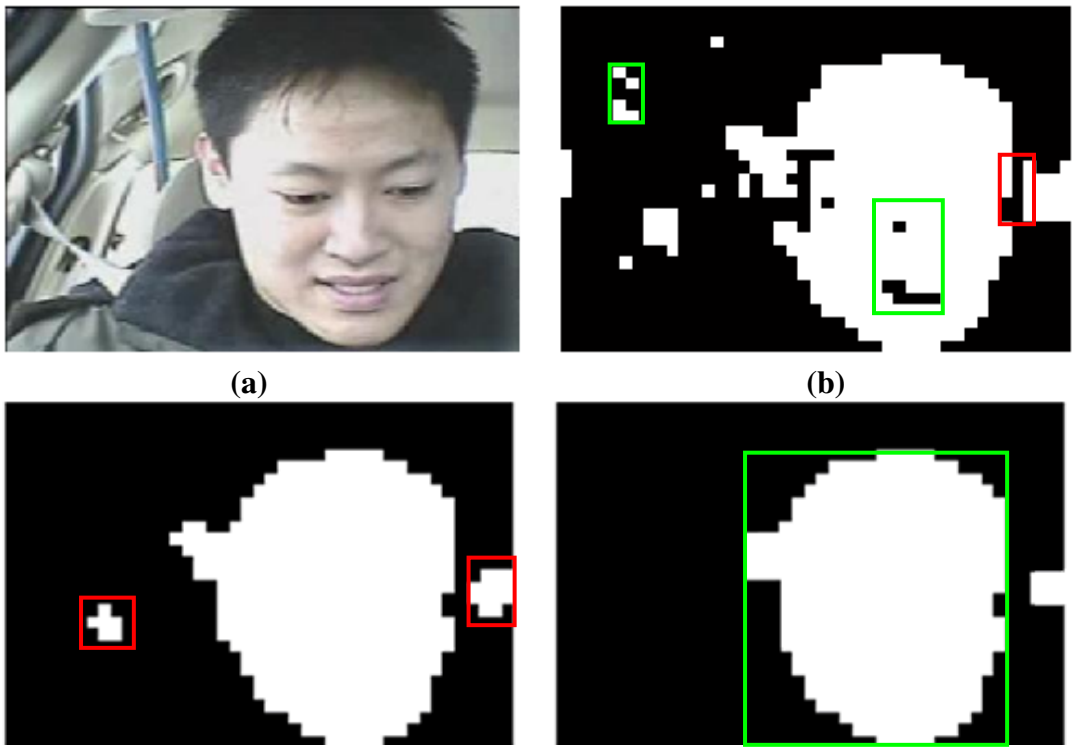
positives were deemed more detrimental to locating the dominant facial skin region within the binary images. Hence, a 33rd percentile order-statistic filter of size 3x3 was selected as a more appropriate filter than the 50th percentile median filter. In essence, the 33rd percentile filter is a nonlinear filter which is passes through the binary image, *BW*. At each pixel location, the pixel and its eight adjacent neighbors (in this 3x3 case) are ordered according to intensity value, returning the intensity value occupying the third lowest position. In equation form, the 3x3 33rd percentile filter can be defined as

$$b = B(p), \qquad B(1) \le B(2) \le \cdots \le (9) \tag{2.15}$$

$$\text{with } p = \frac{9}{3} = 3$$

where b is the output of the filter for the given location, B is the set of nine ordered binary pixel values located within the 3x3 mask arranged from low to high, and p is the percentile index of the filter. Note that the percentile index is derived by taking the index located at the one-third ($\approx 33\%$) mark with respect to the mask's total elements. The size of the filter was set to three in large part due to the coarse resolution resulting from the resampling.

An extra benefit of this filter is that it better separates facial skin regions with skin colored car backgrounds. The red bounding box in Figure 2.8(b) illustrates such a boundary, which is preserved via the 33rd percentile filter from (b) to (c). Had a median filter been applied to this image, the segregation would have disappeared and complicated face candidate localization and subsequent face detection. This is an important performance increase as the cluttered and similarly colored car backgrounds often result in false skin detection.



(a) **(b)**

(c) **(d)**

Figure 2.8: Sample Post-Processing Imagery by Step
(a) Original Image (b) Skin Classified Binary Image
(c) 33rd Percentile Filtered (d) Application of Morphological Operations

Larger regions of false classification can also be problematic when attempting to locate a face within a frame. Figure 2.8(c) outlines such falsely classified skin clusters within the red bounding boxes. To minimize the effect of these larger elements, the binary morphological operations dilation and erosion will be utilized. Binary erosion is in effect the reduction in size of a foreground (binary true) image by a selected structuring element, eliminating a cluster or cluster protrusion completely if it is smaller than said structuring element. According to Gonzalez *et al.* [11] the binary erosion of A by structuring element B, denoted $A \square B$, is defined as

$$A \square B = \{\mathbf{x} \mid (B)_{\mathbf{x}} \subseteq A\}, \; A, B \in Z^2 \qquad \textbf{(2.16)}$$

where Z^2 is the 2D Cartesian integer coordinate space, $(B)_{\mathbf{x}}$ denotes the translation of B by 2D vector $\mathbf{x}$, A is the set of all foreground (binary one) pixel locations within a binary

image, and B is the structuring element [11]. In words, binary erosion is the set of all points z such that B, translated by x, is contained within A. Similarly, binary dilation of A by structuring element B, denoted A⊕B, is defined as

$$A \oplus B = \{ z \mid (\hat{B})_{\mathbf{x}} \subseteq A \}, \; A, B \in Z^2 \qquad \textbf{(2.17)}$$

where $(\hat{B})_{\mathbf{x}}$ indicates reflection of the structuring element about its origin before translation by **x** [11]. Hence, binary dilation can be considered as the set of all displacements **x** such that $\hat{B}$ and A overlap by at least one element.

Referencing the block diagram in Figure 2.7, binary erosion is first applied to the 33rd percentile filtered image, BW_f, to eliminate foreground clusters smaller than the structuring element. This operation is then followed by binary dilation with the same structuring element to return sufficiently large foreground clusters—those larger than the structuring element—to sizes comparable to their pre-erosion dimensions. Even larger clusters are not immune to the elimination of protrusions from the parent cluster that are smaller than the structuring element used. Hence, the selection of the size and shape of the structuring element is important to avoid distorting correctly classified skin classification data, preserving larger clusters that will be considered as face candidates while eliminating falsely classified car background objects. After testing, the finalized structuring element, *SE*, was selected to be square due to the coarse resolution being used. It should be noted that the structuring element can take on any size and shape and are not limited to square or symmetrical masks. The form of the structuring element is outlined in Figure 2.9 below.

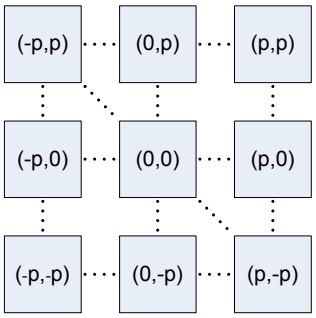


Figure 2.9: Dilation and Erosion Structuring Element

In equation form, the set of all points that comprise the structuring element, *SE*, is

$$\mathbf{SE} = \{(i, j)\} \qquad \forall i, j \tag{2.18}$$

$$\text{for } i, j \in \{-p, -p+1, \ldots p\} \text{ and } p = floor(M_d / 10)$$

where M_d is the size of the downsampled image. Figure 2.8(d) contains the result of the binary erosion and dilation of (c). Notice the elimination of the leftmost background cluster in (c) and the reduction in size of the rightmost cluster which was at least the size of the structuring element at its largest point.

Nonetheless, despite the improved accuracy of the adjusted skin classifier and despite further improvements realized by further filtering, several skin clusters, false and true, may exist in any given image. Since once face is assumed in each image, the largest skin cluster is selected as a region of interest. Selecting the largest skin-classified cluster is accomplished via connected component analysis, or labeling, and subsequently finding the largest labeled region. While the specifics of connected component labeling is not the focus of this report, a rudimentary procedure is as follows:

1. ***First Pass*: Process binary image from top to bottom**
    a. **Label *already processed* neighboring foreground pixels identically using 8-connectedness**
    b. **If no neighbor exists for foreground pixel, create a new label**
2. ***Second Pass*: Recursively combine adjacent labels using 8-connectedness until each independent cluster has only one label**
3. **Calculate cluster areas (pixel counts)**
4. **Return largest cluster bounding box vertices**

Per Figure 2.7, the output of the dilation operation, BW_d, is then clustered and the area of each cluster within the binary image is calculated. Next, the cluster with the most foreground pixels, or largest area, is selected and the cluster's bounding box vertices are returned for further processing on the parent binary image. The largest binary cluster is defined as BW_c and has the dimensions M_cxN_c, where $M_c<M_d$ and $N_c<N_d$. Specifically, this largest cluster will now be input into the face candidate localization algorithm to more accurately bound the facial region for face detection.

2.4 Face Candidate Localization Algorithm and Results

Despite the filtering and classification methods employed, large regions of falsely classified background pixels still comprise part of the largest cluster returned by the pre-processing algorithm outlined in Section 2.3. Resulting from the unconstrained environment, these problem regions include skin-colored car interior regions, such as a car's roof, and window areas. Figure 2.10 illustrates one such distinct, false positive protrusion resulting from a skin colored brick wall behind the car's back windshield. The goal of the face candidate localization algorithm is to simply determine such falsely classified regions attached to the largest cluster and omit them from the ROI's bounding box. Figure 2.11 provides an face candidate localization algorithm flow diagram to be developed within this section.

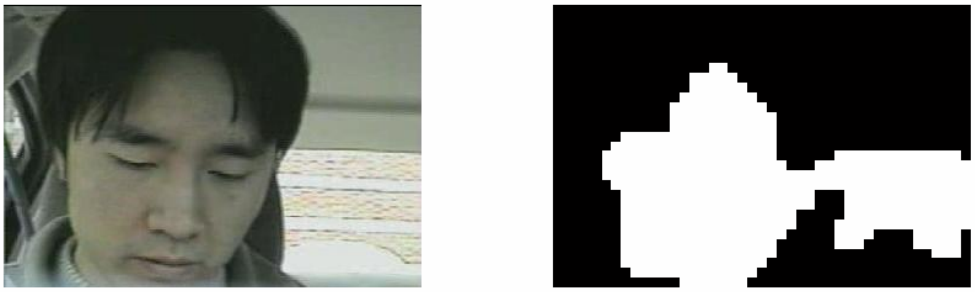


Figure 2.10: Example Face Candidate Protrusion

Per Figure 2.11, the M_cxN_c binary image face candidate, BW_c, is first input to the algorithm. To more effectively separate face candidates which include these background protrusions, an initial candidate screening takes place at the beginning of the algorithm. Sources cite that the average height-to-width ratio of the human face is approximated by the well-known golden ratio of 1.618:1 [24]. Accounting for facial tilt and out-of-frame rotation, typical face candidate ROI height-width ratio were found to exist between values of 1.2:1 and 1.7:1 through database measurements over the training subset utilized in Section 2.2. Hence, all face candidate ROI's whose height-to-width ratio, M_c/N_c, does not fall within the range [1.2,1.7] will be subject to the remainder of the ROI pruning process.

For images which fall outside of the acceptable height-width ratio, further filtering takes place. To eliminate clear protrusions which are comparable in size to the face region itself a two-pass spatial filtering technique was employed. This technique locates sudden deviations in cluster configuration between the top and bottom of the face candidate cluster. While other more accurate methods, such as flood-fill techniques, exist to segment binary clusters, these methods are more computationally intensive, requiring several iterations of initial condition- and parameter-dependent morphological operations. Hence, this computationally inexpensive method was employed to roughly locate distinct

binary cluster protrusions similar to that in Figure 2.10, while preserving the roughly vertically oriented elliptical face region.

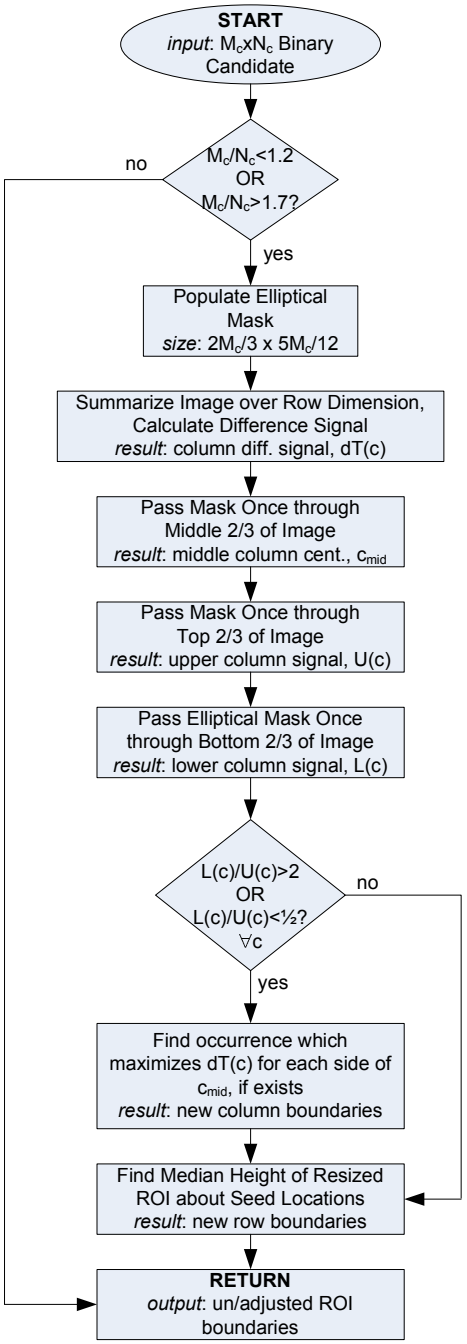


Figure 2.11: Face Candidate Localization Algorithm Flow Diagram

The spatial filtering discussed is the result from passing an elliptical binary mask once through the top two-thirds and bottom two-thirds of the face candidate binary image, BW_c. The height of the elliptical binary mask, called H, was chosen to be two-thirds the input candidate ROI's height, M_c. The width of the ellipse was chosen to mimic the average dimensions of the human face, which is 1.6 times less than its height. Hence, the final size of the elliptical mask is M_h x N_h, where $M_h = floor(2M_c/3)$ and $N_h = floor(5M_c/12)$. The composition of the mask, H, is defined per the following equation

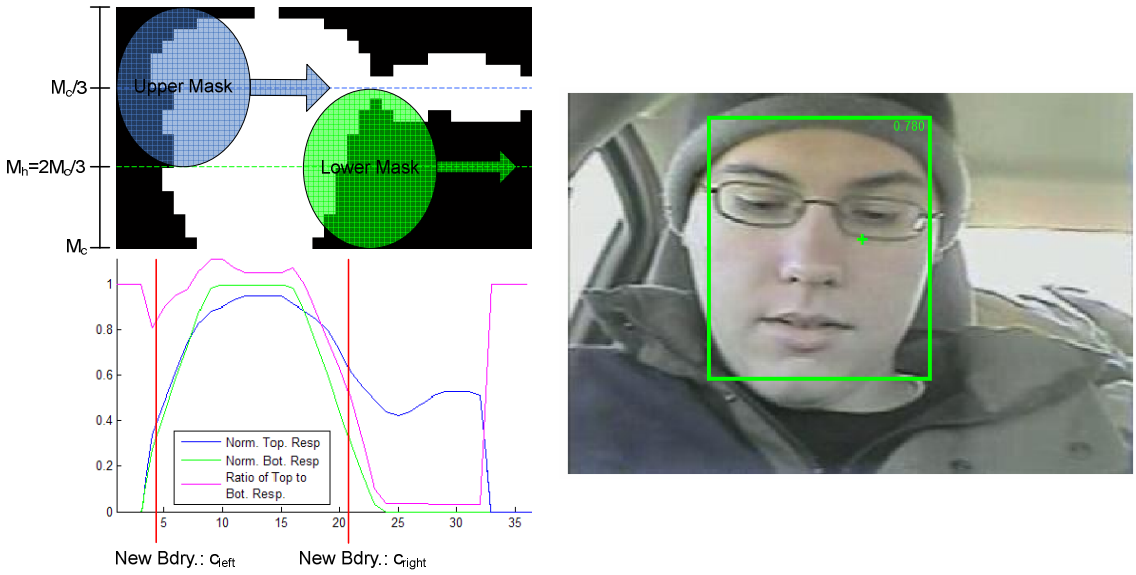
$$H(\mathbf{z}) = \begin{cases} 1 & \text{if} \quad \mathbf{z} \cdot \mathbf{z}^{\mathrm{T}} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.19}$$

$$\text{where } \mathbf{z} = \left[\frac{r - r_{H,cen}}{r_{H,cen}} \quad \frac{c - c_{H,cen}}{c_{H,cen}} \right],\ c_{H,cen} = N_h/2, \text{ and } r_{H,cen} = M_h/2$$

where r and c are the row and column location of the elliptical mask. Thusly defined, the elliptical mask is not convolved with the face candidate binary image in the strictest sense. Rather, the elliptical mask, H, is passed once through the top two-thirds and once through the bottom two-thirds of the of the candidate ROI, centered about one thirds and two-thirds of the candidate ROI's height, respectively. At each column location, the mask and image are multiplied and then summed by element, returning a value equivalent to the total number of skin-classified pixels enclosed within the mask H at that location. Let $U(c)$ and $L(c)$ be the column signals resulting from the upper and lower passes through the candidate ROI, BW_c, respectively. To preserve the accuracy of the spatial filtering, it should be noted that the input binary image, BW_c, was padded column-wise with $N_h/2$ zeros on each side of the largest cluster. Then the ratio of the upper signal to the lower signal is given by:

$$R(c) = \frac{U(c) + \varepsilon}{L(c) + \varepsilon} \qquad c = 1,2,\ldots,N_c \tag{2.20}$$

where ε is a small positive integer introduced to safeguard against $L(c)$=0. This ratio signal effectively shows the relative distribution of the face candidate cluster with $R(c)$>1 indicating a greater concentration at the cluster's top and with $R(c)$<1 indicating a greater concentration at the cluster's bottom. Figure 2.12(a) contains an annotated example of the relative size and shape of the elliptical mask, the resulting upper and lower column signals, U(c) and L(c), as well as the ratio signal, R(c). Note that for clarity this example normalizes each column signal to the area of the elliptical mask.



(a) **(b)**

Figure 2.12: Sample Face Candidate Localization Process
(a) Original Face Candidate Cluster and Spatial Filter and Ratio Responses
(b) Successfully Modified Bounding Box

After the ratio signal has been calculated over the width of the binary image, the binary image is summed across the row dimension yielding a total column vector, T(c). Equivalently, this total signal can be expressed as

$$T(c) = \sum_{r=1}^{M_c} BW_c(r,c) \qquad c = 1,2,...,N_c \tag{2.21}$$

Where r and c are the row and column indices, respectively, from the face candidate binary image. Next, an absolute difference signal, $dT(c)$, is derived from T(c) per the following equation:

$$dT(c) = abs\big(T(c+1) - T(c)\big) \qquad c = 1,2,...,N_c - 1 \tag{2.22}$$

Next, a value of two is chosen to select the factor by which the upper and lower signals can deviate and still be considered part of the facial region. Then, letting $\mathbf{C}$ be the set of all column locations for which $R(c)>2$ or $R(c)<0.5$, the new horizontal boundaries, $c_{c,left}$ and $c_{c,right}$, of the candidate ROI is then selected by the following equation.

$$\begin{aligned} c_{c,left} &= \begin{cases} \arg_c \max\{dT(c) \mid c \in \mathbf{C}\} & 1 \le c \le c_{mid} \quad \text{if} \quad \mathbf{C} < c_{mid} \neq \varnothing \\ 1 & \text{otherwise} \end{cases} \\ c_{c,right} &= \begin{cases} \arg_c \max\{dT(c) \mid c \in \mathbf{C}\} & c_{mid} < c \le N_c \quad \text{if} \quad \mathbf{C} > c_{mid} \neq \varnothing \\ 1 & \text{otherwise} \end{cases} \\ \text{where } c_{mid} &= \text{median}\big\{c \mid T(c) = \max\big(T(c)\big)\big\} \qquad c = 1,2,...,N_c \end{aligned} \tag{2.23}$$

where $c_{c,left}$ and $c_{c,right}$ are the new left and right ROI boundaries, respectively, and c_{mid} is the median value of c for which $T(c)$ is maximum over the candidate's entire width. In words, the new boundaries are selected by maximizing the difference signal for all locations where the upper and lower mask differ by a factor of two. This method effectively selects new boundaries located where an abrupt change in top-bottom concentration occurs.

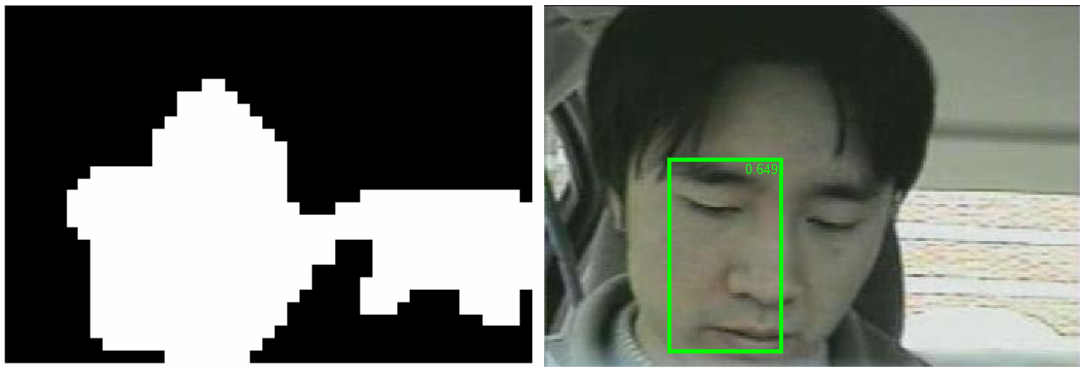
Lastly, new top and bottom boundaries, $r_{c,top}$ and $r_{c,bot}$, are created by median filtering the top and bottom cluster edges within $N_c^{'}/20$ pixels of the new ROI's horizontal center. Hence, the new face candidate ROI is now bounded horizontally over

$[c_{left},c_{right}]$ and vertically over $[r_{top},r_{bot}]$, noting that these ranges are referenced to the origin of the original candidate binary image, BW_c. Figure 2.12(b) illustrates a successfully modified ROI bounding box resultant from this algorithm. Note the correspondence between where the ratio signal drops below one-half and where the new boundaries are located. Also note that these new coordinates are referenced to the downsampled (M_dxN_d) image space and will require conversion back to the original resolution space.

Due to its simplicity, this face candidate localization algorithm achieves only average performance when applied to a test set. The test set was composed of an 160-image subset of the AVICAR database, selected as described in Section 2.2 (page 21), but utilizing a set of 40 different subjects to avoid results contamination. Table 2.3 summarizes the performance of the localization algorithm when applied to this test set, which also generated the overall face detection algorithm performance values in Section 2.6.2 (refer to Table 2.5 on page 56). After skin classification and processing, 39 of the 52 images which required cropping yielded satisfactory results. Satisfactory results are defined as adjusted ROI's which included 75% to 125% of the visible face.

Table 2.3: Face Localization Algorithm Success Rates

Face Localization Algorithm Result	**Instances**	**Percentage**
Tightly Bounds Face (75% to 125% of Visible Face Bounded)	**39**	**75.0%**
Excess Bounds (>125% of Visible Face Area Bounded)	5	9.6%
Under-Bounds (<75% of Visible Face Bounded)	8	15.4%
Total Images Resized	*52*	



(a) (b)

Figure 2.13: Sample Unsuccessful Face Candidate Localization
(a) Skin Classified, Filtered Face Candidate (b) Face Localization Failure

Under-bounding is defined as reducing the size of the ROI such that less than 75% of the face is contained. Excess bounding is defined as bounding the face entirely such that 125% of the face's rectangular footprint is contained within the ROI. Per Table 2.3, the face candidate localization algorithm tends to under-bound candidate faces more often than it excess bounds. Under- and excess-bounding is largely a result of incomplete skin classification, often caused by overly bright or dark lighting conditions or by obscuring of the face. Containing an example of under-bounding, a sample face localization failure is contained in Figure 2.13. Nonetheless, it should be noted that this ROI was still successfully classified as a face.

2.5 Face Model Joint Histogram Estimation

A critical component of face detection is modeling the variable human face such that a given algorithm provides accurate, repeatable, and reliable results. For this reason, selection of a proper feature set and development of an extensive, representative training set is often critical for successful face detection algorithms. Previous work in automatic speech recognition based on the same AVICAR database employed three face models

extracted directly from arbitrarily chosen database images containing light-, medium-, and dark-skinned individuals [4]. This study further applied each of these models to each of five fixed ROI's spread around a given frame. While results from this study was commendable, it is the goal of this work to consolidate the face model into a single, cohesive, and more representative model. Moreover, the intent of this project is to apply this improved model to ROI's of variable shapes, sizes, and locations throughout the frame as they are detected.

This section explores a unique method for modeling the variable human face in the unconstrained car environment. A histogram-based shifted hue and saturation feature space will be used as a basis for face model and candidate joint probability density function (PDF) approximation. Specifically, the face detection scheme employed in this work is founded on the observation that a face's joint distribution is a function of average intensity value. Supporting evidence for this observation, the means by which training data was obtained, and the final face model will be described in the following sections in detail.

2.5.1 The Epanechnikov Kernel

Before statistical approximations of face model and face candidates can be derived for comparison within the face detection algorithm, a spatial weighting of image data must be selected. This spatial weighting is referred to as a Parzen window, or kernel, and can follow any number of well-established distributions. For instance, all of the pixels within the training data or face candidate's rectangular ROI could contribute equally to the statistical representation. Consequently, this simple model would potentially weigh background (non-face) pixels near the ROI's perimeter equally to face

pixels near the ROI's center. The Epanechnikov kernel, on the other hand, weights a given ROI, heavier towards the center and radially less towards the ROI's perimeter. Hence, the Epanechnikov kernel minimizes the effect of background pixels and skin edge pixels which are not always representative of the face itself. Crow utilized the Epanechnikov kernel noting similar advantages and associated performance increases [4]. As will be developed, another benefit of the Epanechnikov kernel is that it is elliptically symmetric about the ROI's central coordinate, mirroring the natural shape of the human face within the ROI.

While the Epanechnikov kernel is defined over any dimensionality, the two-dimensional kernel will be applied as it will be used to weight pixels spatially over a rectangular ROI. The mathematical form of the two-dimensional Epanechnikov kernel is given by

$$K_E(\mathbf{z}) = \begin{cases} K_o\left(1 - \mathbf{z}\cdot\mathbf{z}^{\mathbf{T}}\right) & \text{if} \quad \mathbf{z}\cdot\mathbf{z}^{\mathbf{T}} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.24}$$

$$\text{where } \mathbf{z} = \left[\frac{r - r_{cen}}{r_{cen}} \quad \frac{c - c_{cen}}{c_{cen}}\right],\ c_{cen} = N_{ROI}/2 \text{, and } r_{cen} = M_{ROI}/2$$

where K_o is a normalizing constant, r and c are the row and column indices, respectively, of the pixel location within the ROI, and M_{ROI} and N_{ROI} are the height and width of the given ROI, respectively. Note that this form of the Epanechnikov kernel is radially symmetric and allows for non-square regions of interest ($M_{ROI} \neq N_{ROI}$). Figure 2.14 displays a sample Epanechnikov kernel with height 30 and width 20. Note that the kernel's maximum occurs at the ROI's center of [15,10]**.**

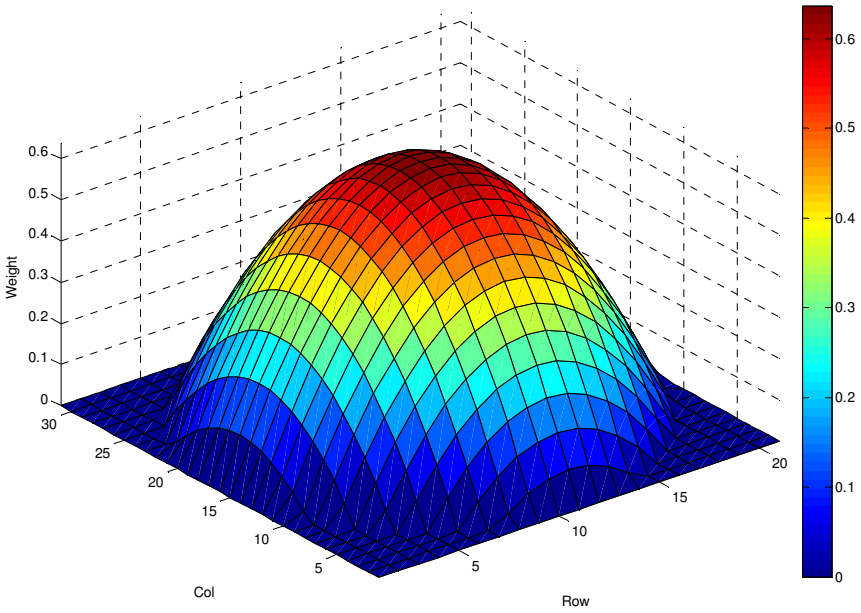


Figure 2.14: Sample Epanechnikov Kernel of ROI Size 30x20

The Epanechnikov kernel will effectively be used as a means to weight training and candidate features based on their spatial location within the ROI. While the logistics will be discussed in the following subsection, a joint shifted hue and saturation feature space will be used to represent candidate face ROI's and face models. Furthermore, a histogram approximation of the model and candidate joint distribution will be employed. Noting that normalization will occur after these distributions are constructed, the constant, K_o, can be disregarded, reducing Equation 2.24 to the form

$$K_E(\mathbf{z}) = \begin{cases} 1 - \mathbf{z} \cdot \mathbf{z}^{\mathbf{T}} & \text{if} \quad \mathbf{z} \cdot \mathbf{z}^{\mathbf{T}} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.25}$$

where $\mathbf{z}$ is the location vector having the same definition the parent equation. This simplification assumes that sum of all joint histogram bins, discussed in Sections 2.5.2 and 2.5.3, will be normalized to unity to be consistent with any valid probability density function.

2.5.2 Face Detection Feature Space

Building upon previous work, a joint shifted hue and saturation feature space was selected as the basis for face detection. This feature space was shown to effectively classify faces over a range of skin tones and lighting conditions [4]. The joint shifted hue and saturation feature space captures skin color information as well as the variation in saturation incurred around facial features such as eyes, nose, and mouth. Monotone car interiors and/or highly saturated background colors, for example, would simply not provide the saturation variance yielded contained in facial regions.

With the feature space selected, another design decision was to approximate the joint probability density function as a histogram which quantizes the discussed two-dimensional feature space into a finite number of bins. Histogram is a nonparametric density estimation method which yield memory efficient and intuitive results. Moreover, quantizing the model and candidate's density functions achieves two important goals. First, the histogram approach reduces the computational complexity of PDF estimation and subsequent comparison. Moreover, histogram approximation maintains a scale- and rotation-invariant comparison environment. In addition to the histogram approximation, the Epanechnikov kernel discussed in the previous section will weight a pixels contribution to the estimated PDF per its spatial location.

Intensity information was not included in the skin detection feature space as illumination-invariant features are more desirable as faces need to be detected in all lighting conditions. While illumination content remains relatively constant within any given image, it will be shown in the following sections that the average illumination within a given ROI directly impacts the distribution of the face within the joint shifted

hue and saturation feature space. Average intensity was chosen as an easily calculable metric which represents the face's ambient lighting conditions. For the sake of consistency, the illumination space was also quantized into a discrete number of bins and the Epanechnikov kernel will weight a pixel's contribution to the average illumination.

To incorporate intensity information, let the two-dimensional histogram-approximated joint probability density function given the ROI average illumination be defined as $P(h,s \mid I_{avg})$, where h and s are the shifted hue and saturation components, respectively, of a pixel's sHSV color triplet and I_{avg} is the average illumination (value component) of the region of interest. It should be noted that the feature space remains two-dimensional despite the incorporation of the illumination information as the shifted hue and saturation joint PDF is conditional on I_{avg}, not a function of it. This particular decision will be justified in the following section. The formula used to calculate kernel-weighted average illumination is given by

$$I_{avg} = \frac{1}{V_E} \sum_{r=1}^{M_{ROI}} \sum_{c=1}^{N_{ROI}} K_E(\mathbf{z}) \cdot V(r,c) \qquad \textbf{(2.26)}$$

$$\text{with } V_E = \sum_{r=1}^{M_{ROI}} \sum_{c=1}^{N_{ROI}} K_E(\mathbf{z})$$

where $K_E(\mathbf{z})$ is the Epanechnikov kernel of Equation 2.25, V_E is the volume of the kernel, and M_{ROI} and N_{ROI} are the height and width, respectively, of the region of interest.

Borrowing from pervious work, the histogram bin count for each feature component, h and s, and the intensity information, I_{avg}, will be segmented into 16 discrete bins uniformly spread about the respective spaces. Letting N_{bin} refer to this bin count of 16, a total of N_{bin}^3 bins comprise an N_{bin}-by-N_{bin} two-dimensional space over N_{bin} distinct illumination bins. This stated value for N_{bin} minimizes storage requirements while

mitigating the risk of overfitting the actual distribution. Nonetheless, the stated bin count should not be considered an optimal choice. Index conversion to bin representation is performed as follows:

$$b(f) = \begin{cases} floor(f \cdot N_{bin}) + 1 & 0 \le f < 1 \\ N_{bin} & f = 1 \end{cases} \text{ for } f \in [0,1] \qquad \textbf{(2.27)}$$

where f is the shifted hue (h), saturation (s), or average illumination (I_{avg}) components to be converted and $floor(\bullet)$ rounds the argument down to the nearest integer. Note that the addition of one to Equation 2.27 is a result of the one-indexing utilized by Matlab, the environment in which all analysis and development was performed.

Let i denote the bin index of the ROI's average illumination resulting from Equations 2.26 and 2.27 and let $b_h(\mathbf{x})$ and $b_s(\mathbf{x})$ denote the bin index of the shifted hue and saturation components, respectively, at the location vector $\mathbf{x}$=[r, c] within the ROI. Finally, let the final illumination-dependent, kernel-weighted, histogram-approximated joint PDF of the M_{ROI}-by-N_{ROI} ROI be defined as $\mathbf{P}_i$, which is calculated via the following series of equations.

$$\begin{aligned} &\text{Initialize,} && \mathbf{P}_i = \mathbf{0} \\ &\text{Populate,} && P_i(b_h(\mathbf{x}), b_s(\mathbf{x})) = P_i(b_h(\mathbf{x}), b_s(\mathbf{x})) + K_E(\mathbf{z}) \quad \forall \mathbf{x} \in \text{ROI} \qquad \textbf{(2.28)} \\ &\text{Normalize,} && \mathbf{P}_i = \frac{\mathbf{P}_i}{\sum_{j=1}^{N_{bin}} \sum_{k=1}^{N_{bin}} P_i(b_j, b_k)} \end{aligned}$$

where $K_E(\mathbf{z})$ is the kernel from Equation 2.25 (with $\mathbf{z}$ being a function of $\mathbf{x}$), $P_i(b_h,b_s)$ is the value of $\mathbf{P}_i$ at bin location $[b_h, b_s]$, and $\mathbf{P}_i$ is of size N_{bin}-by-N_{bin}. Note that the histogram density estimation effectively utilizes a square, uniform window function of dimension $1/N_{bin}$ within the feature space as a result of the flooring operation in Equation 2.27. The following sections will further develop the face model as well as exactly how face model and candidate PDF's are estimated.

2.5.3 Forming the Face Model Joint Density Estimators

Before the face model's dependence on illumination was developed, a study was performed to establish the effect lighting conditions have on the spectral content of the face within the sHSV color space. This study involved the generation of training set which contained 150 images from five individuals of varying skin tone taken under a range of ambient lighting conditions. These subjects were centered in front of a video camera utilizing the same AVI compression and comparable resolution employed by the AVICAR database [17]. Subjects were instructed to maintain a neutral, expressionless face while a series of images were taken under lighting conditions ranging from bright to dark. Care was taken to ensure that across each subject average illumination levels remained within $1/30$ of each of the 30 values uniformly spread over the range [0,1].

For each image within the training set, the kernel-weighted average intensity of Equation 2.26 and the joint PDF histogram of Equation 2.28 were calculated for each image after conversion to the sHSV color space. Selected results obtained by three of the five subjects are detailed in Figure 2.15 through **Figure 2.17** representing light-, medium-, and dark-skinned individuals, respectively. It can be seen that changes in average illumination directly impact the distribution of the largely unimodal (singly peaked)

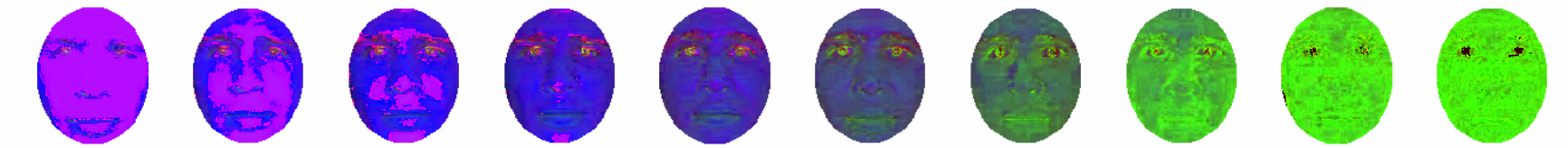
shifted hue and saturation joint PDF. Furthermore, it can be seen across all PDF histograms that a majority of the hue content is contained within three or four histogram bins across all illumination values. However, saturation content varies from more tightly concentrated at low values under high illumination to roughly three times more spread about the saturation axis under low illumination. Differences in the PDF histograms between light and dark skin tones were slight, involving a positive one-bin shift of the general unimodal distribution along the hue axis. Moreover, at high illumination levels spreading about the hue axis occurred largely due to overexposure at the imaging device itself. Hence, the decision was made to replicate this dependence in the final face model.

Hence, the entire 150-image training database was utilized to construct a joint shifted hue and saturation histogram-estimated PDF for each discrete ROI average illumination bin per the discussion in Section 2.5.2. Utilizing the calculated joint PDF histogram and average illumination value for each of the training set images, the final face model set was derived via the following equations:

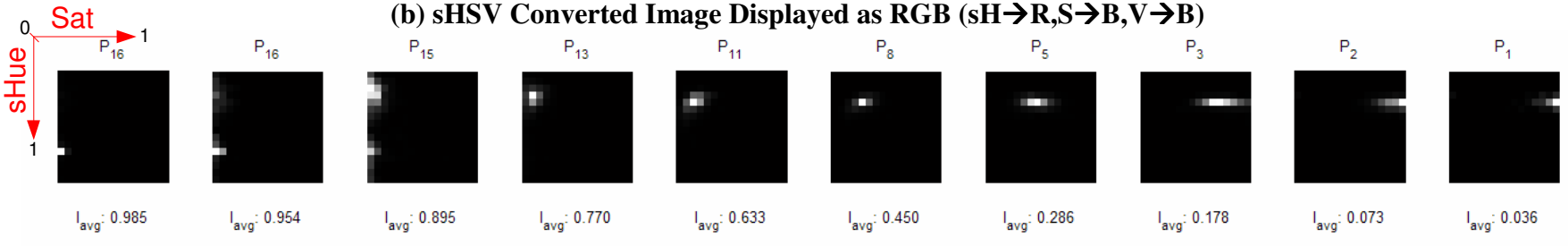
$$\mathbf{Q}_i = \frac{\sum_{n|j=i} \mathbf{P}_{j(n)}}{\sum_{n|j(n)=i} \sum_{b_h=1}^{N_{bin}} \sum_{b_s=1}^{N_{bin}} P_{j(n)}(b_h, b_s)} \qquad \begin{matrix} i = 1, 2, ..., N_{bin} \\ n = 1, 2, ...N_{DB} \end{matrix} \tag{2.29}$$

where $j(n)$ is the average illumination of the training set image n defined in Equation 2.27, N_{DB} is the total number of training set images equal to 150, $\mathbf{P}_{j(n)}$ and $P_{j(n)}(b_h,b_s)$ are defined in Equation 2.28 with $j(n)=i$, and $\mathbf{Q}_i$ is the resulting face model PDF histogram approximation for average illumination level i. In words, the face model histogram set is derived by summing each histogram over the training set whose parent image have the average illumination level and then normalizing each illumination level's PDF histogram independently to unity.

(a) Original Images from Subject 1 Training Set



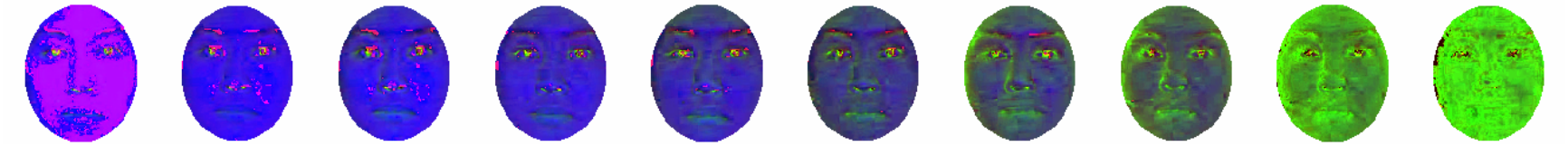
(b) sHSV Converted Image Displayed as RGB (sH→R,S→B,V→B)



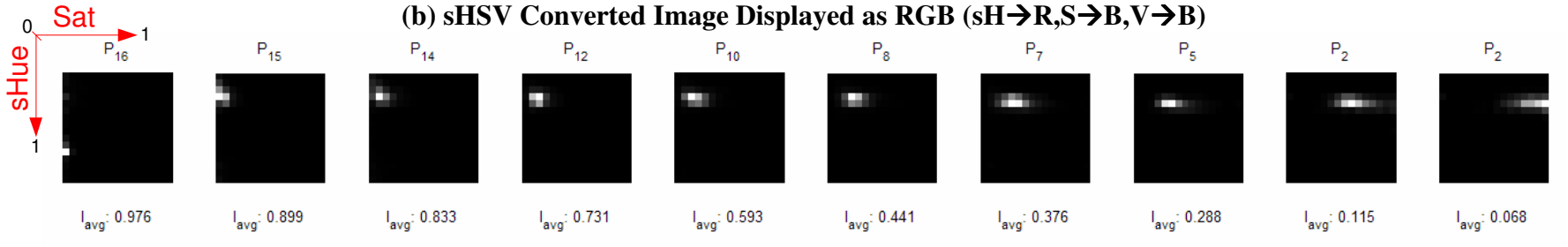
(c) Corresponding Joint Shifted Hue and Saturation Histogram*

Figure 2.15: Face Model Illumination Dependence Training Set, Subject 1
for clarity each histogram has been normalized to it's own maximum value

(a) Original Images from Subject 2 Training Set

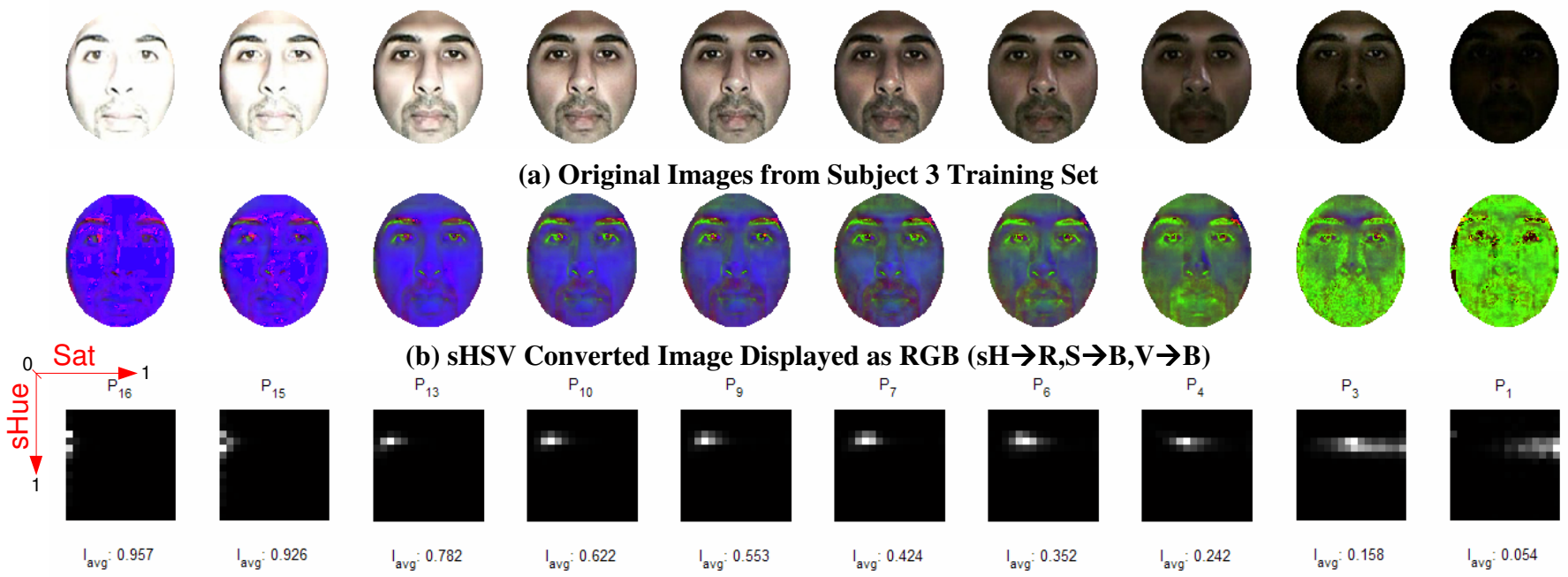


(b) sHSV Converted Image Displayed as RGB (sH→R,S→B,V→B)



(c) Corresponding Joint Shifted Hue and Saturation Histogram*

Figure 2.16: Face Model Illumination Dependence Training Set, Subject 2
for clarity each histogram has been normalized to it's own maximum value



(a) Original Images from Subject 3 Training Set

(b) sHSV Converted Image Displayed as RGB (sH→R,S→B,V→B)

(c) Corresponding Joint Shifted Hue and Saturation Histogram*

Figure 2.17: Face Model Illumination Dependence Training Set, Subject 3
for clarity each histogram has been normalized to it's own maximum value

The resulting face model PDF histogram approximation across each illumination level is displayed in Figure 2.18. Here the value of I_{bin} refers to the illumination component value which corresponds to the center (midpoint) of the discrete illumination bin, i, as calculated in Equation 2.27. This face model histogram set will be stored in memory to be accessed by the face detection algorithm discussed in Section 2.6 to follow.

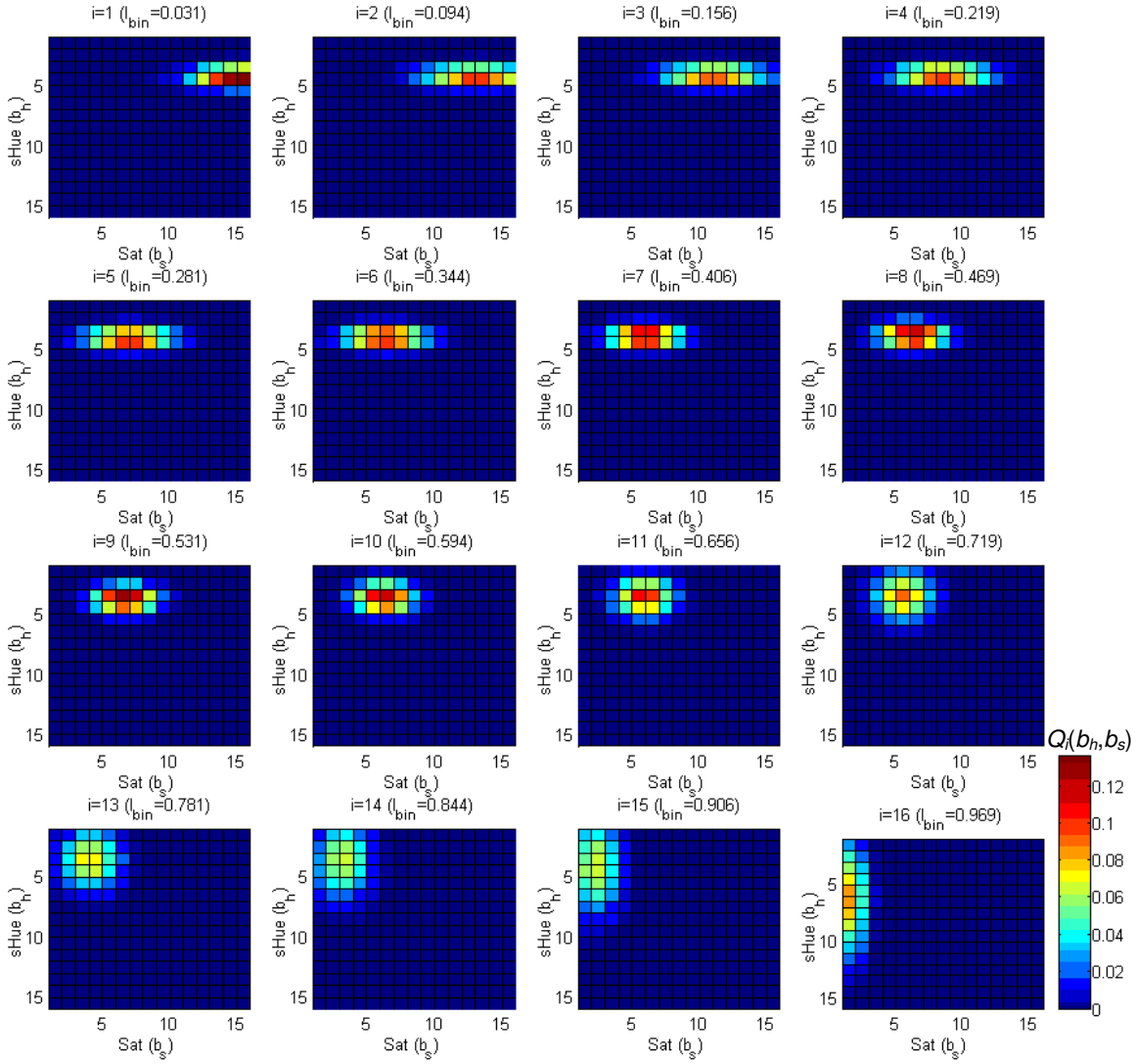


Figure 2.18: Joint sHue and Saturation Histogram-Estimated PDF's over Average Illumination Bin Number

2.5.4 Forming the Face Candidate Joint Density Estimators

With the face model density estimate in place, the face candidate density joint PDF must be constructed so that it can be compared with the model distribution. Derivation of the candidate's histogram approximated joint PDF is straightforward as it only entails the histogram associated with one ROI and its corresponding average illumination value. To complete this task, the face candidate which results from the face candidate localization algorithm (see Section 2.4) must be converted to the original coordinate and resolution space. Next, the converted sHSV ROI will be kernel weighted and the histogram estimation process will take place.

Recall that the output of the face localization algorithm was a series of bounds referenced to the largest cluster binary image, BW_c, before localization. Binary image BW_c is a sub-region of the original skin classified binary image, BW, incurring a translation in image origin. Furthermore, binary image BW is a product of reduced (downsampled) resolution from that of the original image. The boundaries returned from the localization algorithm were a set of row and column locations referenced to the candidate binary image, BW_c. The column boundaries, $c_{c,left}$, $c_{c,right}$, signify the left and right borders of the face candidate. The row boundaries, $r_{c,top}$, $r_{c,bot}$, signify the top and bottom borders of the face candidate. Both the translation and resampling must be corrected in order to bound the candidate within the original sHSV image, I. This conversion of boundary coordinates is as follows:

$$\begin{aligned}[c_{I,left}, c_{I,right}] &= F \cdot \left([c_{c,left}, c_{c,right}] + c_o - 1\right) \\ [r_{I,top}, r_{I,bot}] &= F \cdot \left([r_{c,top}, r_{c,bot}] + r_o - 1\right)\end{aligned} \tag{2.30}$$

where F is the downsampling factor and r_o and c_o are the row and column displacements of BW_c from BW. Note the subtraction of one from each row and column index is a result of one-indexing implemented by Matlab.

With these new boundary coordinates, let the face candidate ROI generated from the original sHSV image, I, be defined as

$$I_c = I(r,c) \quad \text{for} \quad r_{I,top} \le r \le r_{I,bot}, c_{I,top} \le c \le c_{I,bot} \tag{2.31}$$

where I_c is the M_cxN_cx3 face candidate ROI in sHSV color space. Thus, let the weighted and histogram-approximated joint PDF as $\mathbf{P}_i$ defined in Equation 2.28, where i denotes the average ROI illumination level as defined in Equation 2.27. This face candidate joint density estimate, $\mathbf{P}_i$, will be compared with the face model histogram of the same illumination level, $\mathbf{Q}_i$, via the face detection algorithm outlined in the next section.

2.6 Face Detection and Test Results

With a face model and candidate distributions in hand, candidate ROI's output from the skin detection and filtering algorithm can now be processed for the presence of a face. The face detection algorithm implemented in this work utilizes the Bhattacharyya coefficient as a means by which the similarity between the generated face model joint histogram and that of a candidate ROI is measured. Success rates for the face detection scheme as a whole—from skin classification to face detection result—achieved 95% accuracy over a large subset of the AVICAR database. The following sections will detail the Bhattacharyya coefficient, its application to model and candidate PDF comparison, the process of fine tuning the detector's parameters, and the overall performance of the face detection algorithm.

2.6.1 The Bhattacharyya Coefficient

Countless (dis)similarity measures exist for the comparison of probability density functions. Popular measures include the Euclidean and Minkowski distances or statistical measures as well as similarity functions such as spectral angle criteria, histogram intersection, and Kullback divergence. More notably, the Mahalanobis distance metric requires more computationally intensive calculation of the mean vectors and covariance matrices for both known and unknown sets. Moreover, the Bhattacharyya coefficient was proven to be a common method to compare distributions [2][3][4][7] and was adopted in this work.

The major advantage of the Bhattacharyya coefficient is that, unlike the Mahalanobis distance, it requires no statistical measures from each distribution, drastically reducing computation time and complexity. Moreover, face detection via the Bhattacharyya coefficient was deemed superior to methods such as histogram intersection and Kullback divergence [3]. While it is not the focus of this work, the Bhattacharyya coefficient also lends itself well to tracking algorithms which employ the mean shift algorithm [4]. It should be noted that the Bhattacharyya coefficient is merely a measure, not a metric structure, as it violates one of the defining axioms as Derpanis and Comaniciu *et al*. notes [7][3]. While easily calculable metrics have been derived from the Bhattacharyya coefficient, the Bhattacharyya coefficient was shown to be sufficient as a measure for the purposes of face detection within the shifted hue and saturation feature space [4].

The one-dimensional, discrete Bhattacharyya coefficient, ρ, between the m-bin candidate histogram, $\mathbf{p}$, and model histogram, $\mathbf{q}$, is defined as

$$\rho(\mathbf{p},\mathbf{q}) = \sum_{x=1}^{m} \sqrt{p(x) \cdot q(x)} \tag{2.32}$$

where **p** is the candidate vector, **q** is the model vector, and $p(x)$ and $q(x)$ are the density of the candidate and model vectors, respectively, at element x. It can be shown that the one-dimensional Bhattacharyya coefficient has the geometric interpretation of the cosine of the angle between the two length-m unit vectors, also known as the dot product. Remapping the definition of the Bhattacharyya to two dimensions, the Bhattacharyya coefficient can be defined as

$$\rho(\mathbf{P},\mathbf{Q}) = \sum_{h=1}^{m} \sum_{s=1}^{n} \sqrt{P(h,s) \cdot Q(h,s)} \tag{2.33}$$

where $\rho(\mathbf{P}, \mathbf{Q})$ is the Bhattacharyya coefficient between the m-by-n bin candidate histogram **P** and m-by-n bin model histogram **Q**, and $P(h,s)$ and $Q(h,s)$ are the density of the candidate and model histograms, respectively, at bin location $[h, s]$. The form of the Bhattacharyya coefficient in Equation 2.33 also lends itself well to the element-wise matrix multiplication, also called "dot multiplication," utilized by the Matlab environment. As both model and candidate distributions are normalized to one, it can be shown that the domain for Equation 2.32 and 2.33 is [0,1]. When both distributions are equal, the Bhattacharyya coefficient equals unity, meaning a perfect match. Conversely, lower valued Bhattacharyya coefficients indicate a poor match between the two distributions.

Hence, a Bhattacharyya coefficient face detection scheme will be implemented such that sufficiently high ρ values will result in face classification of the candidate ROI. Let *Face* denote the class which indicates the presence of a face within a given ROI and

let *NonFace* indicate the classification for the ROI without a face. Thus let the ROI face classifier be defined as

$$C_{face}(\mathbf{P}_i) = \begin{cases} Face & \rho(\mathbf{P}_i, \mathbf{Q}_i) \geq \rho_{thresh} \\ NonFace & \text{otherwise} \end{cases} \quad \textbf{(2.34)}$$

where ρ_{thresh} is the Bhattacharyya coefficient threshold, $\mathbf{P}_i$ is the face candidate joint PDF estimate defined in Section 2.5.4, and $\mathbf{Q}_i$ is the face model joint PDF estimate defined in Section 2.5.3. In words, the face classifier computes the Bhattacharyya coefficient between the ROI's face candidate histogram and the face model histogram of matching illumination level, classifying the ROI as a face if the coefficient exceeds the predetermined threshold, ρ_{thresh}.

The Bhattacharyya coefficient threshold was selected via iterative analysis over the training set as a means to minimize false negative and false positive error rates. The training set for this analysis was composed of 160 images from the AVICAR database which were skin classified, filtered, and clustered in accordance with Sections 2.2 through 2.3 of this document. However, after connected component labeling, the Bhattacharyya coefficient was calculated for *each* sufficiently large cluster from the processed test set image. In this case, sufficiently large clusters were required to be at least half the area (pixel count) of the corresponding image's largest cluster. Keep in mind that the final algorithm will only considers the largest cluster as the face candidate as discussed. Classification results were then manually verified for each cluster within the set. Table 2.4 contains the result of this analysis over varying Bhattacharyya coefficients.

Table 2.4: Face Detection Failure Rates over Bhattacharyya Coefficient Threshold

Face Classification Error	Bhattacharyya Coefficient Threshold, ρ_{thresh}								
	0.1	**0.3**	**0.4**	**0.45**	**0.5**	**0.55**	**0.6**	**0.7**	**0.9**
False Negative Rate (%)	3.9	4.3	4.3	5.5	**8.6**	12.1	24.2	39.3	52.3
False Positive Rate (%)	68.2	42.2	32.8	24.2	**12.2**	19.7	15.1	12.1	12.1
Average Error Rate (%)	36.1	23.3	18.6	14.9	**10.4**	15.9	19.7	25.7	32.2

(a) ρ_{thresh}=0.7, ρ=0.538

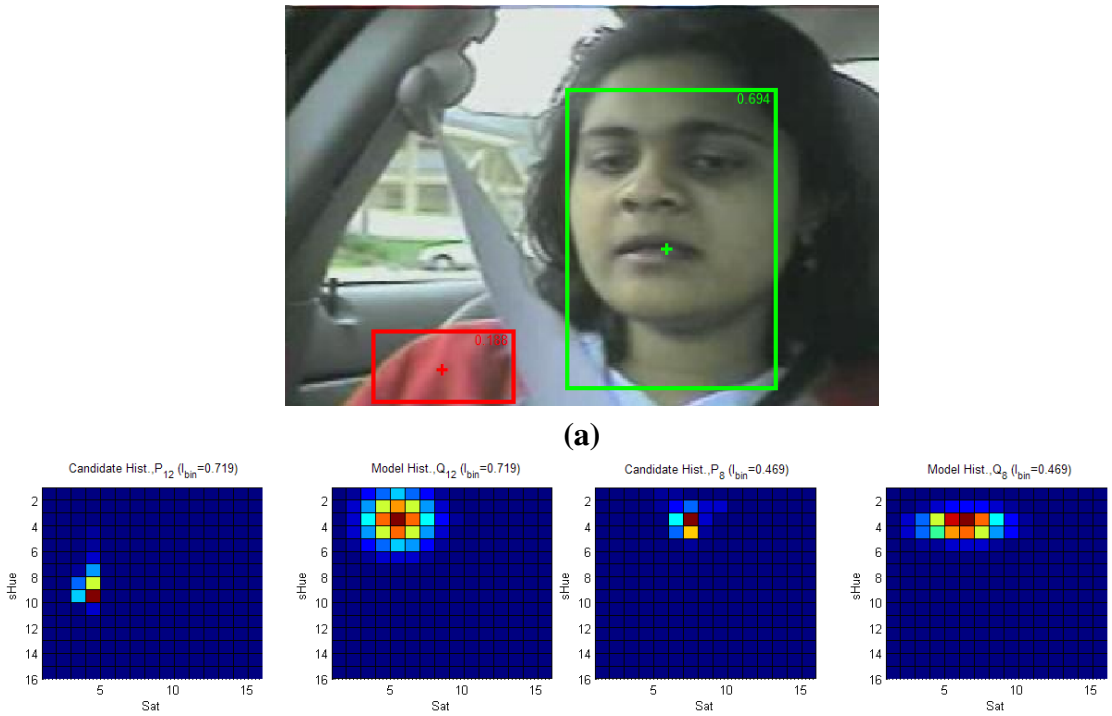
(b) ρ_{thresh}=0.3, ρ=0.380

Figure 2.19: Sample (a) False Negative and (b) False Positive Face Classifications

As expected, false positive error rates decreases as the threshold was increased as higher thresholds effectively increased the similarity measure relative to the face model required for face detection. Conversely, false negative failure rates increased as the threshold was increased as an increased number of candidates failed to adequately compare in similarity to the model distribution. Figure 2.19 illustrates sample false negative and false positive face classification results given the Bhattacharyya coefficient thresholds and actual ρ values listed above the image. The red bounding box indicates a *NonFace* classification while the green bounding box indicates a *Face* classification. From Table 2.4, it can be seen that a Bhattacharyya threshold of 0.5 yielded the lowest average classification error rates. Per this finding, the Bhattacharyya coefficient threshold of 0.5 will be the value utilized in the final detection algorithm. Thus, the final face detection classifier is explicitly defined below

$$C_{face}(\mathbf{P}_i) = \begin{cases} Face & \rho(\mathbf{P}_i, \mathbf{Q}_i) \geq 0.5 \\ NonFace & \text{otherwise} \end{cases} \quad \textbf{(2.35)}$$

Figure 2.20 illustrates the performance of the above classifier as applied to the adjusted algorithm used to generate the Bhattacharyya coefficient threshold data. Figure 2.20(a) contains the original RGB image which contains *multiple* face candidate ROI's. Figure 2.20(b) and (c) contain the ROI's face candidate and model histograms for the face and shirt ROI's, respectively. Figure 2.20(b) results in face classification with ρ=0.694 and (c) results in non-face classification with ρ=0.188. Note that the shirt region's candidate histogram in (c) has reduced saturation and increased hue values than that of the face model of the corresponding illumination level. As expected, when input to the classifier of Equation 2.35, the thresholding operation correctly classified the face region (bounded in green) as a face and the shirt region (bounded in red) as a non-face. Overall performance for this face classifier will now be explored in the following section.



(a)

(b) **(c)**

Figure 2.20: Face Classifier Performance Example
(a) Original RGB Image with Two Candidates* (b) Non-Face Classified ROI Candidate and Model Histograms (c) Face Classified ROI Histograms
note that only the largest skin-classified ROI is face detected in the final algorithm

2.6.2 *Face Detection Algorithm Performance*

This chapter has discussed a novel face detection algorithm for still images, beginning with skin classification and ending with the face classification result itself. The complete face detection algorithm flow diagram can be found in Figure 2.21, noting the different image resolution spaces involved at each step. Recall that the face detection algorithm returns the classification of the largest skin classified cluster only, but could be easily extended to multiple candidates per image if desired for future work.

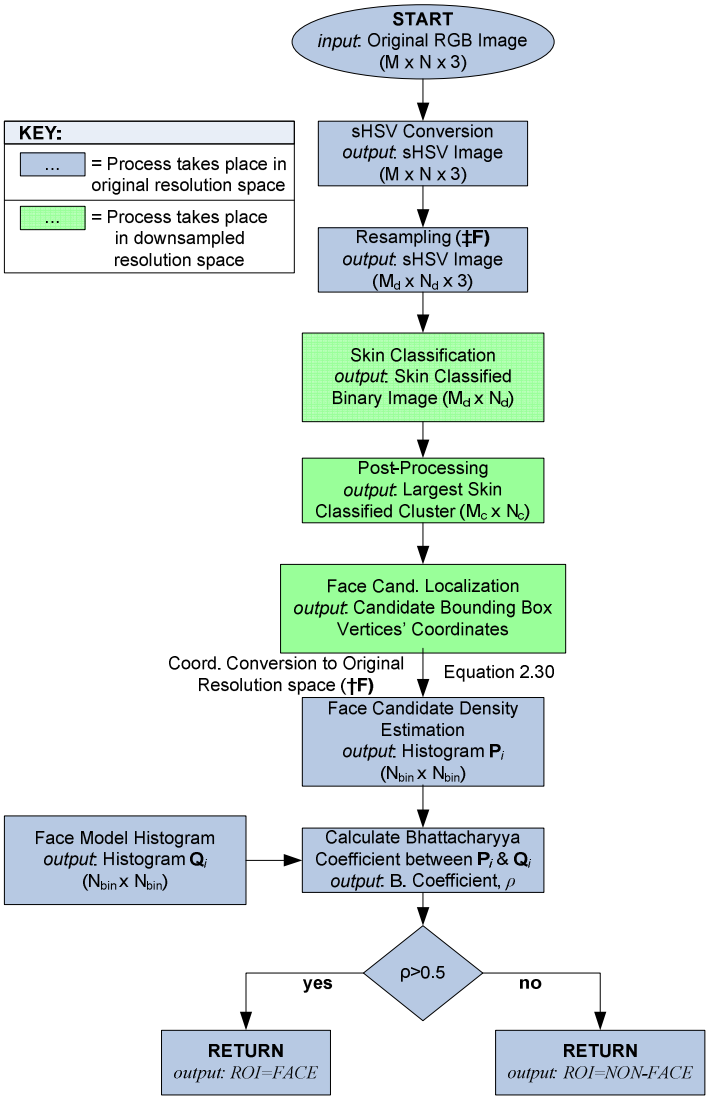


Figure 2.21: Complete Face Detection Algorithm Flow Diagram

To test the performance of the face detector algorithm, another 160-image test set was created from the AVICAR database, not containing any images found in the face-model or skin classification training sets. The test set was composed of 40 subjects at four different time instances throughout the video data. The performance of the face detector using this test set illustrates the success of the algorithm in response to variation in the subject's skin tone as well as any lighting or background changes over time. An important result of the algorithm is that the face candidate ROI output from the face localization algorithm contained an actual face in every single tested image from the test set. Hence the effective true negative and false positive rates cannot be accurately defined given the AVICAR test set without further analysis. Recall that this test set also generated the face localization results from Section 2.4, where 147 of the 160 images incurred successful face localization. Table 2.5 details the true positive and false negative detection rates for both the complete test set and the subset for which the face candidate was successfully localized. Keep in mind that each candidate ROI generated did contain an actual face.

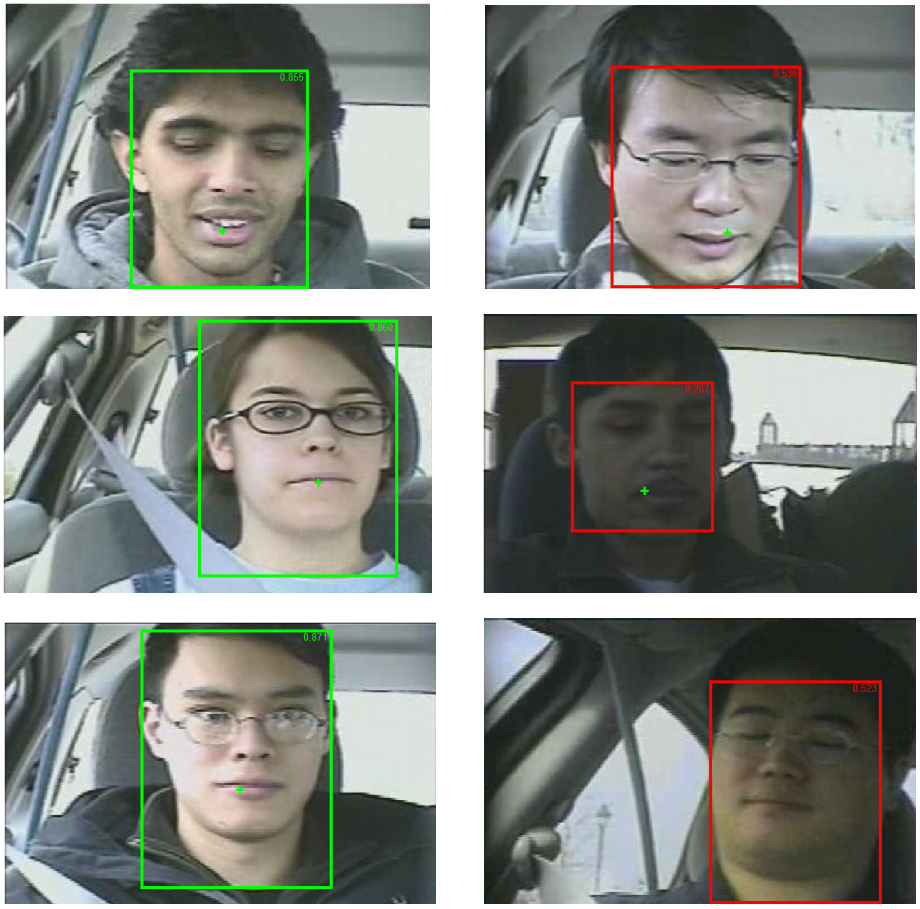
Table 2.5: Face Detection Algorithm Results

	Successful Localization Set*		**Complete Test Set**	
Face Detection Result	**Instances**	**Percentage**	**Instances**	**Percentage**
True Positive ($\rho \geq 0.5$)	139	94.6%	144	90.0%
False Negative ($\rho < 0.5$)	8	5.45%	16	10.0%
Total Images	*147*		*160*	

**Refer to Section 2.4 for definition*

As seen, the face detection algorithm achieved an overall accuracy of 90% across the test set images. The accuracy of the algorithm improves by 5% when the face itself is successfully bounded as a result of the face localization algorithm. Sample positive

(*Face*) and negative (*NonFace*) classifications are contained within Figure 2.22(a) and (b), respectively.



(a) **(b)**

Figure 2.22: Sample (a) Positive Face and (b) Negative Face Detections (RGB)

Significant sources of negative face detections involve changes in lighting conditions, specifically in dark environments. Ninety-percent of false negative classifications resulted from average candidate illumination values less than 0.5, or illumination level 8. Additionally, shifts in light chromaticity (color) away from "pure" white light significantly altered facial candidate's spectral content within the shifted hue and saturation feature space. Time of day and reflective surfaces around the car are two of many factors which have the ability to change the spectral content of ambient (visible)

light. Illustration of this effect can be shown via the contrast in ambient lighting between Figure 2.23(a) and (b). From the relatively white ambient lighting conditions in (a) to the less luminous and yellow-colored light in (b) a noticeable positive, 2-bin hue shift occurs in the candidate's peak histogram density consistent with this change in lighting conditions. Note that the green bounding box and text in Figure 2.23(a) indicates a positive face detection with a Bhattacharyya coefficient of 0.878, while the red bounding box and text in (b) indicates a negative face detection with a coefficient of 0.422.

(a) **(b)**

Figure 2.23: Effect of Ambient Light Chromaticity on Face Detection Original RGB Image, Face Candidate ROI, and Model-Candidate Histogram Pair for (a) Face Detection Success and (b) Face Detection Failure with Same Subject

To mitigate the effects of dark lighting conditions or colored ambient lighting the temporal element of a video sequence must also be incorporated into the face classifier and/or the face model itself. Through tracking algorithms and periodic candidate and model updates, the face classification accuracy of 90% could potentially be increased further. Refer to the Section 4.3 for additional recommendations for future improvement. Nonetheless, the performance of the skin classifier, filtering, face candidate localization,

and face classifier algorithms yielded commendable results in the unconstrained car environment captured within the AVICAR database.

CHAPTER 3
FEATURE EXTRACTION

The ultimate goal of this work is to localize human lips within a still image frame for subsequent tracking and audio-video speech recognition processing. The robust face detection scheme detailed in Chapter 2 is only the first step towards achieving this end. An equally robust feature extraction algorithm must be developed and implemented to localize the lips within the face-classified region of interest yielded by the face detection scheme. Existing methods employ the use of hidden Markov models (HMM's), active contours, and deformable templates to detect and locate regions within an image [10][22]. While the results are commendable, the iterative and extensive calculations demanded by these methods are significant. The lip localization algorithm must minimize both memory and processing time, considering the lip-reading system as a whole is to run in real time. Moreover, a majority of existing lip localization techniques assume ample resolution and controlled lighting conditions, which is not feasible for an unconstrained, in-car lip reading system.

Previous manifestations of this work utilized heuristics as a means to approximate the location of lips within a region of interest, resulting in a 75% lip localization success rate—65% overall when combined with the face detection employed [4]. Under this scheme, over- or under-sized face-classified ROI's would cause the algorithm to scan

erroneous areas of the region based off of these "rules of thumb," possibly failing to find the feature of interest. A more versatile feature localization scheme would mitigate this problem.

To eliminate this dependency on heuristics, a Gabor filter-based feature space is promoted as a means to localize lips within an image based off of shape. This filtered space will be shown to effectively differentiate facial features, including lips, from their backgrounds and to bound the full extent of the lips within a face-classified region of interest. Bounding the entirety of the lips is crucial as the size and shape of the lips over time will aid in automatic speech recognition that occurs downstream.

The following sections will explore the Gabor filter, its properties, and its application to the lip localization algorithm. Section 3.1 of this chapter will define the Gabor filter and its parameters while Section 3.2 will detail creation of a Gabor filter set which accurately represents lip geometry. Section 3.3 will outline the Gabor filtering algorithm as well as the Gabor filter-based feature space that will serve as the basis for lip localization. Lastly Section 3.4 will describe a novel lip central coordinate estimation algorithm and outline its results. Utilizing these coordinates, Section 3.5 will specify the lip localization procedure and summarize results of the entire lip localization algorithm.

3.1 The Gabor Filter and Its Properties

This section will provide an overview of the Gabor filter and its desirable properties toward facial feature extraction. The Gabor filter is a linear filter whose impulse response is defined as a sinusoidal function multiplied by a Gaussian function. The Gabor filter (and corresponding wavelet) is said to more effectively represent natural images than the impulse, δ, or difference of Gaussian (DOG) representations [21]. This

quality alone makes the Gabor filter ideal for locating natural face patterns within a car environment.

The Gabor filter can be defined over any number of dimensions but the two dimensional Gabor filter will be the focus of this work. While the exact definition of the Gabor filter varies, this work's treatment of the function is defined via several parameters. These parameters define the size, shape, frequency, and orientation of the Gabor filter among other characteristics. These parameters and their descriptions are listed below:

- N_x : Width of the Gabor filter mask (pixels)
- N_y : Height of the Gabor filter mask (pixels)
- ϕ : Phase of the sinusoid carrier (radians)
- F_o : Digital frequency of the sinusoid (cycles/pixel)
- θ : Sinusoid rotation angle (radians, counter-clockwise *w.r.t* +*x*-axis)
- γ : Along-Wave Gaussian envelope normalized scale factor
- η : Wave-Orthogonal Gaussian envelope normalized scale factor

Note the spatial frequency of the filter is listed in polar coordinates as opposed to Cartesian *x*- and *y*-axis frequency components. Given these parameters, one definition of the two-dimensional complex Gabor filter in the discrete, spatial domain is given by

$$G(x, y \mid \theta, F_o, N_x, N_y, \gamma, \eta, \phi) = \frac{\gamma \cdot \eta}{\pi} e^{-\left((\alpha x_r)^2 + (\beta y_r)^2\right)} e^{j 2\pi F_o (x_c \cos\theta + y_c \sin\theta + \phi)}$$

$$\forall x \in [1, N_x],\ y \in [1, N_y]$$

$$\text{with } \alpha = F_o / \gamma,\ \beta = F_o / \eta,\ x_o = N_x / 2,\ y_o = N_y / 2 \qquad \textbf{(3.1)}$$

$$\text{and } x_c = x - x_o,\ y_c = y - y_c, x_r = x_c \cos\theta + y_c \sin\theta, y_r = -x_c \sin\theta + y_c \cos\theta$$

where G is the N_y-by-N_x Gabor filter and [*y*,*x*] is the spatial location within the filter synonymous with [*r*,*c*] (row and column indexing, respectively). Sharing this definition

of the Gabor filter, the Gabor Filter Toolbox from Kamarainen *et al.* was used to generate all Gabor filters within the Matlab environment [14]. Refer to Appendix B for a copy of all code used for algorithm development. Figure 3.1 on page 63 contains an example Gabor filter with the stated parameters as visualized in three-dimensions and as a surface and in its two-dimensional environment. Note that this figure displays only the real component of the filter, which is complex in nature. Also note that the peak response of the filter is at the mask's center, $[x_o,y_o]$ and the counter-clockwise rotation of the two-dimensional sinusoid by $\pi/4$ radians (45°).

In addition to its sparse representation of natural images, the Gabor filter has several other attractive properties. Kamaraninen *et al.* notes that a Gabor filter is invariant to illumination, rotation, scale, and translation [13]. In an unconstrained environment these Gabor filter properties, in conjunction with its representativeness of natural images, make the filter an ideal candidate for detecting the facial features in less than desirable circumstances.

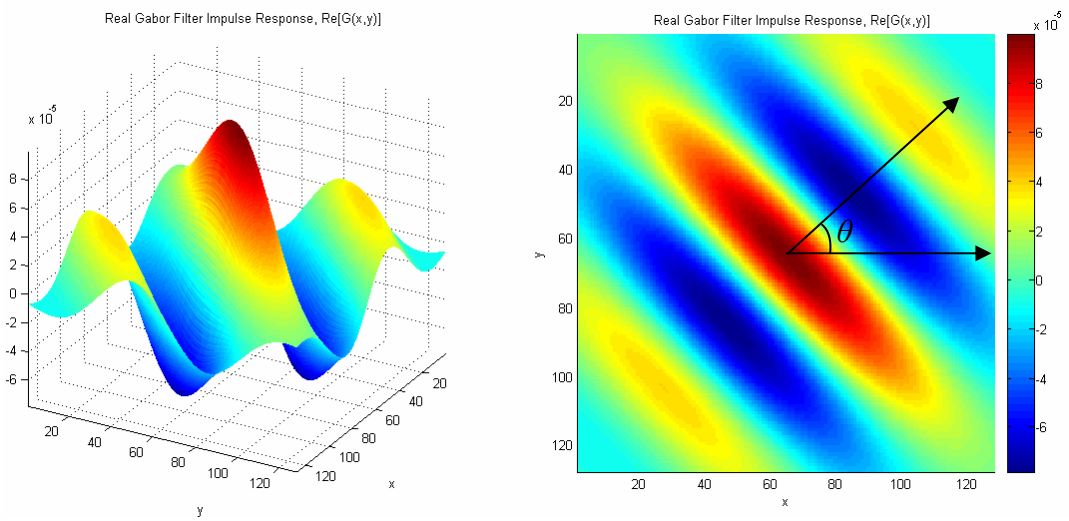


Figure 3.1: Sample Gabor Filter Impulse Response (Real Component) for N_x=N_y=128, θ=π/4, F_o=$\sqrt{2}/80$ φ=0, γ=η=1

3.2 Gabor Filter Set

In addition to its representativeness and invariance properties, the Gabor filter is localized in both the spatial and frequency domains, making it an attractive form for wavelet analysis. However, creation of biorthogonal Gabor wavelets is time consuming and computationally expensive. In practice, filter banks consisting of various Gabor filter configurations are constructed, yielding what is called a "Gabor-space." It has been posited that this Gabor-space is similar to the processes which takes place in human's visual cortex, allowing for rapid recognition of complex patterns in the visual environment.

Hence, the feature extraction process used in this work will also employ the use of multiple Gabor filters to represent facial features of interest. Several studies have successfully utilized Gabor filter sets of varying parameters to locate facial features. Kim *et al.* proposed a so-called "eye model bunch" composed of a total of 40 Gabor filters and classified each pixel's 40-element filter response as an eye via complex distance metrics [15]. While successful, this method was restricted to vertically oriented faces, required a vast training set, and required elevated memory demands and processing time. In fact, a majority of Gabor filter set studies restrict the application to controlled facial imagery, utilizing rotation and scale dependent comparison measures and designs [13].

Utilizing the AVICAR database training set developed in this study, measurements of upper and lower lip thicknesses and orientations were recorded. Upper and lower lip thicknesses, h_{hi} and h_{low}, respectively, were measured in pixels tangentially across the mouth opening. To reduce scale dependency, the lip heights were recorded as the ratio of the pixel heights to the height of the candidate's facial bounding box, M_c. Lip

orientation, $\Delta\theta_{lip}$, was recorded as the absolute rotation of the mouth opening axis from horizontal. Refer to Figure 3.2 for a diagram of how these measurements were calculated. For clarity, any discussion of the lips axial dimension will refer to this diagram with the lip's axis referring to the axis stretching from lip corner to lip corner. Across the 160-image training set, the resulting average measurements are found in Table 3.1. With this data, the Gabor filter set can now be created to more accurately represent the lip region.

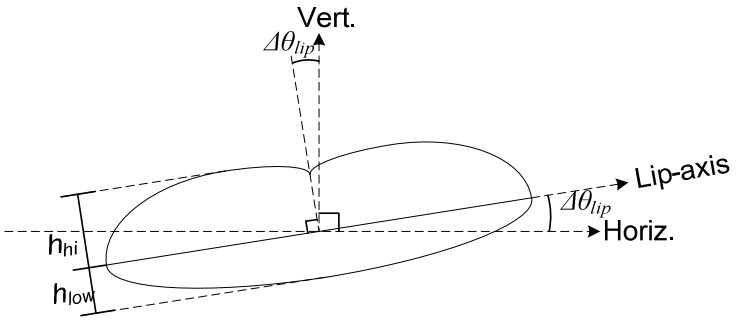


Figure 3.2: Lip Measurement Diagram

Table 3.1: Average Training Set Lip Measurements

Measurement	Average Value
Upper Lip Thickness Ratio, $\frac{h_{hi}}{M_c}$	0.136
Lower Lip Thickness Ratio, $\frac{h_{low}}{M_c}$	0.065
$\lvert\theta_{lip}\rvert$, Absolute Orientation (°)	11.25

In the development of the Gabor filter set, several key simplifications were made to reduce complexity and variability. First, the size of the Gabor filter was kept square such that the x (column) and y (row) dimensions were identical. Moreover, the normalized Gaussian envelop scale factors, γ and η, were kept unity-valued. Lastly, the sinusoid phase offset, φ, was fixed at zero. Using the data from Table 3.1, the remaining

key parameters of the Gabor filter set were selected. Referencing the defining Equation 3.1, the final 12-component Gabor filter set, **G**, is thus defined as,

$$\mathbf{G} = \left\{ G_{n,t,f} = G(x, y \mid \theta = \theta_t, F_o = F_f, N_x = N_n, N_y = N_n, \gamma, \eta, \phi) \right\}$$

$$N_n \in \left\{ \text{floor}\left(\frac{M_c}{8}\right), \text{floor}\left(\frac{M_c}{4}\right) \right\} \quad n = 1,2$$

$$\theta_t \in \left\{ \frac{3\pi}{8}, \frac{\pi}{2}, \frac{5\pi}{8} \right\} \quad t = 1,2,3 \qquad \textbf{(3.2)}$$

$$F_f \in \left\{ \frac{4}{N_n}, \frac{8}{N_n} \right\} \quad f = 1,2$$

$$\text{with } \gamma = \eta = 1 \text{ and } \phi = 0$$

where G is defined in Equation 3.1 and *n*, *t*, and *f* are the set indices of the (square) Gabor filter size, sinusoid angle, and digital frequency sets, respectively. In words, the Gabor filter set, **G**, is the set of Gabor filters for every combination of *n*, *t*, and *f*. The orientation values, $\theta_{t\in 1,2,3}$, were chosen such that the sinusoid orientation was vertically oriented (θ=90° or $\pi/2$ radians) and $\pm 2\Delta\theta_{lip}$ away from vertical, where the factor of two was experimentally determined. Whereas experimental trials showed near-horizontal orientations returned high values for more of the face-line and ear regions, the near-vertical orientations better follows the lip axis itself. The two frequency values, $F_{f\in 1,2}$, were chosen such that the half period of the sinusoid was approximately equal to the average upper and lower lip thicknesses per the face ROI height, M_c, and the ratios in Table 3.1 for the larger filter size, $N=N_2$. The second filter size, $N=N_1$, was experimentally selected such that the finer details of the lip, such as the lip corners, were more easily represented. In addition, the Gabor filter's size, N_n-by-N_n $|_{n\in 1,2}$, was selected such that over 80% of the total energy contained in the unbounded Gabor filter is

contained within the N_n-by-N_n mask for any value of F_f (which depends upon N_n) and θ_t. The relative size and frequency of the Gabor filter to the candidate's height allows for a more scale-invariant design.

Figure 3.3 displays a sample Gabor filter set for a face region of height M_c=235. Note the positive and negative values of the filter which have been mapped to grayscale values per the key on the right. With the establishment of the lip-specific Gabor filter set, processing of the face-classified region of interest can proceed. The following section will detail selection of the color component to be filtered as well as the design of the Gabor filtering process itself.

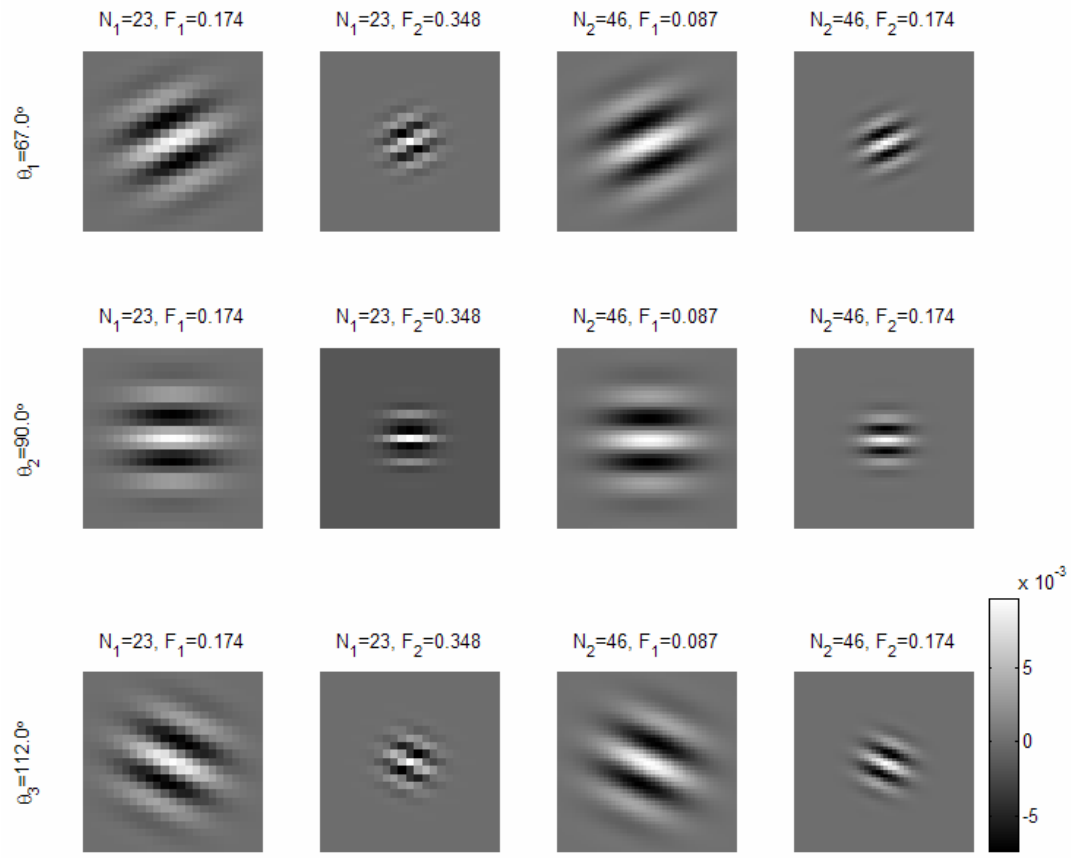


Figure 3.3: Sample 12-Component Gabor Filter Set (M_c=235)

3.3 Gabor Filtering Algorithm

With the lip-specific Gabor filter set in hand, the proper feature space must be chosen to which the Gabor filters will be applied. Previous work noted that mean values for lip and surrounding non-lip regions differed by 0.04 within the (shifted) hue space, 0.05 within the saturation space, and 0.1 within the illumination (value) space [4]. Following this data, it is apparent that skin and lip hues are similar in magnitude, barring any application of cosmetics. In fact, it has already been indicated that the hue and saturation values are a function, in part, of the illumination value itself. It will be shown that the varying value of the lip region and mouth opening provide sufficient contrast with the surrounding face, warranting the sHSV triplet's value component as the feature space of choice for Gabor filtering.

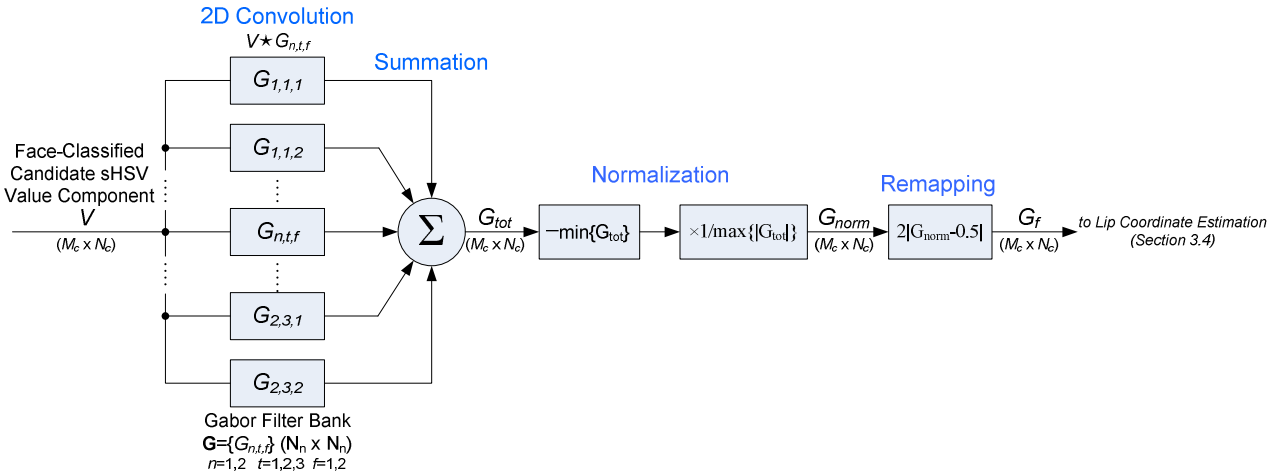


Figure 3.4: Gabor Filtering Process Block Diagram

As a simple, rotationally invariant lip localization space is desirable, the multidimensional Gabor filter set space was reduced to a single dimension. Figure 3.4 contains a block diagram of the entire Gabor filtering process including this space reduction procedure. First, 12 Gabor filter responses are generated by performing two-

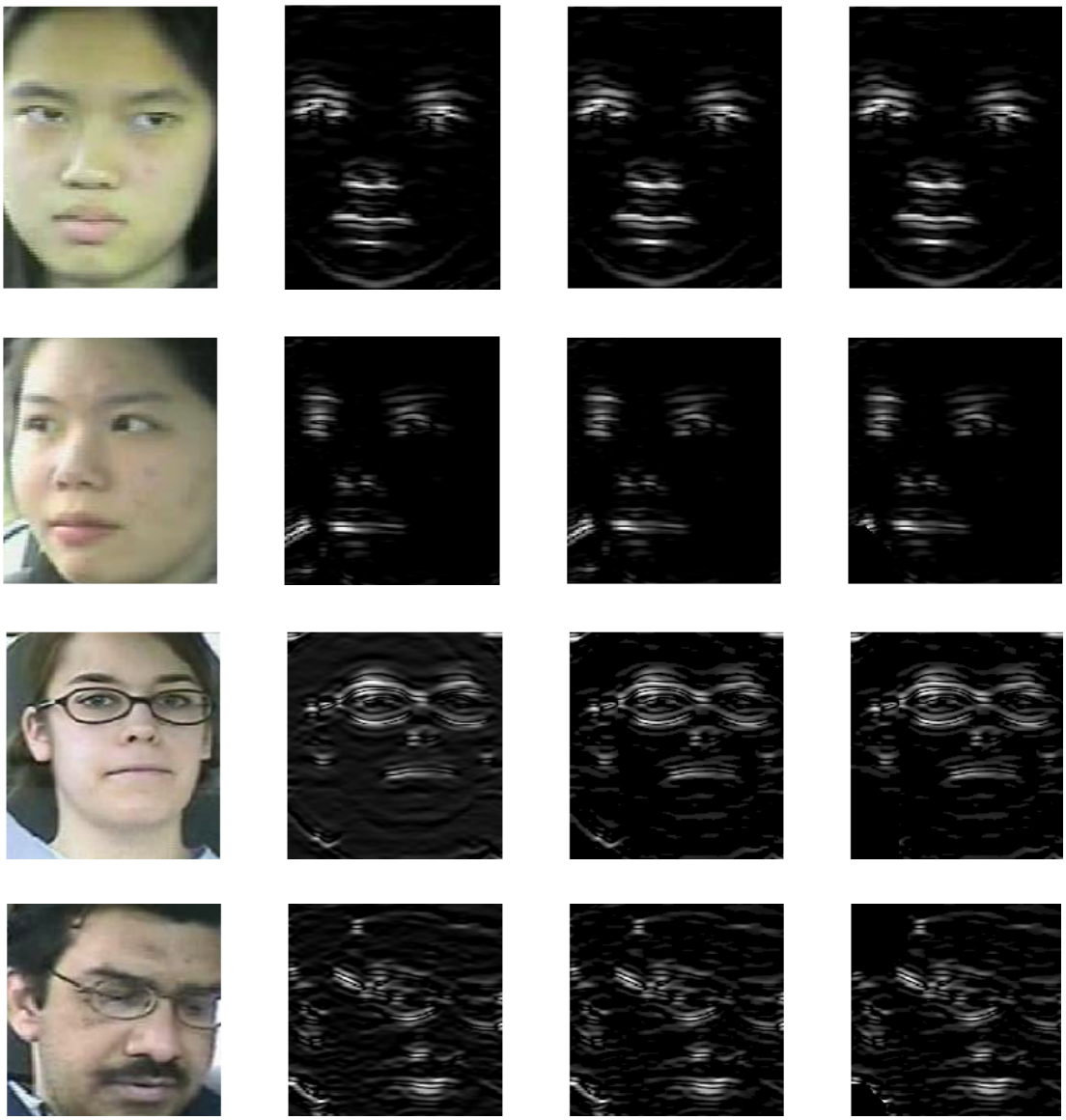
dimensional convolution, denoted by the star operator ($\star$), of the face-classified image's value component, V, independently with each Gabor filter configuration, $G_{n,t,f}$. Next, all 12 Gabor responses are summarized element by element such that the pixel value at any location within the candidate's ROI is the sum of each Gabor responses, also called Gabor jets, at the same location. For the purposes of this document, let the M_c-by-N_c total Gabor response be referred to as, G_{tot}, where M_c and N_c are the row and column sizes of the face candidate, respectively.

Due to the positive- and negative-valued modes of the Gabor filters (refer to Figure 3.1), the total Gabor response is then normalized to the range [0,1] and further remapped to stress the maximal and minimal Gabor jet values. The normalization and remapping procedure is defined below as

$$G_f(r,c) = 2\left|G_{norm}(r,c) - 0.5\right| \qquad \begin{matrix} r = 1,2,...,M_c \\ c = 1,2,...,N_c \end{matrix}$$

$$\text{where } G_{norm}(r,c) = \frac{G_f(r,c) - \min_{r,c}(G_f)}{\max_{r,c}(G_f)} \tag{3.3}$$

Let the final, normalized, and remapped Gabor filter response be defined as G_f, which has size M_c-by-N_c. In this case of zero phase shift, ϕ, normalization preceded remapping as the negative modes of the Gabor filter are attenuated more heavily by the Gaussian envelope than the central, positive mode. Supporting the need for the remapping process, an illumination-invariant design demands detection of absolute changes in achromatic intensity—both from high to low and low to high illumination. Referring to Figure 3.5(a) and (b), the cross section of the lip from chin to the region above the lip involves many such oscillatory changes in illumination value (even with the presence of facial hair).



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

Figure 3.5: Sample Total Gabor Filter Responses
(a) Original RGB Image (b) Total Gabor Response, G_f (c) Mean-Removed Total Response (d) Mean-Removed and Masked Total Response, G_{mr}

As alluded to, Figure 3.5 contains sample Gabor filter responses ranging from the total Gabor response, G_f, in (a) to the mean-removed and skin-classification masked responses in (d) which will be developed in Section 3.4. Note the contrast facial features have against the face's background. Smooth skin surfaces, such as the cheeks, provide minimal response while the mouth opening, lips, nostrils, eyes, and eyebrows provide

much elevated responses. This phenomenon can be attributed to the spatial transitions in illumination (both positive and negative) around these features. Interestingly, facial hair increases contrast between the hair and facial features, as is seen in the bottommost subject in Figure 3.5. Also note that the near-vertical edges of the face provide low responses while the near-horizontal edges, such as the chin region, provide more noticeable responses. With these positive feature qualities, the final Gabor filter response will now be used as the preferred feature space for lip localization, defined in Sections 3.4 and 3.5 to follow.

3.4 Lip Coordinate Estimation

The lip localization techniques employed in this work, utilize a two-step approach. First, the estimated coordinates of the lips are calculated and then the boundaries of the lips are generated about the estimated coordinates. The former step will be the focus of this section while the latter will be outlined in Section 3.5. To estimate the location of the lip's central coordinates, the following algorithm will generate seed points and then determine the most likely central coordinates by finding the seed point which maximizes an established figure of merit.

3.4.1 Seed Point Generation and Seed Parameter Calculation

Given the total Gabor filter response calculated in Section 3.3, a number of possible lip locations, called seeds, will be generated. Following seed generation, key parameters which are indicative of the presence of lips will be calculated. Utilizing these parameters, a figure of merit will then be calculated for each seed point. Figure 3.6 contains a flow diagram for the entire lip central coordinate estimation algorithm.

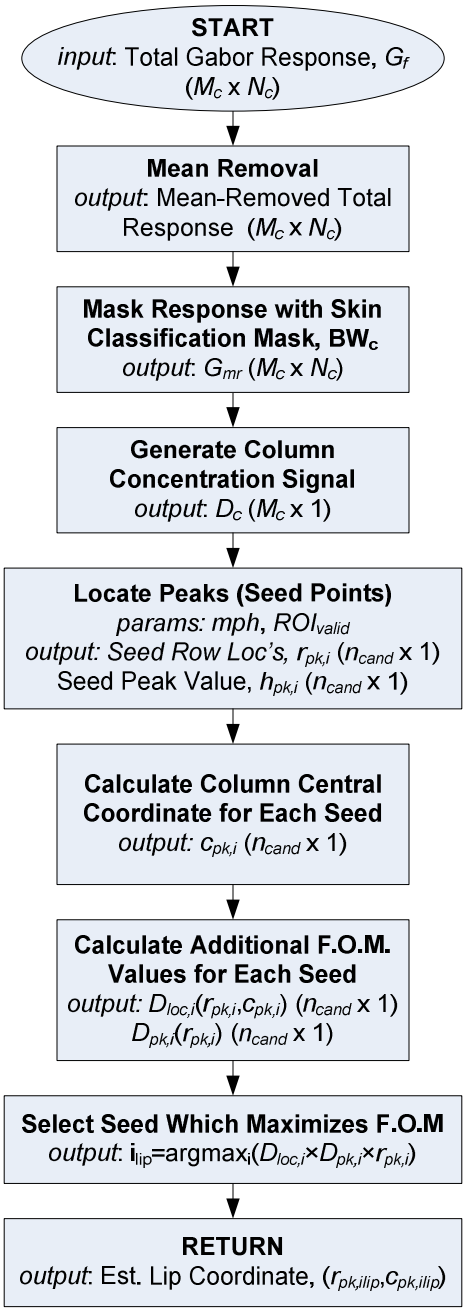


Figure 3.6: Lip Central Coordinate Estimation Algorithm Flow Diagram

Three parameters comprise the figure of merit which is used to select the estimated lip center coordinates. Before the figure of merit can be discussed, the three parameters of the figure of merit must be defined. Referencing Figure 3.6, before any processing is done, the Gabor filter response, G_f, undergoes mean-removal. Contrary to thresholding, mean removal is defined in this work as setting of all response pixel values

to zero if they are less than the total response's sample mean. Conversely, response values above the mean are left unchanged, still existing over the rage [0,1]. In fact, mean removal can be thought of as the element-wise multiplication of the original response with the binary threshold of that response, masking all pixel values with intensities below the sample mean. Furthermore, to remove false positives within the background surrounding the face, the skin-classified binary mask is applied over the mean removed response. Had this mask been applied before filtering this action could introduce hard edges near the skin cluster's border and alter response values, possibly over the lips. Recall that the face candidate's skin-classified binary mask, BW_c, was an element of the downsampled resolution space (refer to Section 2.2.2). Hence, the mean removal and masking operation is given by

$$G_{mr}(r,c) = \begin{cases} G_f(r,c) & \text{if } G_f(r,c) \geq \bar{G}_f \text{ and } BW_c(r',c') = 1 \quad r = 1,2,...,M_c \\ 0 & \text{otherwise} \qquad\qquad\qquad\qquad\quad c = 1,2,...,N_c \end{cases}$$

$$\bar{G}_f = \frac{1}{M_c \cdot N_c} \sum_{r=1}^{M_c} \sum_{c=1}^{N_c} G_f(r,c) \qquad \textbf{(3.4)}$$

$$\text{where } r' = \text{floor}\left(\frac{r}{F}\right) \text{ and } c' = \text{floor}\left(\frac{c}{F}\right)$$

where F is the downsampling factor, $\bar{G}_f$ is the sample mean of the final total Gabor response, G_f, and G_{mr} is the mean-removed and masked total Gabor response of the face-classified image. Note that the *r'* and *c'* values account for the change in resolution via what is essentially zero-one interpolation. Referring back to Figure 3.5(c) on page 70, mean removal effectively eliminates the contribution of background (non-feature) pixels to subsequent processing. The skin-classification masking in (d) within this figure also noticeably reduces the effect of several high-intensity non-face background regions.

To generate seed locations for the lip location an additional signal from the mean-removed response is created. Seed locations refers to possible locations pending subsequent parameter and figure of merit calculation. First, let D_c be the column concentration signal of the mean-removed Gabor response such that

$$D_c(r) = \sum_{c=1}^{N_c} G_{mr}(r,c) \quad \begin{matrix} r = 1,2,...,M_c \\ c = 1,2,...,N_c \end{matrix} \tag{3.5}$$

Even though the subject face within the ROI may not be perfectly vertical, the column concentration still conveys general information about how the filter response is distributed throughout the image's vertical (row) axis. Next, let the row mean be defined as

$$\mu_r = \sum_{r=1}^{M_c} r \cdot \frac{D_c(r)}{N_c} \tag{3.6}$$

Also, let ROI_{valid} denote the region of the candidate's ROI defined by $\mu_r \le r \le \text{floor}\left(p_{bor} \cdot M_c\right)$ and $\text{floor}\left((1-p_{bor}) \cdot N_c\right) \le c \le \text{floor}\left(p_{bor} \cdot N_c\right)$, where p_{bor} is experimentally set to 0.95 to eliminate false response returns at the candidate ROI's border.

Now, the seed point locations can be generated by finding the peaks of the column concentration signal over the region ROI_{valid}. Peak, or local maximum, detection is achieved via locating the concentration signal indices over the stated range, $\mu_r \le r \le \text{floor}\left(p \cdot M_c\right)$, such that the concentration immediately above and below that peak is less. Constant sequential concentration values are handled such that the peak occurs at the midpoint of the "plateau." Moreover, all peaks of height below a minimum peak height, denoted *mph*, are not considered. Let P_j and H_j be the row index and peak

height, respectively, of the j^{th} peak of the column concentration signal, D_c, within the ROI_{valid} space. Now, let the seed point locations for lip coordinate estimation be

$$\begin{aligned} \mathbf{r_{pk}} &= \{r_{pk,i}\} = \{P_j \mid H_j \geq mph\} \quad \forall j \qquad i = 1, 2, ..., n_{cand} \\ \mathbf{h_{pk}} &= \{h_{pk,i}\} = \{H_j \mid H_j \geq mph\} \quad \forall j \\ \text{with } mph &= \frac{1}{M_c} \sum_{r=1}^{M_c} D_c(r) \end{aligned} \tag{3.7}$$

where n_{cand} is the number of returned seeds, $r_{pk,i}$ is i^{th} row index for the peak of height $h_{pk,i}$ within the ROI_{valid} space which is at least as much as mph. Here, the minimum peak height was empirically selected as the sample mean of the column concentration signal. Completing the seed points' Euclidean coordinates, let $c_{pk,i,}$ be the i^{th} element of the set $\mathbf{c_{pk}}$, be defined as the midpoint coordinate of the longest consecutive non-zero chain in row $r_{pk,i}$ of the image's total Gabor response. Hence, the i^{th} seed point now has the location vector $[r_{pk,i},c_{pk,i}]$. For a depiction of this central column point refer to Figure 3.7(e) on page 77.

Combined, the $\mathbf{r_{pk}}$ and $\mathbf{c_{pk}}$ sets convey spatial location within the ROI. Due to the striation of the lip's structure from chin to nose, the concentration of peak Gabor responses along the row-axis is also pertinent to lip localization. Hence, let the peak concentration set be defined as

$$\mathbf{D_{pk}} = \{D_{pk,i}\} = \left\{ \sum_{r=r_{pk,i}-w}^{r_{pk,i}+w} r \in r_{pk,i} \right\} \forall i, \quad w = \text{floor}(M_c / 5) \tag{3.8}$$

where $\mathbf{D_{pk}}$ is the set of n_{cand} concentration values, $D_{pk,i}$, which are the sum of all peaks contained within the $2w$+1-row window centered about $r_{pk,i}$. The windowing value of w was experimentally determined to not exceed the ratio of average lip height to ROI height

and to provide optimal results within the lip region. Lastly, let the local two-dimensional mean-removed Gabor response concentration set be defined as

$$\mathbf{D_{loc}} = \{D_{loc,i}\} = \left\{ \sum_{r=r_{pk,i}-w_r}^{r_{pk,i}+w_r} \sum_{c=c_{pk,i}-w}^{c_{pk,i}+w} G_{mr}(r,c) \right\} \quad \forall i \qquad \textbf{(3.9)}$$

where $w_r = \text{floor}\left(w/2\right)$

where $\mathbf{D_{loc}}$ is the set of n_{cand} local two-dimensional response concentration values, $D_{loc,i}$, at seed index i and w is defined in Equation 3.7. The factor of two which scales the size of the window in the row dimension was based off of the observation by Crow that the width of lip regions is twice the height on average [4].

Figure 3.7 on the next page contains a graphical overview of the lip coordinate estimation algorithm. Figure 3.7 (c) contains the mean-removed and masked Gabor response, G_{mr}, overlaid with the seed locations indicated by the colored crosses. Part (d) contains the concentration signal, $D_c(r)$, while part (e) contains the total Gabor response row signals at the seed row indices, $G_{mr}(r_{pk,i},c)$, respectively, with the row and column seed components, respectively, overlaid on the signal. Developed in the next section, part (f) illustrates the final lip coordinate estimation position within the response G_{mr}.

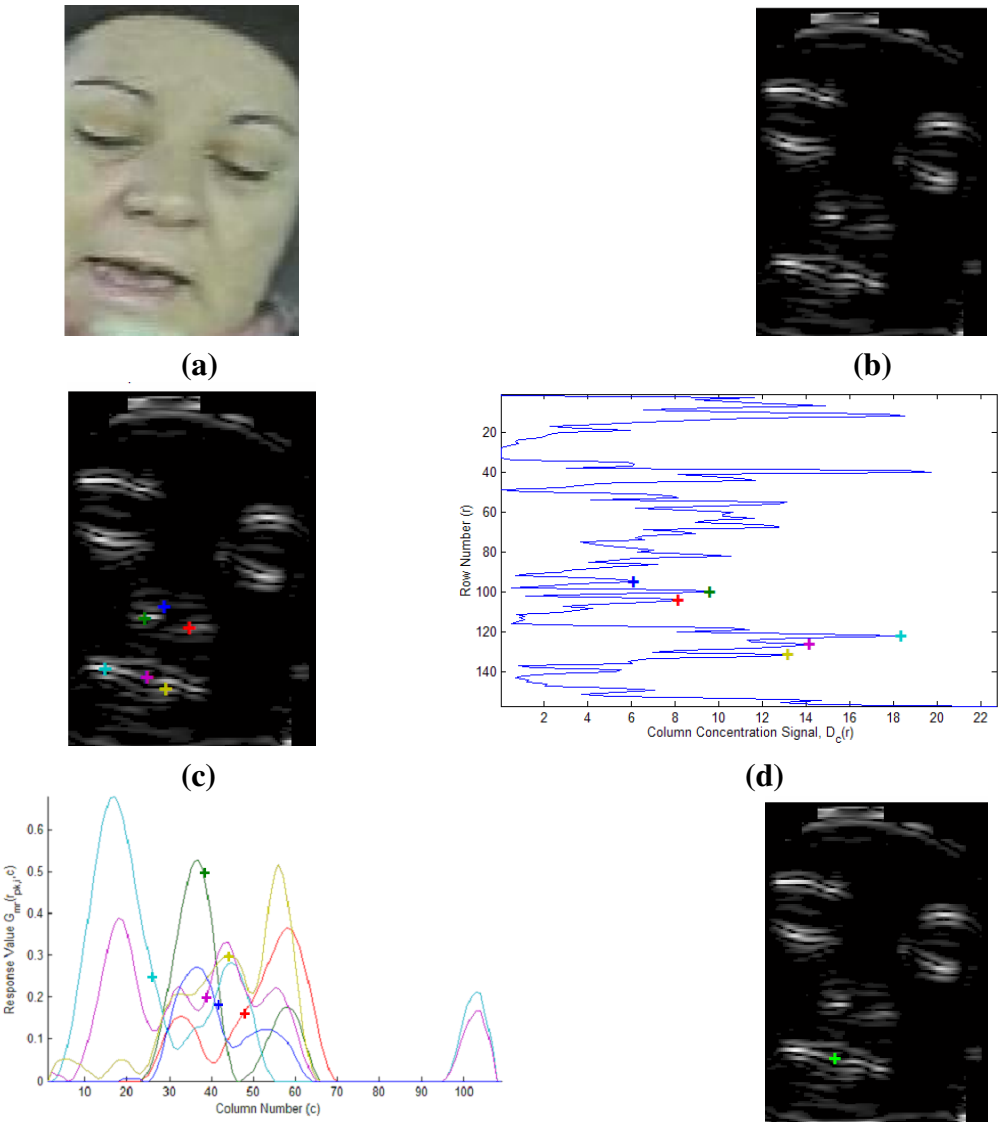


Figure 3.7: Sample Lip Coordinate Estimation Process
(a) Original RGB Face Candidate (b)Mean-Removed, Masked Gabor Response, G_{mr}
(c) Seed Locations within G_{mr} (d) Seed Row Locations Overlaid on D_c Plot
(e) Seed Row's Column Signals, G_{mr} (d) FOM-Maximizing Lip Coordinate

3.4.2 Figure of Merit, Lip Center Estimation, and Results

For the purposes of this work, the term figure of merit refers to a performance measure that represents desirable qualities in a measured sample. At this point many parameters exist for each of the seed locations generated from Equation 3.7. The figure of merit for lip coordinate estimation is a function of the parameters most representative

of a point within the lip region. It was decided through trial and error and iterative processes that the local concentration $D_{loc,i}$, row peak density $D_{pk,}$, and row location $r_{pk,i}$ of Equations 3.6 through 3.8, respectively, maximize both lip representativeness and contrast with non-lip regions. Utilizing these parameters, the final figure of merit equation is defined below as

$$\begin{aligned} &\mathbf{FOM} = \{FOM_i\} = \left\{D_{loc,i} \cdot D_{pk,i} \cdot r_{pk,i}\right\} \\ &D_{loc,i} \geq 1, \;\; D_{pk,i} \geq 1, \;\; r_{pk,i} \in [\mu_r, M_c] \end{aligned} \tag{3.10}$$

where **FOM** is the set of all figure of merit values, FOM_i, at seed index i. The figure of merit utilizes straight multiplication, eliminating the need to normalize each parameter's set to the same range. This figure of merit also equally weights each of the parameter values, a design decision based on empirical evidence. Moreover, note that both $D_{loc,i}$, $D_{pk,i}$, and $r_{pk,i}$ are always greater than one, never allowing a single parameter value to reduce the corresponding FOM or force it to zero. While this range may be easy to deduce for the row location and peak density parameters, the reason $D_{loc,i}$ is bounded below by one is that the column coordinate, $c_{pk,i}$, about which the window is centered requires that at least one nonzero pixel exist within the row (at that location in the limiting case). If a column were to contain strictly zero values, it would not have been returned as a seed point matching or exceeding the minimum peak height.

Conceptually, the figure of merit in Equation 3.9 combines the most visually apparent features of the lips into a single function. The two-dimensional response concentration vector conveys the magnitude of the response in a rectangular region around the lips. Higher local concentration infer coherent and localized response area consistent with a lip region. High-valued peak density parameters also communicate an

increased number of illumination oscillations encountered moving orthogonal to the lip's axis. Lastly, the seed's row location biases the figure of merit per the general spatial location of the lips within the face—high row indices correspond to lower position within the image. Of course, seed positions which lie below the actual lips will have a higher row location value, but the other parameters would ideally not be as elevated.

Hence it has been argued that the lip's central coordinates are the coordinates for which the established figure of merit is maximal. In other words, the estimated lip central coordinate are now given by

$$\mathbf{x}_{\mathbf{est}} = [r_{est}, c_{est}] = [r_{pk,i}, c_{pk,i}] \Big| i = \arg_i \max\left(FOM_i\right) \qquad \textbf{(3.11)}$$

where $\mathbf{x}_{\mathbf{est}}$ is the lip's estimated coordinates relative to the candidate image's coordinates FOM_i as defined in Equation 3.9, and r_{est} and c_{est} are the row and column estimated lip locations, respectively.

The lip center coordinate estimator of Equation 3.11 was applied to the test set used in the previous chapter to analyze face localization and detection. As shown in Table 3.2, the figure of merit and Gabor filter system utilized in the lip coordinate estimate yields comparable results to those of the face detector algorithm of Chapter 2. Of the 139 images for which the face candidate ROI was successfully localized and classified as a face, the algorithm placed the lip coordinates on the lips for 89.2% of the time. When applied to the test set in its entirety, the lip coordinate estimation algorithm placed the estimated coordinate on the lips 83.8% of the time. The major difference between these sets were that the lip coordinates falling more than 15 pixels away from the lips nearly doubled moving from the partial to the complete (overall) test set.

Table 3.2: Estimated Lip Coordinate Location Accuracy

Estimated Lip Coordinate Location Accuracy	Face-Classified Candidates with Successful Localization Set†		Complete Test Set	
	Instances	Percentage	Instances	Percentage
On Actual Lips	**124**	**89.2%**	**134**	**83.8%**
Within 15 pixels of Actual Lips*	6	4.3%	8	5.0%
Beyond 15 pixels of Actual Lips*	9	6.5%	18	11.2%
Total Images	*139*		*160*	

**Euclidean pixel distance from closest lip point*
†Refer to Section 2.4 for definition

(a) **(b)**

Figure 3.8: Sample False Background Response and Resulting Seed Locations (a) Original Image (b) Mean-Removed,Masked Gabor Response and Seed Locations

The most common sources of error for the lip coordinate estimation algorithm is false background responses that are not eliminated by the masking operation. While the face localization algorithm may accurately bound the face region, it does not alter the skin-classified binary image. These false responses occurred mainly towards the edges of an image, but were not masked out across all images for fear of not being able to locate lips on faces that are rotated more towards the profile position. Figure 3.8(b) contains one such example of background noise interfering with the seed point generation (indicated by the colored crosses).

3.5 Lip Localization and Test Results

The ultimate goal of this work is to establish a robust lip localization algorithm applied to still imagery. The face detection algorithm of Chapter 2 and the lip coordinate estimation algorithm discussed thus far in Chapter 3 have been crucial preliminary processes for the final stage of lip localization. The lip localization algorithm will utilize the same total Gabor response feature space as used in lip coordinate estimation (refer to Section 3.2 and 3.3 for a description).

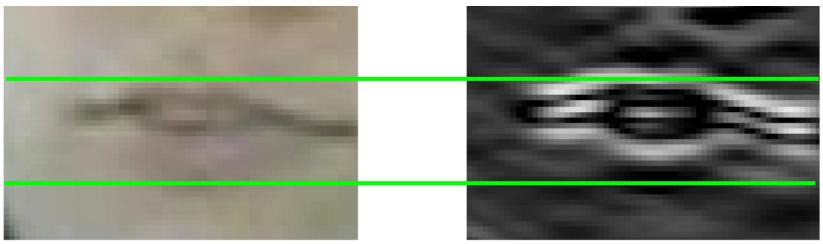


Figure 3.9: Sample Lip Region in (a) RGB and (b) Total Gabor Response, G_f, Spaces

Before discussion of the lip localization algorithm can begin, a few observations must be made about the behavior of lip regions (and sub-regions) within the total Gabor feature space. Figure 3.9 illustrates the appearance of the total Gabor filter response, G_f, over a lip region of the face. In addition to less-than-ideal resolution, note the low hue contrast between lip and non-lip regions in (a). Had the shifted hue space been used as the basis for Gabor filtering instead of the illumination component, facial features would not have been as pronounced. Also notable is the transition from high to low and low to high in the Gabor response when viewed from top to bottom within the upper and boundaries annotated in green. Hence, lips cannot merely be localized by bounding regions of maximum response. Instead consideration of the transition region is required.

Several signals must be defined before lip localization can occur. To simplify implementation, a sub-region of the Gabor total filter response, G_f, is first defined using

the lip coordinate estimate from Equation 3.11. Having dimensions M_{lip}-by-N_{lip}, let this lip region be defined as

$$\begin{aligned} &\mathbf{L} = \{(r,c)\} \ \ \forall r,c \quad \begin{matrix} r_o \le r \le r_e \\ c_o \le c \le c_e \end{matrix} \\ &\text{where} \quad r_o = r_{est} - \tfrac{1}{2} w_{lip}, \ \ c_o = c_{est} - w_{lip} \\ &\qquad\quad r_e = r_{est} + \tfrac{3}{4} w_{lip}, \ c_e = c_{est} + w_{lip} \\ &\qquad\quad w_{lip} = \frac{N_c}{4}, \ M_{lip} = \tfrac{5}{4} w_{lip} + 1, \ N_{lip} = 2w_{lip} + 1 \end{aligned} \tag{3.12}$$

where r_{est} and c_{est} are the estimated lip coordinates from Equation 3.11, r and c are the row and column locations, respectively, within the M_c-by-N_c face candidate region, w_{lip} is the lip window size, and **L** is the set of all points within the given two-dimensional window. The size of the window as well as the factor of one-half and three-quarters applied to the window lengths in the row dimension are a result of empirical testing and general anatomy. The window extends more greatly in the positive row direction as lip movement and lower lip size is greater in that direction. Moreover, the aspect ratio of the lip is greater along the lips axis than orthogonal to that axis in general.

Vertical lip localization within an image is inherently more complex than horizontal localization due to the striation (layers) of the Gabor response in the lip axis direction. Due to this, horizontal lip localization will be performed first to increase accuracy of the vertical localization. To localize the lips in the horizontal axis, let the row concentration over the lip region, **L**, be defined as

$$D_r(c) = \sum_{r=1}^{M_l} G_f(r,c) \quad \forall r,c \in \mathbf{L} \tag{3.13}$$

where $D_r(c)$ is the row concentration signal as a function of column location. Furthermore, let $\mathbf{B_h}$ be the set of all column locations within **L** for which the row

concentration signal is less than 10% of that signal's maximum value above the mean, given by

$$\begin{aligned}&\mathbf{B_h} = \{B_{h,i}\} = \left\{c \middle| D_r(c) < p_{\max} \max(D_r - \bar{D}_r) + \bar{D}_r \right\} \quad \forall c \in \mathbf{L},\ i = 1,2,... \\ &\text{where } \bar{D}_r = \frac{1}{N_{lip}} \sum_{c \in \mathbf{L}} D_r(c) \text{ and } \ p_{\max} = 0.1\end{aligned} \tag{3.14}$$

where $B_{h,i}$ is the i^{th} element of $\mathbf{B_h}$ and p_{max} was empirically derived. The resulting lower threshold from Equation 3.14 was established to locate substantial increases in Gabor response concentration above the lip region's mean. Hence, increased concentration above the lip region's mean is assumed to indicate the presence of lips per the lip-specific Gabor filter. Then, the left and right boundaries of the lips are derived as follows

$$\begin{aligned}c_l &= \max\left\{\mathbf{B_h} \mid B_{h,i} < c_{est}\right\} \quad \forall i \\ c_r &= \min\left\{\mathbf{B_h} \mid B_{h,i} > c_{est}\right\} \quad \forall i\end{aligned} \tag{3.15}$$

where c_{est} is the estimated lip localized column coordinate and c_l and c_r are the left and right lip localized coordinates relative to origin of the face candidate image. Refer to Figure 3.11(c) on page 86 for a graph visualization of this procedure.

After horizontal lip localization, vertical localization is undertaken utilizing the returned left and right boundaries. Let the column concentration signal and column discrete integral signal be defined as

$$\begin{aligned}&D_{c,L}(r) = \sum_{c=c_l}^{c_r} G_f(r,c) \quad \forall r \in \mathbf{L} \\ &S_r(r) = \sum_{p=r_o}^{r} D_{c,L}(p) - \bar{D}_{c,L} \\ &\text{where } \ \bar{D}_{c,L} = \frac{1}{M_{lip}} \sum_{r \in \mathbf{L}} D_{c,L}\end{aligned} \tag{3.16}$$

where $D_{c,L}(r)$ is the column concentration signal over the horizontally localized lip region at row index r and $S_r(r)$ is the integral signal at row index r. The integral signal is the summation of the mean-removed column concentration signal from the top of the lip region (r_o) to row index r. Mean subtraction was performed on the column concentration signal such that lower intensity regions (lines) of pixels would count negatively toward the integral signal and higher intensity regions would positively count toward the signal. Let the maximum integral signal value and row location be given by

$$\begin{aligned} S_{max} &= \max\left(S_r(r)\right) \qquad \forall r \in \mathbf{L} \\ r_{max} &= \arg_r \max\left(S_r(r)\right) \end{aligned} \tag{3.17}$$

where S_{max} is the integral signal's maximum value over the lip region and r_{max} is the row location of the maximum with respect to the face candidate's origin. Furthermore, let the minimum values of the integral signal *spatially* above and below the maximum value be given as

$$\begin{aligned} S_{min,u} &= \min\left(S_r(r)\right) \text{ and } r_{min,u} = \arg_r \min\left(S_r(r)\right) \quad r_o \le r < r_{max} \\ S_{min,b} &= \min\left(S_r(r)\right) \text{ and } r_{min,b} = \arg_r \min\left(S_r(r)\right) \quad r_{max} \le r \le r_e \end{aligned} \tag{3.18}$$

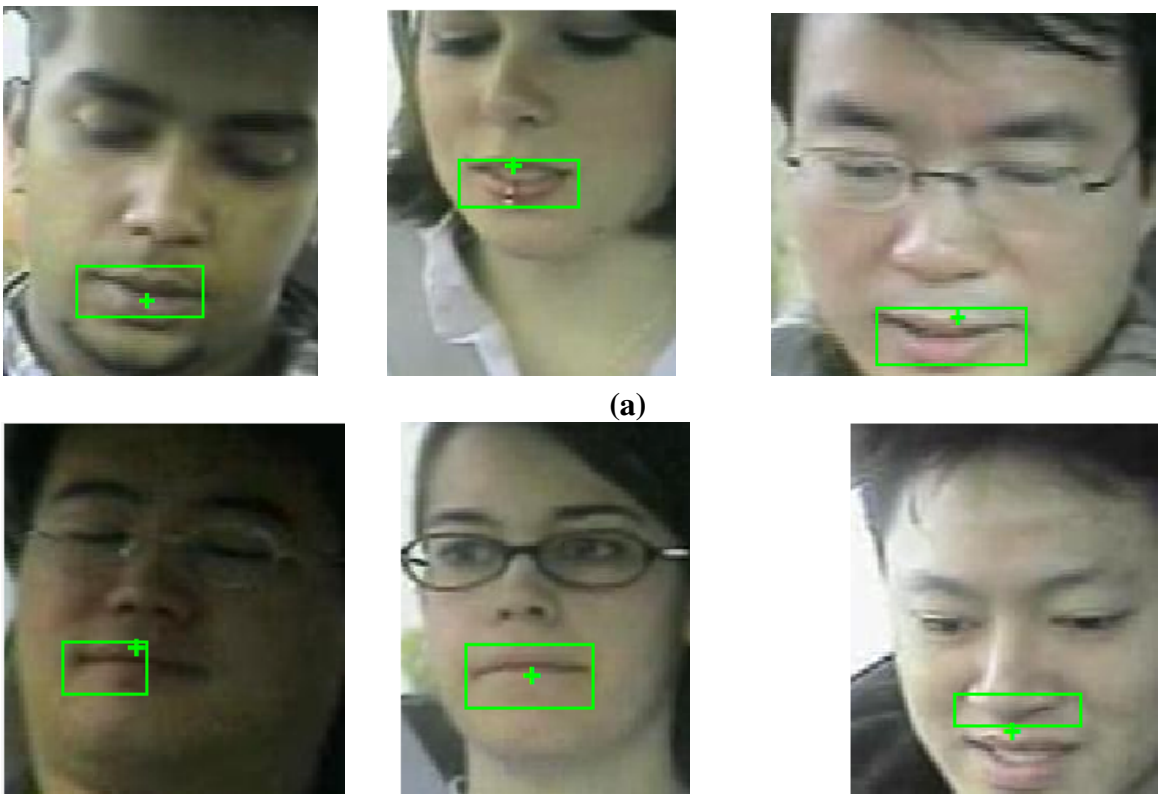
where $S_{min,u}$ and $S_{min,b}$ are the minimum values and $r_{min,u}$ and $r_{min,b}$ are the minimum values row locations of the above and below the maximum, respectively. Hence, let $\mathbf{B_u}$ and $\mathbf{B_b}$ be the set of all points which are less than 10% of S_{max} above and the upper and lower minimum values, respectively. These sets are given by,

$$\begin{aligned} \mathbf{B}_u &= \{B_{u,i}\} = \left\{ r \middle| S_r(r) < p_{\max}(S_{max} - S_{min,u}) + S_{min,u} \right\} \quad \forall r \in \mathbf{L},\ i = 1,2,... \\ \mathbf{B}_b &= \{B_{b,j}\} = \left\{ r \middle| S_r(r) < p_{\max}(S_{max} - S_{min,b}) + S_{min,b} \right\} \quad \forall r \in \mathbf{L},\ j = 1,2,... \\ &\text{where } p_{\max} = 0.1 \end{aligned} \tag{3.19}$$

where $B_{u,i}$ and $B_{b,j}$ are the i^{th} and j^{th} components of sets $\mathbf{B_u}$ and $\mathbf{B_b}$, respectively. Finally, the lip localized upper and lower boundaries are given by

$$r_u = \max\left\{\mathbf{B_u} \mid B_{u,i} < r_{max}\right\} \quad \forall i$$
$$r_b = \min\left\{\mathbf{B_b} \mid B_{b,j} > r_{max}\right\} \quad \forall j \qquad \textbf{(3.20)}$$

where r_u and r_b are the upper and lower lip localized boundaries relative to the origin of the face candidate image. Refer to Figure 3.11 for sample integral signal S_r, column concentration signal $D_{c,L}$, and the resting vertical boundaries overlaid on these signals.



(a)

(b)

Figure 3.10: Sample Lip Localization (a) Success and (b) Failures

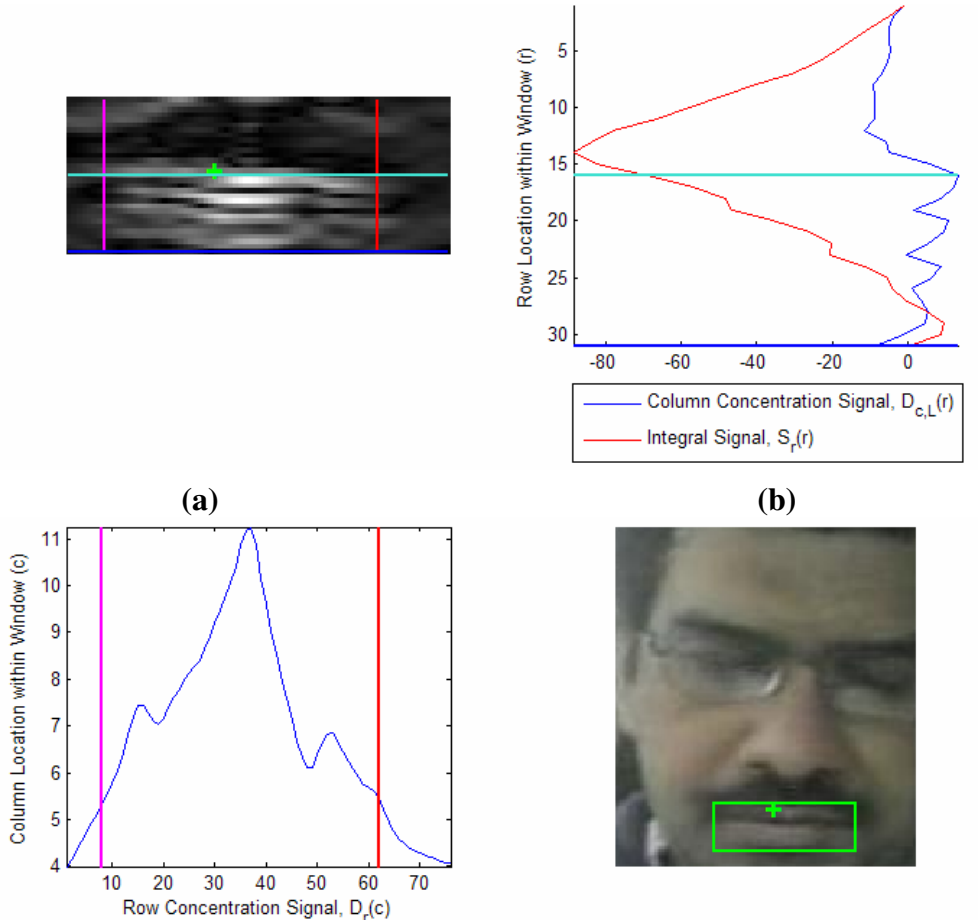


Figure 3.11: Sample Horizontal and Vertical Lip Localization Procedure and Result (a) Gabor Response within Lip Region (b) $D_{c,L}$ and S_r Signals over Lip Region Row (c) D_r Signal over Lip Region Column and (d) Lip Localization Result

The localization process is complete with the final lip localized region being bounded by $r_u \le r \le r_b$ vertically and $c_l \le c \le c_r$ horizontally with respect to the input face candidate image. Conceptually, the lip localization algorithm simply fixes horizontal boundaries such that significant increase in Gabor response roughly indicates a lip boundary horizontally. Vertically, the algorithm is constructed such that the upper and lower boundaries represent a transitional value between horizontal layers of strong and weak Gabor responses relative to minimal Gabor concentrations on the upper and lower regions, respectively.

To test the effectiveness of the lip localization algorithm, the 160-image test set was utilized. Sample lip localization success and failures from this set are contained in Figure 3.10(a) and (b), respectively. The resulting lip localization boundaries were analyzed for proximity to the actual lip boundary in each dimension and as the area of the actual lip contained within the boundaries. Table 3.3 summarizes the findings of this analysis. Note that 125% of actual lip area measure refers to the entire lip region being bounded with 125% of the lip's area contained within the lip localization boundaries.

Table 3.3: Lip Localization Test Set Accuracy

Boundary Dimension	**Localization Accuracy** *(Failure Measures are Italicized)*	**Face-Classified Candidates with Successful Localization Set**[†]		**Complete Test Set**	
		Instances	**Percentage**	**Instances**	**Percentage**
Horizontal[*]	Within 5 pixels of Lip Corner	106	76.3%	118	73.8%
	Beyond 5 pixels of Lips Corner	*22*	*23.7%*	*42*	*26.2%*
Vertical[*]	Within 5 pixels of Closest Lip Point	111	79.9%	124	77.5%
	Beyond 5 pixels of Closest Lip Point	*29*	*20.1%*	*36*	*22.5%*
Both	Area Contains Between 75% and 125% of Actual Lips	**115**	**82.7%**	**121**	**75.6%**
	Contains Less than 75% of Actual Lip Area	*11*	*7.9%*	*35*	*21.9%*
	Contains 125% of Actual Lip Area	*12*	*8.6%*	*4*	*2.5%*
	Total Images	*139*		*160*	

[*] *Manhattan pixel distance from closest lip point in specified axial direction only*
[†] *Refer to Section 2.4 for definition*

Table 3.3 indicates that successful face classification and localization improves lip localization by approximately 7% based off of lip area enclosed. Results also show that the vertical lip localization boundaries were closer to the actual lip than their horizontal counterparts. Also, due to increased lip coordinate estimation failure rates over the complete test set (refer to Table 3.2), failed lip localizations which contained less than 75% of the actual lips exceeded localizations which contained 125% of the lips (excess area) by nearly a factor of ten. Factoring in face detection, the overall accuracy

of 75.6% exceeds that of previous work [4]. While the overall accuracy is less than ideal, the challenges of the sub-optimal image quality and of the unconstrained car environment make this a respectable value. It should also be noted that the lip coordinate estimation need not fall directly on the lip in order to achieve an accurate lip localization due to the sizing of the window parameter, *w*. Nonetheless, as will be discussed in the following chapters, this lip localization algorithm suffers from a disadvantage in that it is based around still imagery and does not benefit from trends observed over time.

From left to right, Figure 3.10(b) contains failed lip localization images representing undersized localization, oversized localization, and false localization (to the nostrils), respectively. Sources of error for the lip localization algorithm include erroneous lip coordinate estimation, incomplete face localizations, as well as drastically different lighting conditions from one horizontal half of the face to another. The left image in Figure 3.11(b) illustrates the latter issue, the most common source of failure in the horizontal lip bounding. While the Gabor filter itself is invariant to regional changes in illumination, image over- and underexposure alter the illumination gradients within the lip region. Over- and underexposure results from overly intense and inadequate incident light upon the face such that the imaging device encodes color in a nonlinear fashion.

Ironically, one of the representative features from the figure of merit in Equation 3.10, column concentration signal peak density, was the most common source of error affecting the vertical localization of the lip. High Gabor responses within non-lip rows have the ability to cause inaccurate maximum response. This causes erroneous localizations of the upper and lower boundaries per Equations 3.16 through 3.20. Within the 21.9% of lip localizations that failed to contain at least 75% of the lip area, a majority

of these regions were centered about the nostrils which also yield notably high Gabor responses. The rightmost lip localization failure of Figure 3.10(b) illustrates this same phenomenon. The lesser extension of the lip windowed region, **L**, above the lip reduced these occurrences for a majority of accurately estimated lip central coordinates.

With the ultimate goal of the lip localization within a still image complete, the Chapter 4 to follow will summarize and expand upon the results of the entire algorithm from skin classification to lip localization. Limitations of the algorithm and recommendations for future algorithm improvement or augmentation will also be discussed.

CHAPTER 4
CONCLUSIONS AND FUTURE WORK

4.1 Overall System Performance

The performance of the lip localization algorithm within the visually noisy unconstrained environment resulted in an accuracy of 75.6% with respect to actual lip area enclosed by the process. The purpose of this section is to summarize the performance of the algorithm by component part and restate the algorithm's performance advantages, specifically over previous incarnations of this system. Sections 4.2 and 4.3 will proceed to discuss system limitations and recommendations for improvement and for future work, respectively.

Table 4.1: Algorithm Component Performance Summary

Component	Avg. Matlab Runtime* (1.760 secs Total)		Overall Accuracy	Measure
	seconds	% Total		
Face Localization	0.978†	55.5%	75.0%	resulting in 75% to 125% of actual face area bounded
Face Detection	0.311	17.7%	90.0%	resulting in true positive
			10.0%	resulting in false negative
Lip Coordinate Estimation.	0.352	20.0%	83.8%	coordinate falling within actual lips
			11.2%	coordinate falling more than 15 pixels away from lips
Lip Localization	0.119	6.8%	**75.6%**	actual lip area contained within localization boundaries

** per input image as performed on a Windows XP, 32-Bit, Pentium 4 Processor with 1.5GB RAM*
† involved RGB to sHSV conversion took 0.201 seconds on average

A results summary of the lip localization algorithm's sub-components is contained within Table 4.1. Keep in mind that each component's accuracy is based upon

the entirety of the extensive test set used throughout this report. As seen, face detection and lip coordinate estimation yielded especially accurate results relative to the 75.6% overall lip localization accuracy. Relative to previous thesis work, positive face detection rates rose form 75% to 90% while effective lip localization rates rose from 65% to 75% when considering face detection as a front end to lip localization [4].

Table 4.1 also highlights the absolute and relative processing time required within the Matlab environment. Of interest is the average total runtime of 1.535 seconds, which is decreased by more than an order of magnitude from previous work. In fact, most of the morphological and spatial filtering performed in the face detection algorithm occurred within the downsampled resolution space towards the end of reducing processing time. Moreover, the previous face detection system utilized three face models for each of five fixed ROI's, requiring the formation of 5 total candidate PDF approximations and 15 Bhattacharyya coefficient calculations. The face detection method in this work established a single, versatile face candidate ROI and a single, working face model, requiring one candidate PDF approximation and one Bhattacharyya coefficient calculation. Consequently, the number of 16-by-16 bin model histograms stored in memory has increased from three to 16, the number of illumination levels used. Nonetheless, required processing time for the face localization and detection algorithms were much reduced for any given image. Moreover, it should also be noted that 0.201 seconds of the face localization algorithm's 0.978 second average runtime was a product of the RGB to sHSV conversion alone. As a consequence of the improved face detection accuracy, the unique illumination-dependent face model and adjusted skin classifier should be considered successful and critical to the stated performance increase.

Lastly, this algorithm proposes a method for feature extraction and lip localization that does not rely upon heuristics. Previous implementations assumed ideal face localization and three-dimensional orientation by searching for and localizing lip regions that lie around the lips theoretical location based on "rule-of-thumb" face dimensions. This algorithm proposed a unique Gabor response feature space which relied upon a figure of merit rather than heuristic approximations, making it more versatile within the unconstrained environment. While still achieving less than desirable results for a pervasive AVASR front-end, the lip coordinate estimation and localization sub-components improved upon existing designs as noted. Consequentially, the established lip coordinate figure of merit and bounding protocols should be deemed a success.

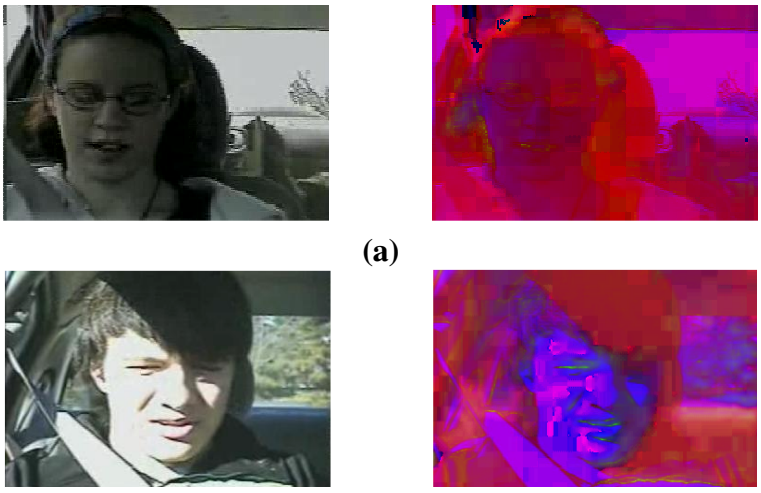
Hence, the proposed lip localization algorithm has yielded tangible improvements in face detection and lip localization accuracy and has drastically reduced processing requirements.

4.2 System Limitations

Despite the stated performance increases, common sources of error throughout the testing process highlight important system limitations of this lip localization algorithm. Among these issues are limited image resolution, skin-colored car environments, and overly bright and dark operating conditions without sufficient image dynamic range. While the lip localization algorithm is designed to mitigate these effects, these important implementation factors severely impact each component of the overall process.

As alluded to throughout the document, over- and under-exposed images significantly impact skin detection, face detection, and lip localization. Nonetheless, realistic, pervasive AVASR systems must be able operate across varying ambient

illuminations over time. Conversely, video hardware and image processing techniques can only provide adequate results sufficient lighting conditions. Naturally, no reliable image processing can occur at night within the visible part of the spectrum. Hence, a feasible low light threshold for valid video data must be established for any AVASR system taking place within the visible spectrum. Within this algorithm, all face candidate average illumination levels below 0.4 failed face classification, so this provides a realistic lower limit.



(a)

(b)

Figure 4.1: Original RGB and sHSV Images Displayed as RGB for (a) Underexposed (b) Overexposed Samples

The lip localization algorithm is heavily dependent on the dynamic range of the imaging device itself as this directly impacts the quality of images input into the algorithm. Without sufficient dynamic range, photodetectors within the camera generate nonlinear color information when incident light falls outside of upper and lower intensity limits. These nonlinear effects have the ability to propagate through the algorithm from sHSV conversion to lip localization itself. Hence, cost-performance analysis is vastly important in choosing video hardware that provides ample dynamic range for processing.

Figure 4.1(a) and (b) contain sample images that represent underexposure and overexposure, respectively. Note the drastic deviations in skin chromatic values from (a) to (b) as well as the inconsistency in these values throughout the facial region in (b). These inconsistencies across facial regions can cause incomplete and inaccurate skin and face classification as well degradation of the final lip localization (refer to Figure 3.10(b) on page 85).

Another significant source of error encountered through system testing was the presence of skin-colored car interiors behind the subject. While non-face clusters by themselves were shown to fail face detection, often non-face regions are included in the largest skin-classified cluster (refer to Figure 2.13 on page 35). While face localization is geared toward eliminating these problem regions, such as car roofs, the inclusion of background noise could cause face detection to fail and cause lip coordinate estimation or lip localization to provide erroneous results. Referring to the future development section to follow, utilizing the time element between frames could help eliminate these problem regions and further improve face detection and lip localization performance.

4.3 Future Development

After research, algorithm development, and careful study of system limitations detailed partially in the previous section, several opportunities for continued system improvement exist. Due to the vast scope of AVASR systems, the contributions of the algorithm specified in this document only lends itself to a small fraction of a complete unconstrained AVASR system. Valuable information and feature parameters generated by the other system-critical sub-components could potentially improve the lip localization algorithm drastically. This section will discuss several refinements to the lip localization

algorithm as well as possible directions that could benefit lip localization and the AVASR system as a whole.

The most notable improvement to the lip localization algorithm would be realized through the inclusion of time into the algorithm. Advanced difference imaging, the detection of movement between frames, would improve face localization and detection while reducing additional processing. Furthermore, face and lip spatial movement are generally orthogonal to each other, aiding the lip localization process even further. Moreover, the lip localization and lip tracking algorithms could mutually benefit one another. Crow implemented a mean shift face tracking algorithm that utilized the Bhattacharyya coefficient as adopted in this document [4]. Time inclusion would also allow for the tracking of face, lip, and environmental parameters that could be used as feedback to this lip localization algorithm, modifying the form of the face model, Gabor filter set, and, especially, estimated lip coordinates.

Advanced face localization methodologies could also be of great benefit to the overall system. The face localization algorithm discussed here was intentionally rudimentary, solving a particular problem, but more advanced face localization algorithms would benefit face detection and subsequent lip localization. Moreover, as the algorithm is implemented, only the largest skin-classified cluster undergoes further processing, but could be easily adapted to consider multiple candidates per frame. In relation to both face and lip localization, the need for elliptical detection arose several times throughout the design process. Following the natural shape of the face and lips, being able to detect the large elliptical shape within the skin-classified image or smaller elliptical shapes form of the total Gabor filter response would be of great benefit. Both

face and lip localization could also benefit from iterative boundary estimation until a selected criteria is satisfied.

To better approximate face models and lip parameters used in the design of this algorithm an audio-video testbed should be constructed such as that which generated the AVICAR database video footage. As the face model was generated on a different optical device, a fully functioning testbed would allow for proper regulation and calibration of the illumination-dependent face model under more controlled circumstances. Furthermore, improvements in image dynamic range and resolution would vastly improve detection and localization rates across the entire algorithm. Establishment of a testbed would benefit continuing work and allow more efficient research into varying audio-visual data fusion methodologies critical to the advancement of the larger AVASR system. Of particular interest is utilizing audio information to help locate corresponding movement within the video frame and more efficiently locate lip regions, again requiring a robust difference imaging algorithm.

REFERENCES

[1] M. Abdel-Mottaleb, A. Elgammal, "Face Detection in Complex Environments from Color Images," *Proceedings of the International Conference on Image Processing*, vol. 3, 1999, pp. 622–626.

[2] C. Chang, S. Chiang, "Anomaly Detection and Classification for Hyperspectral Imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, June 2002, pp. 1314-25.

[3] D. Comaniciu, V. Ramesh, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," *Proceedings on IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, June 13-15, 2000, pp. 142-149.

[4] B. Crow, "Automated Location and Tracking of Facial Features in an Unconstrained Environment," Master's Thesis, California Polytechnic State University, 2008.

[5] B. Crow, H.A. Montoya, X. Zhang, "Finding Lips in Unconstrained Imagery for Improved Automatic Speech Recognition", *Proceedings of the 9th International Conference on Visual Information Systems*, Shanghai, China, June 28-29, 2007.

[6] P. Delmas, M. Lievin, "From Face Features Analysis to Automatic Lip Reading. *Seventh International Conference on Control, Automation, Robotics and Vision*, vol. 3, Dec. 2-5, 2002, pp. 1421-25.

[7] K.G. Derpanis, "The Bhattacharyya Measure," <http://www.cse.yorku.ca/~kosta/CompVis_Notes/bhattacharyya.pdf >, March 20, 2008.

[8] R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification Second Edition*. New York, NY: John Wiley & Sons, Inc, 2001. Pages 23.

[9] A. Elgammal, C. Muang, D. Hu, "Skin Detection – a Short Tutorial," *Encyclopedia of Biometrics*, Springer-Verlag Berlin Heidelberg, 2009.

[10] N. Eveno, A. Capalier, P. Coulon, "Accurate and Quasi-Automatic Lip Tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 5, May 2004, pp. 706-715.

[11] R.C. Gonzalez, R.E. Woods, *Digital Image Processing Third Edition.*. Upper Saddle River, NJ: Pearson Prentice Hall, 2008, pp. 402-3,410, 633-5.

[12] M.J. Jones, J.M. Rehg, “Statistical color models with application to skin detection,” *International Journal of Computer Vision*, vol. 46, no. 1, pp 81–96, 2006.

[13] J. Kamarainen, V. Kyrki, “Invariance Peroperties of Gabor Filter-Based Features—Overview and Applications,” *IEEE Transactions on Image Processing*, vol. 15, no. 5, May 2006, pp. 1088-1099.

[14] J. Kamarainen, V. Kyrki, *The Gabor Features in Signal and Image Processing Toolbox*, March 3, 2003.

[15] S. Kim, S. Chung, S. Jung, D. Oh, J. Kim, S. Cho, “Multi-Scale Gabor Feature Based Localization,” *Proceedings of World Academy of Science, Engineering and Technology*, vol. 21, January 2007, pp. 483-487.

[16] K. Kumar, C. Tsuhan, R.M. Stern, “Profile View Lip Reading,” *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 4, April 15-20, 2007, pp. 429-432,

[17] B. Lee, M. Hasegawa-Johnson, C. Goudeseune, S. Kamdar, S. Borys, M. Liu, T. Huang, “AVICAR: Audio-Visual Speech Corpus in a Car Environment,” *INTERSPEECH2004-ICSLP*, 2004.

[18] 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.

[19] Y. Ming-Hsuan, A. Narendra, “Detecting Human Faces in Color Images,” *Proceedings of the International Conference on Image Processing*, vol. 1, Oct. 4-7, 1998, pp. 127-130.

[20] J.R. Movellan, “Tutorial on Gabor Filters,” <http://mplab.ucsd.edu/tutorials/gabor.pdf>, July 2, 2009.

[21] R. Navarro, A. Tabernero, G. Crístobal, “Image Representation with Gabor Waelets and Its Applications,” *Advances in Imaging and Electron Physics*, Orlando, FL, Academic Press Inc.

[22] G. Potamianos, H.P. Graf, E. Cosatto, “An Image Transform Approach for HMM Based Automatic Lipreading,” *Proceedings of the International Conference on Image Processing*, vol. III, pp. 173-177, 1998.

[23] G. Potamianos, J. Luettin, I. Matthews, “Audio-Visual Automatic Speech Recognition: An Overview,” *Issues in Visual and Audio-Visual Speech Processing*, MIT Press, Ch 10, 2004.

[24] K. Sandeep, A.N. Rajagopalan, “Human Face Detection in Cluttered Color Images sing Skin Color and Edge Information.”

[25] M.C. Shin, K.I. Chang, L.V. Tsap, "Does Colorspace Transformation Make Any Difference on Skin Detection?" *WACV: Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision,* Washington, DC*,* IEEE Computer Society, 2002, pp. 275.

[26] L.G. da Silveira, J. Facon, D.L. Borges, "Visual Speech Recognition: A Solution from Feature Extraction to Words Classification," sibgrapi, *XVI Brazilian Symposium on Computer Graphics and Image Processing*, 2003, pp. 399-405.

APPENDIX A: MATLAB Algorithm Code

A.1: Complete Algorithm Code

```
function lip_loc_alg(rgb)
%--------------------------------------------------------------------------
%PURPOSE:  Detects face witin image rgb and localizes lips
%PROCEDURE:1)Classifies Skin 2) Filter's Image 3)Localizes Face Candidate
%          4)Classifies Candidate 5) Gabor Feature Extraction
%          6)Lip Localization
%OUTPUT:   Multiple Figure windows containing algorithm information,
%          including localized face and lips
%AUTHOR:   Robert Hursig, California Polytechnic State University
%DATE:     7/1/2009
%--------------------------------------------------------------------------
    %Declare Constants
    factor = 8;
    shift_hue = 0.2;
    bins = 16;
    bor = 3;
    M=size(rgb,1);
    N=size(rgb,2);

    %Bhattacharyya Coefficient Data
    t_rho = 0.5;

    %Convert from RGB to sHSV Color Space
    hsv = rgb2hsv(rgb);
    hsv(:,:,1) = mod(hsv(:,:,1)+shift_hue,1);

    %Calculate Centroids, Adjust Metrics Per Scale Factor
    [labels lval cents areas maxs mins]=pre_proc(hsv, factor);
    [max_area I_max] = max(areas);  %column vector length L

    %Overlay Target and Face Boxs, If Exists
    sfigure(1);imshow(rgb);
    if ~isempty(areas)
        candidate = hsv(mins(I_max,1):maxs(I_max,1),mins(I_max,2):maxs(I_max,2),:);
        rho = get_b_coeff(candidate,bins);

        %EXTRAS
        %Overlay Candidate's Bounding Box, B.Coeff
        sfigure(1); hold on
        col = [rho<t_rho,rho>=t_rho,0];
        box_h = maxs(I_max,1)-mins(I_max,1)+1;
        box_w = maxs(I_max,2)-mins(I_max,2)+1;
        rectangle('Position',[mins(I_max,2),mins(I_max,1),box_w,box_h],...
                'LineWidth',bor,'EdgeColor',col);
        text(maxs(I_max,2)-2,mins(I_max,1),sprintf('%4.3f',rho),...
                    'Color',col,'HorizontalAlignment','right',...
                    'VerticalAlignment','top');hold off
        %END EXTRAS

        %Calculate Buffer Size for Gabor Filter Padding
        buf = ceil((maxs(I_max,1)-mins(I_max,1))/6); %M=4*Mp/3,Mp in feature_loc
        lb = [max(1,mins(I_max,1)-buf) max(1,mins(I_max,2)-buf)];
        ub = [min(M,maxs(I_max,1)+buf) min(N,maxs(I_max,2)+buf)];
        feat_cand = hsv(lb(1):ub(1),lb(2):ub(2),:);

        %Zero-One Interpolate Face Candidate Mask
        bw = imfill(labels==lval(I_max),'holes');
        [M_d N_d]=size(bw);
        ri=floor((1:1:M_d*factor)/factor);
        ci=floor((1:1:N_d*factor)/factor);
        ri=[ri.*double(ri~=0)+1*double(ri==0) M_d*ones(1,M-length(ri))];
        ci=[ci.*double(ci~=0)+1*double(ci==0) N_d*ones(1,N-length(ci))];
        bw_i = bw(ri,ci);
```

```
        bw_i_cand = bw_i(lb(1):ub(1),lb(2):ub(2));

        %Locate Features
        [lipcent lip_bdry]=feature_loc(feat_cand,bw_i_cand);

        %Convert Lip Centroid to Original Axes
        rstart=max(1,mins(I_max,1)-buf);
        cstart=max(1,mins(I_max,2)-buf);
        bufp = size(feat_cand,1)/8;
        lipcent = lipcent...
                + [mins(I_max,1)-1 mins(I_max,2)-1].*[rstart~=1 cstart~=1]...
                + [bufp-1 bufp-1].*[rstart==1 cstart==1]...
                -[buf-bufp buf-bufp].*[cstart==1 rstart==1];
        %Convert Lip Localization NW and SE Bounding Box Vertices
        nw_vert=[lip_bdry(1),lip_bdry(3)]...
                + [mins(I_max,1)-1 mins(I_max,2)-1].*[rstart~=1 cstart~=1]...
                + [bufp-1 bufp-1].*[rstart==1 cstart==1]...
                -[buf-bufp buf-bufp].*[cstart==1 rstart==1];
        se_vert=[lip_bdry(2),lip_bdry(4)]...
                + [mins(I_max,1)-1 mins(I_max,2)-1].*[rstart~=1 cstart~=1]...
                + [bufp-1 bufp-1].*[rstart==1 cstart==1]...
                -[buf-bufp buf-bufp].*[cstart==1 rstart==1];
        %EXTRAS
        sfigure(1);
        text(lipcent(2),lipcent(1),'+','Color','g','FontWeight','bold',...
            'FontSize',16,'HorizontalAlignment','center',...
            'VerticalAlignment','middle');
        %END EXTRAS
    end

    %EXTRAS
    sfigure(1);
    %title(sprintf('Subject %u,Pic %u, ratio=%4.3f',sub,ind,ratio))
    hold off
    %EXTRAS
```

A.2: Pre-Processing Code (Skin Classification, Filtering, Clustering)

```
function [labels lval cents areas maxs mins]=pre_proc(hsv_in, factor)
%PURPOSE: Returns "Candidate Face Clusters" and metrics for Pattern
%         Recognition Downstream
%         Pre-processing involves 1)decimation 2)hue blur/thresholding
%         3)Median Filtering 4)Erosion 5)Re-Dilation
%         6)Connected Component Labeling 7)Calculating/Outputting
%         Cluster Metrics and Thematic Map
%INPUTS:  hsv_in: image in sHSV space
%         factor: decimation factor for pre-processing
%                 should be an integer factor of size(hsv_in)
%OUTPUTS: lables: thematic map of size MxN, with L distinct, arb. labels
%         cents: Lx2 column vector of label centroid [x y] tuples
%         areas: column vector of corresponding label area,
%             NOTE areas derived from decimated image, relative areas only
%         maxs:  Lx2 column vector of label's max(SE corner) [x y] tuples
%         mins:  Lx2 column vector of label's min(NW corner) [x y] tuples
%
%AUTHOR:   Robert Hursig, California Polytechnic State University
%DATE:     7/1/2009
%-------------------------------------------------------------------------
    %Declare Constants
    %Skin Parameters
    t_lo = [0.175, 0.2]; %1=sH,2=I
    t_hi = [0.400, 1];   %1=sH,2=I
    t_hys = 0.11;        %hysteresis threshold value
    ptile = 1/3;         %percentile for faked "order statistic" filter
    %Read Input Image Dimensions
    M=size(hsv_in,1);
    N=size(hsv_in,2);
    M_d = floor(M/factor); N_d = floor(N/factor);
    %Face Localization Parameters
    rat_lo = 1.2;

    %Initialize Decimated Images
    sH_d_thresh = zeros(M_d,N_d);

    %Pad Input Image for Blurring
    hsv_pad = [hsv_in(:,1,:) hsv_in(:,:,:) hsv_in(:,N,:)];
```

```
    hsv_pad = [hsv_pad(1,:,:); hsv_pad; hsv_pad(M,:,:)];

    %Decimate, Blur, and Threshold Hue Data, Generate Difference Mask
    for r=1:M_d
        for c=1:N_d
            nhood = hsv_pad(r*factor:r*factor+2,c*factor:c*factor+2,:);
            h_avg = sum(sum(nhood(:,:,1)))/9;
            h_cen = nhood(2,2);
            i_avg = sum(sum(nhood(:,:,3)))/9;
            %Threshold Average Intensity Data
            ind = i_avg>t_lo(2) && i_avg<=t_hi(2);
            if ind
                %Threshold Hue Data
                ind = h_avg>t_lo(1) && h_avg<t_hi(1);
                %If not within range, check if satisfies hysteresis
                if ~ind
                    ind_nhood = sum(sum(nhood(:,:,1)>t_lo(1) & nhood(:,:,1)<t_hi(1)))>0;
                    ind_hys = h_cen>t_lo(1)-t_hys && h_cen<t_hi(1)+t_hys;
                    if ind_nhood && ind_hys
                        ind = 1;
                    else
                        ind = 0;
                    end
                end
            end
            sH_d_thresh(r,c)=ind;
        end
    end

    %Avg. Filter-Faked Percentile Filter for Pixel Noise
    sH_d_erode = (imfilter(sH_d_thresh,ones(3,3),'replicate','same'))>ptile;
    sfigure(2);
    subplot(2,2,3);imshow(sH_d_erode)
    %Erode Binary Image and Re-Dilate
    se = strel('square',floor(M/(factor*10)));
    sH_d_erode = imerode(sH_d_erode,se);
    sH_d_erode = imdilate(sH_d_erode,se);

    %Calculate Centroids, Adjust Metrics Per Scale Factor'
    [labels lval cents areas maxs mins]=bin_label2p(sH_d_erode);
    maxs = maxs*factor;
    mins = mins*factor;
    cents = cents*factor;

    if ~isempty(areas)
        [val I_max] = max(areas);
        cent = floor(cents(I_max,:)/factor);
        BW_max = labels == lval(I_max);

        ratio = (maxs(I_max,1)-mins(I_max,1))/(maxs(I_max,2)-mins(I_max,2));
        if ratio<rat_lo
            [nmin nmax]=trim_bb(BW_max,cent,mins(I_max,:)/factor,maxs(I_max,:)/factor);
            mins(I_max,:)=nmin*factor;
            maxs(I_max,:)=nmax*factor;
        end
    end

    %EXTRAS
    sfigure(2);
    hsv=hsv_in;
    hsv(:,:,1) = mod(hsv_in(:,:,1)-0.2,1);
    subplot(2,2,1);imshow(hsv2rgb(hsv))
    subplot(2,2,2);imshow(sH_d_thresh)
    subplot(2,2,4);imshow(sH_d_erode)
    %END EXTRAS
end
```

A.3: Face Candidate Localization Code

```
function [new_mins new_maxs]=trim_bb(BW,cent,prev_min,prev_max)
%-------------------------------------------------------------------------
%PURPOSE: Crops input candidate binary image, BW, to largest, coherent
%         elliptical shape within the image.  Implements a 3-pass
%         spatial filtering (binary ellipse) to determine cropping points.
%INPUTS:  BW: binary image candidate to be cropped
%         cent: 1x2, centroid in [r,c] of the thresholded cluster in BW
%         prev_min: 1x2, minimal row/col values in [r,c] for cluster
%         prev_max: 1x2, maximal row/col values in [r,c] for cluster
%OUTPUTS: new_mins: 1x2, cropped minimal row/col values in [r,c] for cluster
%         new_maxs: 1x2, cropped maximal row/col values in [r,c] for cluster
%AUTHOR:   Robert Hursig, California Polytechnic State University
%DATE:     7/1/2009
%-------------------------------------------------------------------------

    %Declare Constants, Read Input Image Size, Initialize Vars
    [M N]=size(BW);
    mask_frac = 0.5;    %Percentage of Height Mask Extends Row-wise
    pmax=1/3;   %Lower threshold for top/bottom passes, normalized
    pmid=0.5;   %Lower threshold for center pass, normalized

    %Create Masks
    height=prev_max(1)-prev_min(1)+1;width=prev_max(2)-prev_min(2)+1;
    lheight=floor(mask_frac*height);lwidth=floor(min(lheight/1.2,width));
    lcen=floor([lheight/2 lwidth/2]);
    lmask=zeros(lheight,lwidth);
    dev=floor(width/20);

    %Populate Small Mask
    for r=1:lheight
        for c=1:lwidth
            r_norm = ([r c]-lcen)./lcen;
            lmask(r,c)=(r_norm*r_norm'<1);
        end
    end
    lmask=logical(lmask);
    %Calculate Indices
    left=max(prev_min(2)-lcen(2)+1,1);right=min(N,prev_max(2)+lwidth-lcen(2));
    BW_cand=BW(prev_min(1):prev_max(1),left:right);
    mid_top=floor(height/2-lheight/2);mid_bot=mid_top+lheight-1;

    %Filter
    lpass1=zeros(1,right-left+1);   %Top Pass
    lpass2=zeros(1,right-left+1);   %Bottom Pass
    lpass3=zeros(1,right-left+1);   %Center Pass
    for c=lcen(2):right-left+1-(lwidth-lcen(2))
        lpass1(c)=sum(sum(BW_cand(1:lheight,c-lcen(2)+1:c+lwidth-lcen(2)).*lmask));
    end
    for c=lcen(2):right-left+1-(lwidth-lcen(2))
        lpass2(c)=sum(sum(BW_cand((height-lheight+1):height,c-lcen(2)+1:c+lwidth-lcen(2)).*lmask));
    end
    for c=lcen(2):right-left+1-(lwidth-lcen(2))
        lpass3(c)=sum(sum(BW_cand(mid_top:mid_bot,c-lcen(2)+1:c+lwidth-lcen(2)).*lmask));
    end

    %Crop Columns
    pass_ratio = (lpass2+1)./(lpass1+1);
    pass_thresh = pass_ratio>2 | pass_ratio<0.5;
    [max_mid cmm]=max(lpass3);[row colmid]=find(lpass3<max_mid*pmid);
    if max(lpass1)>max(lpass2)
        [mval cpeak]=max(lpass1);
        max_pass=lpass1;
    else
        [mval cpeak]=max(lpass2);
        max_pass=lpass2;
    end

    [row col]=find(pass_thresh==1);
    [row2 col2]=find(max_pass<mval*pmax);
    tval = left-1+max(col2(col2<cpeak));
    if ~isempty(col(col<cpeak))
        cmin=left-1+max(col(col<cpeak));
        cmin=max(cmin,tval);
    else
        cmin=tval;
    end
```

```
    cmin=max(cmin,left-1+max(colmid(colmid<cpeak)));
    [row2 col2]=find(max_pass<mval*pmax);
    tval=left-1+min(col2(col2>cpeak));
    if ~isempty(col(col>cpeak))
        cmax=left-1+min(col(col>cpeak));
        cmax=min(cmax,tval);
    else
        cmax=tval;
    end
    cmax=min(cmax,left-1+min(colmid(colmid>cpeak)));

    %Crop Rows
    r_mins=zeros(1,3);r_maxs=zeros(1,3);
    ccen=floor((cmax+cmin)/2)-left+1;rcen=cent(1)-prev_min(1)+1;
    for i=1:3
        [rows val]=find(BW_cand(:,ccen+(i-2)*dev)==0);
        vmin=max(rows(rows<rcen));
        if ~isempty(vmin)
            r_mins(i)= vmin+1;
        else
            r_mins(i)=1;
        end
        [rows val]=find(BW_cand(:,ccen+(i-2)*dev)==0);
        vmax=min(rows(rows>rcen))-1;
        if ~isempty(vmax)
            r_maxs(i)= vmax-1;
        else
            r_maxs(i)=height;
        end
    end

    rmin=prev_min(1)-1+min(r_mins);
    rmax=prev_min(1)-1+max(r_maxs);

    %Assign Output
    new_mins=[rmin cmin];
    new_maxs=[rmax cmax];

    %EXTRAS
    sfigure(4);clf
%     subplot(2,2,1);imshow(BW)
    subplot(2,1,1);imshow(BW_cand)
    subplot(2,1,2);
    hold on
    plot(lpass1/max(max_pass),'b'),plot(lpass2/max(max_pass),'g'),axis tight
    plot(lpass3/max(lpass3),'r'),plot(pass_ratio,'m'),axis tight
    hold off
    %END EXTRAS
end
```

A.4: Bhattacharyya Coefficient Calculation Code

```
function rho = get_b_coeff(hsv_in, N_bin)
%-------------------------------------------------------------------------
%PURPOSE: Returns Bhattacharyya Coefficient, rho, for the given area of
%         interest.  The input image will be weighted per the Epanechnikov
%         kernel and the resulting joint hue and saturation histogram,
%         p(h,s), will be normalized and quantized to an N_bin-by-N_bin
%         distrubution. The model's PDF, q(h,s), is a function of the ROI's
%         average intensity.
%INPUTS:  hsv_in: image in sHSV space
%         N_bin:  number of bins used along the hue and saturation
%                 dimensions to generate the joint PDF's
%OUTPUTS: rho: the Bhattacharyya Coefficient, defined over [0 1] with 1
%              being a perfect match to the model's joint PDF, q(h,s)
%AUTHOR:   Robert Hursig, California Polytechnic State University
%DATE:     7/1/2009
%-------------------------------------------------------------------------
    %Initialize Variables
    temp = load('model.mat','sHS_i');
    model_i = temp.sHS_i;
    N_i = 16;

    %Read input image dimensions
    M=size(hsv_in,1);
    N=size(hsv_in,2);
    roi_cen = [M/2 N/2];

    %Generate Candidate Joint PDF
    HSpdf = zeros(N_bin,N_bin);
    norm = 0;
    I_avg = 0;
    for r=1:M
        for c=1:N
            r_norm = ([r c]-roi_cen)./roi_cen;
            r_sq = r_norm*r_norm';
            %Only use elliptical ROI for histogram data
            if r_sq < 1
                i_h = floor(N_bin*hsv_in(r,c,1))+1;
                i_s = floor(N_bin*hsv_in(r,c,2))+1;
                %Safeguards maximally valued (1.0) sH or S in double arith.
                if(i_h > N_bin)
                    i_h = N_bin;
                end
                if(i_s > N_bin)
                    i_s = N_bin;
                end
                HSpdf(i_h,i_s) = HSpdf(i_h,i_s)+ 1 - r_sq;
                norm = norm + 1 - r_sq;
                I_avg = I_avg + hsv_in(r,c,3)*(1 - r_sq);
            end
        end
    end
    %Normalize Model Histogram, Calc Avg. ROI Intensity
    HSpdf = HSpdf/norm;
    I_avg = I_avg/norm;

    %Determine Which Avg. Intensity Bin Candidate Belongs to (ceil
    %eliminates need to add one for Matlab Indexing)
    i_cand = ceil(I_avg*N_i);  if i_cand==0,i_cand=1;end


    %Calculate Bhattacharyya Coefficient, rho
    model = squeeze(model_i(i_cand,:,:));
    rho = sum(sum((HSpdf.*model).^0.5));

    %EXTRAS
    sfigure(3);
    subplot(2,2,1),surf(HSpdf),colormap jet, view(90,90),axis tight
    if i_cand<10,title(sprintf('Candidate Hist., P_%i (I_b_i_n=%4.3f)',i_cand,i_cand/N_i-1/(2*N_i)));
    else title(sprintf('Candidate Hist., P_1_%i (I_b_i_n=%4.3f)',i_cand-10,i_cand/N_i-1/(2*N_i)));end
    xlabel('sHue'),ylabel('Sat')
    subplot(2,2,2),surf(model),colormap jet, view(90,90),axis tight
    if i_cand<10,title(sprintf('Model Hist., Q_%i (I_b_i_n=%4.3f)',i_cand,i_cand/N_i-1/(2*N_i)));
    else title(sprintf('Model Hist., Q_1_%i (I_b_i_n=%4.3f)',i_cand-10,i_cand/N_i-1/(2*N_i)));end
    xlabel('sHue'),ylabel('Sat')
    subplot(2,2,3),surf(model),colormap jet, view(90,90),axis tight
    if i_cand<10,title(sprintf('Model Hist., Q_%i (I_b_i_n=%4.3f)',i_cand,i_cand/N_i-1/(2*N_i)));
```

```
    else title(sprintf('Model Hist., Q_1_%i (I_b_i_n=%4.3f)',i_cand-10,i_cand/N_i-1/(2*N_i)));end
    xlabel('sHue'),ylabel('Sat')
    subplot(2,2,4),surf(HSpdf),colormap jet, view(90,90),axis tight
    if i_cand<10,title(sprintf('Candidate Hist., P_%i (I_b_i_n=%4.3f)',i_cand,i_cand/N_i-1/(2*N_i)));
    else title(sprintf('Candidate Hist., P_1_%i (I_b_i_n=%4.3f)',i_cand-10,i_cand/N_i-1/(2*N_i)));end
    xlabel('sHue'),ylabel('Sat')
    %END EXTRAS
end
```

A.5: Feature Extraction Code (Gabor Filtering, Central Lip Coordinate Estimation)

```
function [lipcent,lip_bdry]=feature_loc(hsv_in,BW_in)
%--------------------------------------------------------------------------
%PURPOSE: Gabor filters intensity component of input sHSV image, hsv_in,
%         applying BW_in as a mask to the sHSV image.
%INPUTS:  hsv_in: face candidate image in sHSV space
%         BW_in:  skin classified face candidate binary image
%OUTPUTS: lipcent:  1x2 lip (central) coordinate estimate
%         lip_bdry: 1x4 vector containing left, right, top, and bottom
%                   indices which approximate lip bounds
%AUTHOR:   Robert Hursig, California Polytechnic State University
%DATE:     7/1/2009
%--------------------------------------------------------------------------
    %Gabor Filter Bunch Parameters
    M=size(hsv_in,1);
    N=size(hsv_in,2);
    sf = 4;
    theta=[3*pi/8, pi/2, 5*pi/8];
    N_gf=floor(sf*M*[1,2]/32);
    fund=[1 2];
    gamma = 1;
    eta = 1;
    buf = ceil(max(N_gf)/2);
    resp_tot = zeros(M,N);
    rs_dev = 1; %rows above and below central row included in row slice avg

    %Fraction of mean for which lip candidate locations are valid
    f_rmean = 1.2;
    %Lip Candidate Bounding Box Scale Factor
    bb_sf = [0.5 1]; %1=vert sf,2=horiz sf, fraction of corr. len value (in each dir!)
    mpd = 2;        %Minimum Peak Distance used in Peak Detection

    %Perform Filtering Over Each Theta,Frequency Combination
    for t=1:length(theta)
        for f=1:length(fund)
            for n=1:length(N_gf)
                freq = sf*fund(f)/N_gf(n);
                g = gfcreatefilter2(freq,theta(t),gamma,eta,N_gf(n));
                resp = imfilter(hsv_in(:,:,3),g,'corr', 'same');
                resp_tot = resp_tot+resp;
            end
        end
    end

    %Normalize (Valid) Total Response,
    resp_tot=real(resp_tot(buf:M-buf+1,buf:N-buf+1));
    BW=BW_in(buf:M-buf+1,buf:N-buf+1);
    [Mp Np] = size(resp_tot);
    BW=imclose(BW,strel('disk',floor(Mp/6)));
    range = [min(min(resp_tot)),max(max(resp_tot))];
    resp_tot=(resp_tot-range(1))/(range(2)-range(1));
    resp_tot=abs(resp_tot-0.5)*2;
    %Calculate and Threshold with Global Mean
    resp_avg = sum(sum(resp_tot))/(Mp*Np);
    resp_mr = resp_tot.*(resp_tot>resp_avg);
    %Mask with Inputted BW Mask
    resp_mr = resp_mr.*BW;

    %Find Lip Candidates via Column-wise Sum and Peak Detection
    col_sum = sum(resp_mr,2);
    row_mean = mean(col_sum)/Np;
    mph = floor(row_mean*Np);  %Minimum Peak Height Used in Peak Detection
    noise = (1:1:Mp)'./(10*Mp);
    col_sum = col_sum+noise;    %eliminates flat-peaks for better detection
    [pks locs]=findpeaks(col_sum,'minpeakdistance',mpd,'minpeakheight',mph);
```

```
%Calc Expectation of Row Number, Start Row of Valid Peaks (*f_rmean)
rows_vec = min(locs):1:max(locs);
sum_tot = sum(col_sum(min(locs):max(locs)));
r_mean = floor(rows_vec*col_sum(min(locs):max(locs))/sum_tot);
N_cand = length(locs);      %Number of Candidates

%Find Max Continuous String of Above Mean Pixels,Calc Corr. Midpoint
rowslices = zeros(length(pks),Np);
cedges=zeros(length(pks),2); %1=min,2=max
lmaxs = zeros(length(pks),1);
for i=1:N_cand
    rowslices(i,:)=mean(resp_mr(locs(i)-rs_dev:locs(i)+rs_dev,:),1);
    rowslices(i,:)=smooth(rowslices(i,:));  %smooth slice for thresh.
    rowslices(i,1)=0;rowslices(i,Np)=0; %safeguards againds bdry conds.
    t_rs = max(mean(rowslices(i,:)),row_mean);
    %Threshold & Determine Zeros Crossings to Find Longest Chain in Row
    [rzc czc]=find(xor(rowslices(i,2:Np)>t_rs,...
                                    rowslices(i,1:Np-1)>t_rs)~=0);
    for k=1:floor(length(czc)/2)
    len = czc(k*2)-czc(k*2-1);
        if len>lmaxs(i)
            lmaxs(i) = len;
            cedges(i,:)=[czc(k*2-1) czc(k*2)];
        end
    end
end
ccents = sum(cedges,2)/2;

%Determine Approximate Area of Candiate Region
%Maximum Row/Column-Wise Deviation from Cand's Valid Median Cent.
c_maxdev = Np/4;
r_maxdev=Mp/20;
%Determines Candidate Flags which Meet Position Criteria
pk_valid = (locs>r_mean*f_rmean)&(locs<Mp-r_maxdev)&...
           (ccents'>c_maxdev/2)&(ccents'<Np-c_maxdev/2);
%indicates candidates less than predetermined threshold from col median
ind_dev = abs(ccents'-median(ccents(pk_valid)))<c_maxdev;

%Calculate Total Response (not BW) of Candidate's Surrounding Region
%Region Dictated by Corr. lmax Value Scaled by bb_sf row/colwise dirs
cand_resp_sum = zeros(1,N_cand);
for i=1:N_cand
    if pk_valid(i)
        dev = floor(bb_sf*lmaxs(i));
        lb = [max(1,locs(i)-dev(1)), max(1,floor(ccents(i))-dev(2))];
        ub = [min(Mp,locs(i)+dev(1)), min(Np,floor(ccents(i))+dev(2))];
        cand_resp_sum(i) = sum(sum(resp_mr(lb(1):ub(1),lb(2):ub(2))));
    end
end

%Calculate Peak Density at Each Candidate Point
w_len = Mp/5;
pk_den = ones(1,N_cand)*Mp;
for i=1:N_cand
    if pk_valid(i)
        [rwin cwin] = find(abs(locs(pk_valid)-locs(i))<w_len/2);
        pk_den(i) = length(cwin);
    end
end

%Determine Most Likely Lip Centroid
fom = cand_resp_sum.*locs.*pks.*ind_dev.*pk_den;
[mfom Imax] = max(fom);
ccent = ccents(Imax);
rcent = locs(Imax);
lipcent = [rcent ccent];
[lip_bdry]=lip_loc(resp_totlipcent);

%--------------------------------------------------------------------
%END CODE, BEGIN FIGURES
%--------------------------------------------------------------------

%Plot Data
sfigure(7);clf
cols=zeros(N_cand,3);
col_hsv=hsv(sum(pk_valid));ci=1;
for i=1:N_cand,if pk_valid(i),cols(i,:)=col_hsv(ci,:);ci=ci+1;end,end
subplot(2,2,1);imshow(resp_mr);
title('Gabor Total Response and Filtered Seed Locations');hold on
```

```
    for i=1:N_cand
        if pk_valid(i)
            text(ccents(i),locs(i),'+','Color',cols(i,:),'FontWeight','bold',...
            'FontSize',16,'HorizontalAlignment','center',...
            'VerticalAlignment','middle');
        end
    end
    hold off
    subplot(2,2,2);plot(col_sum),view(90,90), axis tight
    for i=1:N_cand
        if pk_valid(i)
            text(locs(i),col_sum(locs(i)),'+','Color',cols(i,:),'FontWeight','bold',...
            'FontSize',16,'HorizontalAlignment','center',...
            'VerticalAlignment','middle');
        end
    end
    xlabel('Row Number'),ylabel('Averaged Row Sum')
    subplot(2,2,3);xlabel('Column Number'),ylabel('Row Averaged Value')
    hold on,
    for i=1:N_cand,
        if pk_valid(i), plot(rowslices(i,:),'Color',cols(i,:)),end,
    end
    hold off,axis tight
    subplot(2,2,4);imshow(resp_mr)
    title('Gabor Total Response and Estimated Lip Coordinates')
    text(ccent,rcent,'+','Color','g','FontWeight','bold',...
            'FontSize',16,'HorizontalAlignment','center',...
            'VerticalAlignment','middle');

    sfigure(8);
    shsv = hsv_in(buf:M-buf+1,buf:N-buf+1,:);
    shsv(:,:,1) = mod(shsv(:,:,1)-0.2,1);
    subplot(1,4,1);imshow(hsv2rgb(shsv))
end
```

A.6: Lip Localization Code

```
function [verts]=lip_loc(resp,cent_est)
%--------------------------------------------------------------------------
%PURPOSE: Localizes lip within Gabor response of face candidate region
%INPUTS:  resp: Face candidate Gabor filtering response
%         cent_est:  estimated lip coordinates within resp
%OUTPUTS: verts: 1x4 vector containing left, right, top, and bottom indices
%                which approximate lip bounds
%AUTHOR:   Robert Hursig, California Polytechnic State University
%DATE:     7/1/2009
%--------------------------------------------------------------------------
    %Declare Constants
    rsf = [0.5 1];
    M = size(resp,1);
    N = size(resp,2);
    buf = floor(rsf*N/4);
    r_nv = 3;   %top/borrom invalid border for max detection
    lb = floor([max(1,cent_est(1)-buf(1)) max(1,cent_est(2)-buf(2))]);
    ub = floor([min(M,cent_est(1)+buf(1)) min(N,cent_est(2)+1.5*buf(2))]);
    cent_est_p = floor(cent_est) - lb.*[lb(1)~=1 lb(2)~=1]...
                -(buf-lb).*[lb(1)==1 lb(2)==1];
    lipresp=resp(lb(1):ub(1),lb(2):ub(2));
    Mp=size(lipresp,1);
    Np=size(lipresp,2);

    %Get Column (Horizontal) Boundaries
    ppk_h = [0.1 0.1];        %percentage of min/max cropped to,1=L,2=R
    rs = sum(lipresp,1);      %Row Sum Vector (Horizontal Proj)
    mpd = floor(Np/6);
    [cpks clocs] = findpeaks(rs,'minpeakdistance',mpd);
    [tval i_close]=min(abs(clocs-cent_est(2)));
    Imax = clocs(i_close);vmax=cpks(i_close);
    lrmin = [min(rs(1:Imax)) min(rs(Imax+1:Np))];
    thr = lrmin+(vmax-lrmin)*ppk_h';
    %Get Right Boundry
    [rtmp rloc]=find(rs(Imax+1:Np)<thr(2),1,'first');rloc=rloc+Imax;
    if isempty(rloc),[rtmp rloc]=min(rs(Imax+1:Np));rloc=rloc+Imax;end
    %Get Left Boundry
    [ltmp lloc]=find(rs(Imax:-1:1)<thr(1),1,'first');
    if isempty(lloc),[ltmp lloc]=min(rs(1:Imax));else lloc=Imax-lloc+1; end
```

```
    %Get Row (Vertical) Boundaries
    ppk_v = [0.05 0.15];
    %Windowed Column Sum Vector
    cs = sum(lipresp(:,lloc:rloc),2); cs=cs-mean(cs);
    i_cs = zeros(1,Mp);for i=1:Mp,i_cs(i)=sum(cs(1:i));end
    [vmax Imax]=max(i_cs(r_nv:Mp-r_nv+1)); Imax=Imax+r_nv;%determine max of integral vector

    [vmin Imin]=min(i_cs(Imax:Mp));Imin=Imin+Imax-1;  %determine min of integral vector
    [btmp bloc]=find(i_cs(Imax:Imin)<(vmax-vmin)*ppk_v(2)+vmin,1,'first');bloc=bloc+Imax-1;
    if isempty(bloc),bloc=Imax;end  %defaults to max

    [vmin Imin]=min(i_cs(1:Imax));  %determine min of integral vector
    [ttmp tloc]=find(i_cs(Imin:Imax)>(vmax-vmin)*ppk_v(1)+vmin,1,'first'); tloc=tloc+Imin;
    if isempty(tloc),tloc=Imin;end  %defaults to min

    verts=[lloc,rloc,tloc,bloc]+[lb(2)-1,lb(2)-1,lb(1)-1,lb(1)-1];

    %EXTRAS
    sfigure(10);clf
    subplot(2,2,1);imshow(lipresp);axis tight
    text(cent_est_p(2),cent_est_p(1),'+','Color','g','FontWeight','bold',...
        'FontSize',16,'HorizontalAlignment','center',...
        'VerticalAlignment','middle');
    line([lloc lloc],[1 Mp],'LineWidth',2,'Color','m')
    line([rloc rloc],[1 Mp],'LineWidth',2,'Color','r')
    line([1 Np],[tloc tloc],'LineWidth',2,'Color',[0.251 0.878 0.816])
    line([1 Np],[bloc bloc],'LineWidth',2,'Color','b')
    subplot(2,2,2);hold on, plot(cs),plot(i_cs,'r'), hold off;view(90,90);axis tight
    xlabel('Row Location within Window (r)')%ylabel('Row Integral Signal, S_r(r)'),
    legend('Column Concentration Signal, D_c_,_L(r)','Integral Signal, S_r(r)')
    line([tloc tloc],[-100 100],'LineWidth',2,'Color',[0.251 0.878 0.816])
    line([bloc bloc],[-100 100],'LineWidth',2,'Color','b')
    subplot(2,2,3);plot(rs);axis tight
    xlabel('Row Concentration Signal, D_r(c)'),ylabel('Column Location within Window (c)')
    line([lloc lloc],[1 Mp],'LineWidth',2,'Color','m')
    line([rloc rloc],[1 Mp],'LineWidth',2,'Color','r')
    subplot(2,2,4);imshow(rgb)
    text(cent_est(2),cent_est(1),'+','Color','g','FontWeight','bold',...
        'FontSize',16,'HorizontalAlignment','center',...
        'VerticalAlignment','middle');
    rectangle('Position',[verts(1)-2,verts(3)-2,verts(2)-verts(1)+4,...
            verts(4)-verts(3)+4],'LineWidth',2,'EdgeColor','g');
    %END EXTRAS
end
```

APPENDIX B: Gabor Filter Toolbox Code

```
function [g]=gfcreatefilter2(f0,theta,gamma,eta,n,varargin)
% GFCREATEFILTER2 Create normalized 2-D Gabor filter in the spatial domain.
%
%   G = GFCREATEFILTER2(F0,THETA,GAMMA,ETA,N,...) creates a
%   two-dimensional normalized Gabor filter G with frequency F0,
%   orientation THETA, normalized width GAMMA along the wave,
%   normalized width ETA orthogonal to the wave, and size N.
%   If N is a scalar, G will have equal number of rows and
%   columns. Also a two element vector N=[NX NY] can be used to
%   specify the size.
%
%   G = GFCREATEFILTER2(...,'PF',PF) determines that at least
%   P percent of the Gaussian envelope of the filter must be
%   included in the filter in frequency domain. For default,
%   PF=0.998.
%
%   G = GFCREATEFILTER2(...,'PT',PT) determines that at least
%   P percent of the Gaussian envelope of the filter must be
%   included in the filter in spatial domain. For default,
%   PT=0.998.
%
%   Examples
%
%   See also GFCREATEFILTERF2, GFCHECKFILTER2, GFCREATEFILTERF.
%
% References:
%   [1] Kamarainen, J.-K., Kyrki, V., Kalviainen, H., Gabor
%       Features for Invariant Object Recognition, Research
%       report 79, Department of Information Technology,
%       Lappeenranta University of Technology
%
% Author(s):
%    Joni Kamarainen <Joni.Kamarainen@lut.fi>
%    Ville Kyrki <Ville.Kyrki@lut.fi>
%
% Copyright:
%
%   The Gabor Features in Signal and Image Processing Toolbox is
%   Copyright (C) 2000 by Joni Kamarainen and Ville Kyrki.
%
%
%   $Name: V_0_4 $ $Revision: 1.9 $  $Date: 2003/03/03 10:51:17 $
%
pt=0.998; % corresponds approximately to (1-1/512)
pf=0.998;

if mod(length(varargin),2)>0,
  error('Each parameter must be given a value.');
end;

currentarg=1;
while length(varargin)>currentarg,

  [param,value]=deal(varargin{currentarg:currentarg+1});

  switch lower(param)
   case 'pt'
    pt=value;
   otherwise
    error(['Unknown parameter ''' param '''.']);
  end;

  currentarg=currentarg+2;
end;
```

```
%gfcheckfilter2(f0,theta,gamma,eta,n,pt,pf);

alpha=f0/gamma;
beta=f0/eta;

if length(n)>1,
  nx=n(1);
  ny=n(2);
else
  nx=n;
  ny=n;
end;

% Parittomalla pituudella indeksit -(n-1)/2:(n-1)/2
% Parillisella -(n/2):(n/2-1)
% Esim. 9 -> -4:4, 8 -> -4:3
if mod(nx,2)>0,
  tx=-((nx-1)/2):(nx-1)/2;
else
  tx=-(nx/2):(nx/2-1);
end;

if mod(ny,2)>0,
  ty=-((ny-1)/2):(ny-1)/2;
else
  ty=-(ny/2):(ny/2-1);
end;

[X,Y]=meshgrid(tx,ty);

g=abs(alpha*beta)/pi*exp(-alpha^2*(X*cos(theta)+Y*sin(theta)).^2-...
            beta^2*(-X*sin(theta)+Y*cos(theta)).^2 +...
            j*2*pi*f0*(X*cos(theta)+Y*sin(theta)));
```