Content-Based Representation of Sign Language Video Sequences

Nariman Habili

B.Sc., B.Eng. (Flinders)

A thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

in the

School of Electrical and Electronic Engineering

The University of Adelaide

Australia

September, 2002

Contents

List of Figures

List of Tables

Abstract

Sign language is a visual language used by deaf or hearing-impaired people to communicate. For distant communication, deaf people commonly use the text telephone, which is at least 10 times slower than sign language. Moreover, sign language is the first language of many pre-lingually deaf individuals, and its speed is comparable to that of normal speech. Video communication would allow deaf individuals to communicate remotely via sign language, providing them the equivalent of the telephone for individuals of normal hearing. Therefore, video communication would be a boon to the deaf community.

Block-based video coding strategies, the cornerstone of the H.261 and H.263 coding standards for video conferencing, are unsuitable for the transmission of sign language video over affordable low bit-rate channels. This is mainly due to the presence of rapid hand and arm motion in sign language video, as well as the necessity of smooth motion perception. Accordingly, sign language video will require content-based coding strategies to achieve the image quality and frame rate necessary for accurate perception. Using content-based coding, video sequences are typically segmented into different objects which may be independently coded and transmitted. More resources are allocated to the perceptually important objects, which in the case of sign language, are the face and hands.

In this thesis, a methodology is devised for the segmentation of the face and hands in sign language video sequences. As well as an improved coding performance, the content-based representation of video data would allow other functionalities, such as improved error-robustness and scalability. The proposed algorithm employs color and motion cues to segment the face and hands. First, a color segmentation algorithm is devised to locate skin-color regions in each frame. Second, we note that sign language is characterized by the motion of the hands and the face. Based on this observation, the proposed face and hand segmentation

methodology employs motion information to locate the moving skin-color regions in each frame. To this end, a statistical change detection method is proposed based on the F test and block-based motion estimation. In addition to the face and hand segmentation methodology, a face detection and temporal tracking method is also presented. This has applications in lip-reading, where more coding resources are allocated to the face.

The performance of the skin-color segmentation algorithm is demonstrated by simulations carried out on both still images and video sequences. The proposed change detection method is tested on four video sequences. The simulation results demonstrate the effectiveness of the proposed face and hand segmentation methodology, and the face detection and tracking method.

Statement of Originality

I hereby declare that this work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I give consent to this copy of my thesis, when deposited in the University Library, being available for loan and photocopying.

Nariman Habili

1 September 2002

Acknowledgments

I would like to express my gratitude and sincere thanks to my supervisors, Dr. Cheng Chew Lim and Dr. Alireza Moini, for their guidance, interest, and technical assistance, and also for reading my thesis. I remain indebted to Dr. Lim for taking me under his wings after the departure of Dr. Moini from the University. I would also like to thank Professor Neil Burgess for offering me a postgraduate research scholarship, and for supervising this research before leaving for greener pastures in the UK. Neil, the next beer is on me.

I would like to thank all staff and postgraduates in the School of Electrical and Electronic Engineering for their assistance, and generally for making my studies here an enjoyable experience. My appreciation goes to Mr. Hooman Nikmehr for proofreading my thesis, and for his help with my thesis while I was in Singapore.

I would also like to express my gratitude to Mr. Gunnar Hellström of Omnitor (Stockholm, Sweden) for the informative discussions on sign language video communication, and to Dr. Richard Schumeyer for providing me with a copy of his PhD thesis.

I must pay tribute to members of the School's soccer team ("Adelaide Aardvarks FC"), and especially to Mr. David Bowler for guiding us to cup victory in 2000.

Thanks are also due to my family and friends, who supported and encouraged me during the course of my studies. Finally, I would like to acknowledge the postgraduate research scholarship from the Australian Research Council and the School of Electrical and Electronic Engineering.

Dedication

You only live twice,

or so it seems,

one life for yourself,

and one for your dreams.

–From "You Only Live Twice", sang by Nancy Sinatra.

This thesis is dedicated to my late grandmother.

Publications

1. **N. Habili**, C. C. Lim and A. R. Moini. Segmentation of the face and hands in sign language video sequences using color and motion cues. *Submitted to IEEE Trans. on Circuits and Systems for Video Technology*, February 2002.

2. **N. Habili**, C. C. Lim and A. R. Moini. Automatic human skin segmentation based on color information in the YCbCr color space. In *Proc. Information, Decision and Control*, Adelaide, Australia, February 2002.

3. **N. Habili**, C. C. Lim and A. R. Moini. Hand and face segmentation using motion and color cues in digital image sequences. In *Proc. IEEE International Conference on Multimedia and Expo*, Tokyo, Japan, August 2001.

4. **N. Habili**, A. R. Moini and N. Burgess. Histogram based temporal object segmentation for VOP extraction in MPEG-4. In *Proc. IEEE Pacific Rim Conference on Multimedia*, pages 310-313, Sydney, Australia, December 2000.

5. **N. Habili**, A. R. Moini and N. Burgess. Automatic thresholding for change detection in digital video. In *Proc. SPIE Visual Communications and Image Processing*, pages 133-142, Perth, Australia, June 2000.

6. **N. Habili**, A. R. Moini, and N. Burgess. A variable search count block-matching algorithm for video coding. In *Proc. IEEE Region Ten Conference*, pages 108-111, Cheju, Korea, September 1999.

List of Principal Symbols

$\mathcal{A}$	aspect ratio
$\mathcal{C}$	connected component
H_0	null hypothesis
H_1	alternative hypothesis
I	moment of inertia
$\mathcal{O}$	compactness
P_D	probability of detection
P_{error}	probability of error
P_F	probability of false alarm
P_M	probability of miss
$P(\omega_S)$	*a priori* probability of the skin class
$P(\omega_{\bar{S}})$	*a priori* probability of the non-skin class
R_M	miss rate
R_F	false alarm rate
$\mathbf{R}$	rotation matrix
$\mathcal{R}_S$	skin decision region
$\mathcal{R}_{\bar{S}}$	non-skin decision region
S^2	sample variance
$\mathcal{S}$	solidity
T_i	tristimulus value
$\mathbf{U}$	matrix containing eigenvectors
$\mathbf{c}$	feature vector
d	Mahalanobis distance

$\mathbf{e}$	eigenvector
k	frame number
$p(\mathbf{c}\|\omega_S)$	the conditional probability density of $\mathbf{c}$ given ω_S.
$p(\mathbf{c}\|\omega_{\bar{S}})$	the conditional probability density of $\mathbf{c}$ given $\omega_{\bar{S}}$.
v_x, v_y	motion vectors
$\mathbf{\Sigma}$	covariance matrix
$\mathbf{\Upsilon}$	diagonal matrix containing eigenvalues
α	significance level
θ	threshold value
μ	population mean
$\xi_{p,q}$	(p, q) central moments
$\boldsymbol{\mu}$	mean vector
ρ_S	spectral surface reflectance
ρ_B	spectral body reflectance
σ^2	population variance
τ	threshold value
υ	eigenvalue
ϕ	angle
$\varphi(.)$	threshold function
ω_S	skin class
$\omega_{\bar{S}}$	non-skin class

List of Abbreviations

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
BDF	Binary Difference Frame
bps	Bits Per Second
CDM	Change Detection Mask
CIE	Commission Internationale de l'Eclairage
CIF	Common Intermediate Format
DF	Difference Frame
EM	Expectation-Maximization
FHSM	Face and Hand Segmentation Mask
GOV	Group of Video Object Planes
HVS	Human Visual System
ISDN	Integrated Services Digital Network
LL	Low Low
LMS	Least Median of Squares
MAE	Mean Absolute Error
MPEG	Moving Picture Experts Group
NTSC	National Television Standards Committee
PAL	Phase Alternation by Line
PSNR	Peak Signal to Noise Ratio
QCIF	Quarter Common Intermediate Format
ROC	Receiver Operating Curve

VO	Video Object
VOL	Video Object Layer
VOP	Video Object Plane
SDM	Skin Detection Mask
SLF	Segmentation Label Field
SR	Skin Region

Chapter 1

Introduction

"The beginning is the most important part of the work."

- Plato, The Republic, Book II, 377B

This chapter provides an overall introduction to the thesis. Content-based representation is reviewed, and the research objectives defined. The major contributions of the thesis are listed, and an outline of the thesis is provided.

1.1 Content Based Representation: An Overview

The text telephone is a commonly used device by deaf or hearing impaired individuals for distant communication. However, text communication is susceptible to misunderstandings and omissions [Hel00], and its speed is at least 10 times slower than that of sign language [Hel97]. Moreover, sign language is the first language of many pre-lingually deaf individuals, and its speed is comparable to that of normal speech [IT99]. As a result, affordable and effective video communication would be a boon to the deaf population. Unfortunately, videoconferencing and videotelephony technology accessible to individuals of normal hearing, are seldom suited to sign language communication. This puts deaf and hearing-impaired people at a disadvantage, and creates obstacles to education, job opportunities, and social life [Hel97].

Digital video communication is characterized by the generation, manipulation, and transmission of an enormous amount of data. Video sources usually generate more data than can be transmitted over current low-cost communication channels. For example, the approximate bit-rate required to transmit a color video sequence at 30 frames per second (fps) with a frame-size of 352 $\times$ 288 pixels[1] and a pixel resolution of 24 bits is 73 million bits per second (bps). To get some idea of the significance of this rate, note that the current telephone modem rate is 56 kbps. Transmitting one second of this color video sequence over a 56 kbps modem would require 21.73 minutes (more if the telephone line is too noisy). A more expensive solution is to transmit the video sequence over an ISDN (integrated services digital network) line. The transmission time for one second of the sequence over a 128 kbps ISDN line is 9.51 minutes. Due to the vast amount of data associated with video, compression is a key requirement for its digital transmission. Based on current trends, the demand for video communication will continue to outpace increases in channel capacity. Hence, the importance of video coding (also called video compression) is unlikely to diminish, in spite of the promises of unlimited bandwidth [GGKV98].

Current video coding standards can be categorized into block-based[2] (Section 2.2.1) and content-based (Section 2.2.2) video coding. In block-based video coding, a video sequence

[1] This is the common intermediate format (CIF) luminance image size. See Appendix A.

[2] Also known as waveform based video coding.

is compressed without any regard to its semantical content. On the other hand, content-based video coding identifies regions and semantical objects in a video sequence and compresses those.

1.2 Sign Language Video Communication

Sign language video sequences have characteristics different from those of a typical head-and-shoulder sequence. Therefore, block-based video coding strategies, the cornerstone of the H.261 [IT93] and H.263 [IT98] coding standards for video communication (i.e., video-conferencing and videotelephony), are unsuitable for the transmission of sign language video sequences over affordable low bit-rate channels [Sch98]. This is mainly due to the presence of rapid hand and arm motion in sign language video, as well as the necessity of smooth motion perception. Accordingly, sign language video will require content-based coding strategies, as advocated in the MPEG-4 standard [Gro01], to achieve the image quality and frame rate necessary for accurate perception [Sch98]. Using content-based coding, video sequences are typically segmented into semantically meaningful objects which may be independently coded and transmitted. More resources are allocated to the perceptually important objects. As well as improved coding efficiency, content-based representation enables other functionalities, such as improved error-robustness, and scalability.

The language carrying components in sign language are movements and positions of the hands, the eyes, the mouth, and the face [IT99, Sch98]. Therefore, the perceptually significant objects in sign language video are the face (which includes the eyes and the mouth) and the hands.

1.3 Research Objectives

The primary goal of this thesis is to devise a methodology for the segmentation of the face and hands in sign language video sequences. The segmentation of the face and hands will provide a content-based representation of sign language video for different content-based functionalities.

In this thesis, we employ two cues to segment the face and the hands. The first cue is

color. The human skin has a special color distribution that differs significantly (although not entirely) from those of the background objects [CN99]. Color information is used to localize skin-color regions in each frame.

As noted above, sign language is characterized by the motion of the hands, the eyes, the mouth, and the face. Accordingly, the second cue that we employ is motion. A statistical hypothesis test is employed to segment the video frames into "changed" and "unchanged" regions with respect to the previous frame. Changed regions represent the moving and occlusion regions, and unchanged regions represent the stationary background. The moving skin-color regions (i.e., the face and the hands) are then identified based on the color and motion information.

Having segmented the face and the hands, it may sometimes be necessary to discriminate between the two. This is required in applications such as lip-reading, where more coding resources are allocated to the face. Therefore, the other objective of this thesis is to detect the face in a video frame, and then to track it throughout the sequence.

The block diagram of the face and hand segmentation methodology is shown in Figure 1.1, and the block diagram of the face detection and tracking methodology is shown in Figure 1.2. The flowcharts are intended to give the reader an overview of the scope covered in this thesis. Basically, hand and face segmentation is the result of color segmentation and change detection. The face and hand segmentation mask ($FHSM$) is then employed for the purpose of face object detection and tracking. The skin-segmentation and change detection methods presented in this thesis are self-contained and independent; each can be utilized as a component of another algorithm, e.g., face recognition for skin-color segmentation and security surveillance for change detection. Therefore, besides sign language video sequences, we have also tested our proposed methods on other video sequences and still images.

1.4 Contributions of the Thesis

The major contributions made in this thesis are:

- An analysis of the dichromatic reflection model and its relation to skin-color.

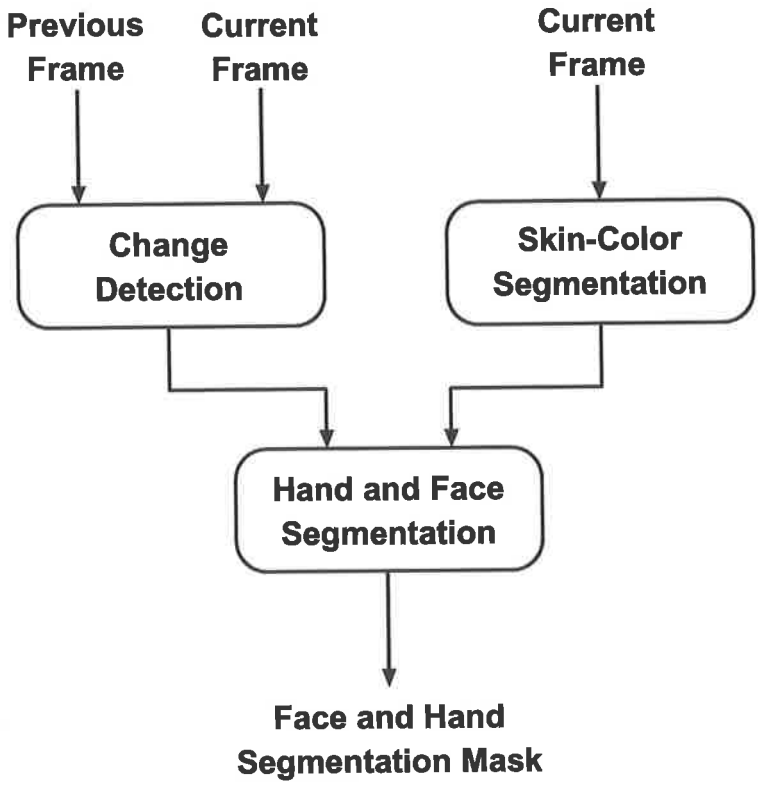


Figure 1.1: Block diagram of the face and hand segmentation methodology.

- The development of a methodology for the segmentation of the face and hands in sign language video sequences. To the best of our knowledge, the only other study on face and hand segmentation for sign language video communication in the framework of MPEG-4 is by Schumeyer [Sch98]. Unlike Schumeyer's approach, our face and hand segmentation methodology is intended to work on a range of skin colors, lighting conditions, and background complexities. Moreover, our algorithm does not require a separate face detection algorithm to generate a skin-color model. The following were developed as part of the face and hand segmentation methodology:
  - A new skin-color segmentation algorithm.
  - A new change detection technique based on the F test and block-based motion estimation.
- The development of a new methodology for face detection and tracking. As part of the face detection and tracking methodology, the following techniques were developed:

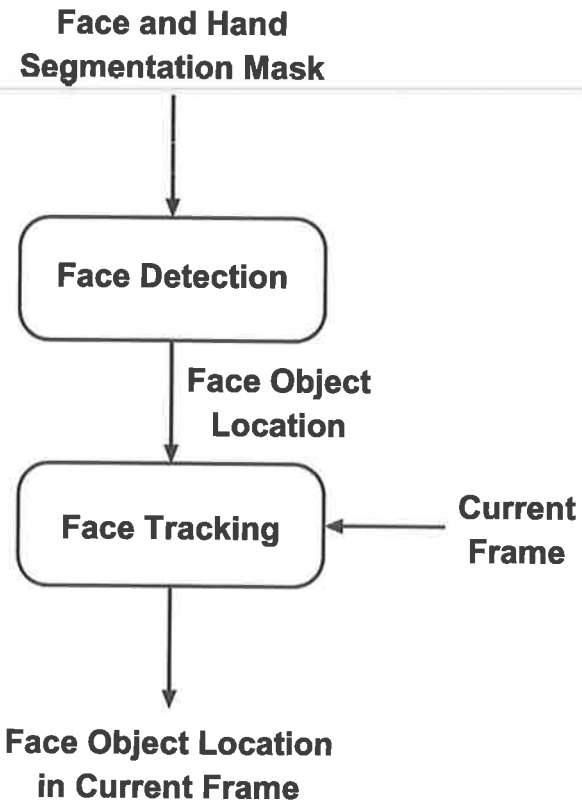


Figure 1.2: Block diagram of the face detection and tracking methodology.

- A new technique for face detection based on shape features.
- Two different face tracking techniques.

1.5 Outline of the Thesis

Chapter 2 presents background information on segmentation, video coding, and sign language. In particular, we discuss the differences between block-based and content-based video coding, and emphasize the significance of video communication for deaf and hearing-impaired people.

Chapter 3 presents an introduction to color. We review the basics of light and color, and explore certain aspects of the human visual system, such as eye anatomy, color perception, and the opponent color model of chromatic vision. We also consider the trichromatic theory of color mixture, and the dichromatic reflection model. The color spaces, CIE XYZ, YUV,

YIQ, and YCbCr, are also reviewed.

Chapter 4 presents an introduction to motion in video. We discuss different camera models, and then consider two-dimensional and three-dimensional motion models. We also distinguish between two-dimensional motion and apparent motion.

Chapter 5 presents our skin-color segmentation algorithm. For skin-color segmentation, we employ the YCbCr color space, and obtain training data by manually segmenting training images into skin and non-skin classes. The skin class training pixels are modeled as a bivariate normal distribution in the CbCr plane. Image pixels are classified as skin or non-skin based on their Mahalanobis distance. A segmentation threshold is derived for the classifier. The performance of the algorithm is illustrated by simulations carried out on still images, and on sign language video sequences. A literature survey of different skin-color segmentation algorithms is also provided.

Chapter 6 discusses statistical change detection. A change detection technique based on the F test and block-based motion estimation is proposed. Simulation results are presented for four different video test sequences. Previous research pertaining to change detection is also discussed.

Chapter 7 discusses the generation of the hand and face segmentation mask. The $FHSM$ is a binary map that indicates the face and hand regions in a frame. Techniques are also introduced for the detection and tracking of the face, which may be required in applications such as lip-reading.

Chapter 8 provides the overall conclusions of the thesis, and discusses some avenues for future research.

Chapter 2

Segmentation, Video Coding, and Sign Language: An Overview

"While the task of image segmentation hardly has a counterpart in human visual experience, it is a nontrivial task in digital image analysis."

- Kenneth R. Castleman in Digital Image Processing, 1996

This Chapter presents background information on segmentation, video coding, and sign language. In Section 2.1, we define image and video segmentation, and discuss a general scheme for segmentation algorithms. Video coding is discussed in Section 2.2, the characteristics of sign language video are discussed in Section 2.3, and the chapter is summarized in Section 2.4.

2.1 Segmentation

The decomposition of an image into non-overlapping parts is an effortless process for a human being. Humans use numerous cues, such as color, motion, texture, and shape to aid the segmentation process. These cues are then analyzed and matched against objects stored in the memory. Unfortunately, the task of image or video segmentation is far from effortless in digital image analysis. Autonomous segmentation is one of the most difficult tasks in image processing [GW92]. The state of the art has still to be improved to lead to robust segmentation algorithms able to deal with generic images and video sequences. Image segmentation has been widely and actively studied for the last few decades, with applications such as image understanding, robot vision and face recognition. Moreover, in recent years, research activity in segmentation has intensified as a result of its applications being extended towards image and video coding.

The goal of image segmentation is to partition an image into a set of non-overlapping regions whose union is the entire image. The purpose is to decompose an image into parts that are meaningful with respect to a particular application. For example, in 2D object recognition, segmentation might be performed to separate a 2D object from the background. Figure 2.1(a) shows a gray-level image of four coins, and Figure 2.1(b) shows its gray-level histogram. One obvious way to extract the coins from the background is to select a threshold that separates the two modes in the histogram. Figure 2.1(c) shows the result of segmenting Figure 2.1(a) by using a threshold of 170. The segmented coins are shown in white (i.e., binary "1"), and the background is shown in black (i.e., binary "0").

It is very difficult to define what constitutes a "meaningful" segmentation. However, general segmentation procedures tend to obey the following rules [HS92]:

1. Regions of a segmented image should be uniform and homogeneous with respect to some characteristics, such as grey-level or texture.

2. Region interiors should be simple and without many small holes.

3. Adjacent regions of a segmented image should have significantly different values with respect to the characteristics on which they are uniform.

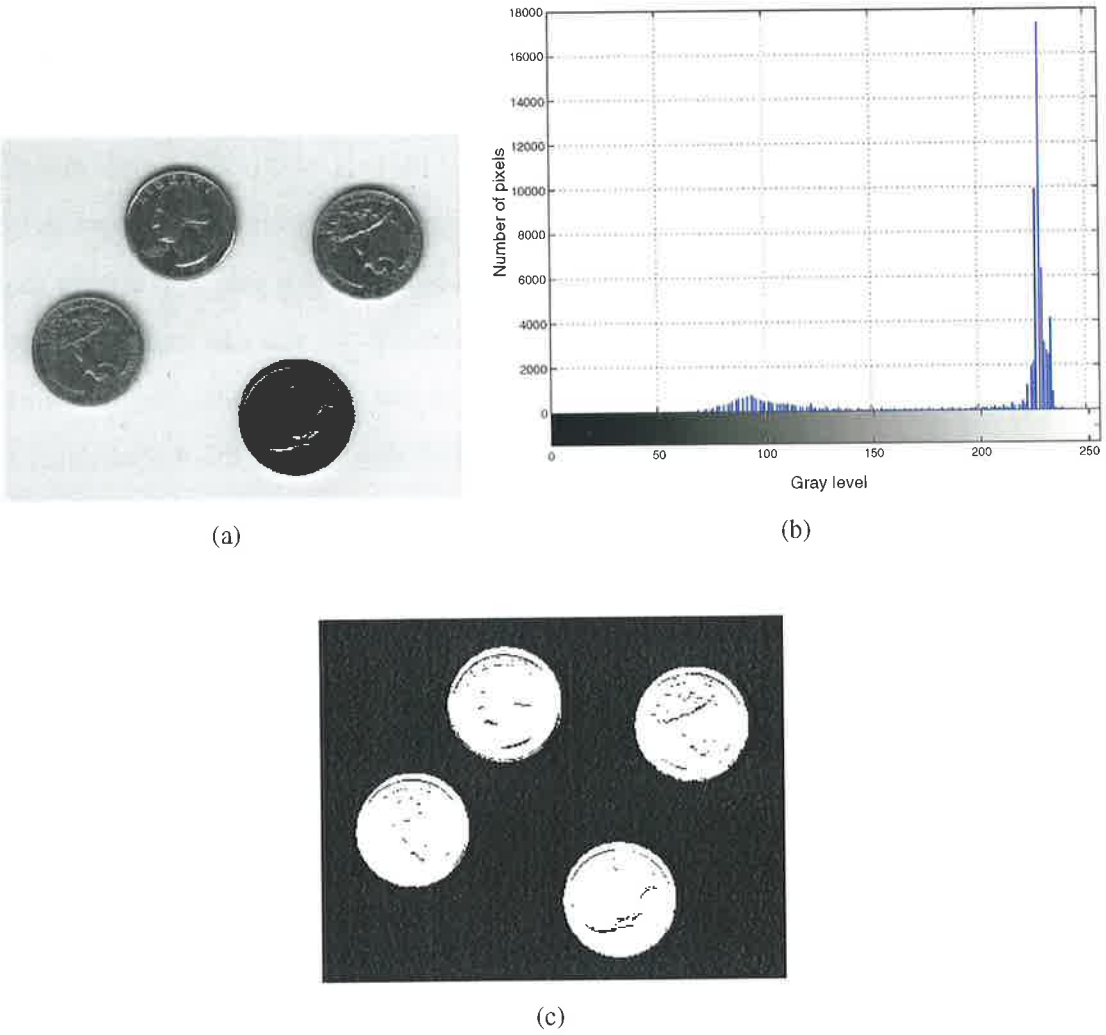


Figure 2.1: Example of gray-level thresholding. (a) Original image, (b) the histogram of the image, and (c) result of segmentation.

4. Boundaries of each segment should be simple, not ragged, and must be spatially accurate.

Achieving all these desired properties is difficult because strictly uniform and homogeneous regions are typically full of small holes and have ragged edges. Also, insisting that adjacent regions have large differences in values can cause regions to merge and boundaries to be lost. While the above properties refer to still image segmentation, the temporal extension of segmentation is straightforward when referred to video sequences. In this case, motion parameters are included among the characteristics that are considered in the segmentation process. The resulting video partition is required to show the same properties of

uniformity and homogeneity that were required in the case of still images.

It is important to distinguish between *regions* and *objects*. A region is defined as a set of connected pixels that is homogeneous with respect to a given quantitative criterion, such as gray-level[1], color, texture, motion, or a combination of those [Cas98]. The formal definition of *connectedness* is as follows: between any two pixels in a connected set, there exists a connected path wholly within the set, where a connected path is a path that always moves between neighboring pixels [Cas96]. Thus in a connected set you can trace a connected path between any two pixels without ever leaving the last. Meanwhile, our definition of object is in accordance with the definition of the video object in MPEG-4 (Section 2.2.2), i.e., "a video object in a scene is an entity that a user is allowed to access (seek, browse) and manipulate (cut and paste)" [Gro98]. Objects are characterized by their semantical meaning, unlike regions. An object may be formed by the union of several regions.

Salembier and Marqués [SM99] have advocated a general scheme for segmentation. The scheme is the concatenation of three steps, namely simplification, feature extraction, and decision. The steps are illustrated in Figure 2.2 and summarized below.

- **Simplification:** Most often, the original data pertaining to an image or a video sequence contains information that is irrelevant for a given application. The data can be simplified by removing or filtering the irrelevant information. The simplification step controls the amount and nature of the information that is preserved. Furthermore, the simplified data should contain areas that are easier to segment. For example, simplification can reduce the complexity of textured areas or remove small details. Note that simplification should not modify the boundary information that is relevant for the application.

- **Feature extraction:** Segmentation relies on specific features of the data. The selection of the *feature space* determines the type of homogeneity that is expected in the final partition. In some applications, the pixel values correspond directly to the feature of interest (e.g. color segmentation), and in other applications, the feature has to be estimated from the data. Note that, in some cases, feature estimation has to be performed on a region of support which should be homogeneous in terms of the same

[1] The gray-level is the luminance component of an image.

feature. As a result, a loop is sometimes introduced (the dashed line in Figure 2.2) in the segmentation process so that the estimation depends on the segmentation results. The final result is obtained through an iterative process.

- **Decision:** The feature space is analyzed to obtain a partition of the data. The decision step decides on the position of the boundaries that form the partition in the temporal (1D), spatial (2D), or spatio-temporal (3D) decision space. The boundaries separate data areas that contain elements sharing the same characteristics in the selected feature space.

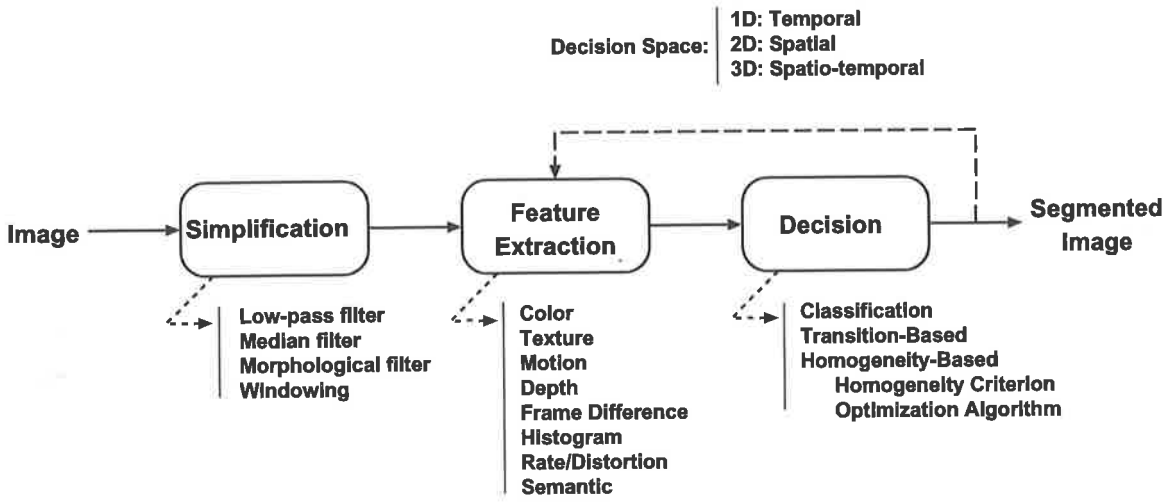


Figure 2.2: The major segmentation steps and the typical tools used.

Segmentation algorithms often use more than one feature. This can be achieved either through the definition of a complex criterion combining several features, or through the use of several segmentation steps that use different criteria. For example, the application of various degrees of simplification allows the analysis to be performed at several levels of resolution and, at each resolution level, a specific feature space may be used. The feature space can be complex to define if the segmentation process allows user interaction. In such cases, a user can implicitly introduce semantic notions which might not be easily obtained by any automatic analysis of the data. As a result, it is often not possible to classify the segmentation algorithms as a function of the feature space they use [SM99].

2.1.1 Approaches to Still Image Segmentation

Approaches to image segmentation range from purely mathematically based solutions to higher level systems of symbolic representation and manipulation. It is generally the case that the more flexible an image segmentation approach needs to be, the less accurate it becomes. Some common approaches to image segmentation include:

- Gray-level thresholding.
- Clustering.
- Edge detection.
- Region growing.
- Region splitting and merging.
- Statistical classification.

Gray-level Thresholding

Gray-level thresholding [NR79, KI86, CHY89, HS92, PB93, JW97, Ros99] is a popular image segmentation technique. The simplest thresholding technique is to partition an image histogram by using a single threshold (e.g., Figure 2.1). Segmentation is then accomplished by scanning the image pixel by pixel and labeling each pixel as foreground or background, depending on whether the gray-level of that pixel is greater or less than the threshold. This technique is known as *global thresholding*, and its success depends entirely on how well the histogram can be partitioned [GW92]. The counterpart of global thresholding is *local thresholding*. In local thresholding, a threshold is selected based on some local property (e.g., neighborhood average) of the pixel.

Clustering

Image segmentation methods based on clustering are also popular [Pap92, HD94, TG97, RT99, Tur01, Com02]. Clustering, in the context of image segmentation, refers to the classification of pixels into groups according to certain properties of the pixels [Tek95]. In image segmentation, it is expected that feature vectors from regions with a similar appearance

would form groups, known as clusters, in the feature space. If we consider the segmentation of an image into K classes, then the segmentation label field, (SLF), assumes one of the K values at each pixel, i.e., $SLF(x, y) = l, l = 1, ..., K$. The parameters x and y are the spatial coordinates of the image. If the features are scalar, for example pixel intensities, clustering can be considered as a method of determining the $K - 1$ thresholds that define the decision boundaries in the 1D feature space. With L-dimensional vector features, the segmentation corresponds to partitioning the L-dimensional feature space into K regions. A standard procedure for clustering is to assign each sample to the class of the nearest mean [Tek95]. In the unsupervised mode, clustering can be achieved by an iterative procedure known as the K-means algorithm [Rom84, Fuk90, KR90] since the means are initially unknown. A generalized version of the K-means algorithm is the *fuzzy* K-means algorithm [Bez81, BKKP99]. Instead of classifying the feature vectors as belonging to one class or another, in the fuzzy K-means approach, the feature vectors possess a degree of belongingness to each class. An analysis of some work on clustering is given in [Fas99].

Edge Detection

Edge detection [eated93, HSSB96, Fas97, KC97] is commonly employed for detecting discontinuities in gray-level images. An edge is the boundary between two regions in an image with relatively distinct gray-level properties. An *edge image* or an *edge map* is an image in which the gray-levels reflect how strongly each pixel meets the requirements of an edge pixel. This can also be displayed as a binary image showing the location, but not the magnitude, of the edge pixels. An edge map usually shows the outline of each image, but the outline seldom forms closed, connected boundaries required for image segmentation. *Edge linking* is the process of associating nearby edge pixels so as to create a closed, connected boundary. Common edge detection techniques are the Roberts, Sobel, Prewitt, and Canny edge operators [Cas96]. Figure 2.3 shows the binary edge map of Figure 2.1(a) using the Sobel edge operator.

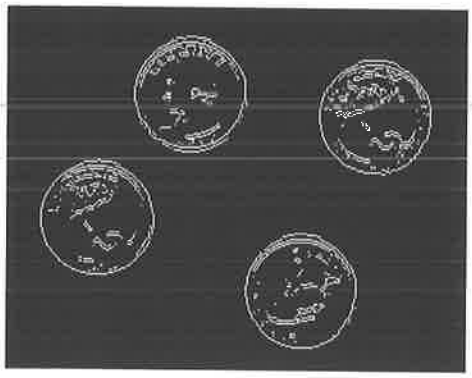


Figure 2.3: Example of edge detection using the Sobel edge operator.

Region Growing

Region growing is a conceptually simple approach to image segmentation [gana98, Til98, GSH01, FYEA01, SGT02]. Segmented regions are formed by grouping together neighboring pixels with similar properties. Region growing techniques usually proceed as follows: the image is partitioned into connected regions by grouping neighboring pixels of similar intensity levels. Adjacent regions are then merged under some criterion involving perhaps homogeneity or sharpness of region boundaries. Region growing methods strongly depend on the estimation of the measure of similarity between pixels and regions, as well as the rules used to establish the level of pixel connectivity (e.g., 4- or 8- neighborhood connectivity). The watershed transform [Cas96] is a popular region growing based image segmentation algorithm.

Region Splitting and Merging

In region splitting and merging [OPR78, GW92], an image is initially subdivided into a set of disjoint regions. The regions are then merged and/or split until each region satisfies some condition indicating that it is homogeneous with respect to some criteria. For example, it is possible to split an image into four quadrants. If the homogeneity condition is not satisfied for a quadrant, then the quadrant is further subdivided into four smaller quadrants. It is also possible to merge neighboring regions whose combined pixels satisfy the condition of connectedness. The process is terminated when no further splitting or merging is possible.

Statistical Classification

Statistical classification based image segmentation methods have also been studied [STB97]. Statistical classification is akin to pattern recognition. The first step in statistical classification is to decide which properties of pixels (e.g., gray-level, color) best distinguishes the pixel types, and how to measure them. The classifier is then designed by establishing a mathematical basis for the classification algorithm, and selecting the type of classifier structure to be used. Once the basic decision rules of the classifier have been established, one must determine the particular threshold values that separate the classes. This is generally achieved by training the classifier on a group of known pixels. The training set is a collection of pixels from each class that have been previously identified by some accurate method (e.g., manual segmentation). Pixels in the training set are then modeled, usually via maximum-likelihood (ML) estimation [DHS01] or the expectation-maximization (EM) algorithm [DLR77], and the feature space is partitioned into regions that maximize the accuracy of the classifier when it operates on the training set. The Bayes classifier [DHS01] is commonly used to classify the image pixels.

2.1.2 The Use of Motion in Video Segmentation

In video segmentation, motion information can be included among the criteria used in the segmentation process. For further information on motion, the reader is referred to Chapter 4. Spatial and motion information are usually combined to segment moving objects in video sequences. Thus, motion segmentation is an integral part of video segmentation. Besides video segmentation, motion segmentation also finds use in many other image sequence analysis problems, including improved optical flow estimation, 3D motion and structure estimation in the presence of multiple moving objects, and higher-level description of the temporal variation and/or the content of video imagery [Tek95].

Tekalp [Tek95] classifies motion segmentation into three categories: direct methods (change detection), optical flow segmentation, and simultaneous estimation and segmentation. Our approach to motion segmentation is based on change detection, which is described in Chapter 6. In change detection, a video frame is segmented into "changed" versus "unchanged" regions with respect to the previous frame. The changed and unchanged regions

are marked in a so-called change detection mask (CDM). Change detection is a popular and robust approach to motion segmentation, however it provides limited information on the motion characteristics of individual components in a scene. More sophisticated motion segmentation algorithms need to address the following requirements:

- Estimation of the number of motion components in a scene.
- Estimation of the motion characteristics of each component.
- Estimation of the spatial support of each component.

The proposal in [KCK+99] employs the CDM and performs spatio-temporal segmentation of moving objects in a video sequence. To this end, regions for which a majority of pixels are classified as changed are assigned to moving objects. In [Wan98], the watershed transform is employed to spatially segment an image into homogeneous regions based on pixel intensity. Adjacent regions with coherent motion are then merged to form a moving object. The object is then tracked in subsequent frames of the sequence.

A semi-automatic video object segmentation approach was described in [TTE00]. Assuming that the boundary of an object of interest is interactively marked on some key-frames, the method finds the boundary of the object in all other frames automatically by tracking the 2D mesh representation of the object in both forward and backward directions. Semi-automatic segmentation is also exploited in [KJK+01].

In [MN98b] and [MN99], a 2D binary model of moving object is derived and tracked throughout the sequence. The edge pixels of the binary model are detected by the Canny edge operator. The main assumption of the approach is the existence of a dominant global motion that can be assigned to the background. Individual objects that do not follow the background motion indicate the presence of independently moving objects. A morphological motion filter or a CDM is used to identify such objects.

A video segmentation approach based on multiple features was presented in [CEK98] and [Cas98]. The extraction of regions is based on a multidimensional analysis of several image features by a spatially constrained fuzzy K-means algorithm. The features considered were position, motion, texture, and color. The local level of reliability of the features is taken into account in order to adaptively weight the contribution of each feature in the

segmentation process. In order to track the image regions in the sequence, an average feature vector for each region is evaluated. The correspondence among regions is assessed by comparing their average feature vectors. Another video segmentation method based on multiple features was presented in [KS01]. In this paper, the authors employed position, motion, and color within a maximum *a posteriori* framework. Weights were assigned to each feature and then adjusted for every pixel based on a confidence measure of the feature. The authors did not employ tracking to find correspondence among the regions in the sequence. A comprehensive literature review on motion segmentation is provided in [ZL01].

2.2 Video Coding

Compression is a process intended to yield a compact digital representation of a signal. The goal of video compression (also called video coding) is to reduce the bit-rate of a video sequence so that it is feasible to transmit the sequence in real-time over a given communication channel. In addition to video communication, compression is also necessary for the storage and retrieval of video data, where different storage media have different storage capacities and access rates, thus demanding varying amounts of compression. Due to the wide range of data rates and applications, different video coding algorithms have been developed. A detailed discussion on compression is outside the scope of this thesis, however it is worth summarizing the main attributes of video coding. For more information on video coding, the reader is referred to texts such as [BK95, MPFL96, Sol97, ES98, GBL+98].

Video coding can be categorized into *lossless* and *lossy* compression [BK95]. Lossless compression techniques seek to eliminate statistical redundancy in a video sequence by exploiting spatial correlation among neighboring pixels, and temporal correlation between consecutive frames [ES98]. Adjacent pixels in the same video frame usually change smoothly and are therefore correlated. Temporal correlation refers to the fact that consecutive frames of a video sequence usually show the same physical scene, occupied by the same objects that may have moved. Statistical redundancy can be removed without destroying any information. That is, the original uncompressed data can be covered exactly by various inverse operations. Therefore, the reconstructed and the original video sequences are identical. Unfortunately, a video coding algorithm based solely on lossless compression will not achieve

the large compression ratios required for video transmission.

As well as spatial and temporal correlation, lossy compression techniques also take advantage of the subjective or perceptual redundancies inherent in a video sequence. For example, the human visual system (HVS) is more sensitive to changes in brightness than to changes in color. Also, the HVS is more sensitive to the low frequency components of an image than to its high frequency components. Unlike statistical redundancy, the removal of information based on the limitations of the HVS is irreversible. The original video data cannot be recovered following such removal. Therefore, lossy compression techniques introduce some distortion in the reconstructed video sequence. It is however possible to construct a lossy compression technique such that the difference between the original and the reconstructed sequence is barely perceptible, and still provide the large compression ratios required for video communication. Almost all video coding techniques in use today are lossy.

The components of a video coding technique are determined to a large extent by the *source model* that is adopted for modeling the video sequence. The video coder seeks to describe the contents of a video sequence by means of its source model. The source model may make assumptions about the spatial and temporal correlation between pixels of a sequence. It might also consider the shape and motion of objects or illumination effects. The basic components of a video coding system are shown in Figure 2.4[2]. In the encoder, the digitized video sequence is first described using the parameters of the source model. If a source model of statistically independent pixels is employed, then the parameters of this source model would be the luminance and chrominance amplitudes of each pixel. On the other hand, if the model describes a scene as several objects, the parameters would be the shape, motion, and texture of the individual objects. Next, the parameters of the source model are quantized into a finite set of symbols. The quantization parameters depend on the desired trade-off between the bit-rate and distortion. The quantized parameters of the source model are finally mapped into binary codewords using lossless coding techniques, which further exploit the statistics of the quantized parameters. The resulting bit-stream is then transmitted over a communication channel (with some added noise) [PS94]. The decoder retrieves the quantized parameters of the source model by reversing the binary encoding and quantization processes of the encoder.

[2]Adapted from [WOZ01], Figure 8.1

Finally, the decoded video frame is synthesized using the quantized parameters of the source model.

Current video coding standards can be categorized into *block-based* and *content-based* video coding. In block-based video coding, a video sequence is compressed without any regard to its semantical content. On the other hand, content-based video coding identifies regions and semantical objects in a video sequence and compresses those.

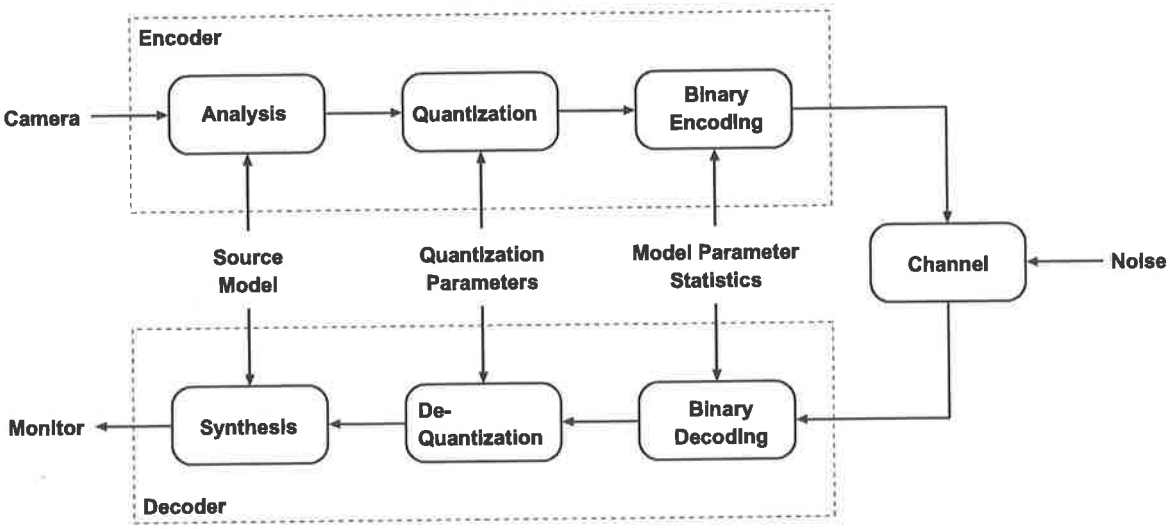


Figure 2.4: Overview of a video compression system.

2.2.1 Block-Based Video Coding

In 1992, the Moving Picture Experts Group (MPEG) completed the ISO/IEC[3] MPEG-1 video coding standard and approved the MPEG-2 standard in 1994 (see http://www.cselt.it/mpeg). These standards made interactive video on CD-ROM and digital television possible. The International Telecommunications Union-Telecommunications (ITU-T) organization established the H.261 standard in 1990 and H.263 in 1995, which are especially targeted to videophone communications. The ITU-T standards made videophone image sequence transmission possible at rates of approximately 64 kbps.

The video coding standards, H.261, H.263, MPEG-1, and MPEG-2, are based on *motion-compensated hybrid coding*, which combines predictive coding with transform coding. This

[3]ISO is the International Organization for Standardization. IEC is the International Electronics Commission.

coding technique subdivides each image into fixed sized blocks of 8×8 or 16×16. Each block in frame k is synthesized using a block of the same size at a displaced position in the previous frame $k-1$. This is performed for all blocks of frame k. The resulting image is called the *predicted image*. The 2D motion vectors for all blocks are transmitted by the encoder to the decoder so that the decoder can compute the same predicted image. The encoder subtracts this predicted image from the original image, resulting in the *prediction error image*. If the synthesis of a block using the predicted image is not sufficiently accurate, then the encoder uses a transform coder (based on the discrete cosine transform) for transmitting the prediction error of this block to the decoder. The decoder adds the prediction error to the predicted image and thus synthesizes the decoded image. In addition to the luminance and chrominance information encoded as transform coefficients of the prediction error, motion vectors have to be transmitted.

The following is a list of the problems caused by the block-based nature of such coding schemes.

- The semantic content of the frame is not taken into account; the partitioning of the frames into blocks results in visible degradation (e.g., blockiness), specially when high compression rates are desired.
- The motion models are applied to square blocks of pixels, which have little resemblance with the motion of objects in the scene.
- Unnatural motion arises when the limited bandwidth forces the frame rate to fall below that required for smooth motion.

Hence, there is a need for new coding schemes that have improved coding efficiency and produce acceptable quality video at very low bit-rates. In the next section, we will describe content-based video coding, which recognizes the problems associated with block-based video coding.

2.2.2 Content-Based Video Coding

To overcome the problems associated with block-based video coding, content-based video coding schemes have been proposed, e.g., [SM95, AH95, LCL+97, TOB97, MS97, TAB97].

Content-based coding schemes segment a video frame into regions corresponding to different semantic objects, and then compress those objects independently. The following is a list of the functionalities enabled by content-based manipulation of video data [Cas98].

- **Multiplex functionalities:** The different semantic objects can be multiplexed separately in the bitstream, allowing the receiver to manipulate each object independently. In addition, the decoder can choose to download a subset of the objects (e.g., if there is bandwidth shortage), and still obtain a semantically meaningful scene. This is commonly referred to as *object scalability*.

- **Improved coding performance:** Different coding strategies can be implemented for different objects in the scene. For example, different objects in a video sequence can be compressed at different rates, depending on their significance to the overall scene.

- **Improved error-robustness:** Parts of the bitstream corresponding to different objects can be protected with different levels of error resilience, both at the source and at the channel coding level, according to their relative relevance to the overall scene. This will guarantee a higher level of subjective quality in error-prone environments.

- **Scalability:** In addition to object scalability, the object-based structure of the bitstream would allow two other types of scalability, namely *temporal* and *spatial* scalability. The bitstream can be structured so as to allow the decoder to retrieve the same object at different levels of spatial and/or temporal resolution.

- **Content description:** Access to the semantic constituents of a scene would allow the description of the scene content in a much more efficient way. This will in turn allow faster access and retrieval of the desired information.

In order to benefit from the different functionalities of content-based video coding, a video frame needs to be decomposed into semantically meaningful objects. Therefore, integral to any content-based coding scheme is video segmentation.

MPEG-4 Standard

This section provides a very brief overview of the MPEG-4 video standard. In 1993, MPEG launched the MPEG-4 standard [Sik97, Gro98, Bra99, Gro01], approving version 1 in Octo-

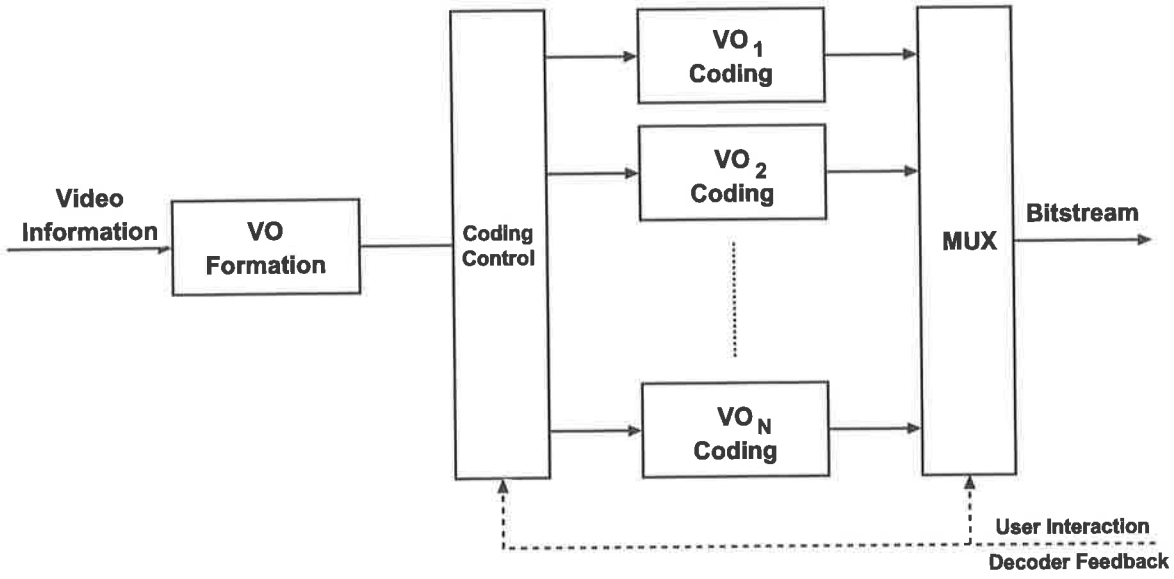
ber 1998 and version 2 in December 1999. MPEG-4 is the first audio-visual representation standard to model a scene as a composition of objects with specific characteristics and behavior. The MPEG-4 standard allows a much more flexible approach to video coding than its predecessors (MPEG-1 and MPEG-2). MPEG-4 is designed to address the requirements of a new generation of highly interactive multimedia applications while supporting traditional applications as well. The standard provides tools for object-based coding[4] of natural and synthetic audio and video, as well as graphics.

MPEG-4 enables content-based interactivity with video objects by coding objects independently using motion, texture, and shape. At the decoder, the objects are composed into a scene and displayed. An MPEG-4 scene consists of several video objects (VOs). A VO corresponds to a particular 2D object in the scene. Each VO can be encoded in a scalable (multi-layer) or a non-scalable form (single layer), depending on the application, represented by the video object layer (VOL). The VOL provides support for scalable compression. A video object can be encoded using spatial or temporal scalability, from coarse to fine resolution. Depending on parameters such as bandwidth, computational power, and user preferences, the desired resolution can be made available to the decoder. A video object plane (VOP) is a time sample of a VO. The VOPs can be either encoded independently of each other, or dependently on each other by using motion compensation. A conventional video frame can be represented by a single VOP with a rectangular shape. VOPs can be grouped together to form group of video object planes (GOV). GOVs provide points in the bitstream where VOPs are encoded independently of each other, and can thus provide random access points into the bitstream. The standard does not prescribe a method for creating the VOs. Depending on the application, VOs may be created in a variety of ways, such as spatio-temporal segmentation of natural scenes, or from parametric descriptions used in computer graphics.

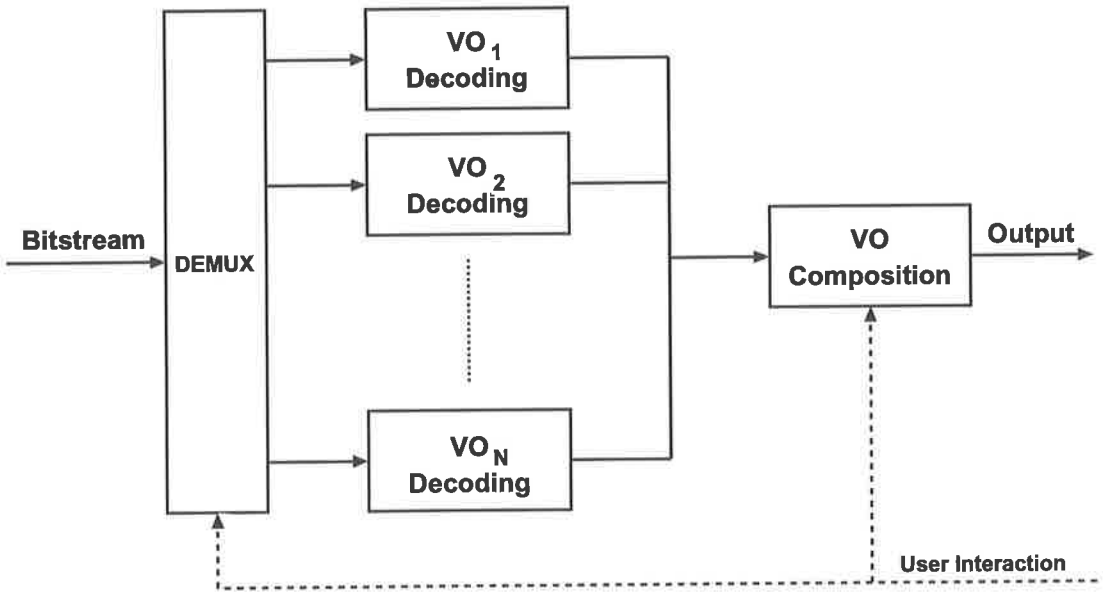
The basic structure of the MPEG-4 codec (i.e., coder-decoder) is shown in Figure 2.5. The encoder, shown in Figure 2.5(b), is subject to MPEG-4 standardization. Methods used to obtain the VOs are not standardized. Similarly, the encoder (Figure 2.5(a)) itself is not standardized, as compliance with the standard is required only at the bitstream level.

A VOP can be utilized in several different ways. In the most common way, the VOP contains encoded video data at a time sample of a VO. The encoded video data includes

[4]In this thesis, we regard object-based and content-based representation as synonymous.



(a)



(b)

Figure 2.5: Structure of the MPEG-4 codec. (a) Decoder, and (b) encoder.

motion parameters, shape information, and texture data. This information is then encoded. The VOPs can also be used to code a *sprite*. A sprite is a VO that is usually larger than the displayed video, and persists over time. It can be modified slightly by changing its brightness, or by warping it to take into account spatial deformation. A sprite is used to represent large, more or less static areas, such as backgrounds.

Figure 2.6(a) shows a frame from the *Akiyo* sequence. The scene consists of a foreground person and a stationary background. The frame is decomposed into a foreground VOP (VOP_1) and a background VOP (VOP_2). The contents of the two VOPs are shown in Figures 2.6(b) and (c).

(a) (b) (c)

Figure 2.6: The concept of the video object plane. (a) Original frame from the *Akiyo* sequence, (b) VOP_1, and (c) VOP_2.

In addition to video coding, MPEG-4 also provides tools for still image coding, mesh animation, and face and body animation.

2.3 Sign Language

Sign language is a visual language used by deaf or hearing-impaired people to communicate. A common device for distant communication between deaf individuals is the *text telephone*. The text telephone is an assistive technology that allows people who are deaf or hard of hearing to communicate directly over a telephone line using standard telephone equipment. This technology is also useful for individuals who have difficulty communicating using a standard telephone due to speech impairment. Unfortunately, the speed of text conversation is limited by a person's typing ability, and is at least 10 times slower than sign language [Hel97]. Moreover, sign language is the first language of many pre-lingually deaf individuals, and

its speed is comparable to that of normal speech [IT99]. Therefore, affordable video communication would greatly benefit the deaf community. In this section, we will summarize the characteristics of sign language and the quality requirements for effective sign language video communication. For further information, the reader is referred to [Hel97] and the international telecommunications union (ITU) supplement H.Supp-1 [IT99].

2.3.1 Characteristics of Sign Language

Sign language has been in existence in some form for thousands of years [Inc01]. Contrary to popular belief, there is no single universal sign language. For example, Australian sign language or Auslan, is the sign language used by the Australian deaf community, and BSL, or British sign language, is one used by the British deaf community. Most sign languages are linguistically complex and sophisticated, with their own grammar and lexicon [SSW99]. Sign language uses distinct movements called signs in place of spoken words and sentences. Although some signs are iconic (i.e., based on natural gesture), most signs are arbitrary (i.e., have no links between the sign and its referent). The signs are based on movements and positions of the hands. Facial expressions, similar to vocal intonation in spoken language, and body language, are also significant.

Sign language is often supplemented by lip-reading and finger-spelling. Lip-reading is a method by which deaf people read the speech of others from the movements of the lips and mouth. On the other hand, finger-spelling allows the spelling of words via different hand-arrangements representing the letters of the alphabet. Words that are finger-spelt do not have a sign (e.g., names of people and places). Finger-spelling presents a considerable challenge for videophone communication, because the finger movements are extremely rapid, and in some cases are recorded in one frame only. Figure 2.7 depicts three frames of the *Irene* sequence, depicting the finger-spelling of the letter "k". Eye blinks are also typical grammatical components of sign language, and are used as sentence delimiters. The blinks are rapid, and in many cases are recorded in one or two frames only.

(a) (b) (c)

Figure 2.7: Three consecutive frames of the *Irene* sequence showing the spelling of the letter "k".

2.3.2 Sign Language Video

Most videophone research conducted to date has concentrated on head-and-shoulder video sequences [CN94, EJ95, Ito96, Zha98, CN99, MW00a, EWG00]. Sign language video sequences, however, have characteristics different from those of a typical head-and-shoulder sequence. While sign language video shares many real-time constraints with videoconferencing, there are additional challenges inherent in the transmission of sign language video. These challenges include the presence of increased motion and the necessity of smooth motion perception. A significant difference in arm position or shape would mean that consecutive video frames in the sequence are less alike. This would result in less compression (or a deterioration in video quality if coded at the same compression rate) since the consecutive frames are less correlated. Sign language video is characterized by the motion of the head, eyes, mouth, and the rapid motion of the arms and hands. Generally there is no global motion (e.g., camera panning, zooming, etc.) in sign language video.

The quality requirements of sign language video for distant communication have been studied by Hellström [Hel97, Hel00]. Hellström observed that for accurate comprehension, the frame rate of sign language video should be at least 20 frames per second (fps) at CIF resolution. As well as video, sign language communication systems are also required to process and transmit text and audio information. People who have impaired hearing, but are not completely deaf, sometimes use voice as well as sign language to communicate.[5] Accordingly, the transmission of sign language video over low-bit rate channels would require

[5]See http://www.omnitor.se for specifications on the "Total Conversation" system.

significant compression.

In his study, Schumeyer [SHB97, SB98, Sch98] compared the coding performance between two QCIF (see Appendix A) test sequences: *Akiyo*, a head-and-shoulder sequence, and *Silent*, a sign-language sequence using American sign language. Schumeyer's observed that the *Silent* sequence is not as compressible as the *Akiyo* sequence utilizing the H.263 coder. For a fixed frame rate and quantizer, the *Silent* sequence had lower peak signal-to-noise ratio (PSNR) and higher bit-rate. Alternatively, for a fixed bit-rate, the *Silent* sequence had lower PSNR and lower frame rate. Schumeyer concluded that for transmission over low bit-rate channels, sign language video would require content-based coding strategies to achieve the necessary image quality and frame rate for accurate perception. In sign language video, the hands and face are perceptually important, while other regions are less important [IT99, Sch98]. Therefore, to enable content-based manipulation of sign language video, the face and the hands must be extracted from the rest of the frame. Figure 2.8(a) shows frame 221 from the *Silent* sequence. The objects of interest, namely the face and hand objects, are indicated in Figure 2.8(b). Note that parts of the hair and neck may also be included in the face object. Details of the video test sequences used in this thesis are provided in Appendix B.

(a) (b)

Figure 2.8: The objects of interest. (a) Frame 221 of the *Silent* sequence, and (b) the objects of interest.

2.4 Summary

In this chapter, we reviewed and discussed segmentation, video coding, and the different characteristics of sign language. The following list summarizes the main points in this chap-

ter:

- The goal of image segmentation is to partition an image into a set of non-overlapping regions whose union is the entire image. In the case of video segmentation, motion information can be employed in the segmentation process. A region is defined as a set of pixels that is homogeneous with respect to a given quantitative criterion, while objects are characterized by their semantical meaning.

- The goal of video coding is to reduce the bit-rate of a video sequence so that it is feasible to transmit the sequence in real-time over a given communication channel. Compression can be characterized into lossless and lossy compression. Almost all video coding techniques in use today are lossy.

- Block-based video coding schemes suffer from blockiness and unnatural motion. To overcome these problems, content-based coding schemes have been proposed. Content-based coding schemes segment a video frame into different semantic objects, and then compress these objects independently.

- MPEG approved version 1 of the MPEG-4 standard in October 1998 and version 2 in December 1999. MPEG-4 is the first audio-visual representation standard to model a scene as a composition of objects with specific characteristics and behavior.

- Sign language is a visual language used by deaf or hearing-impaired people to communicate. The challenges inherent in the transmission of sign language video include the presence of increased motion and the necessity of smooth motion perception.

- For transmission over low bit-rate channels, sign language video would require content-based coding strategies to achieve the image quality and frame rate required for accurate perception. In sign language video, the face and hands are perceptually important, and must be segmented to enable a meaningful content-based representation of the sequence.

Chapter 3

Color

"Indeed rays, properly expressed, are not colored."

- Sir Isaac Newton

We employ color as a cue to segment the face and the hands. A video signal is a sequence of two dimensional frames projected from a three dimensional scene onto the image plane of a camera. The color at a point in a frame records the emitted or reflected light at a particular point in the dynamic 3D scene. In this chapter, we review the basics of light and color (Section 3.1), and the human visual system (Section 3.2). The trichromatic theory of color mixture is discussed in Section 3.3, and the dichromatic reflection model is discussed in Section 3.4. Some commonly employed color spaces are reviewed in Section 3.5, and the chapter is summarized in Section 3.6.

3.1 Light and Color

Color is the perceptual result of light in the visible spectrum, with wavelengths in the region of 380 nm to 780 nm [NH95]. The *radiant intensity* of light refers to its radiant power (flux) in a particular, specified direction. Formally, radiant intensity is the rate at which energy is transferred, per unit solid angle, and is measured in watts per steradian (Wsr^{-1}). Radiant intensity has a large extent, but imaging systems use pixels with a small area. Thus, it is not appropriate to use radiant intensity as a metric for image data. A more suitable quantity is *radiance*, defined as radiant intensity per unit projected area [Poy95b]. Radiance is measured in watts per steradian per meter squared.

The color of light depends on its wavelength composition. For example, light that has its energy concentrated near 700 nm appears red, and light that has equal energy in the entire visible spectrum appears white. In general, light of a single wavelength (or narrow bandwidth) is referred to as a *spectral color*. On the other hand, white light is referred to as *achromatic*.

There are two types of light sources: *illuminating sources*, which emit electromagnetic waves, and *reflecting sources*, which reflect incident waves. Examples of illuminating light sources are the sun, and television (TV) sets. The perceived color of an illuminating light source depends on the range of wavelengths emitted by the source. Illuminating light follows an additive rule, that is the perceived color of several mixed illuminating sources depends on the sum of the spectra of all light sources. For example, white color is created by combining red, green, and blue lights in appropriate proportions.

As already mentioned, reflecting light sources are those that reflect an incident light, which could itself be reflected. When light hits an object, the energy in a certain wavelength is absorbed, while the rest is reflected. The color of reflected light depends on the spectral content of the incident light and the wavelengths that are absorbed. The perceived color of several mixed reflecting light sources depends on the unabsorbed wavelength range, i.e., reflecting light sources follow a subtractive rule. For example, if the incident light is white, a dye that absorbs the wavelength near 400 nm (blue) would appear yellow. This is further discussed in Section 3.4.

3.2 The Human Visual System

In this section, we will review the different attributes of the human visual system. The anatomy of the eye, color perception, and the opponent-color model are covered.

3.2.1 Anatomy of the Eye

The human eye, shown in Figure 3.1[1], is a complex anatomical device. The eye is essentially an opaque eyeball filled with a water-like fluid. In the front of the eyeball is a transparent opening known as the *cornea*. The cornea has the dual purpose of protecting the eye and refracting light as it enters the eye. After passing through the cornea, a portion of the light passes through an opening known as the *pupil*. The diameter of the pupil is controlled by the *iris*. The iris functions like the aperture of a camera, enlarging in dim light and contracting in bright light. Light that passes through the pupil opening enters the *lens*. The lens is made of a fibrous, jelly-like material with a refractive index of 1.42 to 1.47 [WS82]. The shape of the lens changes to fine-tune vision, a process known as *accommodation*.

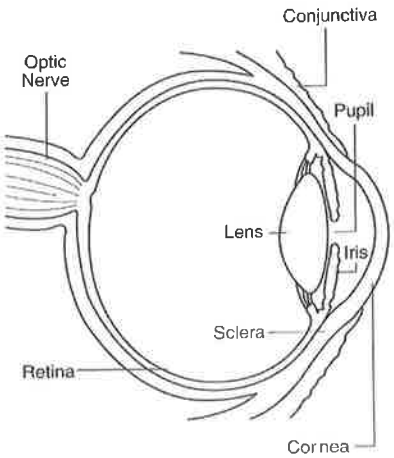


Figure 3.1: The human eye.

The inner surface of the eye is known as the *retina*. The retina consists of receptors sensitive to light called *photoreceptors*. The photoreceptors contain chemical pigments that absorb light and initiate a neural response. There are two types of photoreceptors, *rods* and *cones*. Rods are responsible for low light vision, while cones are responsible for details and

[1] From the National Eye Institute, http://www.nei.nih.gov.

color under normal light conditions (e.g., daylight). The visual information from the retina is passed via optic nerves to an area of the brain called the *visual cortex*. Visual processing and understanding occurs in the visual cortex.

3.2.2 Color Perception

In the previous section, we noted that the retina of the human eye contains photoreceptors called cones that are responsible for color vision. There are three types of cones that have overlapping pass-bands in the visible spectrum, with peaks at blue (near 445 nm), green (near 535 nm), and red (near 570 nm). Each type of cone integrates the energy in the incident light at various wavelengths in proportion to their sensitivity to light to that wavelength. The three resulting number are primarily responsible for color sensation. This is the bases for the *trichromatic theory of color vision* [WS82], first described by Thomas Young [You02]. The theory states that the color of light entering the eye can be specified by only three numbers, rather than a complete function of wavelengths over the visible spectrum.

Color sensation is described by *luminance* and *chrominance*. Luminance is proportional to the light energy emitted per unit projected area in the visible band, and is closely related to the perception of brightness. Moreover, the same amount of light energy produces different sensations of brightness at different wavelengths. This phenomenon is characterized by the *relative luminous efficiency function*, $a_y(\lambda)$. The eye is most sensitive to green, less sensitive to red, and least sensitive to blue light. The luminance (denoted by Y) of any given spectral distribution $C(\lambda)$ is given by:

$$Y = k_m \int C(\lambda) a_y(\lambda) d\lambda, \tag{3.1}$$

where k_m is a constant. Luminance is measured in candelas per meter squared (cdm^{-2}). However, luminance is often normalized to 1 or 100 units with respect to the luminance of a specified or implied white reference. For example, a studio broadcast monitor has a white reference whose luminance is about 100 cdm^{-2}, and $Y = 1$ refers to this value [Poy95b]. Human vision has a nonlinear perceptual response to luminance, e.g., a source having only 18% of a reference luminance appears about half as bright. The perceptual response to luminance is called *lightness* (denoted by L^*), and is defined by the Commission Internationale

de l'Eclairage (CIE) as [MPFL96]:

$$L^* = \begin{cases} 116(Y/Y_n)^{\frac{1}{3}} - 16, & \text{if } Y/Y_n > 0.008856 \\ 903.3(Y/Y_n), & \text{otherwise} \end{cases} \tag{3.2}$$

where Y_n is the white reference luminance. The relationship between lightness and luminance is plotted in Figure 3.2.

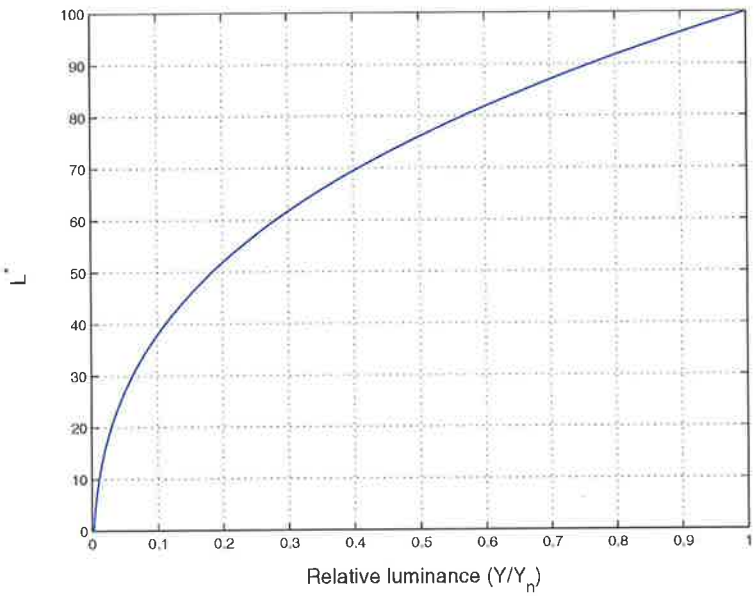


Figure 3.2: The relationship between perceived brightness (i.e., lightness) and luminance.

Chrominance is related to the perception of color *hue* and *saturation*. Hue describes the color tone, and depends on the peak wavelength of the light. On the other hand, saturation specifies how pure the color is, and depends on the bandwidth of the light spectrum. Figure 3.3[2] shows a perceptual representation of the color space. Brightness varies along the vertical space, hue varies along the circumference, and saturation varies along the radial distance. For a fixed brightness, the symbols R, G, and B show the relative locations of red, green, and blue spectral colors, respectively.

[2] Adapted from [Jai89], Figure 3.10.

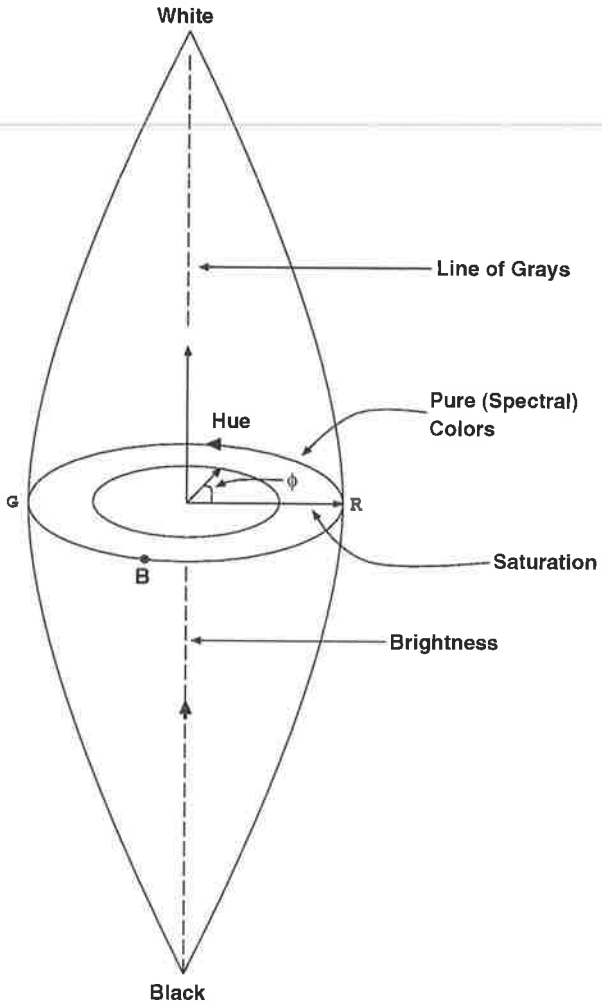


Figure 3.3: Perceptual representation of the color space.

3.2.3 The Opponent-Color Model of Chromatic Vision

A secondary stage in the human visual system converts the three color values obtained by the cones into values that are proportional to luminance and chrominance. This is known as the *opponent-color model* of chromatic vision [NH95, MPFL96]. The opponent-color model hypothesizes the interconnectivity represented in Figure 3.4[3]. The large numbers of interconnections in the visual system are modeled as simple functional blocks with two basic types of synaptic action, excitation (indicated by the add sign) and inhibition (indicated by the minus sign). According to the opponent-color model, the light absorbed by the three types of cones first undergoes a logarithmic transformation. Then, by sums and differences, three opponent

[3] Adapted from [MPFL96], Figure 4.6.

systems are developed, blue-yellow, red-green, and black-white. The summation of red and green provide the luminance channel and the other two are chrominance channels.

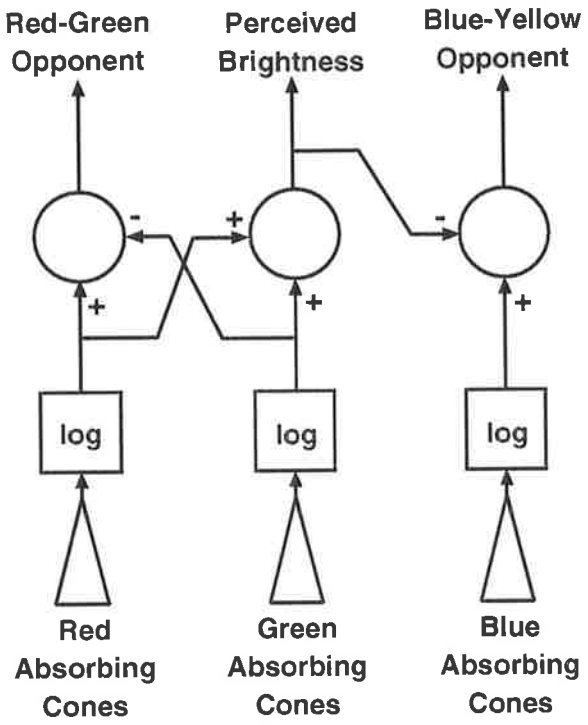


Figure 3.4: The opponent color model.

3.3 The Trichromatic Theory of Color Mixture

The *trichromacy theory of color mixture* is the counterpart of the trichromacy of vision. The theory was first demonstrated by Maxwell in 1855 [NH95], and states that light of any color can be synthesized by an appropriate mixture of three properly chosen primary colors. Let C_i, $i = 1, 2, 3$ represent the colors of the three primary color sources, and C a given color. Then the theory can be stated as:

$$C = \sum_{i=1}^{3} T_i C_i, \tag{3.3}$$

where the T_i's are the amounts of the three primary colors required to match color C. The T_i's are called the *tristimulus values*. In general, some of the tristimulus values can be negative. The most popular primary set for illuminating lights is the *RGB primary*. The *CIE*

RGB primary system consists of colors at 700 nm (R), 546.1 nm (G), and 435.8 nm (B) [WOZ01].

For a given primary set, the tristimulus values for any color can be evaluated by determining the *color matching functions*, $m_i(\lambda)$, for primary colors C_i, $i = 1, 2, 3$. These functions describe the tristimulus values of spectral colors with unit intensity. The tristimulus values for any color with spectrum $C(\lambda)$ is determined by:

$$T_i = \int C(\lambda) m_i(\lambda) d\lambda, \; i = 1, 2, 3. \tag{3.4}$$

By convention, tristimulus values are expressed in normalized form for a reference white color (equal energy in all wavelengths) with unit energy.

The tristimulus representation described above mixes the luminance and chrominance attributes of color. The chrominance information (i.e., hue and saturation) of light can be measured by employing normalized quantities called *chromaticity coordinates*, defined as:

$$c_i = \frac{T_i}{T_1 + T_2 + T_3}, \; i = 1, 2, 3. \tag{3.5}$$

Note that $\sum_{i=1}^{3} c_i = 1$, therefore two chromaticity coordinates are sufficient to specify the chrominance of a color.

3.4 The Dichromatic Reflection Model

The light reflected from an object $L(\psi, \lambda)$ is determined by its reflectance and the light it is exposed with (i.e., the incident light). The reflected light is a function of λ, and the photometric angles ψ, including the viewing angle, the phase angle, and the illumination direction angle. For dielectric non-homogeneous materials this is often modeled by the dichromatic reflection model, which described the reflected light $L(\psi, \lambda)$ as an additive mixture of the light L_S reflected at the material's surface (*interface* or *surface reflection*) and the light L_B reflected from the material's body (*body*, *diffuse*, or *matt reflection*) [SAG01]:

$$L(\psi, \lambda) = m_S(\psi) L_S(\lambda) + m_B(\psi) L_B(\lambda), \tag{3.6}$$

where $m_S(\psi)$ and $m_B(\psi)$ are geometrical scaling factors for the surface and body reflections, respectively. For materials with a high oil or water content, the reflected light at the surface

has approximately the same spectral power distribution as the light that it is exposed with, i.e., it has the same color as the illuminant.

The light that is not reflected at the surface penetrates into the material body where it is scattered and selectively absorbed at wavelengths that are characteristic of the material. A fraction of this light arrives back at the surface and exits the material. The body reflection provides the characteristic color of the material.

The reflected lights, L_S and L_B, from the surface and the body, are a product of the incident light spectrum E and the spectral surface reflectance ρ_S and body reflectance ρ_B of the material, respectively [SAG01]:

$$L_S(\lambda) = E(\lambda)\rho_S(\lambda), \tag{3.7}$$

and

$$L_B(\lambda) = E(\lambda)\rho_B(\lambda). \tag{3.8}$$

A non-homogeneous dielectric material, depicting the photometric angles and reflection components, is shown in Figure 3.5[4].

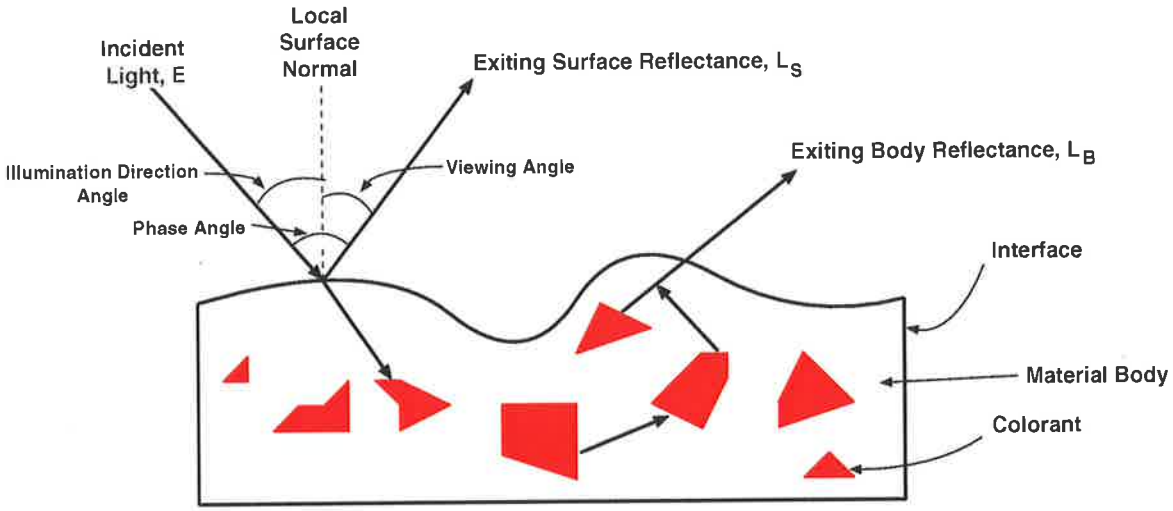


Figure 3.5: Photometric angles and reflection components from a non-homogeneous material.

[4] Adapted from [SAG01]

3.5 Color Spaces

The purpose of a color space is to facilitate the specification of colors in some standard, generally accepted way [GW92]. Most color spaces are oriented either towards hardware (e.g., monitors and printers) or towards applications where color manipulation is a goal (e.g., color graphics for animation). In the following sections we will review four commonly employed color spaces. A more comprehensive treatment of color science and color spaces is given in [Poy96].

3.5.1 CIE XYZ Color Space

To facilitate cooperation among scientists, CIE has defined a standard observer and the XYZ primary, in which the Y tristimulus value directly measures luminance, normalized to equal energy white. The tristimulus values and the color matching functions in the CIE XYZ primary are non-negative (Figure 3.6), which is a very desirable feature. The problem with the CIE XYZ primary is that the X, Y, and Z colors are not realizable by actual color stimuli. Therefore, the CIE XYZ primary is not used directly for color production, rather it is employed to define other primaries and for the numerical specification of colors [WOZ01].

Following from (3.5), the chromaticity coordinates for CIE XYZ are:

$$x = \frac{X}{X+Y+Z}, \tag{3.9}$$

and

$$y = \frac{Y}{X+Y+Z}. \tag{3.10}$$

A color can be specified by the two chromaticity coordinates x, y and luminance. Figure 3.7 shows the chromaticity diagram for this system.

RGB values in a particular set of primaries can be transformed to and from CIE XYZ by a three-by-three matrix transform. To transform from XYZ to ITU-R recommendation BT.709-4 RGB[5] (with D_{65} white point), the following matrix transform is employed

[5]The chromaticity coordinates for the RGB primaries and the D_{65} white point are specified in ITU-R recommendation BT.709-4 [IR00].

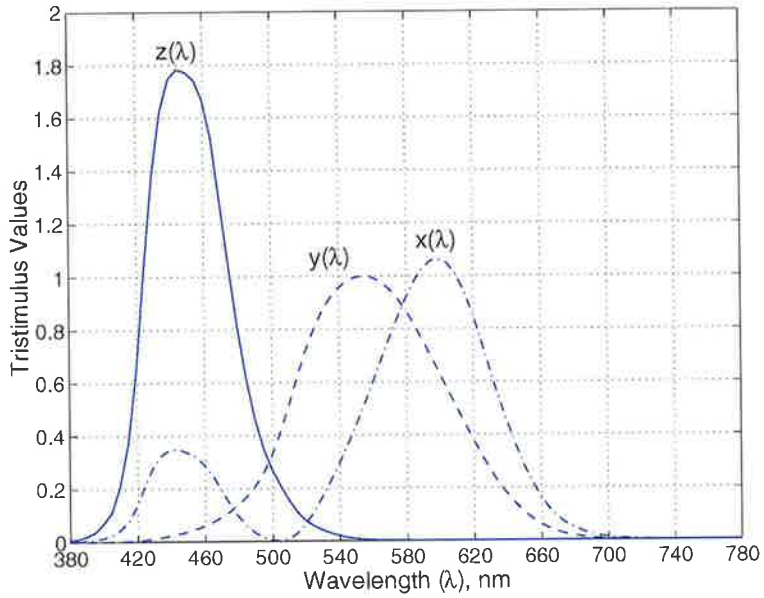


Figure 3.6: Color matching functions $x(\lambda)$, $y(\lambda)$, and $z(\lambda)$ of the CIE 1931 standard colorimetric system.

[Poy95a]:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.240 & -1.537 & -0.499 \\ -0.969 & 1.876 & 0.042 \\ 0.056 & -0.204 & 1.057 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{3.11}$$

The range for valid R, G, B values is [0,1]. Note, the above matrix has negative coefficients. Consequently, some XYZ colors may be transformed to RGB values that are negative or greater than one. This means that not all visible colors can be produced using the RGB system. The inverse transform is:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412 & 0.358 & 0.180 \\ 0.213 & 0.715 & 0.072 \\ 0.019 & 0.119 & 0.950 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{3.12}$$

3.5.2 YUV Color Space

It is often desirable to define color in terms of its luminance and chrominance components. This will enable a more efficient processing and transmission of color signals. Various three-component color spaces have been developed, in which one component reflects the lumi-

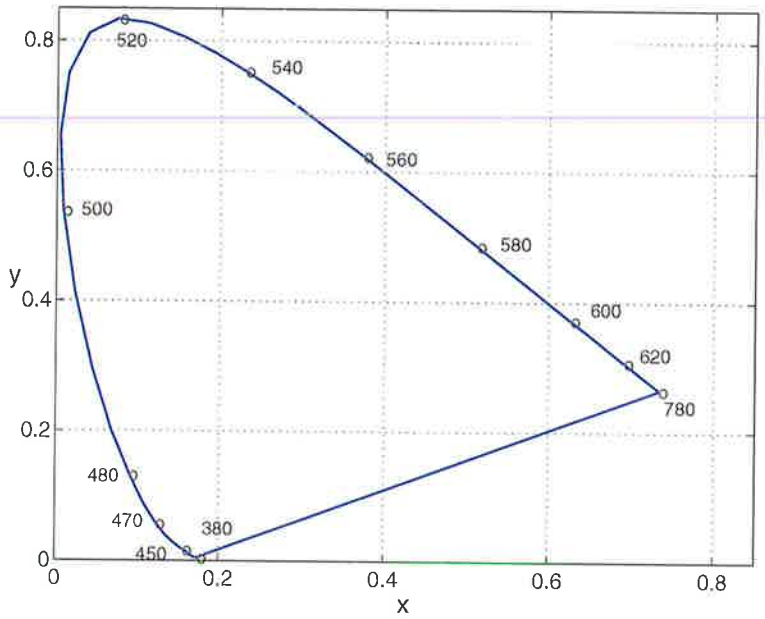


Figure 3.7: Chromaticity diagram for the CIE XYZ color space.

nance and the other two collectively characterize hue and saturation. YUV is one such color space.

The YUV color space is used in the PAL (Phase Alternation by Line) TV system. The PAL system is used mainly in western Europe, most of Asia (including Australia), and the Middle East. The YUV space is related to the PAL RGB primary values by [NH95]:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}, \tag{3.13}$$

and

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1.000 & 0.000 & 1.140 \\ 1.000 & -0.395 & -0.581 \\ 1.000 & 2.032 & 0.001 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix}, \tag{3.14}$$

where R', G', B' are normalized gamma-corrected values, so that $(R', G', B') = (1, 1, 1)$ corresponds to the reference white defined in the PAL system. Gamma correction is performed because most display devices suffer from a non-linear relationship between the input voltage signals and the displayed color intensity.[6]

[6]Gamma-correction also approximates the lightness response of vision using $R'G'B'$ signals that are each

Since Y is derived from gamma-corrected R', G', B' values, it is referred to as *luma* [Poy01]. However, in common with most literature on video compression and segmentation, we will use the term luminance instead of luma. The two chrominance components, U and V, are proportional to color differences, $B - Y$ and $R - Y$, scaled to have a desired range.

3.5.3 YIQ Color Space

The NTSC (National Television Standards Committee) TV system (used mainly in North America and Japan) uses the YIQ color space, where the I and Q components are the rotated versions (by 33°) of the U and V components. As a result of the rotation, the I component corresponds to colors in the orange-to-cyan range, and the Q component corresponds to colors in the green-to-purple. The human visual system is less sensitive to changes in the green-to-purple range than it is to changes in the yellow-to-cyan range. Therefore, the Q component can be transmitted with less bandwidth than the I component [WOZ01]. The YIQ values are related to the NTSC RGB values by:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & -0.311 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}, \tag{3.15}$$

and

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.620 \\ 1.000 & -0.271 & -0.647 \\ 1.000 & -1.108 & 1.700 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}. \tag{3.16}$$

In the YIQ color space, $\tan^{-1}(Q/I)$ approximates the hue, and $\sqrt{I^2 + Q^2}/Y$ reflects the saturation.

3.5.4 YCbCr Color Space

Due to an increasing demand for digital approaches in image and video processing, the ITU-R BT.601 [IR98] recommendation defined the YCbCr color space. The Y, Cb, and Cr components are scaled and shifted versions of the analog Y, U, and V components,

subject to approximately a 0.5-power function. This is comparable to the 1/3-power function defined by L^* in (3.2).

respectively, where the scaling and shifting operation results in the components having a range of (0,255). Assuming that the RGB (Red, Green, Blue) values are in the range of 0 to 255, the YCbCr values can be derived from the RGB values by:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{bmatrix} \begin{bmatrix} R_d \\ G_d \\ B_d \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}. \tag{3.17}$$

The inverse relation is:

$$\begin{bmatrix} R_d \\ G_d \\ B_d \end{bmatrix} = \begin{bmatrix} 1.164 & 0.0 & 1.596 \\ 1.164 & -0.392 & -0.813 \\ 1.164 & 2.017 & 0.000 \end{bmatrix} \begin{bmatrix} Y - 16 \\ Cb - 128 \\ Cr - 128 \end{bmatrix}. \tag{3.18}$$

In the above relations, $R_d = 255R'$, $G_d = 255G'$, $B_d = 255B'$ are the digital equivalents of the normalized, gamma-corrected RGB primaries. In the YCbCr color space, Y reflects the luminance (actually luma) and is scaled to a range of 16 to 235. This places black at level 16 and white at level 235. In doing so, it reserves the extremes of the range for signal processing footroom and headroom. The chrominance components, Cb and Cr, are scaled versions of color differences $B - Y$ and $R - Y$, respectively. Cb and Cr have a range of 16 to 240, inclusive.

In Chapter 5, we show that the YCbCr color space provides an effective use of chrominance information for modeling the human skin-color. This, coupled with the fact that the YCbCr color space is employed in digital video (and therefore digital video coding) was the main motivation for choosing this color space in this thesis.

3.6 Summary

The first cue that we employ to segment the face and the hands in sign language video sequences is color. In this chapter, we reviewed the basics of light and color, the human visual system, the trichromatic theory of color mixture, and the dichromatic reflection model. The CIE XYZ, YUV, YIQ, and YCbCr color spaces were also reviewed. We summarize the important points below:

- Color is the perceptual result of light in the visible spectrum, with wavelengths in the region of 380 nm to 780 nm. The color of light depends on its wavelength composition.
- Any color can be created by mixing three primary colors. This is know as the trichromacy theory of color mixture. The most common primary set includes red, green, and blue colors.
- The human eye perceives color by photoreceptors (cones) in the retina that are sensitive to red, green, and blue wavelengths. The color sensation can be described by three attributes, namely luminance (i.e., brightness), hue (color tone), and saturation (purity). The eye is most sensitive to luminance, followed by hue, and then to saturation.
- A secondary stage in the human visual system converts the three color values obtained by the cones into values that are proportional to luminance and chrominance. This is known as the opponent-color model of chromatic vision.
- Color can be specified by three numbers. These numbers either correspond to the contributions of the three primary colors (i.e., tristimulus values), or a luminance and two chrominance values.
- The dichromatic reflection model describes reflected light as an additive mixture of the light reflected from the material's surface, and the light reflected from the material's body.
- CIE has defined a standard observer and the XYZ primary color space. The Y tristimulus value directly measures luminance, and is normalized to equal energy white. The tristimulus values and the color matching functions in the CIE XYZ primary are non-negative.
- The YUV color space is used in the PAL TV system, while the YIQ color space is used in the NTSC TV system.
- The YCbCr color space is defined in the ITU-R BT.601 recommendation. In this thesis, we employ the YCbCr color space for skin-color segmentation. The YCbCr color space is considered since it is employed in digital video and is effective for modeling the human skin-color.

Chapter 4

Motion

"Never mistake motion for action."

- Ernest Hemingway

Motion is the second cue that we employ to segment the face and the hands. In this chapter, we summarize the main attributes of motion. In order to relate changes in the real world to temporal changes in a video sequence, we need parametric models to describe the real world and the image generation process. Using the parametric models and their estimated parameters, we can reconstruct a model that is an approximation of the real world.

In the following sections, we describe the camera model (Section 4.1), three-dimensional motion (Section 4.2.1), and two-dimensional motion (Section 4.2.2). The scene model is discussed in Section 4.3, and we distinguish between 2D motion and apparent motion in Section 4.4. The chapter is summarized in Section 4.5.

4.1 Camera Models

To understand how vision can be modeled computationally and replicated on a computer, we need to understand the image acquisition process. The role of the camera is analogous to that of the eye (Section 3.2.1). The camera model describes the projection of real objects onto the imaging plane of the camera.

The *pin-hole camera* is a widely used approximation of the camera function, shown in Figure 4.1. In Figure 4.1, F denotes the *focal length*, and **C** the *focal center*. The projected position **p** of a 3D point **P** is the intersection of the line connecting **P** and **C** with the imaging plane. The image position is reversed from its true 3D position, since the imaging plane is behind the focal center. Usually, reversed image positions are avoided by placing the image plane and the object on the same side of the focal center. This scenario is shown in Figure 4.2.

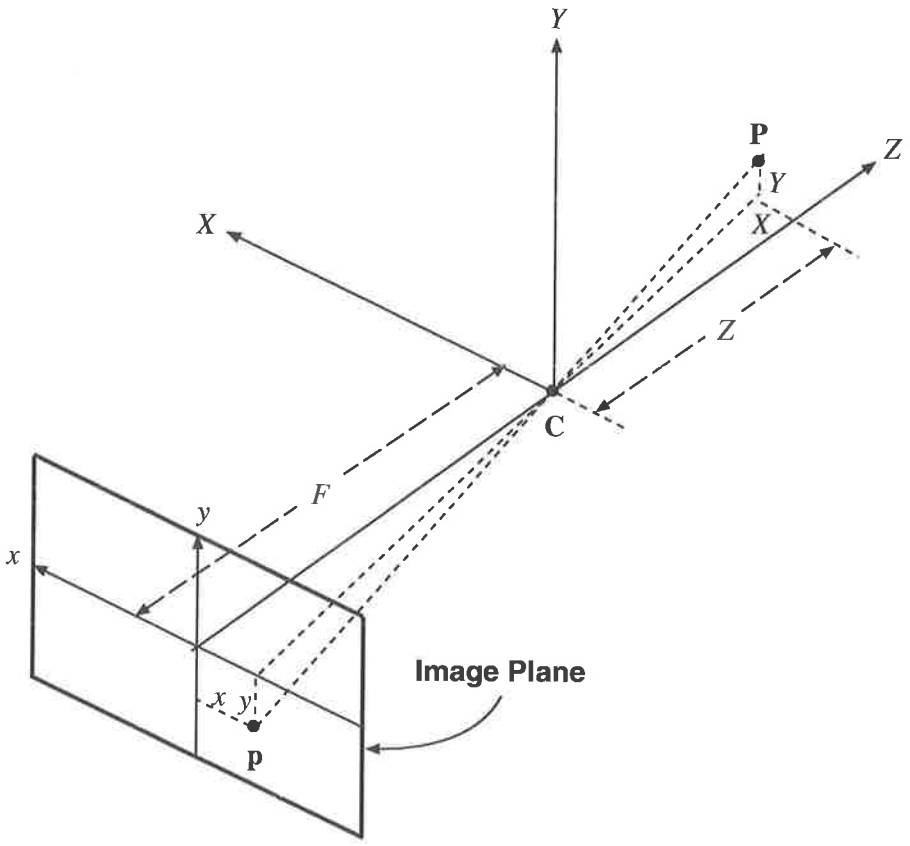


Figure 4.1: Perspective projection by a pin-hole camera. The image plane is behind the focal center.

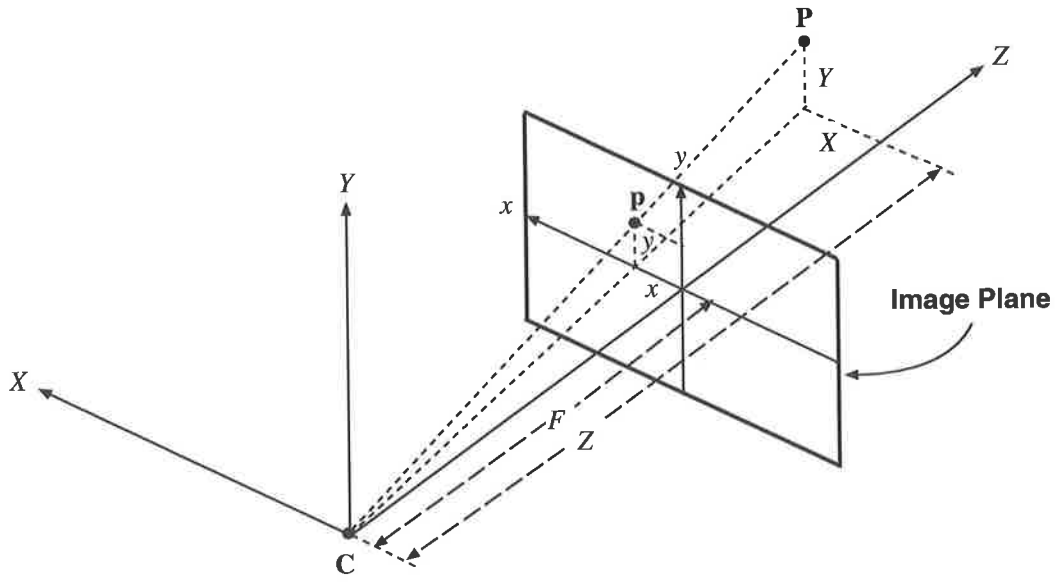


Figure 4.2: Perspective projection by a pin-hole camera. The image plane and the object are on the same side of the focal center.

It is assumed (cf. Figure 4.2) that the origin of the 3D world coordinate is located at the focal center, its XY-plane is parallel to the imaging plane, and the scene coordinate (X, Y, Z) follows the right-hand rule, with the positive direction of the Z-axis being in the imaging direction [WOZ01]. Also, it is assumed that the imaging plane uses the same distance unit as the 3D coordinate. From Figure 4.2, we have:

$$\begin{aligned} \frac{x}{F} &= \frac{X}{Z}, \\ \frac{y}{F} &= \frac{Y}{Z}, \end{aligned} \tag{4.1}$$

or

$$\begin{aligned} x &= F\frac{X}{Z}, \\ y &= F\frac{Y}{Z}. \end{aligned} \tag{4.2}$$

The relation (4.2) is known as *perspective projection*. Note that there is an inverse relation between the projected x and y values and Z. Therefore, the image of an object is smaller if it is farther away from the camera.

The other type of projection is the *orthographic* or *parallel projection*. Orthographic projection, depicted in Figure 4.3, is an approximation of perspective projection where it is assumed that all the rays from the 3D object to the image plane travel in parallel [Tek95]. Provided that the image plane is parallel to the XY plane of the world coordinate system, orthographic projection can be described in Cartesian coordinates as:

$$x = X, \; y = Y. \tag{4.3}$$

The distance of the object from the camera does not affect the image plane intensity distribution in orthographic projection. That is, the object will always yield the same image no matter how far it is from the camera. Orthographic projection provides a reasonable approximation to the actual image formation process if the distance between the objects and the camera is large compared to the depth of the object [MN98b]. The relation (4.3) is much simpler than (4.2) and greatly simplifies 3D and 2D transformation.

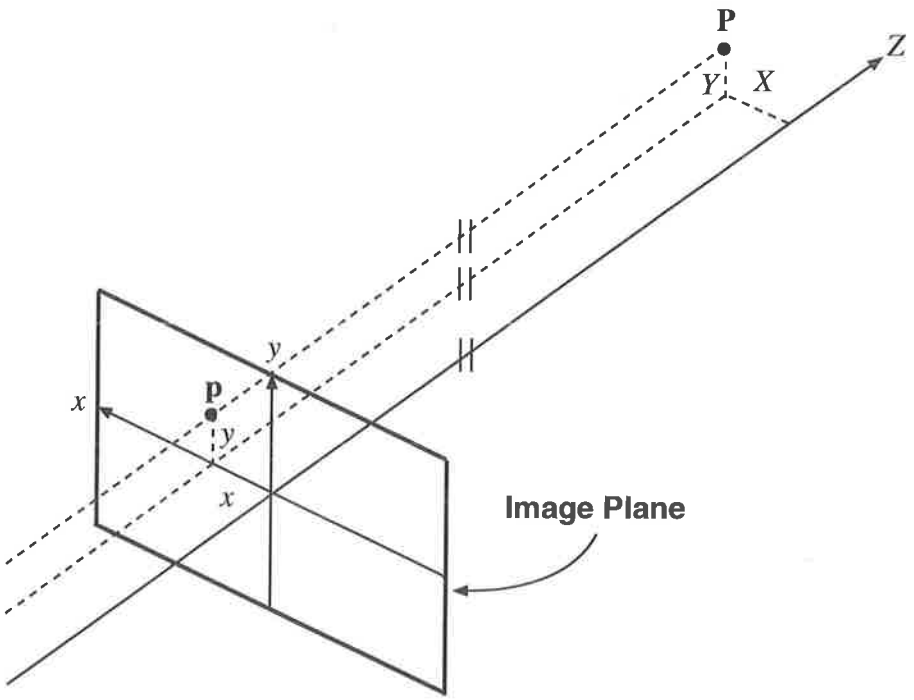


Figure 4.3: Orthographic projection as an approximation of a pinhole camera.

Having discussed the camera model, it must be pointed out that the pin-hole camera and its perspective projection are only an approximation of real cameras. For example, it does not consider misalignment of the camera axis and the image center, the lowpass filter effect

of the finite size aperture of a real lens, the finite exposure time, and other distortions of the lens [WOZ01].

4.2 Motion Models

This section introduces the motion models used in describing the assumptions that we make about the real world. We will look at both 3D motion and 2D motion models.

4.2.1 Three-Dimensional Motion

The 3D motion of a rigid object can be described in terms of a translation vector $\mathbf{T} = (T_x, T_y, T_z)$ and a rotation matrix $[\mathbf{R}]$. The translation vector $\mathbf{T}$ describes a displacement of a point from $\mathbf{P}$ to $\mathbf{P}'$ by T_x, T_y, T_z in the direction of the coordinate axes X, Y, Z, respectively:

$$\mathbf{P}' = \mathbf{P} + \mathbf{T}. \tag{4.4}$$

Equation (4.4) holds for all points of a translated object.

The rotation matrix $[\mathbf{R}]$ describes the rotation of the points of an object around the origin of the 3D space, i.e.,

$$[\mathbf{R}] = [\mathbf{R}_x] \cdot [\mathbf{R}_y] \cdot [\mathbf{R}_z]. \tag{4.5}$$

Equation (4.5) rotates a point in 3D space around the axes X, Y, and Z in this order. The rotation matrix is computed from the rotation matrices that rotate a point just around one axis. The individual rotation matrices are:

$$[\mathbf{R}_x] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_x & -\sin\phi_x \\ 0 & \sin\phi_x & \cos\phi_x \end{bmatrix}, \tag{4.6}$$

$$[\mathbf{R}_y] = \begin{bmatrix} \cos\phi_y & 0 & \sin\phi_y \\ 0 & 1 & 0 \\ -\sin\phi_y & 0 & \cos\phi_y \end{bmatrix}, \tag{4.7}$$

and

$$[\mathbf{R}_z] = \begin{bmatrix} \cos\phi_z & -\sin\phi_z & 0 \\ \sin\phi_z & \cos\phi_z & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.8}$$

where ϕ_x, ϕ_y, and ϕ_z are the rotation angles with respect to each axis. Therefore, the rotation matrix is:

$$[\mathbf{R}] = \begin{bmatrix} \cos\phi_y\cos\phi_z & \sin\phi_x\sin\phi_y\cos\phi_z - \cos\phi_x\sin\phi_z & \cos\phi_x\sin\phi_y\cos\phi_z + \sin\phi_x\sin\phi_z \\ \cos\phi_y\sin\phi_z & \sin\phi_x\sin\phi_y\sin\phi_z + \cos\phi_x\cos\phi_z & \cos\phi_x\sin\phi_y\sin\phi_z - \sin\phi_x\cos\phi_z \\ -\sin\phi_y & \sin\phi_x\cos\phi_y & \cos\phi_x\cos\phi_y \end{bmatrix}. \tag{4.9}$$

Since $[\mathbf{R}]$ is an orthonormal matrix, it satisfies:

$$[\mathbf{R}]^T = [\mathbf{R}]^{-1}, \tag{4.10}$$

and

$$\det[\mathbf{R}] = \pm 1. \tag{4.11}$$

The motion of a point on the object surface from $\mathbf{P}$ to $\mathbf{P}'$ can be expressed as:

$$\mathbf{P}' = [\mathbf{R}] \cdot \mathbf{P} + \mathbf{T}. \tag{4.12}$$

For many motion estimation algorithms, the non-linear rotation matrix according to (4.5) has to be linearized with respect to the rotation angles [WOZ01]. Assuming small rotation angles such that $\cos\phi \approx 1$ and $\sin\phi \approx \phi$, the rotation matrix can be simplified to:

$$[\mathbf{R}] \approx [\mathbf{R}'] = \begin{bmatrix} 1 & -\phi_z & \phi_y \\ \phi_z & 1 & -\phi_x \\ -\phi_y & \phi_x & 1 \end{bmatrix}. \tag{4.13}$$

Equation (4.12) rotates the point $\mathbf{P}$ on the object surface around the center of the world coordinate system. If the object is far away from the center of the coordinate system and is just rotating around its own center, a local coordinate system for each object can be defined. Rotation and translation are then defined with respect to the object's own center $\mathbf{C} = (C_x, C_y, C_z)^T$, which is also the center of the local coordinate system. The 3D motion equation can now be expressed as:

$$\mathbf{P}' = [\mathbf{R}] \cdot (\mathbf{P} - \mathbf{C}) + \mathbf{C} + \mathbf{T}. \tag{4.14}$$

If an object is non-rigid, it can be described by decomposing it into two or more rigid components. Each component would then have its own set of motion parameters according to (4.14).

4.2.2 Two-Dimensional Motion

Object or camera motion in 3D space leads to *2D* or *projected motion*. When an object moves from $\mathbf{P} = [X\ Y\ Z]^T$ at time t_1 to $\mathbf{P}' = [X'\ Y'\ Z']^T = [X + V_X\ \ Y + V_Y\ \ Z + V_Z]^T$ at time $t_2 = t_1 + v_t$, its projected image changes from $\mathbf{p} = [x\ y]^T$ to $\mathbf{p}' = [x'\ y']^T = [x + v_x\ \ y + v_y]^T$. This is shown in Figure 4.4. Using the notations in [WOZ01], the 3D displacement, $\mathbf{V}(\mathbf{P}; t_1, t_2) = \mathbf{P}' - \mathbf{P} = [V_X\ V_Y\ V_Z]^T$, is called the *3D motion vector* at $\mathbf{P}$. Likewise, the 2D displacement, $\mathbf{v}(\mathbf{p}; t_1, t_2) = \mathbf{p}' - \mathbf{p} = [v_x\ v_y]^T$, is called the *2D motion vector* at $\mathbf{p}$. Motion vectors (MVs) are in general position-dependent. The *2D motion field* from t_1 to t_2 is represented by $\mathbf{v}(\mathbf{p}; t_1, t_2)$.

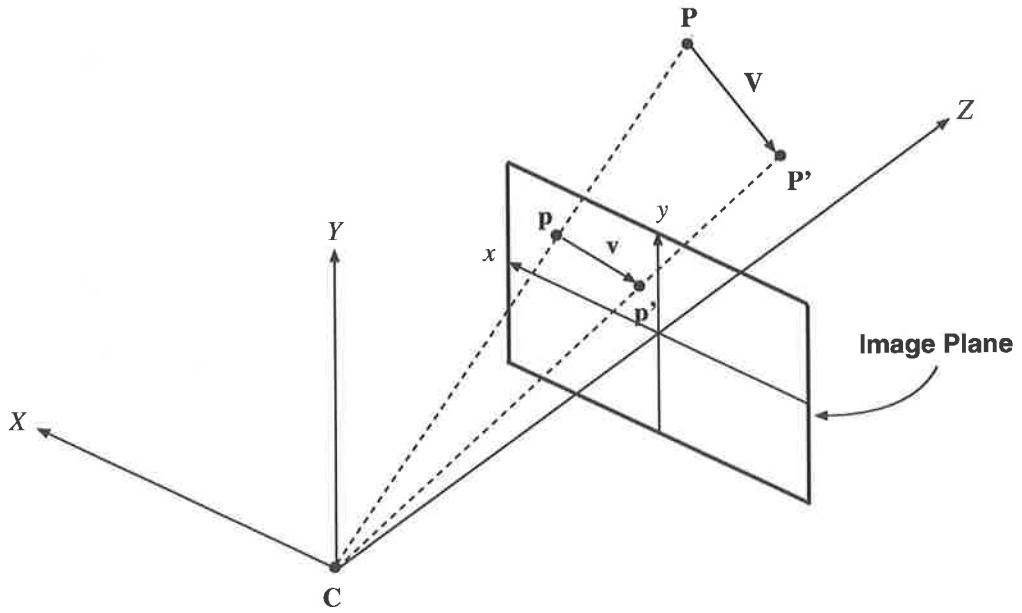


Figure 4.4: Projection of a moving object.

It is sometimes more convenient to specify for each point $\mathbf{p}$ at time t_1, its corresponding position at t_2, $\mathbf{w}(\mathbf{p}; t_1, t_2) = \mathbf{p}'$. The *mapping function* is represented by $\mathbf{w}(\mathbf{p}; t_1, t_2)$. Note that the mapping function is uniquely related to the motion field by $\mathbf{w}(\mathbf{p}; t_1, t_2) = \mathbf{p} + \mathbf{v}(\mathbf{p}; t_1, t_2)$.

This thesis deals with digital video signals that have a finite and discrete image domain, described by a truncated lattice, Λ. The notation $\mathbf{x} = [x\ y]^T \in \Lambda$ represents a pixel index. Furthermore, we assume that the time interval $v_t = t_2 - t_1$ is either equal to the temporal

sampling interval or an integer multiple of this interval. The motion field for a given time interval is a finite set of 2D vectors in a 2D array, in the same order as the pixels.

Velocity vectors can also be used to characterize motion. The velocity vectors, or *flow vectors*, are defined as $\mathbf{f} = \frac{\partial \mathbf{v}}{\partial t} = \left[\frac{\partial v_x}{\partial t} \; \frac{\partial v_y}{\partial t}\right]^T$. If v_t is small, the motion within the interval can be assumed to be constant, i.e. $\mathbf{f} = \mathbf{v}/v_t$. Similar to motion fields, the *flow field* can be defined over the entire image domain as $\mathbf{f}(\mathbf{p}; t_1, t_2), \mathbf{p} \in \Lambda$.

Let us now consider the 2D motion induced by an arbitrary 3D rigid motion. Without loss of generality, we will assume that the object is undergoing rigid motion and the camera is stationary. As mentioned in Section 4.2.1, the 3D motion of an object can be described by:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 \\ s_7 & s_8 & s_9 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}. \tag{4.15}$$

The rotation matrix can be completely determined by three rotation angles. As well as the three translation parameters, there are six parameters in total.

The perspective displacement field can be derived by substituting (4.2) into (4.15) to obtain the relation between the image coordinates before and after motion:

$$\begin{aligned} x' &= F\frac{(s_1x + s_2y + s_3F)Z + T_xF}{(s_7 + s_8y + s_9F)Z + T_zF}, \\ y' &= F\frac{(s_4x + s_5y + s_6F)Z + T_yF}{(s_7x + s_8y + s_9F)Z + T_zF}. \end{aligned} \tag{4.16}$$

If the translational parameters T_x, T_y, T_z and the depth Z are scaled by the same factor, the correspondence between (x, y) and (x', y') will not change. This indicates that based on the correspondence of image coordinates, the parameters T_x, T_y, T_z are unique only up to a scaling factor.

On the other hand, the orthographic displacement field can be derived by substituting (4.3) into (4.15) to obtain:

$$\begin{aligned} x' &= s_1x + s_2y + (s_3Z + T_x), \\ y' &= s_4x + s_5y + (s_6Z + T_y). \end{aligned} \tag{4.17}$$

The model (4.17) is an affine mapping of the pixel (x, y) at time t_1 to the pixel (x', y') at time t_2 defined in terms of the six parameters $s_1, s_2, (s_3Z + T_x), s_4, s_5$, and $(s_6Z + T_y)$.

It is common to model 3D objects by (piecewise) planar patches whose points line on a plane described by [MN98b]:

$$aX + bY + cZ = 1, \tag{4.18}$$

where $[a\,b\,c]^T$ denotes the normal vector of this plane. The 3D displacement model (4.15) can be rewritten as [Tek95]:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \mathbf{R} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{T}[a\,b\,c] \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{4.19}$$

or

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \mathbf{A} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{4.20}$$

where

$$\mathbf{A} = \mathbf{R} + \mathbf{T}[a\,b\,c]. \tag{4.21}$$

If we map the 3D displacement onto the 2D image plane using perspective geometry and normalize $a_9 = 1$ due to the well known scale ambiguity, we obtain the image plane mapping from t_1 to t_2 given by the eight-parameter model [Tek95]:

$$\begin{aligned} x' &= \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1}, \\ y' &= \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1}. \end{aligned} \tag{4.22}$$

If the imaging geometry is approximated by an orthographic projection, a planar patch undergoing 3D rigid motion can be described by an affine model, expressed as:

$$\begin{aligned} x' &= a_1 x + a_2 y + a_3, \\ y' &= a_4 x + a_5 y + a_6. \end{aligned} \tag{4.23}$$

Both the eight-parameter (4.22) and affine (4.23) models are very popular, however many other transformations exist depending on the assumptions made [MN98b].

4.3 Scene Model

Having discussed motion models, we are now ready to consider the modeling of an image scene. If the objects in a sequence are in motion, we distinguish between four frame areas: *unchanged (stationary) background*, *moving object*, *uncovered background*, and *covered or occluded background*.

Figure 4.5 shows two frames with a moving object. Comparing frames k and $k+1$, we can distinguish between the *changed regions* and *unchanged regions*. The unchanged regions show the stationary background in both frames. The moving object is part of the changed regions in frames k and $k+1$. In frame k, the changed region is defined as the area of the moving object and the *background to be covered* region in frame $k+1$ due to object motion. In frame $k+1$, the changed region is defined as the area of the moving object and the *uncovered background* region that was not visible in frame k. Note that in real video sequences, motion vectors are not always defined everywhere. For example, motion vectors are not defined over uncovered regions.

4.4 2D Motion Versus Apparent Motion

Motion is perceived by identifying corresponding points at different times. The correspondence is usually determined by assuming that the color or brightness of a point does not change after the motion. Under certain circumstances, the observed 2D motion can be different from the actual projected 2D motion. Figure 4.6 illustrated one such example. A sphere with a uniform flat surface is rotating under a constant ambient light. Since every point on the sphere reflects the same illumination, the eye cannot perceive any change in the color pattern of the imaged sphere and therefore considers the sphere as being stationary.

The actual projected 2D motion is referred to as 2D motion, while the observed 2D motion is referred to as *apparent motion* [Tek95]. For example, the apparent motion of the sphere in Figure 4.6 is zero. In computer vision literature, apparent motion is referred to as *optical flow*. Moving objects must contain sufficient texture to generate optical flow, because the luminance in the interior of moving objects with uniform intensity remains constant. For further information on optical flow and the derivation of the optical flow equation, the reader

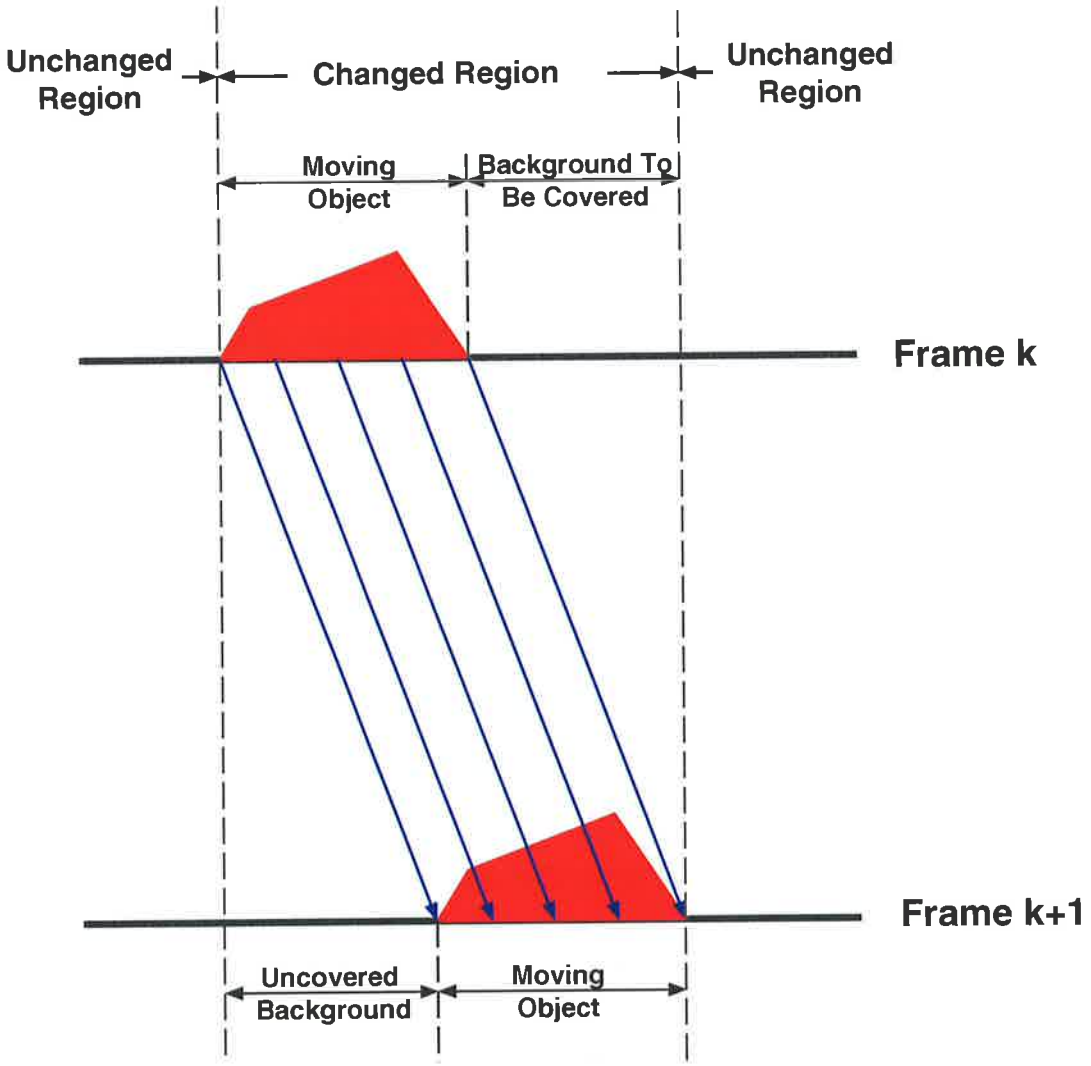


Figure 4.5: The separation of changed regions into moving objects, uncovered background, and background to be covered.

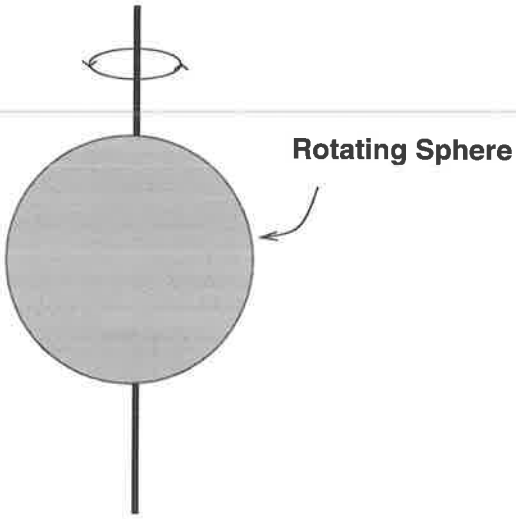


Figure 4.6: A sphere rotating under constant ambient illumination.

is referred to [LK81, HS81, Tek95].

4.5 Summary

In this chapter, we reviewed the different attributes of motion as they relate to video sequences. Motion is the second cue we employ to segment the face and the hands in sign language video sequences. The following list summarizes the main points in this chapter:

- The camera model describes the projection of the 3D world on to the image plane of a camera. Depending on the application, camera models of different complexity can be used. If the objects in the 3D world are far away from the camera, a simple camera model with orthographic projection can be employed. On the other hand, perspective projection enables us to describe the change of object size in an image sequence as an object changes its distance from the camera.

- The 3D motion of an object can be expressed by means of a 3D translation vector and a three-by-three rotation matrix. The rotation matrix is computed from the rotation angles around the three coordinate axes.

- Object or camera motion in 3D space leads to 2D motion. If perspective geometry is assumed and a 3D object is modeled by planar patches, the relation between the image

coordinates before and after motion can be described by the eight-parameter model. If the imaging geometry is approximated by an orthographic projection, a planar patch undergoing 3D rigid motion can be described by an affine model.

- In the scene model, four different frame regions can be distinguished: unchanged (static) background, moving object, uncovered background, and covered or occluded background.

- The projection of 3D motion onto the image plane is referred to as 2D motion. Apparent motion is what the human visual system perceives as motion.

Chapter 5

Skin-Color Segmentation

"Everything should be as simple as possible, but not simpler."

- Albert Einstein

This chapter presents our skin-color segmentation algorithm. The YCbCr color space is employed in digital video. It also provides an effective use of chrominance information for modeling the human skin-color. This was also observed in [CN99]. To obtain training data, we manually segment training images into skin and non-skin classes. The skin-color distribution in the CbCr plane is modeled as a bivariate normal distribution. A pixel is classified as skin or non skin based on its Mahalanobis distance. We derive a segmentation threshold for the classifier. The skin and non-skin regions of an image or frame are represented in a skin detection mask (SDM). The performance of our algorithm is illustrated by simulations carried out on still images and video sequences.

A literature survey of some existing skin-color segmentation algorithms is presented in Section 5.2. The proposed skin-color model is described in Section 5.3, and the SDM generation method is described in Section 5.4. Simulations results are presented in Section 5.5, and the chapter is summarized in Section 5.6.

5.1 Introduction

The use of color cues to segment skin regions in image and video sequences has gained increasing popularity in recent years. Skin-color segmentation is feasible because the human skin has a special color distribution that differs significantly (although not entirely) from those of the background objects [CN99]. Skin segmentation has been mainly employed for face segmentation in digital images and video. Some major uses for face segmentation include content-based representation in MPEG-4, face recognition, and face tracking. Skin-color segmentation is usually performed using the chrominance components of image pixels and not the luminance component. The reason for this is twofold: (a) by utilizing the chrominance components only, skin-color segmentation algorithms will remain relatively invariant to changes in brightness (e.g., shadow versus no shadow); and (b) it has been widely observed that apparent differences in skin-color among different races (e.g., dark skin versus fair skin) are characterized by the difference in the brightness of the color, which is governed by the luminance component of light and not the chrominance components [WC97, SP98, CN99, MW00b]. Another reason is that, by considering the chrominance components only, the feature space is reduced from 3D to 2D, thus reducing the computational complexity of the segmentation algorithm.

The human skin is composed of a thin surface layer, called the *epidermis*, followed by a thicker layer, called the *dermis*. Surface reflection takes place at the epidermis, and is approximately $\rho_S = 5\%$ (see Section 3.4 for notations) of the incident light, independent of its wavelength [SAG01]. The rest of the incident light (i.e., 95%) enters the skin, where it is absorbed and scattered within the two skin layers, and then eventually reflected (body reflectance). The epidermis has the property of an optical filter, and absorbs light. The light is transmitted depending on its wavelength and the melanin concentration in the epidermis. In the dermis, the light is scattered and absorbed, and the absorption is mainly dependent on the content of blood and its ingredients, such as hemoglobin, bilirubin, and beta-carotene. The optical properties of the dermis are basically the same for all humans. Skin-color is therefore determined by the epidermis transmittance, which depends mainly on its melanin concentration. Differences in melanin concentration affect the intensity of the light reflected from the skin, but not its hue [JR99]. Variations in the blood content of the dermis are

independent of human ethnicity.

There are some limitations to any skin-color segmentation algorithm that must be considered. Accurate and reliable results are obtained if there is reasonable contrast between skin-color and those of the background objects. Note that in the context of hand and face segmentation, other parts of the body, including clothing, are also considered as background. Stationary background regions with color similar to that of skin do not pose a serious problem, since they can be identified by change detection (Chapter 6). However, articles of clothing that undergo motion and have similar color to that of skin, may pose problems. As well as poor color contrast, there are other limitations of color segmentation when an input image is taken under some particular lighting conditions. The color segmentation process may encounter difficulties if the input image has the following characteristics [CN99]:

- A "bright spot" on the subject's face or hands due to the reflection of an intense light source.
- A dark shadow is present on the face or hands as a result of strong directional lighting that has partially blackened the skin region.
- A colored filter has been used to capture the image.

Bright spots and strong shadows pose great technical challenges to skin color segmentation. The effect of a colored filter can be overcome if skin training pixels for that particular colored filter are available.

We have considered the YCbCr color space in our study since it is typically used in video coding, and provides an effective use of chrominance information for modeling the human skin-color [CN99]. Also, since digital video is stored and processed in the YCbCr color space, our algorithm does not require color space conversion. Conversion from one color space to another is computationally expensive.

The block diagram of our skin-color segmentation algorithm [HLM01, HLM02] is shown in Figure 5.1. The portion within the dashed line is performed during classifier training. The algorithm follows the general segmentation scheme proposed by Salembier and Marqués [SM99] (Section 2.1). The algorithm is automatic in the sense that it does not require any manual adjustment of the design parameters during the skin-color segmentation process.

Also, our algorithm is intended to work on a range of skin types and its underlying assumptions are minimal. The following list describes the steps in the algorithm, and indicates the section in which that step is developed:

1. A universal skin-color model is generated from training images depicting people of different ethnicity (Section 5.3).

2. A segmentation threshold is derived by considering the probability of classification error (Section 5.4.3).

3. A median filter is applied to the Cb and Cr components of the input image (Section 5.4.1).

4. The pixels are classified as skin or non-skin based on their *Mahalanobis distance*. If the Mahalanobis distance of a pixel is below the segmentation threshold, the pixel is classified as skin, otherwise it is classified as non-skin (Section 5.4.2).

5. A skin detection mask (SDM) is generated, indicating skin and non-skin color regions in an image or video frame (Section 5.4.2).

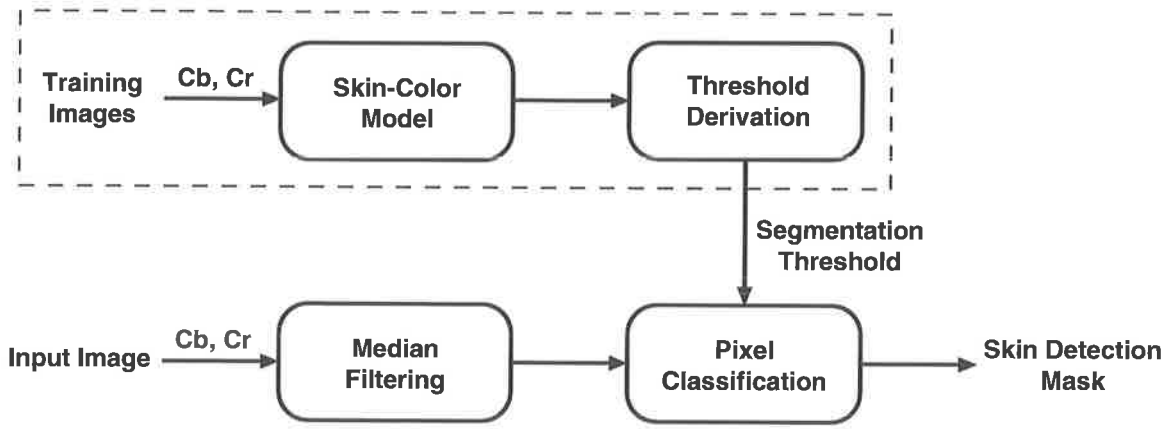


Figure 5.1: Block diagram of the skin-color segmentation algorithm.

5.2 Previous Research

This section reviews some skin-color segmentation algorithms proposed in the literature for different applications. A good review of skin-color segmentation is provided in [Zar99].

In their study, Chai and Ngan [CN99] employed the YCbCr color space (Section 3.5.4) to automatically segment a human face from a given image with a complex background scene. The skin-color pixels are set to a range of [133 173] for the Cr values, and [77 127] for the Cb values. The same set of Cr and Cb values are used for all human races. This set of Cr and Cb values forms a square region in the CbCr plane, as shown in Figure 5.2. Based on the spatial distribution of the detected skin color pixels and their corresponding luminance values, the algorithm employs a set of regularization processes to reinforce regions of skin color pixels that are more likely to belong to the facial regions, and eliminate those that are not. The authors then employ the face segmentation algorithm to improve the perceptual quality of a videophone sequence encoded by a H.261-compliant coder. In a later study, Chai and Bouzerdoum [CB00] considered the Bayesian decision rule for minimum cost to classify image pixels into skin and non-skin classes. The authors tested their algorithm on images of different subjects, head poses, background complexities, and lighting conditions.

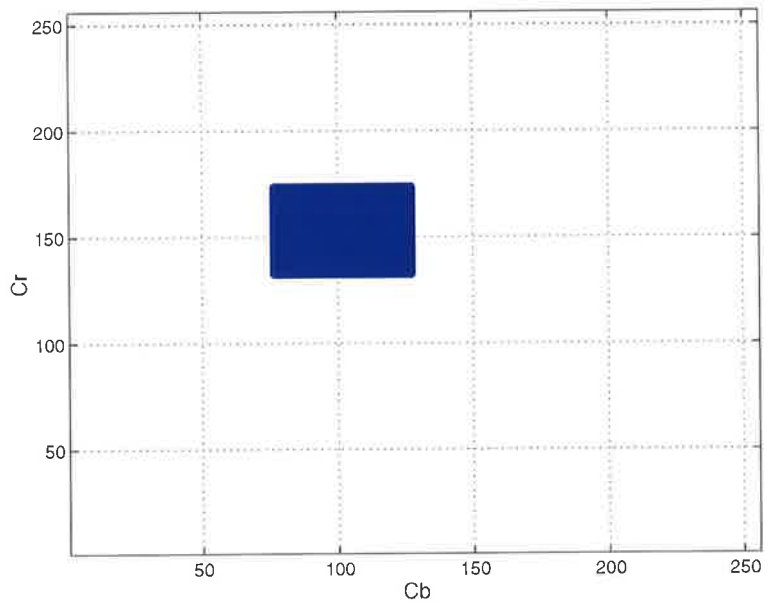


Figure 5.2: Skin-color region in the CbCr plane according to Chai and Ngan, 1999.

Wang and Chang [WC97] proposed an algorithm to detect human face regions in MPEG video sequences. The algorithm takes the inverse quantized discrete cosine transform coefficients as the input, and outputs the location of the detected facial regions. The authors

argue that by detecting faces directly in the compressed domain, there is no need to carry out the inverse discrete cosine transform, enabling the algorithm to run faster. The algorithm uses the Bayesian decision rule for minimum cost to classify a color into the skin class or the non-skin class. Therefore, the classification problem becomes finding the class that gives the minimal cost, considering different cost weightings on classification decisions. The algorithm can be applied to JPEG unconstrained images or motion JPEG video.

Menser and Wien [MW00b] modeled the the skin-color distribution in the CbCr plane as a bivariate normal distribution. Instead of binary classification (i.e., either skin pixel or non-skin pixel), a skin probability image is created. Connected component operators are then applied to the skin probability image to reduce the number of face candidate regions. The number of face candidates regions are then further reduced by employing shape-based operators. To this end, the solidity, aspect ratio, and compactness of each region are evaluated. Finally, texture information is employed to eliminate any remaining non-face regions. The authors applied their algorithm to a H.263 region of interest coding system [MW00a]. The authors claim that the bit-rate can be significantly reduced while retaining a high perceptual quality.

Garcia and Tziritas [GT99] considered both the YCbCr and the HSV (hue, saturation, and value) spaces in their study. The authors proposed a scheme for the detection of faces under unconstrained scene conditions, such as the presence of a complex background and uncontrolled illumination. Clustering and filtering using approximations of the YCbCr and HSV skin color subspaces are first applied to an image, providing quantized skin color regions. The algorithm then iteratively merges the set of homogeneous skin color regions in the color quantized image, in order to provide a set of potential face regions. To detect faces, face shape and size constraints are considered, and then texture analysis is performed on each face area candidate by wavelet packet decomposition. The authors reported a good detection rate in images depicting different face appearances.

The HSV color space was also employed by Sobottka and Pitas [SP96, SP98]. The authors deemed a region of the HS space (shaded area in Figure 5.3) to contain skin-colors. The following parameters, indicated in Figure 5.3, were defined: $S_{min} = 0.23$, $S_{max} = 0.68$, $H_{min} = 0°$, and $H_{max} = 50°$. To discriminate between the face and other skin-color regions, connected components analysis is first performed to remove isolated and small

regions. Then, assuming that the shape of the face can be approximated by an ellipse, shape analysis is performed to discriminate between the face and other regions.

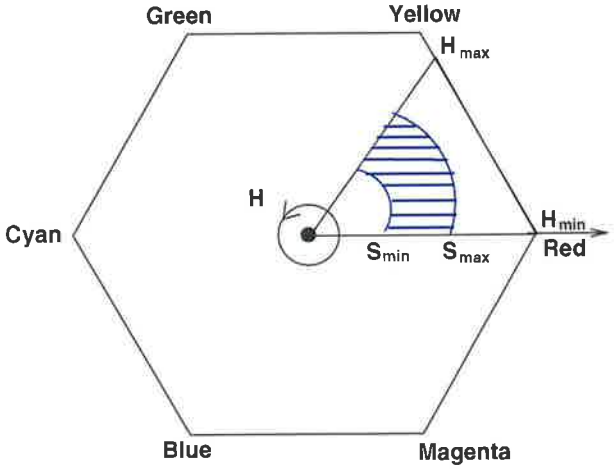


Figure 5.3: The HS space, indicating the region that contains skin-color pixels.

Instead of segmenting the face, Zhu *et al.* [ZYW00] employed the HSV color space to segment the hands for the purpose of gesture recognition. A hand color model and a background color model are generated for each image and the expectation-maximization (EM) algorithm [DLR77] is employed to train a normal mixture model. Pixels are then classified as skin or non-skin using the Bayes decision theory. The authors claim that their proposed algorithm is capable of segmenting hands of arbitrary color in a complex scene. We have found that the performance of the EM algorithm is dependent on the starting parameter values and the number of components in the normal mixture. Usually these values are difficult to estimate.

In his study, Schumeyer [SHB97, SB98, Sch98] considered the problem of segmenting the face and the hands in sign language video sequence, which is also the subject of this thesis. Schumeyer employed the CIE L*a*b*[1] color space for skin color segmentation. The proposed algorithm maps each possible YCbCr value to an a*b* value, and stores the a*b* values in a look-up table. The look-up table is then quantized to reduce the memory size. The skin color distribution is modeled as a normal mixture in the a*b* space. The mean vector and the covariance matrix of each component of the normal mixture are then estimated by the

[1] L^* is lightness defined in (3.2) (i.e., the perceptual response to brightness), and a^* and b^* are the chrominance components.

EM algorithm. A separate skin localization algorithm is required for distribution training. In contrast, our skin-color segmentation algorithm does not require a separate skin localization algorithm. Schumeyer did not take advantage of any motion information to enhance the segmentation results either. The skin-segmentation algorithm along with a proposed perceptual rate controller were incorporated into a H.263 coder for sign language video communication.

The CIE L*u*v* color space (where u^* and v^* are the chrominance components) has also found use in skin-color segmentation. In [YA99], an algorithm is proposed for the segmentation of skin color for applications in image and video databases. Skin color is modeled as a normal mixture in the u*v* color space and the EM algorithm is employed to estimate the mean vectors and covariance matrices of the components. McLachlan's bootstrap method [MB88] is used to determine the number of components in the mixture. In [YA99], skin-color is modeled as a bivariate normal distribution in the u*v* plane. A pixel is classified as skin if its probability in the normal distribution is greater than 0.5.

A color classification algorithm was proposed by Saber *et al.* [STEK96]. The YES color space is employed for color classification. Training pixels for each class are obtained from a set of training images, and each class is modeled as a bivariate normal distribution in the ES plane (i.e., the chrominance plane). The Mahalanobis distance (Section 5.3.2) is then employed to classify image pixels as sky, skin, or grass. A universal threshold is selected for each class based on the the receiver operating characteristics (ROC) (Section 5.4.3) of the training set. The classification results are improved by adapting the universal threshold to the characteristics of the individual pixels based on histogram cluster analysis. Finally, if a pixel is found to belong to more than one class, a maximum *a posteriori* probability (MAP) rule is employed to resolve the ambiguity.

An algorithm for the detection of faces and facial features was introduced by Saber and Tekalp [ST98]. First, using the algorithm introduced in [STEK96] (see above), image pixels are classified as skin or non-skin to obtain a color-classification map. The color-classification map is then smoothed by Gibbs random field model-based filters to define skin regions. Symmetry-based cost functions are then employed to search for facial features, such as the tip of the nose and the center of the eyes.

Research has also concentrated on skin-color segmentation using the RGB color space [JR98, JR99]. Note that in the RGB color space, the luminance component and the chromi-

nance components are not decoupled. Also, the feature space is 3D for RGB as opposed to 2D for chrominance skin-color segmentation. In [JR98] and [JR99], the authors compared the performance of histogram and normal mixture models in skin detection and found histogram models to be superior in accuracy and computational cost. A training set is first obtained from images on the world wide web, and the normal mixture model is trained using the EM algorithm. As mentioned in Section 3.3 the chrominance information (i.e., hue and saturation) of light can be measured by employing chromaticity coordinates. An algorithm for the detection of the hands in sign language video sequences was presented in [AA01]. The authors employed the RGB skin-color segmentation approach presented in [JR98] and [JR99] to detect skin-color regions in the video frames. To eliminate false alarms, motion information obtained from motion history images is employed. The authors did not explain how the face and the hands can be differentiated. Bergasa *et al.* [BMG+00] considered the problem of skin-color segmentation using the normalized RGB space, where:

$$r = \frac{R}{R+G+B}\,, \tag{5.1}$$

and

$$g = \frac{G}{R+G+B}\,. \tag{5.2}$$

The proposed algorithm employs a clustering method based on vector quantization.

An algorithm for finding faces in the wavelet domain for content-based coding of color images was introduced by Karlekar and Desai [KD00]. The YUV color space (see Section 3.5.2) is employed in the algorithm, and three level wavelet transform is performed on each component (Y, U, and V) of the input image. The lower resolution LL (low, low) subimage corresponding to chrominance components (U,V) of wavelet transform is used for skin-color classification. A multilayer perceptron (MLP) is trained to classify each pixel as skin or non-skin. The result is a binary map indicating skin and non-skin regions in the image. The binary map is further processed by using a median filter to eliminate noise and to fill holes. In the second stage of the algorithm, shape constraints are applied to the connected components in the binary map to eliminate false alarm regions. The final stage of the algorithm examines the wavelet coefficients of component Y. The eyes, nose, and lips, give rise to high frequency coefficients in the wavelet domain. If a skin-color region in the binary map does not have sufficient high frequency coefficient, it is deemed a false alarm and discarded. The block

diagram of the algorithm is shown in Figure 5.4.

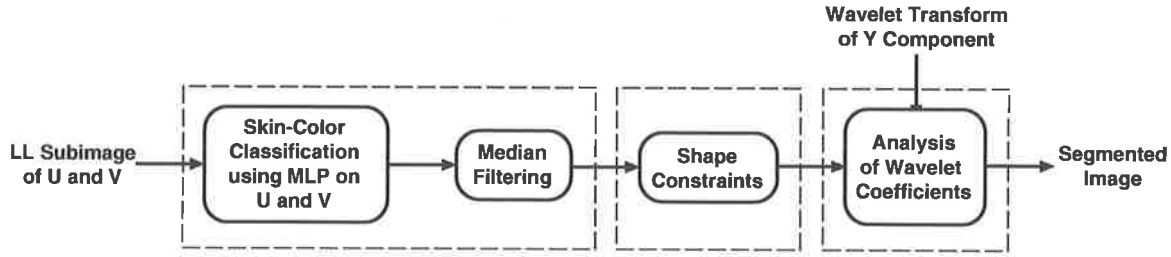


Figure 5.4: Block diagram of the wavelet based face segmentation algorithm introduced by Karlekar and Desai (2000).

Note that algorithms that advocate color space other than the YCbCr would incur additional computational cost because digital video processing is usually performed in the YCbCr color space.

5.3 Generation of the Skin-Color Model

A skin-color model can be derived in three ways. One approach is to predefine a skin-color model for a particular race, lighting condition, or camera system. For example, Terrillon *et al.* [TDA98] predefined a skin-color model for a particular camera system and race. Another approach is to define a skin-color model for an individual (e.g., [Sch98]). The third approach is to predefine a universal skin-color model that encompasses different races and lighting conditions. Among the three approaches, the first two are likely to produce more accurate segmentation results since a more precise skin-color model is employed. However, improved segmentation results are realized at the expense of a skin segmentation algorithm that is either too restrictive because it uses a model that is suited only to a particular condition, or that requires human interaction to manually define the necessary model. Therefore, the third approach is more practical since it attempts to cater for a wide range of skin-colors and lighting conditions. In this thesis, we have opted for the third approach.

5.3.1 Manual Segmentation of Training Images

This section describes the method used to obtain labeled training pixels from training images. The training images were downloaded over the internet, and were selected randomly from various sites. Each image was visually examined to assess whether it was modified by a colored filter. If an image was modified by a colored filter, for example a person's skin appeared too red or yellow, the image was excluded from the training set. The training images were of different subjects (with different ethnicities, e.g., European, Asian, and African), body poses, background complexities, and lighting conditions (e.g., outdoor, indoor, and studio images).

Each image was manually segmented in the following manner: regions of skin pixels were marked using the Jasc Paint Shop Pro™ software tool [Jas01]. In the labeled image, the eyes, hair, eyebrows, and the mouth opening were excluded. In most images, it was often difficult to define the boundary between skin and non-skin regions (e.g., forehead obscured by hair), therefore only the easily identifiable skin pixels were segmented. This strategy was employed to avoid the contamination of the skin training pixels with non-skin pixels. The marked skin pixels were then added to the skin training set, and the non-skin pixels were similarly added to the non-skin training set.

The manual segmentation process is depicted in Figure 5.5. The original image is shown in Figure 5.5(a) and the labeled image is shown in Figure 5.5(b). Figure 5.5(c) shows the binary mask with the skin pixels set to binary "1" and non-skin pixels set to binary "0".

(a) (b) (c)

Figure 5.5: Example of manual image segmentation. (a) Original image, (b) labeled image, and (c) binary mask.

Figures 5.6(a), (b) and (c) show the distribution of the skin training pixels in the CbCr

plane for people of European, African and Asian descent, respectively. Notice that the skin training pixels occupy similar regions in the CbCr plane. The fact that the skin-training pixels occupy similar regions in the CbCr plane suggests that the chrominance components of skin-color are invariant across different races. The skin training pixels for all three races is shown in Figure 5.7. We notice that the skin training pixels form a small and compact cluster in the CbCr plane (Figure 5.7(a)), and the Y component (luminance) has little influence on the distribution of the training pixels in the CbCr plane (Figure 5.7(b)). This demonstrates that an effective skin-color model can be derived based on the Cb and Cr components of the input image.

5.3.2 The Skin-Color Model

This section discusses the proposed skin-color model. Let $\mathbf{c}$ denote the feature vector formed by the Cb and Cr components of a pixel (i.e., $\mathbf{c} = [Cb\ Cr]^T$), and $\mathbf{c}$ is in a 2D Euclidean space $\mathbf{R^2}$, called the feature space. The skin and non-skin classes are denoted by ω_S and $\omega_{\bar{S}}$, respectively. We model the skin-color distribution in the CbCr plane (Figure 5.7) as a bivariate normal distribution:

$$p(\mathbf{c}|\omega_S) = \frac{1}{2\pi|\mathbf{\Sigma}_S|^{\frac{1}{2}}}\exp[-\frac{1}{2}(\mathbf{c}-\boldsymbol{\mu}_S)^T\mathbf{\Sigma}_S^{-1}(\mathbf{c}-\boldsymbol{\mu}_S)], \tag{5.3}$$

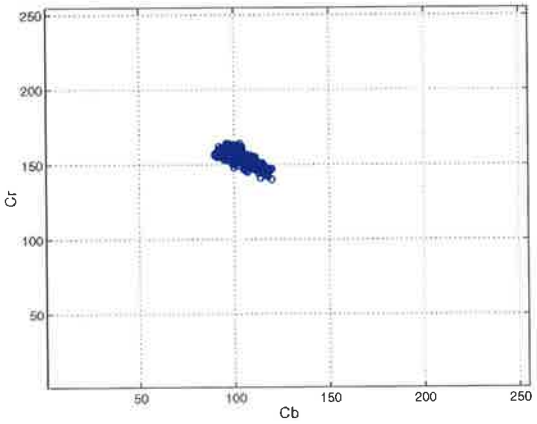
where $\boldsymbol{\mu}_S$ and $\mathbf{\Sigma}_S$ are the mean vector and covariance matrix of the distribution, respectively, and $|\mathbf{\Sigma}_S|$ is the determinant of $\mathbf{\Sigma}_S$. These parameters are estimated from the skin training pixels. Consider an ith skin training pixel. The sample mean vector and covariance matrix are given by [DHS01]:

$$\hat{\boldsymbol{\mu}}_S = \frac{1}{N_S}\sum_{i=1}^{N_S}\mathbf{c}_i, \tag{5.4}$$

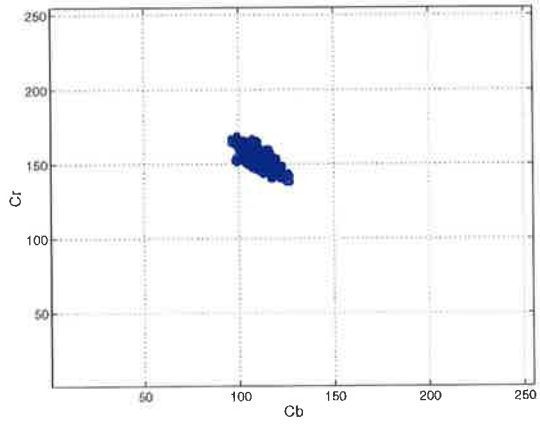
and

$$\hat{\mathbf{\Sigma}}_S = \frac{1}{N_S-1}\sum_{i=1}^{N_S}(\mathbf{c}_i-\hat{\boldsymbol{\mu}}_S)(\mathbf{c}_i-\hat{\boldsymbol{\mu}}_S)^T, \tag{5.5}$$

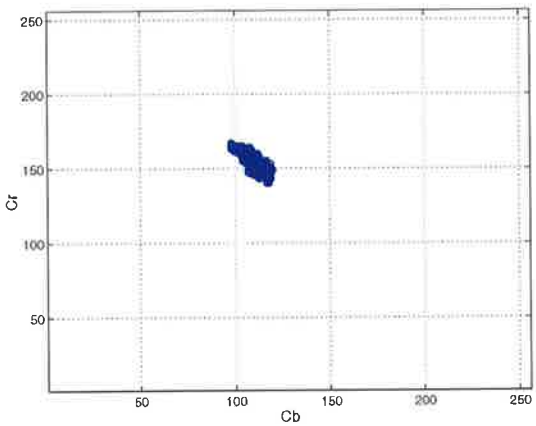
where N_S is the number of skin training pixels and is in the order of ninety thousand pixels. The results are given in Table 5.1.



(a)

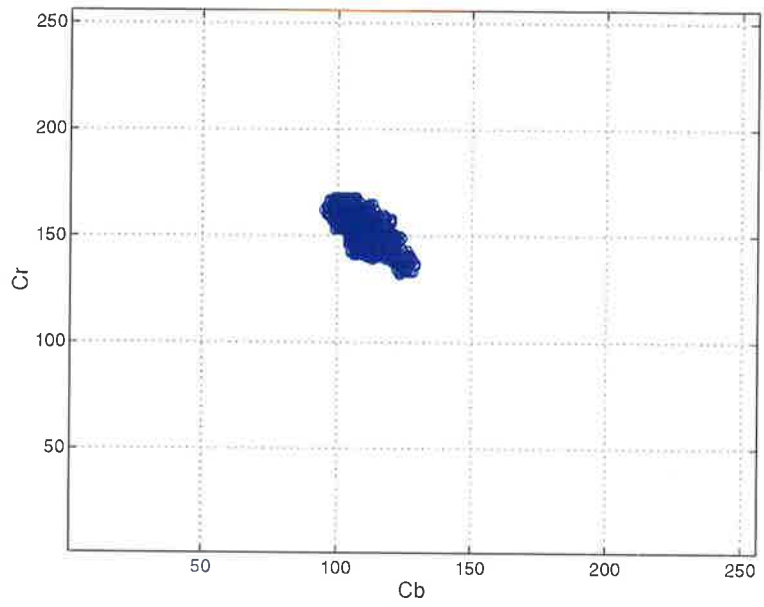


(b)

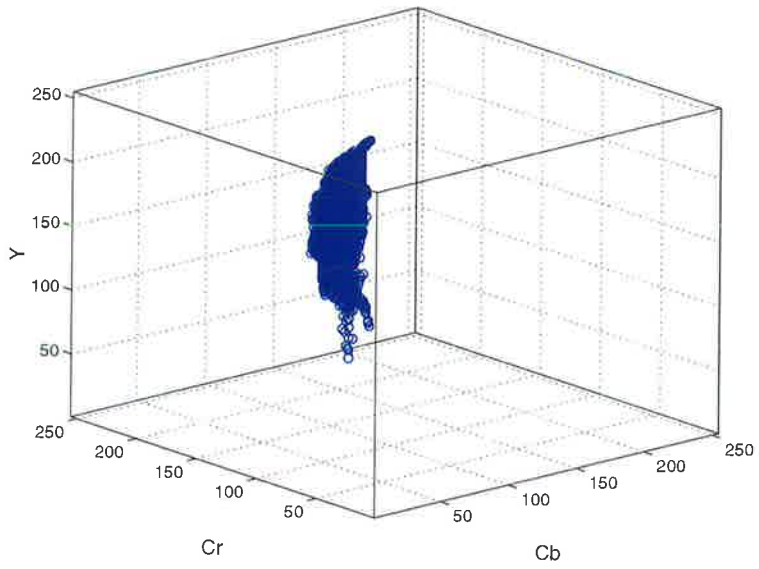


(c)

Figure 5.6: Skin training pixels in the CbCr plane. (a) European skin, (b) African skin, and (c) Asian descent.



(a)



(b)

Figure 5.7: Skin training pixels for people of European, Asian, and African descent in the (a) CbCr plane, and the (b) YCbCr cube.

Parameter	Value
$\hat{\boldsymbol{\mu}}_S = \left[\begin{array}{cc} \hat{\mu}_{Cb} & \hat{\mu}_{Cr} \end{array} \right]^T$	$\left[\begin{array}{cc} 109.3 & 151.5 \end{array} \right]^T$
$\hat{\boldsymbol{\Sigma}}_S = \left[\begin{array}{cc} \hat{\sigma}_{CbCb} & \hat{\sigma}_{CbCr} \\ \hat{\sigma}_{CrCb} & \hat{\sigma}_{CrCr} \end{array} \right]$	$\left[\begin{array}{cc} 31.4 & -21.8 \\ -21.8 & 47.0 \end{array} \right]$
$\mathbf{U} = \left[\begin{array}{cc} \mathbf{e}_1 & \mathbf{e}_2 \end{array} \right]$	$\left[\begin{array}{cc} 0.82 & -0.58 \\ 0.58 & 0.82 \end{array} \right]$
$\boldsymbol{\Upsilon} = \left[\begin{array}{cc} \upsilon_1 & 0 \\ 0 & \upsilon_2 \end{array} \right]$	$\left[\begin{array}{cc} 16.1 & 0 \\ 0 & 62.3 \end{array} \right]$

Table 5.1: Estimated model parameters for the skin class density function.

The Mahalanobis Distance

The quantity d in

$$d^2 = (\mathbf{c} - \boldsymbol{\mu}_S)^T \boldsymbol{\Sigma}_S^{-1} (\mathbf{c} - \boldsymbol{\mu}_S) \tag{5.6}$$

is the *Mahalanobis distance* from $\mathbf{c}$ to $\boldsymbol{\mu}_S$. It follows from (5.3) that the contours of constant density are ellipses for which d is constant. If the features are uncorrelated and the variances in all directions are the same, these contours are circles, and the Mahalanobis distance is equivalent to the Euclidean distance. We define $\mathbf{U}$ to be a 2×2 matrix whose columns are the orthonormal eigenvectors ($\mathbf{e}_1$ and $\mathbf{e}_2$) of $\boldsymbol{\Sigma}_S$,

$$\mathbf{U} = [\mathbf{e}_1 \; \mathbf{e}_2], \tag{5.7}$$

and $\boldsymbol{\Upsilon}$ the diagonal matrix of the corresponding eigenvalues (υ_1 and υ_2),

$$\boldsymbol{\Upsilon} = \left[\begin{array}{cc} \upsilon_1 & 0 \\ 0 & \upsilon_2 \end{array} \right]. \tag{5.8}$$

Note that υ_1 and υ_2 are the eigenvalues associated with $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively (values given in Table 5.1). The axes of the ellipses are given by the principal components (the eigenvectors) of $\boldsymbol{\Sigma}_S$, and the length of these axes are given by $d\sqrt{\upsilon_1}$ (along $\mathbf{e}_1$) and $d\sqrt{\upsilon_2}$ (along

e_2). These concepts are illustrated in Figure 5.8. The ellipse is at constant Mahalanobis distance d from μ_S in the CbCr plane.

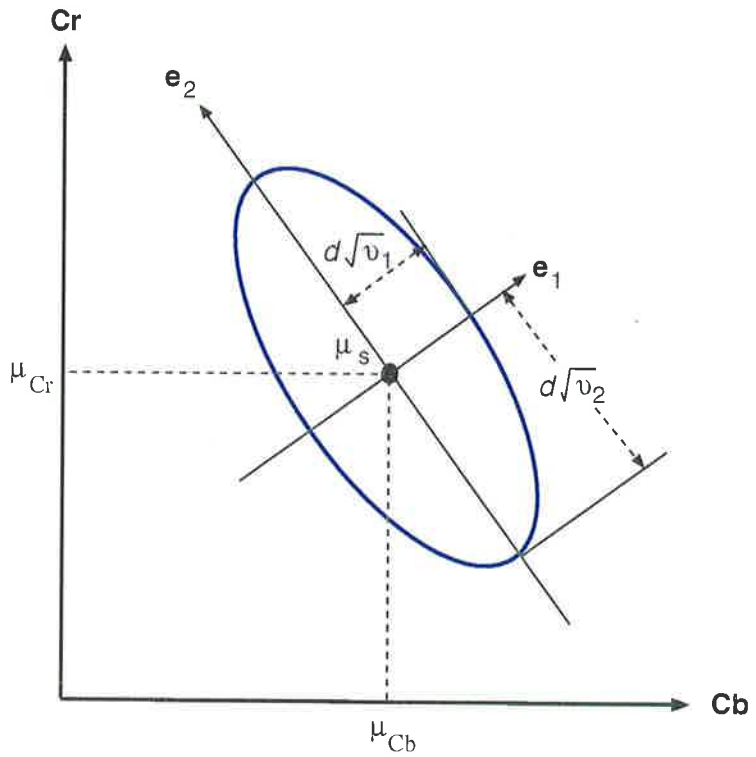


Figure 5.8: Contour of constant Mahalanobis distance d from μ_S.

Equation (5.6) provides a mapping from the 2D feature space to a 1D distance space. Within this framework, skin pixels can be classified as skin or non-skin based on their Mahalanobis distance. The value of d is related to the probability that a given pixel belongs to class ω_S. A small value of d indicates a high skin pixel probability and vice-versa.

5.4 Generation of the Skin Detection Mask

In this section, we describe the classification method employed to classify pixels as skin or non-skin. The method is analogous to the single hypothesis classifier [Fuk90]. Single hypothesis schemes have been proposed to solve problems in which one class is well defined while others are not.

5.4.1 Median Filtering

Prior to pixel classification, a median filter [GW92] is applied to the Cb and Cr components of each image that is to be segmented. In median filtering, the chrominance-level of each pixel is replaced by the median of the chrominance-levels in the neighborhood of that pixel. The filter removes outliers in skin regions, while preserving edges (if the size of the kernel is small, see below).

The median med of a set of values is such that half the values in the set are less than med and half are greater than med. In order to perform median filtering in a neighborhood of a pixel, the values of the pixel and its neighbors are first sorted, the median is determined, and the median value is assigned to the pixel. For example, in a 3×3 kernel the median is the 5th largest value, in a 5×5 kernel the median is the 13th largest value, and so on. When several values in a neighborhood are the same, all equal values have to be grouped. For example, suppose that a 3×3 kernel has values (16, 21, 21, 21, 11, 21, 21, 200, 26). These sorted values are (11, 16, 21, 21, 21, 21, 21, 26, 200), which results in a median of 21. Therefore, the principal function of median filtering is to force points with distinct intensities to be more like their neighbors, actually eliminating intensity spikes that appear isolated in the neighborhood.

The size of the kernel was chosen based on an empirical study of sign language video frames (in QCIF format). Our experimental data suggests that if the size of the kernel is large (i.e., 7×7 pixels or larger), the face and hand objects would merge if they are close to each other. This is not desirable for face detection. Alternatively, a small kernel size (i.e., 3×3 pixels) would be ineffective in eliminating outliers. We have found a kernel size of 5×5 pixels to be effective in eliminating outliers without merging nearby skin-color regions. For larger frame sizes, the size of the kernel should be increased accordingly. For example, if the frame size is CIF (i.e., double of QCIF, see Appendix A), the kernel size should be set to 10×10 pixels.

Examples of the effect of using different kernel sizes in median filtering are shown in Figure 5.9. Figure 5.9(a) shows frame 2 of the *Irene* sequence, and Figures 5.9(b), (c), (d), and (e) show the effect of using kernels of different sizes. For a kernel size of 3×3 pixels some residual noise is present in the skin detection mask, while for a kernel size of 7×7

pixels the hand objects have merged. For larger kernel sizes (i.e., greater than 11×11 pixels), an added disadvantage is that the edges of the face and hand regions become distorted (Figure 5.9(e)).

5.4.2 Pixel Classification

As discussed in Section 5.3.2, the skin-color distribution in the CbCr plane is modeled as a bivariate normal distribution. Recall that for the bivariate normal distribution, the contours of constant density are ellipses of constant Mahalanobis distance to $\boldsymbol{\mu}_S$. To classify a pixel as skin or non-skin, we first measure the Mahalanobis distance of the pixel, i.e., the Mahalanobis distance between the feature vector of the pixel and $\boldsymbol{\mu}_S$. Next, we compare the Mahalanobis distance against a predetermined threshold. If the Mahalanobis distance is less than or equal to the predetermined threshold, the pixel is classified as skin, otherwise it is classified as non-skin.

The skin detection mask (SDM) is defined as:

$$SDM(x, y, k) = \begin{cases} 1, & \text{if } d_{x,y,k} \leq \tau \\ 0, & \text{otherwise,} \end{cases} \tag{5.9}$$

where τ is the segmentation threshold. The Mahalanobis distance $d_{x,y,k}$ is defined as:

$$d^2_{x,y,k} = (\mathbf{c}_{x,y,k} - \hat{\boldsymbol{\mu}}_S)^T \hat{\boldsymbol{\Sigma}}_S^{-1} (\mathbf{c}_{x,y,k} - \hat{\boldsymbol{\mu}}_S). \tag{5.10}$$

The parameter $\mathbf{c}_{x,y,k}$ denotes the feature vector of a pixel in frame k, at spatial location (x, y). Therefore, SDM is a bitmap where binary "1" indicates a skin class pixel and a binary "0" indicates a non-skin class pixel. The derivation of a suitable segmentation threshold is discussed next.

5.4.3 Derivation of the Segmentation Threshold

Let $\mathcal{R}_S$ denote the region in the feature space where the classifier decides ω_S, and likewise for $\mathcal{R}_{\bar{S}}$ and $\omega_{\bar{S}}$. There are two ways in which a classification error can occur; either an observation $\mathbf{c}$ falls in $\mathcal{R}_S$ and the true class is $\omega_{\bar{S}}$, or $\mathbf{c}$ falls in $\mathcal{R}_{\bar{S}}$ and the true class is ω_S. Since these events are mutually exclusive and collectively exhaustive, the probability of error

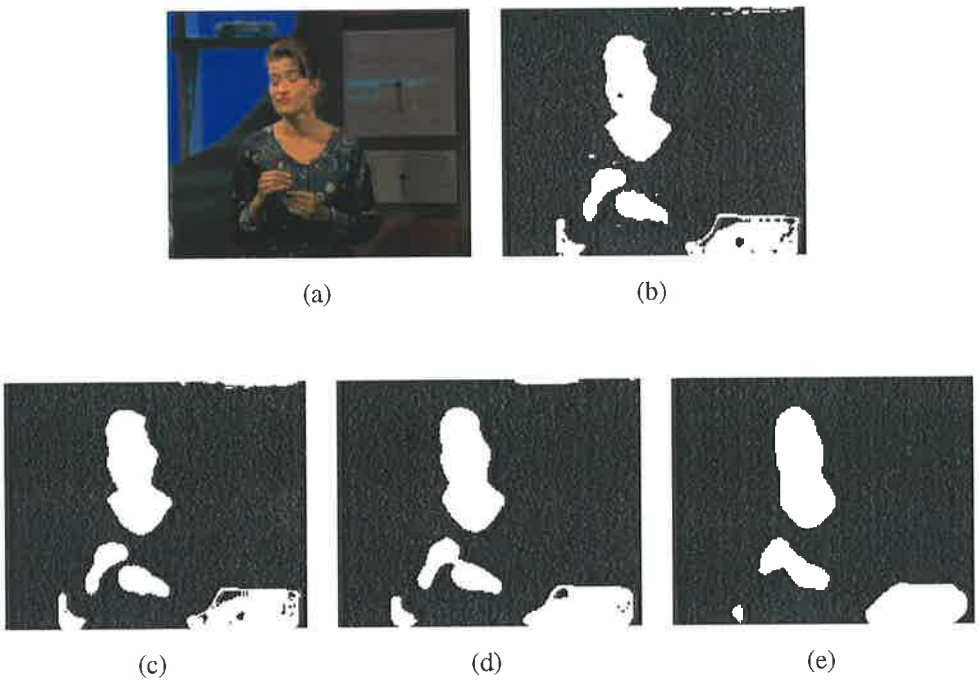


Figure 5.9: The effect of using different kernel sizes in median filtering. (a) Frame 2 of the *Irene* sequence, (b) kernel with a size of 3×3 pixels, (c) kernel with a size of 5×5 pixels, (d) kernel with a size of 7×7 pixels, and (e) kernel with a size of 17×17 pixels.

is

$$\begin{aligned} P_{error} &= P(\mathbf{c} \in \mathcal{R}_{\bar{S}}, \omega_S) + P(\mathbf{c} \in \mathcal{R}_S, \omega_{\bar{S}}) && (5.11) \\ &= P(\mathbf{c} \in \mathcal{R}_{\bar{S}} | \omega_S) P(\omega_S) + P(\mathbf{c} \in \mathcal{R}_S | \omega_{\bar{S}}) P(\omega_{\bar{S}}) && (5.12) \\ &= \int_{\mathcal{R}_{\bar{S}}} p(\mathbf{c}|\omega_S) P(\omega_S) + \int_{\mathcal{R}_S} p(\mathbf{c}|\omega_{\bar{S}}) P(\omega_{\bar{S}}), && (5.13) \end{aligned}$$

where $P(\omega_S)$ and $P(\omega_{\bar{S}})$ denote the *a priori* probabilities of the skin and non-skin classes, respectively, and $P(\omega_S) + P(\omega_{\bar{S}}) = 1$. The probabilities $p(\mathbf{c}|\omega_S)$ and $p(\mathbf{c}|\omega_{\bar{S}})$ denote the conditional probability densities of $\mathbf{c}$ given ω_S and $\omega_{\bar{S}}$, respectively. For the remainder of this chapter, the following notations, borrowed from radar terminology, will be used

$$\begin{aligned} P_F &= \int_{\mathcal{R}_S} p(\mathbf{c}|\omega_{\bar{S}}) P(\omega_{\bar{S}}) \\ P_D &= \int_{\mathcal{R}_S} p(\mathbf{c}|\omega_S) P(\omega_S) \\ P_M &= \int_{\mathcal{R}_{\bar{S}}} p(\mathbf{c}|\omega_S) P(\omega_S). && (5.14) \end{aligned}$$

P_F, P_D and P_M are the probabilities of false alarm, detection and miss, respectively. Note that $P_M = 1 - P_D$. The concepts of false alarm, detection, and miss as related to an image, are illustrated in Figure 5.10. The box represents a skin region (SR) and the circles indicate the regions identified by the classifier as skin. Detection occurs when the SR coincides with a region, or part of a region, identified as skin. When the SR, or part of it, is not identified as skin, miss occurs. False alarm occurs when a skin identified region, or part of it, is outside the SR.

Based on the above notations, we define the probability of error as

$$P_{error} = P_M(\theta) P(\omega_S) + P_F(\theta) P(\omega_{\bar{S}}), \quad (5.15)$$

where θ is a threshold. The probability of error is a function of θ and the *a priori* probabilities, $P(\omega_S)$ and $P(\omega_{\bar{S}})$. The relationship between θ, $\mathcal{R}_S$, and $\mathcal{R}_{\bar{S}}$ is illustrated in Figure 5.11. The ellipse represents the decision boundary, and is at Mahalanobis distance θ from $\boldsymbol{\mu}_S$. Note that region $\mathcal{R}_S$ is inside the ellipse while region $\mathcal{R}_{\bar{S}}$ is outside the ellipse.

The probabilities P_D and P_F are evaluated using the skin and non-skin class training

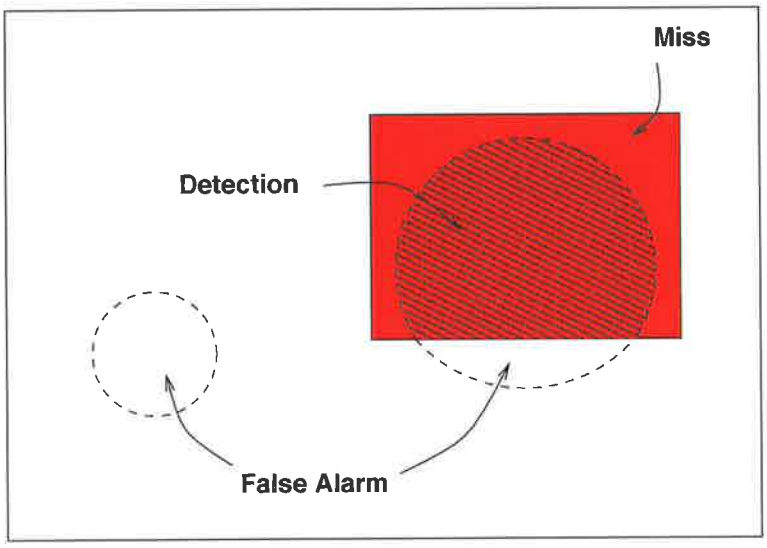


Figure 5.10: The concepts of false alarm, detection, and miss as related to an image. The box represents a skin region (SR) and the circles indicate the regions identified by the classifier as skin.

data. For the set of training images I_j, $j = 1, ..., J$,

$$P_D(\theta) = \frac{1}{N_S} \sum_{j=1}^{J} \sum_{(x,y) \in I_j} \alpha(\mathbf{c}_{x,y,j}, \theta), \tag{5.16}$$

and

$$P_F(\theta) = \frac{1}{N_{\bar{S}}} \sum_{j=1}^{J} \sum_{(x,y) \in I_j} \beta(\mathbf{c}_{x,y,j}, \theta), \tag{5.17}$$

where $\alpha(\mathbf{c}_{x,y,j}, \theta)$ and $\beta(\mathbf{c}_{x,y,j}, \theta)$ are defined as:

$$\alpha(\mathbf{c}_{x,y,j}, \theta) = \begin{cases} 1, & \text{if } \mathbf{c}_{x,y,j} \in \omega_S \text{ and } d_{x,y,j} \leq \theta \\ 0, & \text{otherwise,} \end{cases} \tag{5.18}$$

and

$$\beta(\mathbf{c}_{x,y,j}, \theta) = \begin{cases} 1, & \text{if } \mathbf{c}_{x,y,j} \in \omega_{\bar{S}} \text{ and } d_{x,y,j} \leq \theta \\ 0, & \text{otherwise.} \end{cases} \tag{5.19}$$

The parameter $\mathbf{c}_{x,y,j}$ denotes the feature vector of a pixel in training image I_j, $j = 1, ..., J$, at spatial location (x, y). The Mahalanobis distance $d_{x,y,j}$ is defined as:

$$d^2_{x,y,j} = (\mathbf{c}_{x,y,j} - \hat{\boldsymbol{\mu}}_S)^T \hat{\boldsymbol{\Sigma}}_S^{-1} (\mathbf{c}_{x,y,j} - \hat{\boldsymbol{\mu}}_S). \tag{5.20}$$

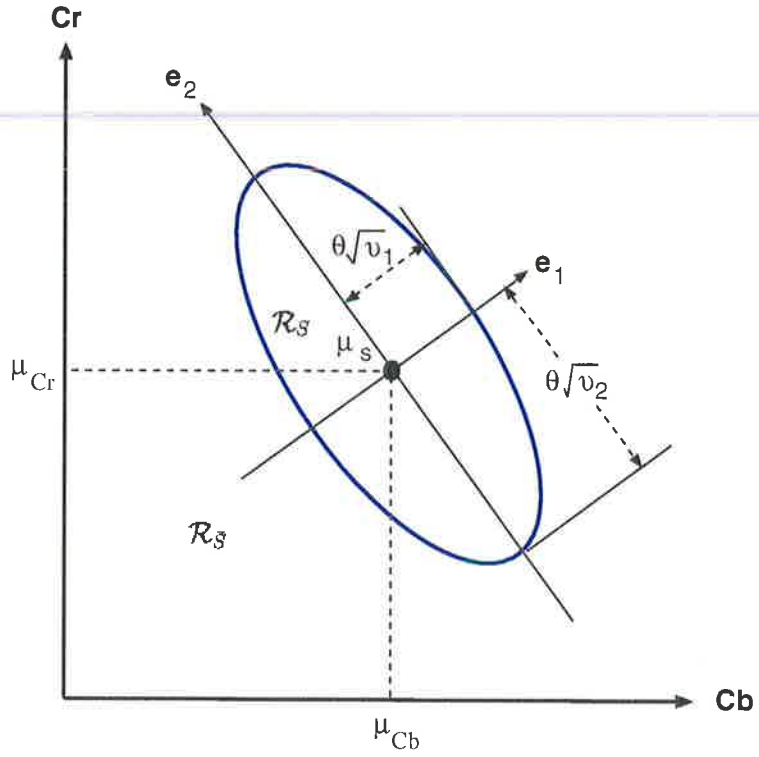


Figure 5.11: Contour of equal density at Mahalanobis distance θ from μ_S, where υ_1 and υ_2 are the eigenvalues associated with e_1 and e_2, respectively. Region $\mathcal{R}_S$ is inside the ellipse while region $\mathcal{R}_{\bar{S}}$ is outside the ellipse.

The problem is how to derive a segmentation threshold. Two methods for deriving the segmentation threshold are described next. In both methods, the probability of error will guide the τ selection process.

Case One: $P(\omega_S)$ is Known

Consider if $P(\omega_S)$ is known. The segmentation threshold can be derived by minimizing (5.15),

$$\tau = \arg \min_{\theta} \left(P_M(\theta) P(\omega_S) + P_F(\theta) P(\omega_{\bar{S}}) \right). \tag{5.21}$$

To solve (5.21), we plot θ against P_{error} and find the minimum P_{error}. For generic images and video sequences, $P(\omega_S)$ is difficult to estimate. However, for specific images or video sequences (e.g., passport photos or sign language video sequences), it is possible to estimate $P(\omega_S)$. The skin class prior $P(\omega_S)$ is simply the number of skin pixels in an image divided

by the number of pixels in an image. Using the dimensions of a QCIF frame (Appendix A), the number of image pixels is $176 \times 144 = 25\,344$. This leaves only the number of skin pixels to be estimated. For a sign language video frame, Schumeyer [Sch98] assumed a face size of 50×50 pixels and a hand size of 25×25 pixels. This results in $P(\omega_S) = 0.15$ and $P(\omega_{\bar{S}}) = 0.85$. Based on our experimental data, $P(\omega_S) = 0.15$ seems appropriate for sign language video. Figure 5.12 shows the probability of error versus θ for $P(\omega_S) = 0.15$. The minimum P_{error} is indicated in the graph and corresponds to $\theta = 2.1$. The corresponding skin-color region in the CbCr plane is shown in Figure 5.13.

We now briefly compare the decision boundary (in the CbCr plane) proposed in this thesis to the one proposed by Chai and Ngan (CN) [CN99], and reviewed in Section 5.2. As eluded to in Section 5.3.2, our proposed decision boundary forms an ellipse in the CbCr plane. Based on the distribution of the skin training pixels in the CbCr plane (Figure 5.7), an elliptical decision boundary seems more appropriate than the square decision boundary advocated by CN (Figure 5.2). To see the effect that the different decision boundaries have on skin-color segmentation, we segment frame 33 of the *Silent* sequence, shown in Figure 5.14(a). Notice that the amount of false alarms in the SDM (with the identified skin-color pixels shown) using the CN decision boundary (Figure 5.14(b)) is much higher than the amount of false alarms in the SDM using our elliptical decision boundary (Figure 5.14(c)).

The following is a summary of the procedure used to derive the segmentation threshold when $P(\omega_S)$ is known:

1. Find an expression for the probability of classification error as a function of $P(\omega_S)$ and θ.

2. Evaluate $P_D(\theta)$ and $P_F(\theta)$ for the set of training images.

3. Estimate $P(\omega_S)$.

4. For the given $P(\omega_S)$, the segmentation threshold is taken to be the value of θ that minimizes the probability of error.

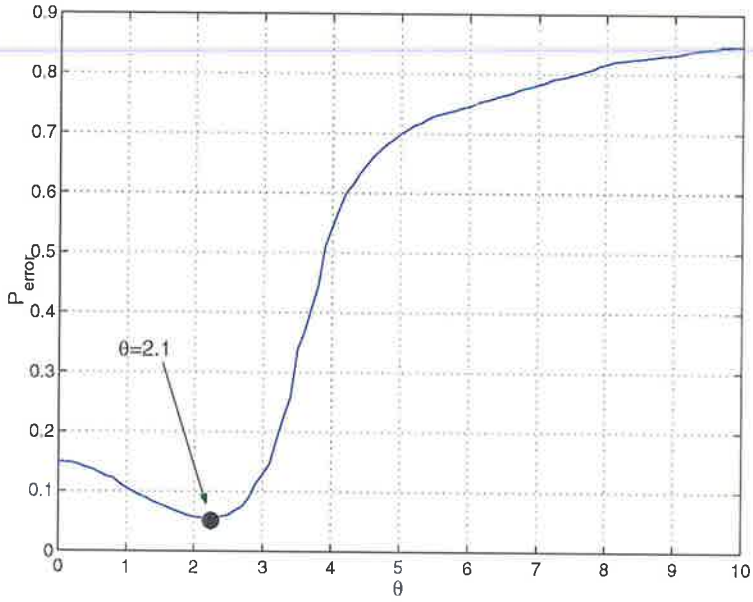


Figure 5.12: Probability of classification error versus θ for $P(\omega_S) = 0.15$ and $P(\omega_{\bar{S}}) = 0.85$.

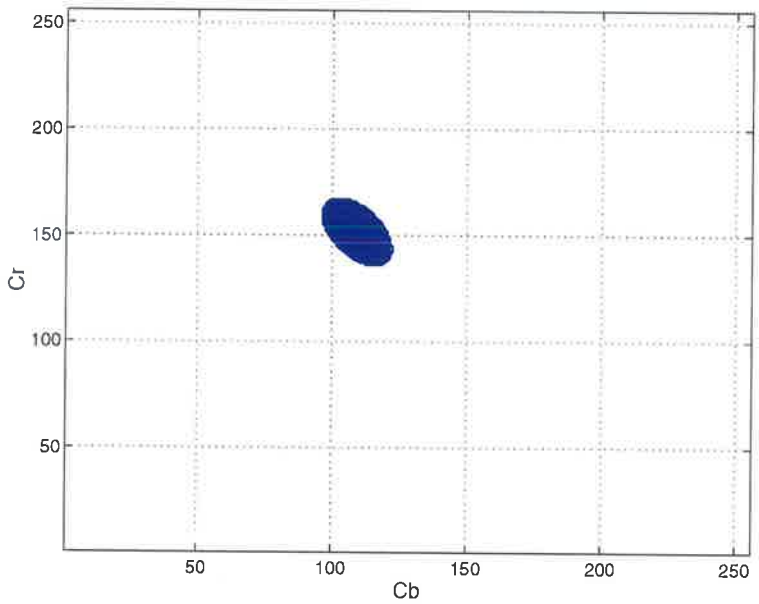


Figure 5.13: Skin-color region in the CbCr plane when $\tau = 2.1$.

(a)

(b)

(c)

Figure 5.14: The effect of using different decision boundaries on the SDM. (a) Frame 33 of the *Silent* sequence, (b) SDM obtained using the CN decision boundary, and (c) SDM obtained using our proposed elliptical decision boundary.

Case Two: $P(\omega_{\bar{S}})$ is Unknown

The expression for the error probability in (5.15) shows that once θ is fixed (hence P_M and P_F are fixed), P_{error} is a linear function of the *a priori* probabilities, $P(\omega_S)$ and $P(\omega_{\bar{S}})$. Therefore, to select the θ that minimizes P_{error}, $P(\omega_S)$ must be known beforehand. Unfortunately, $P(\omega_S)$ can vary quite considerably among different images. Factors such as a person's distance from the video camera and the orientation of the body can influence $P(\omega_S)$. The minimax test [Tre68, Fuk90, DHS01] is designed to protect the performance of the classifier from variations in $P(\omega_S)$.

Inserting $P(\omega_{\bar{S}}) = 1 - P(\omega_S)$ into equation (5.15),

$$\begin{aligned} P_{error} &= P_M(\theta)P(\omega_S) + P_F(\theta)(1 - P(\omega_S)) \\ &= (P_M(\theta) - P_F(\theta))P(\omega_S) + P_F(\theta). \end{aligned} \tag{5.22}$$

In Figure 5.15, the curved line at the bottom shows minimum P_{error} plotted against $P(\omega_S)$ (i.e., θ is selected optimally for each $P(\omega_S)$). If θ is fixed at some threshold, say $\theta = 2.1$, and $P(\omega_S)$ allowed to change, P_{error} will change as a linear function of $P(\omega_S)$, as indicated by the dashed-dot line. The maximum P_{error} will occur at the extreme value of the *a priori* probability, $P(\omega_S) = 1$. To minimize the maximum P_{error}, θ should be set to make the coefficient of $P(\omega_S)$ in (5.22) zero, regardless of $P(\omega_S)$. That is, we need to solve:

$$\begin{aligned} P_M(\theta) - P_F(\theta) &= 0, \\ P_M(\theta) &= P_F(\theta), \end{aligned} \tag{5.23}$$

for θ. This choice of θ would render P_{error} independent of $P(\omega_S)$, as indicated by the horizontal dashed line (line of equal error), and hence guarantee that the maximum error probability is minimized regardless of any changes in $P(\omega_S)$.

In order to find the segmentation threshold based on the minimax test, we need to show the miss and false alarm probabilities as a function of θ (Figure 5.16). The point where $P_M(\theta) = P_F(\theta)$ is indicated in the graph and corresponds to $\theta = 2.6$. The data of Figure 5.16 also appears in the corresponding receiver operating curve (ROC) of Figure 5.17. The receiver operating curve shows $P_M(\theta)$ versus $P_F(\theta)$. Based on the information in Figures 5.16 and 5.17, $\tau = 2.6$ is derived for the minimax test. We advocate the use of $\tau = 2.6$

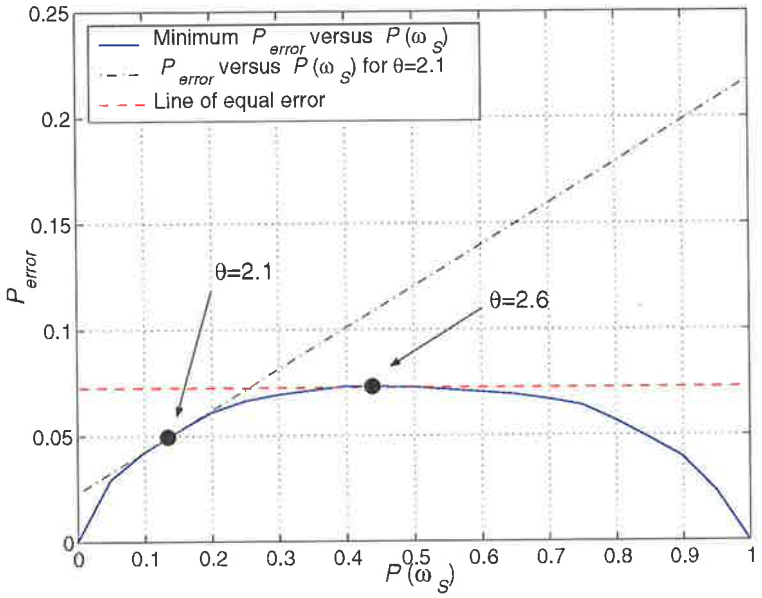


Figure 5.15: Graph showing minimum P_{error} versus $P(\omega_S)$, P_{error} versus $P(\omega_S)$ for $\theta = 2.1$, and the line of equal error.

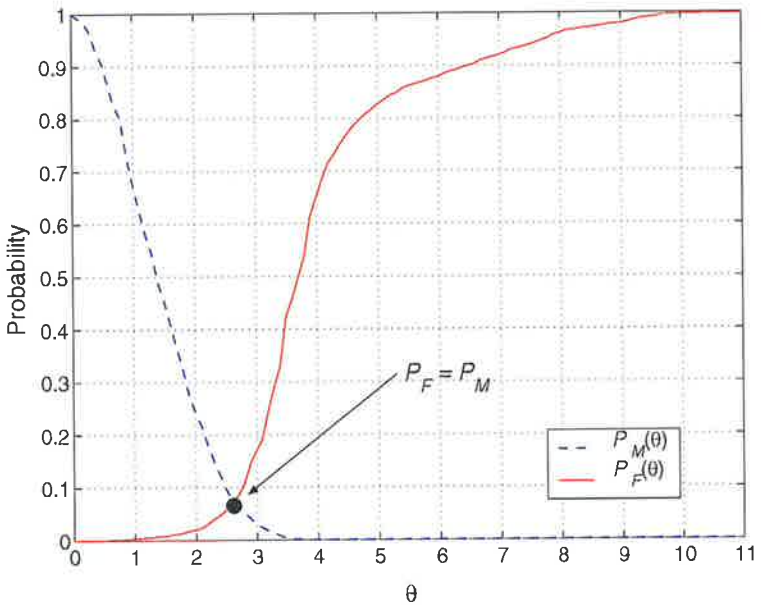


Figure 5.16: The probability of miss and false alarm as a function of θ.

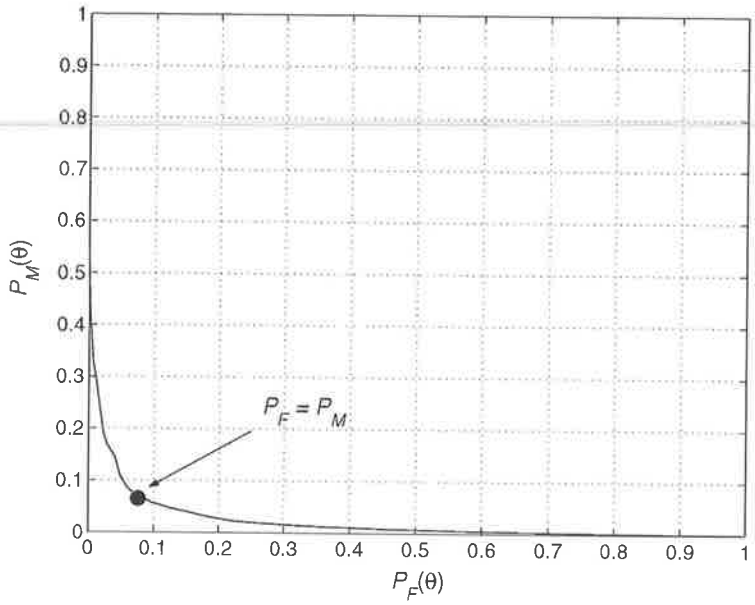


Figure 5.17: The receiver operating curve for the set of skin training pixels.

when $P(\omega_S)$ is unknown, for example in segmenting generic images or video sequences. The corresponding skin-color region in the CbCr plane is shown in Figure 5.18.

The following is a summary of the procedure used to derive the segmentation threshold based on the minimax test:

1. Find an expression for the probability of classification error as a function of $P(\omega_S)$ and θ.

2. Evaluate $P_D(\theta)$ and $P_F(\theta)$ for the set of training images.

3. Derive the segmentation threshold by solving $P_M(\theta) = P_F(\theta)$ for θ.

5.5 Simulation Results and Discussions

This section presents the simulation results for the skin-color segmentation algorithm. The proposed algorithm is intended to work on a range of skin colors. To this end, the test set was chosen to contain people of European, Asian and African descent. The results are presented in two sections. In the first section, skin-color segmentation results for still images

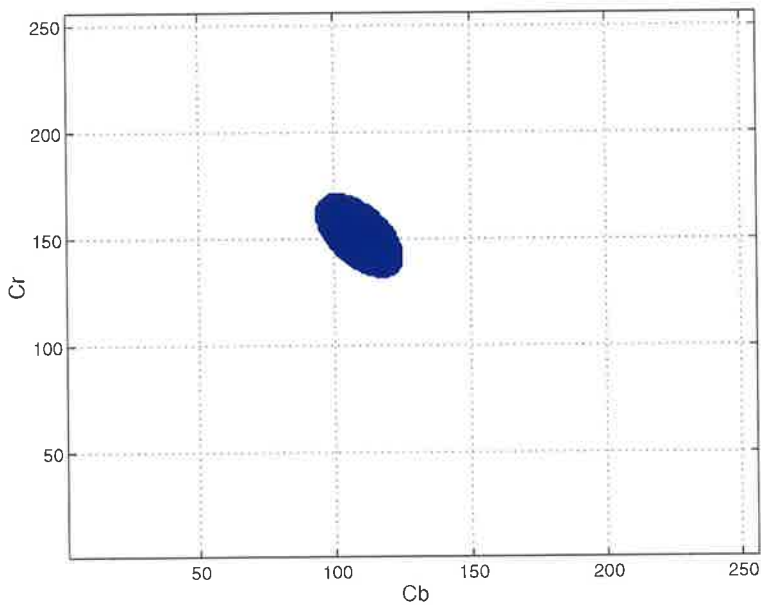


Figure 5.18: Skin-color region in the CbCr plane when $\tau = 2.6$.

are presented.[2] In the second section, results for the *Irene* and *Silent* video sequences are presented.

5.5.1 Performance Evaluation

Researchers in the field of skin segmentation usually provide a qualitative (i.e., visual) evaluation of their segmentation results. In common with other researchers, we will also provide a qualitative evaluation of our results. However, due to the lack of space, it is not possible to include a large number of images in the thesis. Therefore, quantitative evaluation is also provided. Quantitative evaluation is intended to give the reader an insight into the performance of the algorithm without the need to include a large number of images in the thesis. To quantitatively evaluate the accuracy of the proposed segmentation algorithm, each image (or frame) was manually segmented into skin and non-skin classes.[3] The manually segmented images serve as a reference (i.e., benchmark) to which the automatically segmented images are compared. The false alarm rate (R_F) and miss rate (R_M) are evaluated for each image.

[2]These images do not include any of the training images.

[3]The manual segmentation of the training images, test images, and the video sequences took four days to accomplish.

Image	False Alarm Rate (%)	Miss Rate (%)
John	3.2	0.2
Alex	1.3	7.8
Latiena	7.6	8.2

Table 5.2: Miss and false alarm rates for the *John*, *Alex*, and *Latiena* images.

R_F and R_M are given by

$$R_F = \frac{\textit{Number of false alarm pixels}}{\textit{Number of non-skin pixels}} \times 100, \tag{5.24}$$

and

$$R_M = \frac{\textit{Number of miss pixels}}{\textit{Number of skin pixels}} \times 100. \tag{5.25}$$

R_F and R_M are expressed as a percentage. The concepts of false alarm and miss are illustrated in Figure 5.10.

5.5.2 Still Images

Testing was carried out on 100 test images of different subjects, body poses, lighting conditions, and background complexities. The images were obtained either from the internet, or with a digital camera. Three images from the test set are shown in Figure 5.19. The *John* and the *Latiena* images were obtained from the world wide web. The *Alex* image was taken with a digital camera. Note that the images have varying degrees of background complexity. Since $P(\omega_S)$ is unknown, the segmentation threshold was set to $\tau = 2.6$.

The results, shown in Figure 5.19, indicate that the skin regions in each image have been effectively segmented. This demonstrates that the chrominance distributions are consistent across each race. The false alarm and miss rates are given in Table 5.2. The false alarm rates for the *John* and *Alex* images are both low. It is slightly higher for the *Latiena* image since parts of the subject's clothing and hair have been classified as skin. The miss rate for the *John* image is the lowest, and higher for the *Alex* and *Latiena* images. This is partly due to the strong shadow cast under the chin area of both subjects.

(a) *John*

(b) *Alex*

(c) *Latiena*

Figure 5.19: Results of the segmentation algorithm on three images depicting people of different descent. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

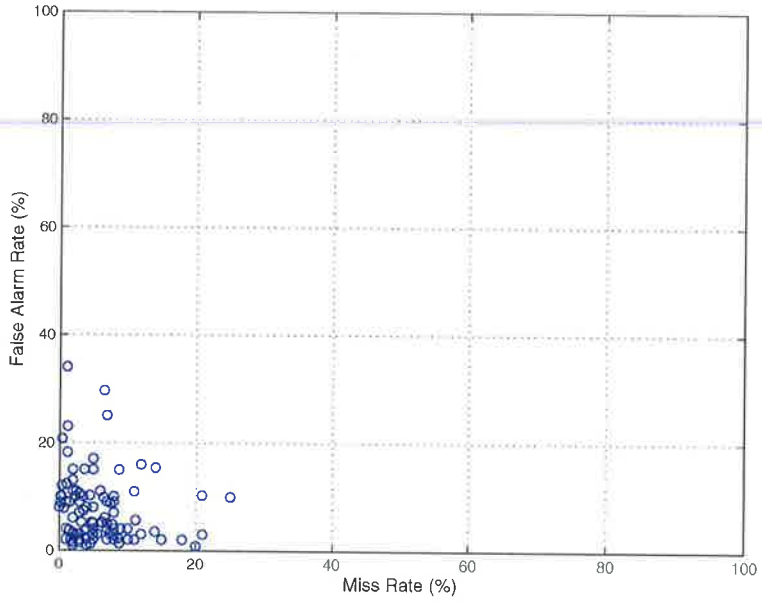


Figure 5.20: False alarm versus miss rates for 100 different images.

The average false alarm and miss rates for the 100 test images were 7.5% and 5.8%, respectively. The low average miss rate demonstrates the effectiveness of the proposed classifier in detecting skin-color pixels. Moreover, we have found that the proposed skin-color model is immune to moderate illumination changes and shading, as those conditions do not significantly alter the chrominance characteristics of the skin-color model. The relatively higher average false alarm rate is due to the inability of the algorithm to distinguish between actual skin and background objects with skin-color appearance. The false alarm rate can be reduced by postprocessing tasks. The scatter plot of miss rate versus false alarm rate is shown in Figure 5.20. Note that the points are concentrated near the lower left-hand corner of the scatter plot. Additional simulation results are provided in Appendix C.

5.5.3 Video Sequences

Figures 5.21 and 5.22 show the segmentation results for six consecutive frames of the *Carphone* and *Foreman* video sequences, respectively. The *Carphone* and *Foreman* sequences were considered as generic, and thus the threshold was set to $\tau = 2.6$.

Our aim, with respect to the *Carphone* and *Foreman* video sequences, is to segment the face regions. For the *Foreman* sequence, the subject's face has been effectively segmented,

however some background regions have also been segmented as skin. For the *Carphone* sequence, although the amount of false alarm regions is less, small regions of the subject's neck and face have been classified as non-skin. These observations are reflected in the plots of Figures 5.25(a) and (b). The plots show the the false alarm and miss rates for 60 consecutive frames of the *Carphone* and *Foreman* sequences. Notice that the false alarm rate for the *Foreman* sequence is consistently higher than that of the *Carphone* sequence. The opposite is true for the miss rates. The miss rate for the *Carphone* sequence is higher than that of the *Foreman* sequence. The average false alarm and miss rates for the 60 frames tested are given in Table 5.3.

We now turn our attention to sign language video sequences. The segmentation results for six consecutive frames of the *Silent* and *Irene* video sequences are shown in Figures 5.23 and 5.24, respectively. For still images (Section 5.5.2) and the *Carphone* and *Foreman* video sequences , we assumed that $P(\omega_S)$ is unknown, and the segmentation threshold was selected ($\tau = 2.6$) to minimize the maximum error probability regardless of any changes in $P(\omega_S)$ (Figure 5.15). In Section 5.4.3, we estimated the skin and non-skin *a priori* probabilities for a sign language video frame as $P(\omega_S) = 0.15$ and $P(\omega_{\bar{S}}) = 0.85$, respectively. Since we can reasonably estimate $P(\omega_S)$ for sign language video frames, the segmentation threshold is set to $\tau = 2.1$ (Figure 5.12).

The face and hands of the subjects in the *Silent* and *Irene* sequences have been effectively segmented, however some background regions have also been detected as skin. False alarm regions can be removed by change detection and connected components analysis, which will be described in the next two chapters.

Figures 5.26(a) and (b) show the false alarm and miss rates for 60 consecutive frames of the *Silent* and *Irene* sequences, respectively. The false alarm and miss rates for both sequences are reasonably low. Note that both sequences have complex background scenes. The average false alarm and miss rates for the first 60 frames are given in Table 5.3. We observe that for the *Silent* sequence, the average false alarm rate is lower than the average miss rate. The strong shadow cast under the chin of the subject in the *Silent* sequence contributes to the miss rate. For the *Irene* sequence, the average false alarm rate is higher than the average miss rate. The main contributors to the false alarm rate are two large background regions, at the bottom of each frame, detected as skin. The *Silent* sequence has a higher average miss

rate than the *Irene* sequence. In contrast, the *Irene* sequence has a higher average false alarm rate. Overall, the average false alarm and miss rates are low enough to indicate that the face and the hands can be extracted from the sequence reasonably well.

Sequence	Average R_F (%)	Average R_M (%)
Carphone	2.8	9.9
Foreman	9.7	0.5
Silent	4.3	6.8
Irene	6.2	3.2

Table 5.3: Average miss and false alarm rates for 60 consecutive frames of the *Carphone*, *Foreman*, *Silent*, and *Irene* sequences.

5.6 Summary

Our skin-color segmentation algorithm was presented in this chapter. Training images of different subjects, body poses, lighting conditions, and background complexities, were manually segmented into skin and non-skin classes. The skin-color distribution in the CbCr plane was modeled as a bivariate normal distribution. Pixels were classified as skin or non-skin based on their Mahalanobis distance. A segmentation threshold was derived for the classifier.

Simulation results for both still images and video sequences demonstrated that the algorithm is capable of segmenting skin regions quite effectively. The algorithm was found to be tolerant to different lighting conditions, and skin-color of people of different ethnicity.

Figure 5.21: Results of the skin-color segmentation algorithm for the *Carphone* sequence. Left column: Frames 10 to 15. Center column: SDM. Right column: Identified skin pixels.

Figure 5.22: Results of the skin-color segmentation algorithm for the *Foreman* sequence. Left column: Frames 1 to 6. Center column: SDM. Right column: Identified skin pixels.

Figure 5.23: Results of the skin-color segmentation algorithm for the *Silent* sequence. Left column: Frames 22 to 27. Center column: SDM. Right column: Identified skin pixels.

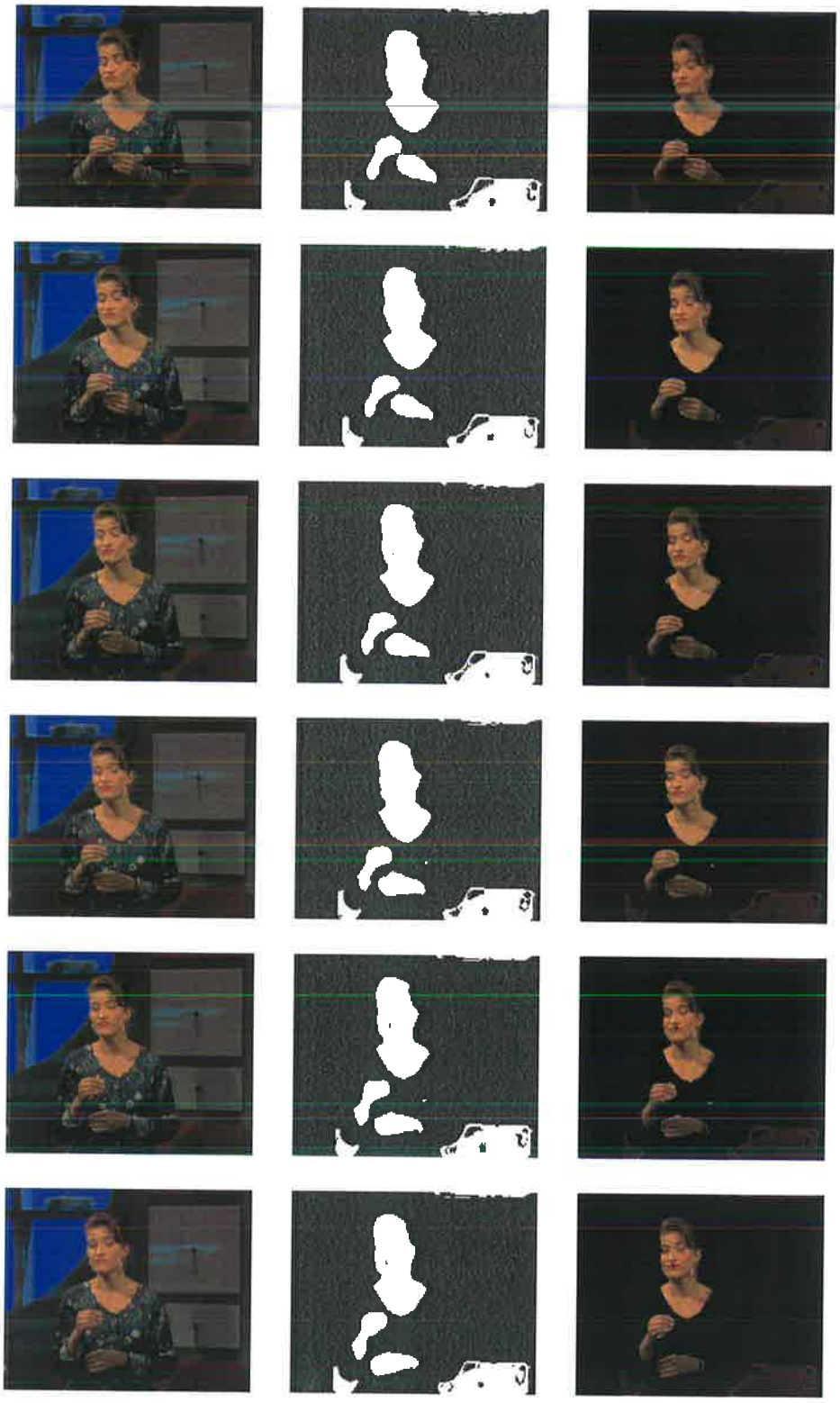
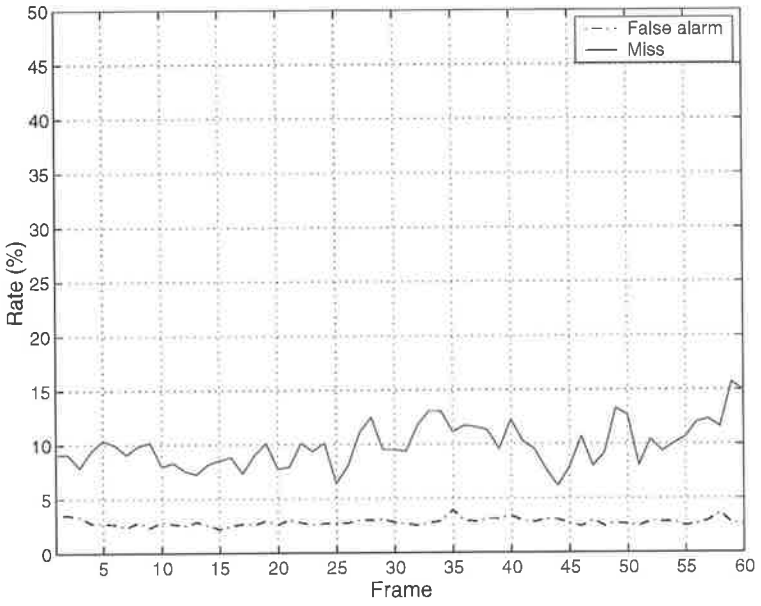
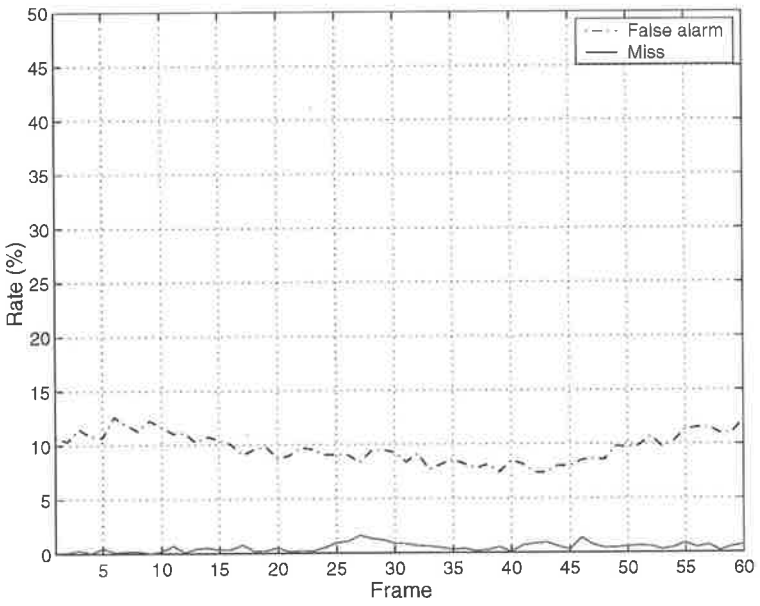


Figure 5.24: Results of the skin-color segmentation algorithm on the *Irene* sequence. Left column: Frames 12 to 17. Center column: SDM. Right column: Identified skin pixels.

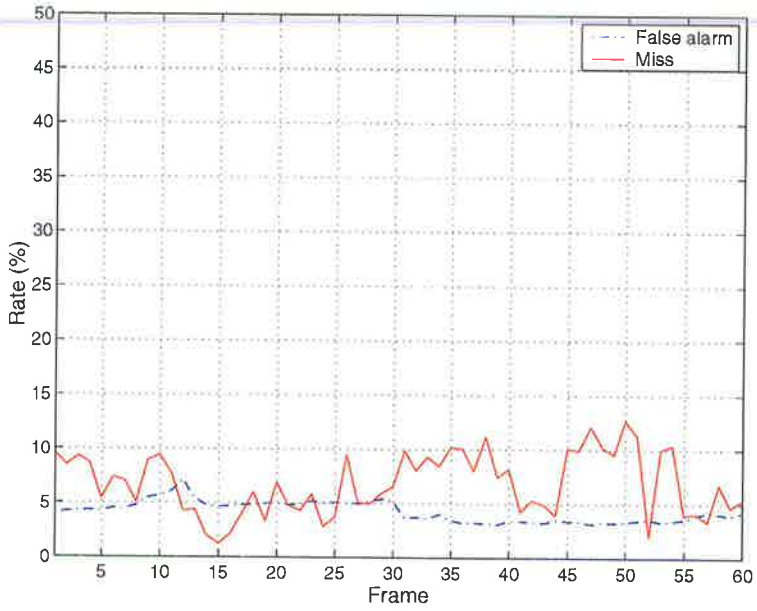


(a)

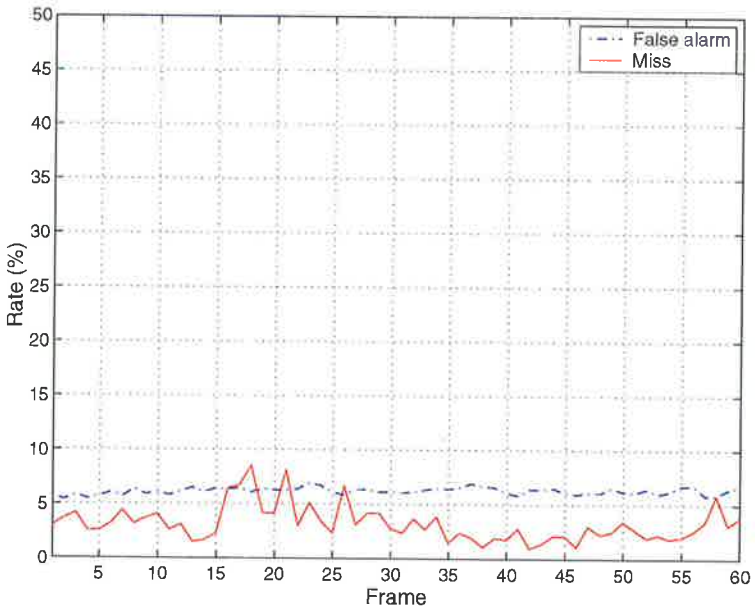


(b)

Figure 5.25: Miss and false alarm rates for 60 consecutive frames of the (a) *Carphone* sequence, and (b) *Foreman* sequence.



(a)



(b)

Figure 5.26: Miss and false alarm rates for 60 consecutive frames of the (a) *Silent* sequence, and (b) *Irene* sequence.

Chapter 6

Statistical Change Detection

"Nothing endures but change."

- Heraclitus

In this chapter, we present a new statistical change detection technique based on the F test and block-based motion estimation. Change detection is employed for segmenting video frames into "changed" and "unchanged" regions with respect to the previous frame. The unchanged regions denote the stationary background, while the changed regions denote the moving and occlusion regions. An advantage of the proposed change detection technique is that it is automatic in the sense that no manual input from a user is required.

A literature survey of previous research is provided in Section 6.2, the proposed statistical change detection method is discussed in Section 6.3, simulation results are presented in Section 6.4, and the chapter is summarized in Section 6.5.

6.1 Introduction

We derive temporal information by segmenting a video sequence into moving and stationary regions. Since we are only interested in determining which regions in a frame have changed due to motion, the direction of object motion and its velocity are not required. Therefore, motion estimation and optical-flow methods, cited extensively in the literature, provide excessive information and would not necessarily improve the accuracy of our hand and face segmentation algorithm. Besides, motion estimation and optical-flow methods are computationally expensive, and would inhibit real-time segmentation.

If the background is stationary (i.e., no camera panning or zooming)[1] and there are no changes in the image acquisition parameters (i.e., camera focus etc.), taking the color or gray-level (i.e., the luminance component) difference between two frames is an efficient way to detect changed regions with respect to the previous frame. The gray-level difference frame (DF) between frames $F(x, y, k)$ and $F(x, y, k-1)$ is defined as:

$$DF_{k,k-1}(x, y) = F(x, y, k) - F(x, y, k-1). \tag{6.1}$$

Assuming that the illumination remains constant from one frame to the next, pixel locations where $DF_{k,k-1}(x, y)$ differ from zero indicate objects that are moving or changing their shape. Moreover, the intensity at each pixel in the current frame is either a displaced value from the previous frame (i.e., a moving pixel), the same value as in the previous frame (i.e., a stationary pixel), or an uncovered-background value (i.e., an uncovered-background pixel). These different pixel types are depicted in Figure 6.1. Furthermore, unless the objects are textured, only the boundaries of moving objects can be observed, and not the objects themselves. In sign language video, the moving eyes, nose, mouth and fingers add texture to the face and hands.

Non-zero differences can also occur due to camera or quantization noise. Figures 6.2(a) and (b) show frames 13 and 14 (grayscale) of the *Silent* sequence, respectively. The binary difference frame (BDF) is shown in Figure 6.2(c), and is defined as

$$BDF_{k,k-1}(x, y) = \begin{cases} 1, & \text{if } |DF_{k,k-1}(x, y)| > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{6.2}$$

[1] This is usually the case for sign language video sequences.

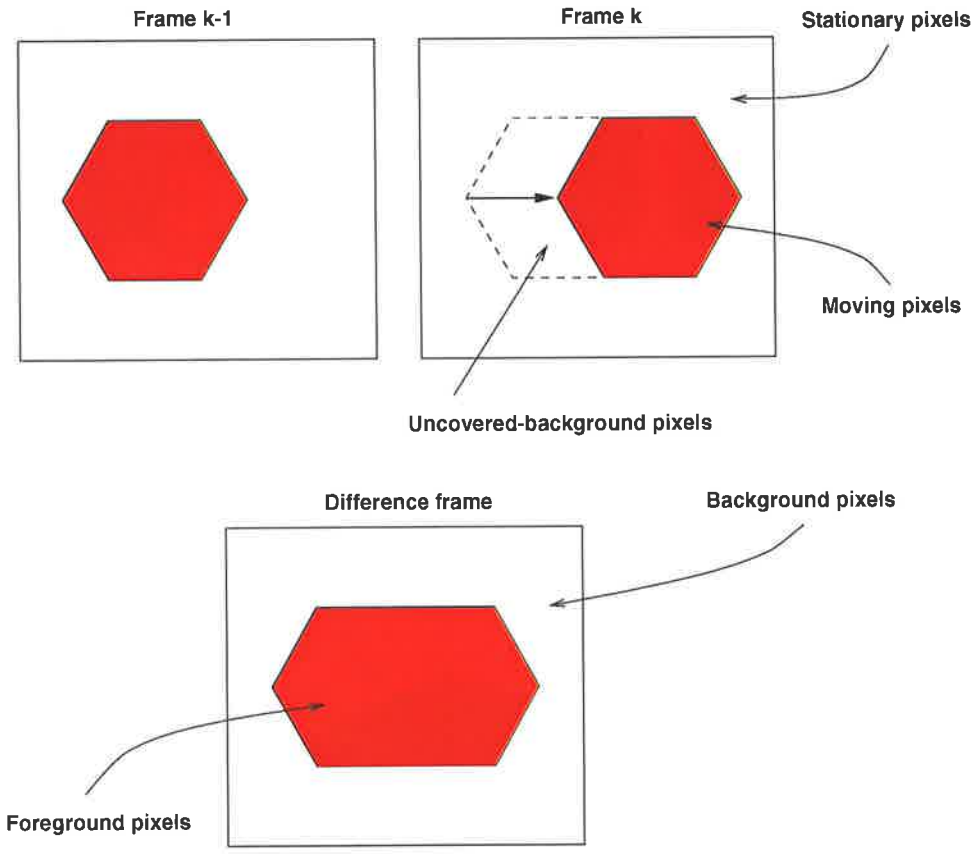
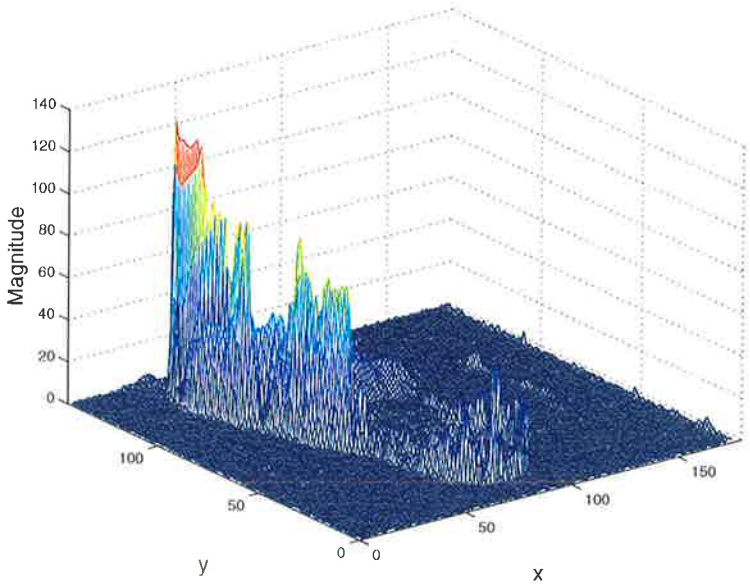


Figure 6.1: Different pixel types inherent in object motion.

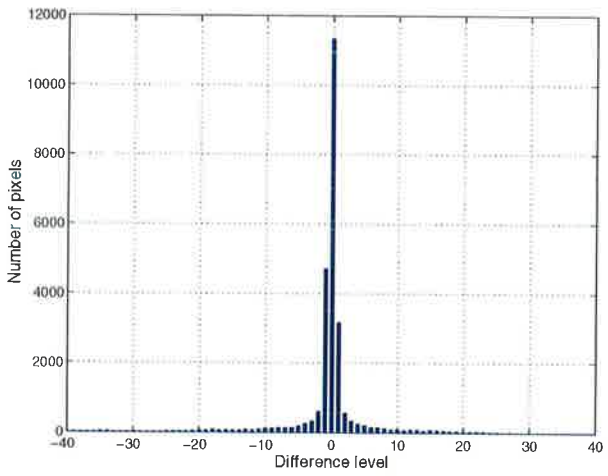
That is, a binary "1" is allocated to non-zero differences, and a binary "0" is allocated to zero differences. Since the background is stationary, one would expect the background to consist entirely of binary "0" pixels, however due to noise, the background appears very noisy. The notion of noise is also illustrated in Figure 6.2(d), which shows the 3D plot of the absolute difference levels in the difference frame. The moving pixels have a much higher difference level than the stationary pixels. The histogram of the difference levels is shown in Figure 6.2(e).

The image noise is usually modeled as an additive zero-mean normal distribution (i.e., additive white Gaussian noise) [KCK+99, Tek95, HMB00a, HMB00b]. Note that the histogram in Figure 6.2(e) appears to follow a normal distribution, at least near zero. This gives weight to the assumption that the background pixels follow a zero-mean normal distribution. The objective of change detection is to distinguish between temporal variations caused by noise from those caused by object motion. We refer to changed pixels as foreground pixels

(a) (b) (c)



(d)



(e)

Figure 6.2: Example of frame differencing. Frames 13 (a) and 14 (b) of the *Silent* sequence, (c) BDF, (d) 3-D plot of the absolute difference levels, and (e) histogram of the difference levels.

and stationary pixels as background pixels. In order to distinguish between foreground and background pixels, the difference frame can be thresholded to form a change detection mask (CDM). Thresholding may be viewed as an operation that involves tests against a function of the form:

$$\varphi = \varphi(x, y, p_{k,k-1}(x, y), DF_{k,k-1}(x, y)), \tag{6.3}$$

where $p_{k,k-1}(x, y)$ denotes some local property of the difference pixel in $DF_{k,k-1}(x, y)$ (e.g., average gray-level or median). With φ as threshold, the change detection mask is defined as:

$$CDM_{k,k-1}(x, y) = \begin{cases} 1, & \text{if } |DF_{k,k-1}(x, y)| > \varphi \\ 0, & \text{otherwise.} \end{cases} \tag{6.4}$$

Foreground pixels are assigned a binary "1" and background pixels are assigned a binary "0". In practice, thresholding may still yield isolated 1's in the CDM, which can be eliminated by post-processing; for example, forming 4- or 8-connected components, and discarding any component with a predetermined number of pixels [Tek95]. The difficulty with discarding components if they are below a certain size is deciding what size constitutes a foreground region, and what size constitutes residual noise.

In the next section, we present a literature survey of common change detection methodologies.

6.2 Previous Research

The change detection method described in [GW92] uses memory to ignore changes that occur only sporadically over a frame sequence and can therefore be attributed to random noise. An accumulative difference frame is formed by comparing a reference frame with every subsequent frame in the sequence. A counter for each pixel location is incremented every time a difference occurs at that pixel location between the reference and a frame in the sequence. Thus when the kth frame is being compared with the reference, the entry in a given pixel of the accumulative frame gives the number of times the gray-level at that position was different from the corresponding pixel value in the reference frame.

Rosin [Ros97, Ros98] described four different methods for selecting change detection thresholds. Either the noise or signal (i.e., foreground pixels) is modeled, and the model

covers either the spatial or intensity distribution characteristics. The methods are: (a) a normal model is used for the noise intensity distribution; (b) signal intensities are tested by making local intensity distribution comparisons in the two frames; (c) the spatial properties of the noise are modeled by a Poisson distribution; and (d) the spatial properties of the signal are modeled as a stable number of regions (or stable Euler number). In method (a), the noise is modeled as a zero mean normal distribution. The variance of the noise is estimated by a robust estimation technique based on the least median of squares (LMS) method [RL87]. A suitable threshold is chosen for an acceptable proportion of false foreground pixels (i.e., false alarms). Note that the decision is based on the difference level of a single pixel, not the statistical properties of an observation window. We have tested the LMS method on a number of synthetic (i.e., noise variance is known *a priori*) and real video sequences, and found that the LMS method does not accurately estimate the noise variance, especially when the noise variance is low. In method (b), a non-parametric method is used so that no assumptions about the intensity distributions need to be made. The Kolmogorov-Smirnov and the Cramer-von Mises tests are used to compare the pixel intensities in two observation windows of the original (pre-differenced) frames. The spatial distribution of the noise is modeled as a Poisson distribution in method (c). Since a Poisson distribution has its mean equal to its variance, the ratio of the sample variance to the sample mean is a natural test for that distribution, and is called the relative variance. The threshold is chosen such that the relative variance is maximized, thereby maximizing "clumpiness" (regions of change) and minimizing the Poisson distribution (noise). In method (d), a frame's *Euler number* is used to select a suitable threshold. The Euler number is the number of regions in the BDF minus the number of holes in those regions. At low threshold values, there will be many regions and holes in the BDF caused primarily by the noise, and the Euler number will alter rapidly with threshold. At high threshold values, there will be few regions in the BDF, and the Euler number will be stable. Therefore, a suitable threshold is when the Euler number becomes stable.

Mech and Wollborn [MW97] employed change detection to segment moving objects in video sequences. Initially, a CDM is generated by taking the difference between two successive frames using a global threshold. The CDM is then refined in an iterative relaxation method that uses a locally adaptive threshold to enforce spatial continuity. Temporal stabil-

ity is increased by incorporating a memory such that each pixel is labeled as changed if it belonged to an object at least once in a certain number of previous CDMs. The simplification step involves the morphological closing operator, and the elimination of small regions to obtain the final CDM.

An automatic change detection algorithm was proposed by Neri *et al.* [NCRT98]. In the preliminary stage, potential foreground regions are detected by applying a higher order statistics test to a group of frame differences. The non-zero values in the difference frames are either due to noise or moving objects, and the noise is assumed to be Gaussian in contrast to the moving objects, which are highly structured. In the case of moving background, the frames are first aligned by motion compensation. For all difference frames, the zero-lag fourth order moments are calculated because of their capability to suppress Gaussian noise. These moments are then thresholded, resulting in a preliminary segmentation map containing moving objects and uncovered background. To identify the uncovered background, the motion analysis stage computes the displacement of pixels that are marked as changed. The displacement is calculated at different lags from the fourth-order moment maps by block-matching. If the displacement of a pixel is estimated at different lags, it is classified as background, otherwise it is classified as foreground. Finally, a regularization phase applies morphological opening and closing operators to achieve spatial continuity and to remove small holes inside moving objects of the segmentation map.

Meier and Ngan [MN99] employed change detection to detect independently moving objects in their motion segmentation algorithm. Connected components analysis is used to suppress noise in the BDF since pixels belonging to moving objects are connected, while noisy pixels form isolated clusters. If the size of a connected component exceeds a threshold, it is assumed that the connected component belongs to a moving object. Unfortunately, this method will only work if the amount of noise in the BDF is little. For example, it would be difficult to accurately detect any foreground regions in the BDF shown in Figure 6.2(c).

A motion detection algorithm based on spatial-temporal entropy was presented in [MZ01]. The color variation in successive frames is measured and a spatio-temporal entropy image (STEI) is formed. Morphological operators are then employed to extract the moving objects from the STEI. The authors claim that since STEI is a statistical measurement of variation, the method is more robust to noise than methods based on change detection.

Aach *et al.* [AKM93] proposed several statistical change detection methods. In one proposal, the local sum of absolute differences is used as a test statistic. The local sum of pixel differences can be traced back to the assumption of Laplacian noise in the difference frame. In another proposal, the camera noise is modeled as a zero-mean normal distribution, and the chi-square test is used to detect changed regions in the difference frame. The chi-square test requires the variance of the background population σ_0^2 in DF. The background population variance is estimated offline for the used camera. Since the camera noise is uncorrelated between different frames, σ_0^2 is equal to twice the variance of the assumed Gaussian camera noise σ_c^2, i.e.,

$$\sigma_0^2 = 2\sigma_c^2. \tag{6.5}$$

A recursive method that automatically estimates the background population variance was proposed by Ziliani [Zil00]. First, σ_0^2 is estimated for the used camera, and then change detection is performed on the DF. The areas in the DF that are declared as background are used to estimate a new σ_0^2. This allows σ_0^2 to be automatically adapted to the Gaussian noise. Unfortunately, it is difficult to estimate the variance of the Gaussian camera noise for the used camera system.

Noting the difficulty of estimating the variance of the background population, Kim *et al.* [KCK+99] proposed a change detection method based on the F test. Instead of the background population variance, the F test requires a sample variance of the difference pixels in a background region. The authors did not explicitly explain how a background region in the DF can be selected. To make the change detection technique automatic in the sense that no manual manipulation is required, we employ block-based motion estimation to find difference pixels in a background region of DF.

6.3 Change Detection Based on the F Test

This section presents a method for thresholding the gray-level difference frame based on the F test [JW82, Hay88, ASW94, oST01]. In order to make change detection less sensitive to noise, thresholds are usually calculated based on the statistics of a small region in DF, rather than the difference level of a single pixel. Therefore, the hypothesis test is based on the statistical properties of the samples in a square observation window, W. To form a CDM, a

binary "1" is allocated to the center pixel in W if the null hypothesis (i.e., the hypothesis that the difference pixels in W are drawn from the background population) is rejected, otherwise a binary "0" is allocated. The use of a window for thresholding corresponds to applying a low-pass filter to the difference frame. This will cause a blurring effect in the CDM because changes in the observation window are attributed to the center pixel in W, regardless of precisely where the changes occur. Figure 6.3 shows the effect of increasing the size of W. As the size of W is increased, the blurring effect becomes more profound. The blurring effect can be reduced by the Markov random field based refining method described in [AKM93]. However, we have found that the blur does not adversely affect the outcome of our segmentation results if a window size of 3×3 pixels is used (for frames of size QCIF; see Appendix A).

Another parameter that must be considered is the significance level, α. The significance level is the probability of detecting background pixels as foreground. The value of α is critical, since too high a value will swamp the CDM with spurious changes, while too low a value will suppress significant changes. A significance level of 0.01 was found to be appropriate.

6.3.1 The F Test

The F test is used to test if the standard deviations of two populations are equal. Suppose that the difference pixels in W are drawn from a normal population with variance σ_1^2. The F hypothesis test is defined as:

$$\begin{aligned} H_0 &: \sigma_0^2 = \sigma_1^2, \\ H_1 &: \sigma_0^2 < \sigma_1^2. \end{aligned} \tag{6.6}$$

The null hypothesis, H_0, implies that σ_0^2 and σ_1^2 are equal, while the alternative hypothesis, H_1, implies that σ_0^2 is less than σ_1^2. The hypothesis test is based on the notion that the intensity variation induced by a moving object is greater than that of the background due to the higher intensity gradient at the edge and within a moving object.

Let S_0^2 (respectively, S_1^2) be the unbiased estimator of σ_0^2 (respectively, σ_1^2), and n_0 (re-

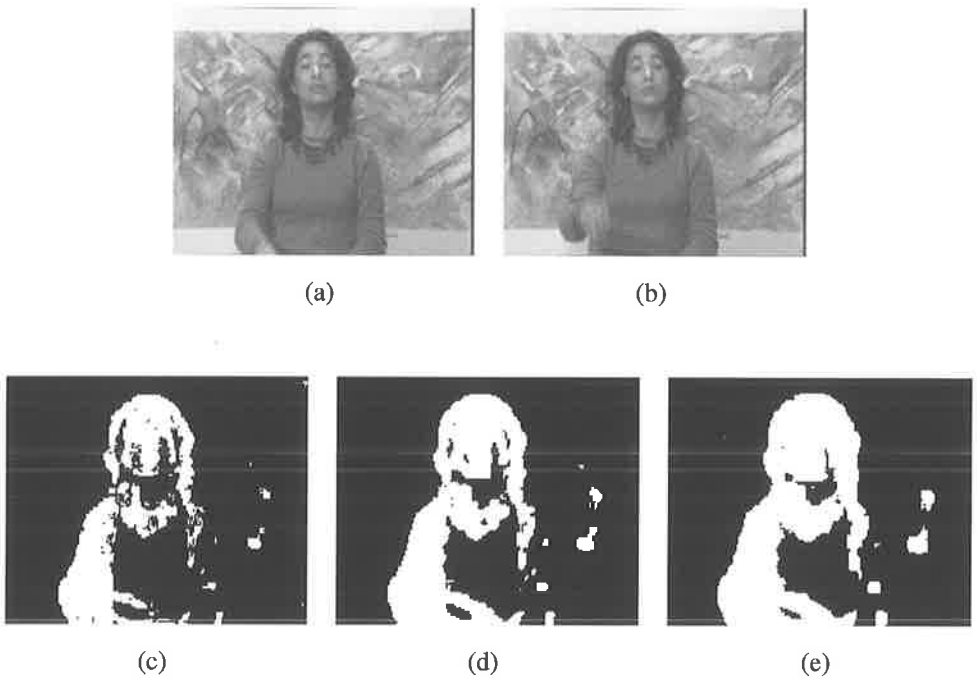


Figure 6.3: The effect of increasing the size of W. Frames 14 (a) and 15 (b) of the *Irene* sequence, (c) $W = 3 \times 3$ pixels, (d) $W = 5 \times 5$ pixels, and (d) $W = 7 \times 7$ pixels.

spectively, n_1) be the sample size. If the null hypothesis is true, then the ratio

$$F = \frac{S_1^2}{S_0^2} \tag{6.7}$$

has an F-distribution with $n_0 - 1$ and $n_1 - 1$ degrees of freedom [Wei99]. The F-distribution is the ratio of two independent chi-square random variables, each divided by its degrees of freedom $n_0 - 1$ and $n_1 - 1$, i.e.,

$$F = \frac{\chi^2_{(n_1-1)}/(n_1 - 1)}{\chi^2_{(n_0-1)}/(n_0 - 1)}, \tag{6.8}$$

where

$$\chi^2_{(n_0-1)} = (n_0 - 1)\frac{S_0^2}{\sigma_0^2} \tag{6.9}$$

and

$$\chi^2_{(n_1-1)} = (n_1 - 1)\frac{S_1^2}{\sigma_1^2} \tag{6.10}$$

are chi-square distribution with $n_0 - 1$ and $n_1 - 1$ degrees of freedom, respectively.

Since hypothesis test (6.6) is an upper one-tailed test, the null hypothesis is rejected if $F > F_{(\alpha,n_1-1,n_0-1)}$, where $F_{(\alpha,n_1-1,n_0-1)}$ is the critical value of the F-distribution with $n_0 - 1$ and $n_1 - 1$ degrees of freedom, and a significance level of α. Note that the F test does not require the background population variance.

The sample variance of the background population must be derived from an area in DF that does not contain any moving regions. To this end, we advocate a method based on block-based motion estimation. The procedure is described in the next section.

6.3.2 Estimation of the Background Sample Variance

Block-based motion estimation techniques are commonly used in video coding schemes to reduce temporal redundancies between frames. Indeed, block-based motion estimation is a core component of the H.261, H.263, MPEG-1, MPEG-2, and MPEG-4 video coding standards. Given a reference frame (frame $k - 1$) and an $N \times N$ block in the current frame (frame k), the objective of block-based motion estimation is to seek the $N \times N$ block in the reference frame that best matches (according to a given cost function) the characteristics of the block in the current frame. The relative displacement between a block in the current frame and a block in the reference frame is described by a motion vector (v_x, v_y). To reduce

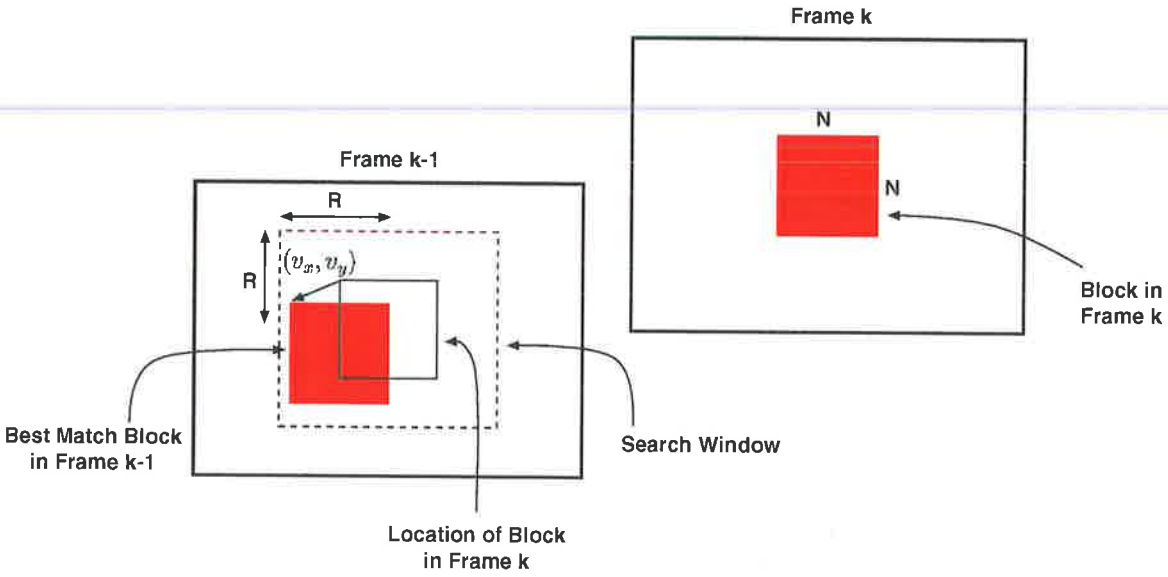


Figure 6.4: Block-based motion estimation.

the computational complexity, the search is usually restricted to a search region around the original location of the block in the current frame (Figure 6.4). The full search algorithm exhaustively searches the entire search window in the reference frame to find the optimal match. However, this is at the expense of higher computational cost. Various other sub-optimal block-based motion estimation techniques have been proposed [KIH+81, GM90, CCJC91, LZL94, PM96, TRRK98, HMB99] that aim to reduce the computational cost.

Let the search range be $\pm R$ pixels in both horizontal and vertical directions. With a stepsize of one pixel, the total number of candidate matching blocks in the search window is $(2R+1)(2R+1)$. Our block-based motion detection strategy is conceptually simple. We first choose a $N \times N$ block in frame k, and define a search window in frame $k-1$. An $N \times N$ block located at the upper border of a frame is usually chosen, since we do not expect motion at the upper border.[2] Since the probability of motion at the upper border is low, the value of R is set to 7. Usually a value of $R = 15$ is chosen for head and shoulder type scenes [BK95], however a large R value would substantially increase the computational cost of motion estimation, i.e., from 225 searches for $R = 7$ to 961 searches for $R = 15$. The full search algorithm is then employed to find the best matching block in frame k. The

[2]Note that if the block is at the border of the frame, the number of candidate matching blocks is reduced.

procedure is depicted in Figure 6.4. The matching of the blocks can be quantified according to various criteria including the maximum cross-correlation, the minimum mean square error, the minimum mean absolute error, and maximum matching pixel count. The mean absolute error (MAE) was chosen as the matching cost function because it is a popular choice for hardware implementation [Tek95]. The MAE is defined as:

$$MAE(v_x, v_y) = \frac{1}{N^2} \sum_{(v_x,v_y)\in\mathcal{B}} |F(x, y, k) - F(x + v_x, y + v_y, k - 1)|, \tag{6.11}$$

where $\mathcal{B}$ denotes an $N \times N$ block, for a set of candidate motion vectors (v_x, v_y). The estimate of the motion vector is taken to be the value of (v_x, v_y) that minimizes the MAE, i.e.,

$$[\hat{v}_x\ \hat{v}_y]^T = \arg \min_{(v_x,v_y)} MAE(v_x, v_y). \tag{6.12}$$

If the motion vector is zero, i.e., $[\hat{v}_x\ \hat{v}_y]^T = [0\ 0]^T$, we assert that the corresponding $N \times N$ block in DF does not contain any foreground pixels. However, if the motion vector is non-zero, we assume that the corresponding block in DF contains foreground pixels. If the motion vector is non-zero, another block in frame k is chosen, and the above procedure repeated. Since a block was chosen at the upper border of a frame where the probability of motion is low, motion estimation usually had to be performed only once (for block sizes of 8×8, see below). The procedure rarely had to be performed more than twice.

The assumption of a common displacement (v_x, v_y) for all pixels in the block implies that a local smoothness constraint is imposed on the motion vector field. The local smoothness constraint is only satisfied for small block sizes. The choice of the dimensions of the block is the result of tradeoffs among three conflicting requirements, specifically [BK95],

1. a small value for N is preferable, since the smoothness constraint would be easily met at this resolution;

2. a small value for N would reduce the reliability of the motion vector (v_x, v_y), since few pixels would participate in the matching process; and

3. fast algorithms for finding motion vectors are more efficient for larger values of N.

In block-based motion estimation, block-sizes of 4×4, 8×8, and 16×16 pixels are typically considered. In order to determine which block-size to employ, we performed full

search motion estimation on various video sequences and analyzed the resulting motion vectors. For a block-size of 4×4 pixels, we found that non-zero motion vectors often occur in regions where motion is not expected. An example is shown in Figure 6.5. The motion vectors (blue arrows) are superimposed on top of frame 11 of the *Salesman* sequence. Blue dots indicate a motion vector of $[0 \ 0]^T$ for that particular block (with reference to frame 12). In the *Salesman* sequence, the subject moves while the background is stationary. Even though the background is stationary, a significant proportion of the motion vectors in the background regions are non-zero. This is because small block-sizes reduce the reliability of the motion vectors. As a result, the motion estimation procedure may have to be performed a number of times. This adds to the computational cost of the change detection procedure and is undesirable.

In Figure 6.6, block-based motion estimation has been performed using block-sizes of 8×8 pixels. Notice that the proportion of non-zero motion vectors in the stationary regions is significantly less. The motion estimation results for block-sizes of 16×16 pixels are shown in Figure 6.7. We found that for block-sizes of 16×16 pixels, the motion vectors in moving regions are unreliable. In Figure 6.7, even though the hand is moving, the motion vector for the hand region indicates no motion.

Based on the above discussions, a block-size of $N \times N = 8 \times 8$ pixels (i.e., 64 samples) was employed to perform motion estimation. Therefore, from (6.7), the F-distribution has $n_0 - 1 = 8$ and $n_1 - 1 = 63$ degrees of freedom. For a significance level[3] of 0.01, and $n_0 - 1 = 8$ and $n_1 - 1 = 63$ degrees of freedom, the critical value of F is $F_{(0.01,63,8)} = 5.02$[4]. The following is a summary of the change detection procedure based on the F test.

1. Compute the background sample variance S_0^2 using block-based motion estimation.
2. Compute the sample variance S_1^2 of the difference pixels in observation window W.
3. Compute the test statistic (6.7), where the degrees of freedom are 8 and 63.
4. If $F > 5.02$, the center pixel in W is declared a foreground, otherwise it is declared a background.

[3]Recall that the significance level is the probability of detecting background pixels as foreground.

[4]The critical value was derived using the Matlab "finv.m" function.

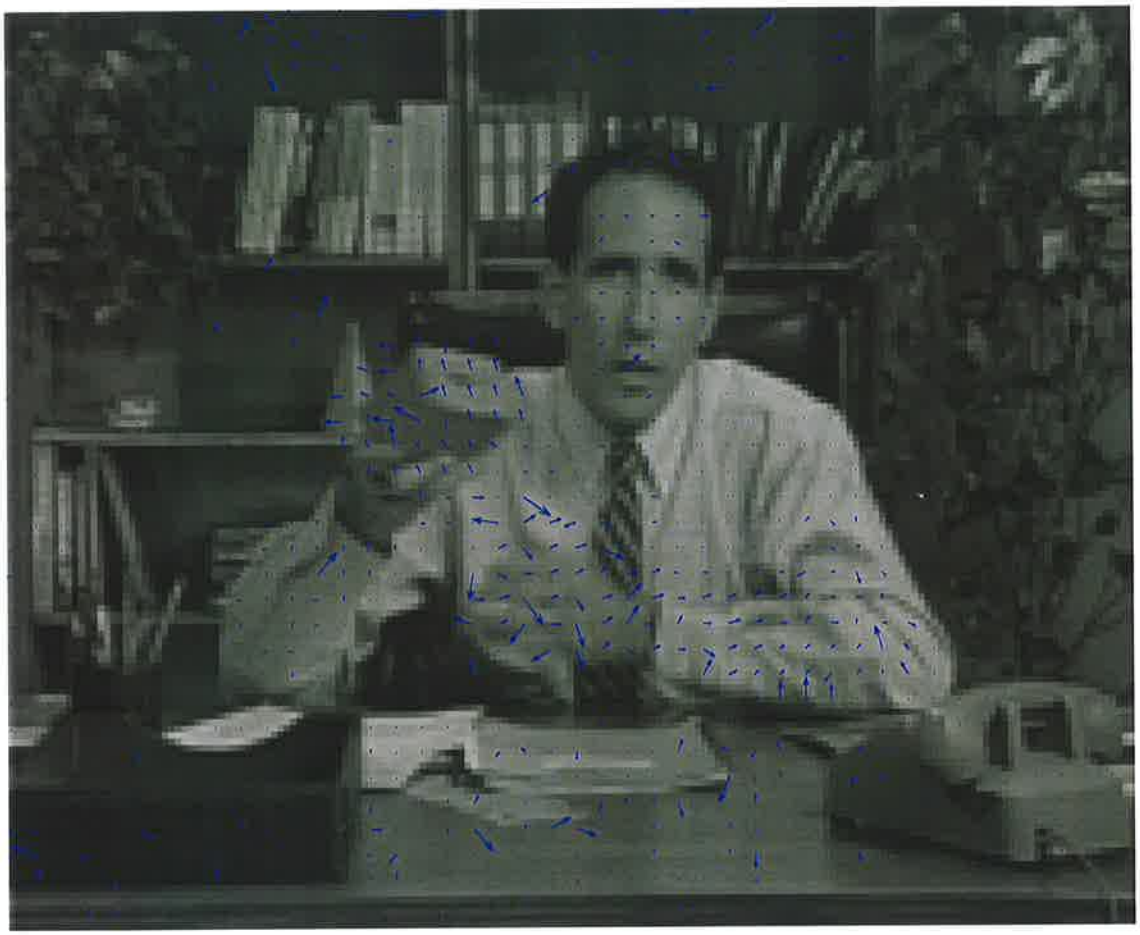


Figure 6.5: Block-based motion estimation between frames 11 and 12 of the *Salesman* sequence using block-sizes of 4×4 pixels.

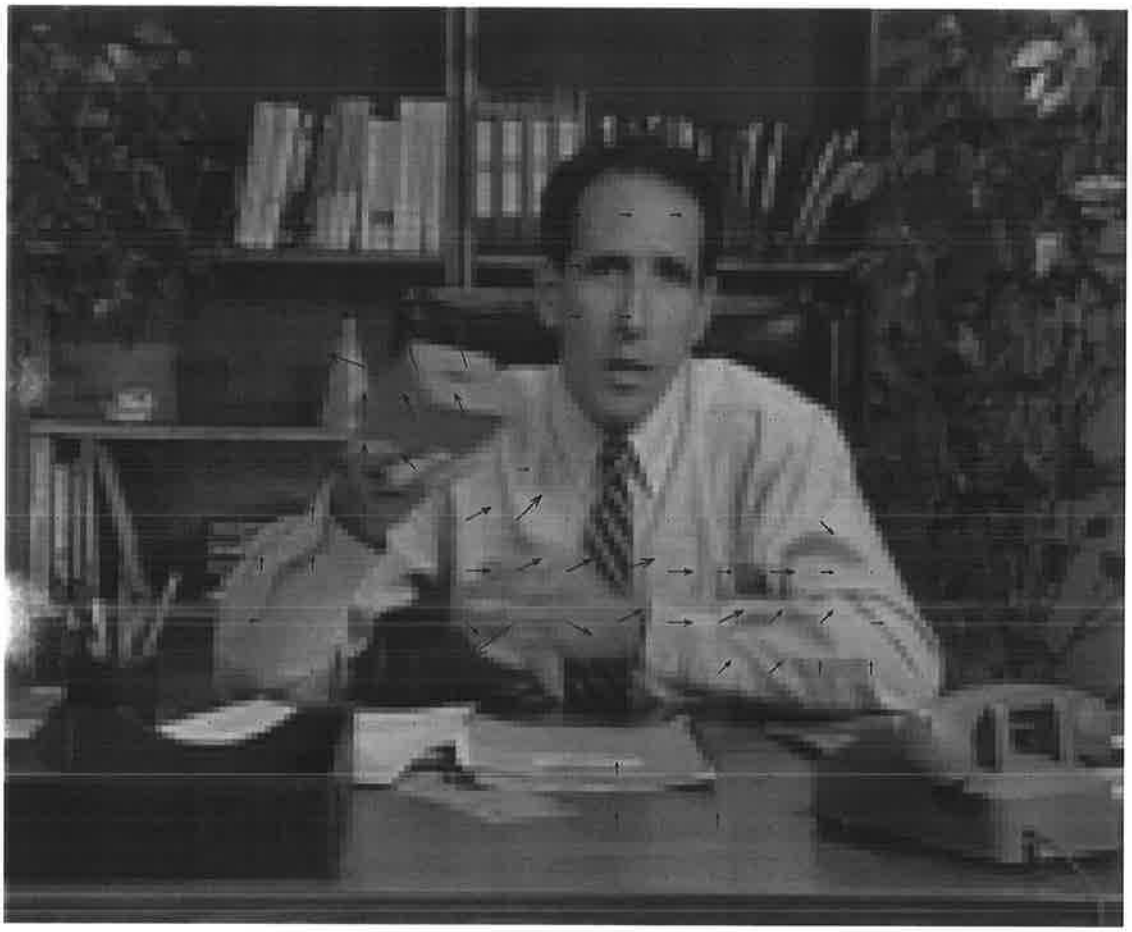


Figure 6.6: Block-based motion estimation between frames 11 and 12 of the *Salesman* sequence using block-sizes of 8×8 pixels.

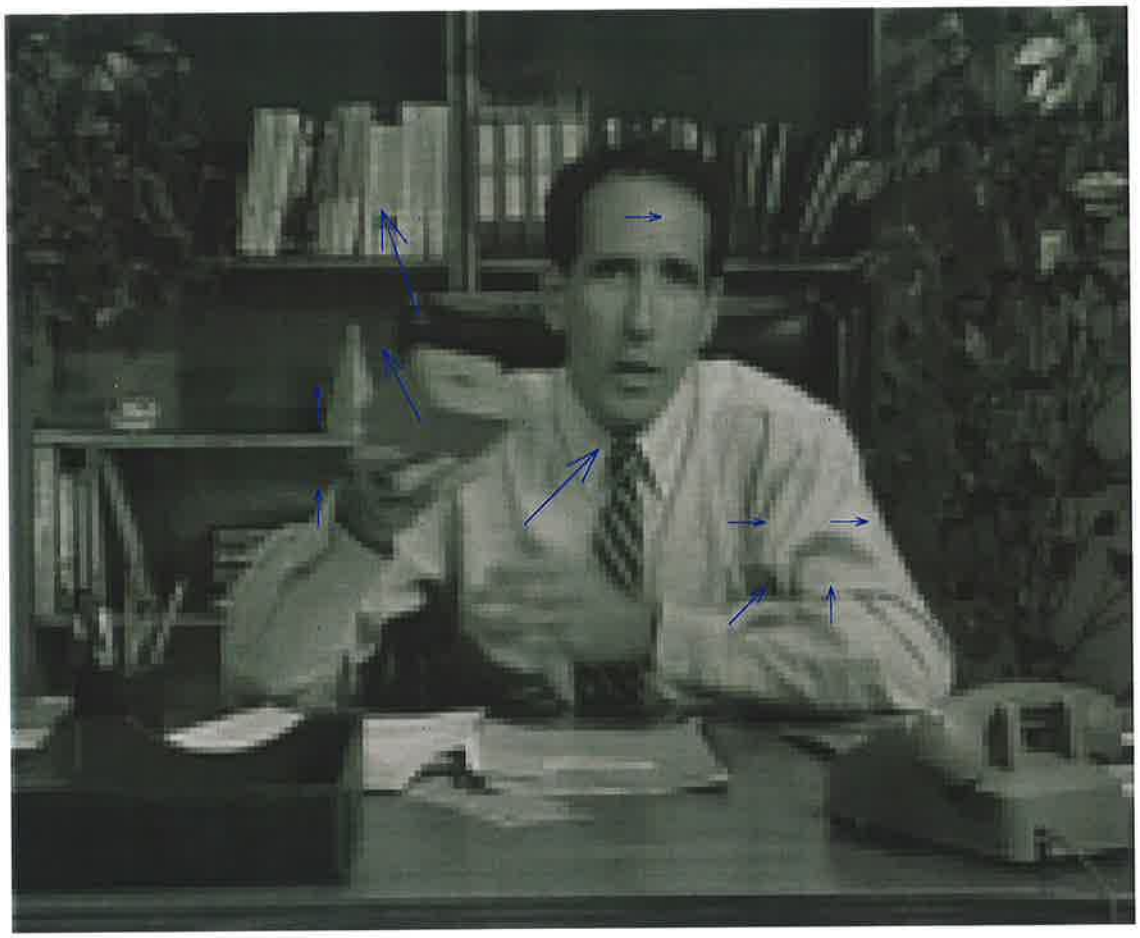


Figure 6.7: Block-based motion estimation between frames 11 and 12 of the *Salesman* sequence using block-sizes of 16×16 pixels.

5. Repeat steps 2 to 4 for all pixel locations in DF.

6.4 Simulation Results and Discussions

In this section, we present the simulation results. In Section 6.4.1, simulation results for synthetic frames are presented, and in Section 6.4.2 simulation results for real video sequences are presented.

6.4.1 Synthetic Frames

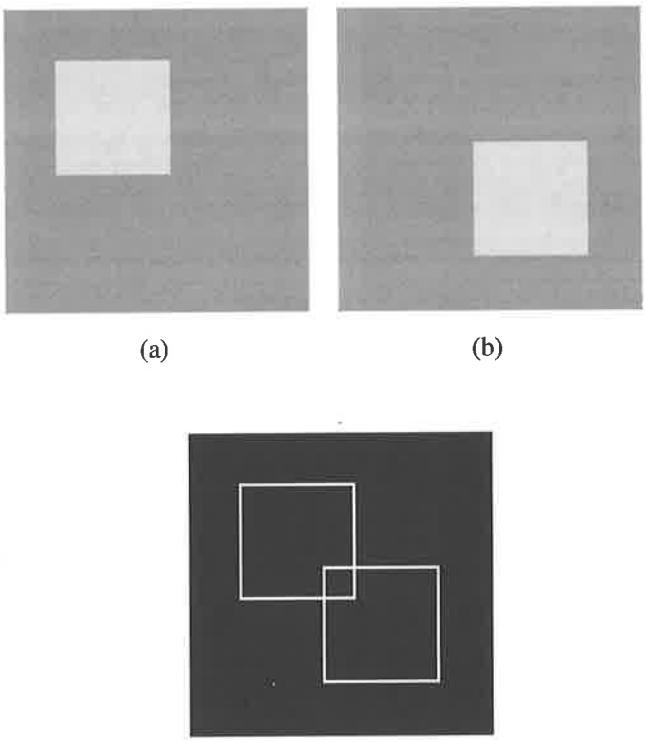
Two pairs of synthetic frames (SF_1 and SF_2) were created. The size of each frame was set to 200×200 pixels. For SF_1, the stationary background was set to a constant gray-level of 128, and the moving object (of size 75×75 pixels) was set to a constant gray-level of 200. Gaussian noise of mean-zero and variance 2 was then added to each frame.[5] The S_0^2 values are given in Table 6.1. The original frames are shown in Figures 6.8(a) and (b), and the change detection mask is shown in Figure 6.8(c). The bright areas indicate the foreground regions. Note that only the occlusion regions of the moving object are marked as changed. This is because there is a large intensity gradient at the edge of the moving object, but no intensity gradients within the moving object. Thus, regions within a moving object will be marked as changed only if they are textured.

For SF_2, the stationary background was also set to a constant gray-level of 128. However, instead of a non-textured moving object, a textured moving object (of size 75×75 pixels) was used. The original frames are shown in Figures 6.9(a) and (b), and the change detection mask is shown in Figure 6.9(c). Gaussian noise of mean-zero and variance 2 was also added to each frame. This time regions within the moving object are also marked as changed since the moving object is textured.

6.4.2 Real Frames

This section presents the simulation results for real video test sequences. Simulations results for 10 consecutive gray-level frames of the *Silent*, *Irene*, *Salesman*, and *Mother & Daughter*

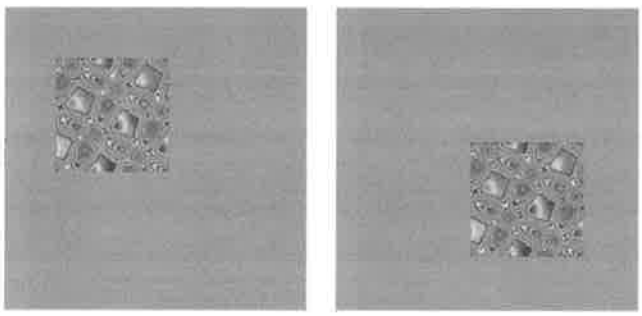
[5]Therefore the variance of the background population in DF is 4.



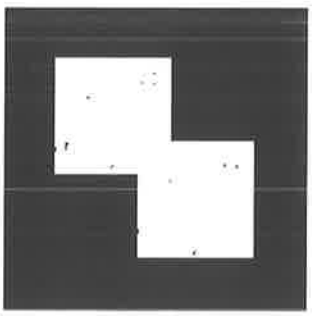
(a) (b)

(c)

Figure 6.8: Synthetic frames SF_1. (a) Frame 1, (b) frame 2, and (c) CDM.



(a) (b)



(c)

Figure 6.9: Synthetic frames SF_2. (a) Frame 1, (b) frame 2, and (c) CDM.

Sequence	S_0^2
SF_1	3.4
SF_2	3.7

Table 6.1: Sample variance values for the synthetic frames tested.

Sequence	S_0^2
Silent	0.35
Irene	0.62
Salesman	0.92
Mother & Daughter	0.58

Table 6.2: Sample variance values for the real sequences tested.

test sequences are presented. The S_0^2 values for the sequences are given in Table 6.2. The same S_0^2 value was used to obtain the CDMs for each sequence, i.e., motion estimation was performed only once for each sequence. This would reduce the computational cost of change detection since block-based motion estimation is computationally expensive. In addition to the four test sequences mentioned above, we have also tested other video sequences, however due to the lack of space, these results are not presented here.

Change detection results for the *Silent* and *Irene* sign language test sequences are shown in Figures 6.10 and 6.11, respectively. The bright areas indicate the foreground regions. Since the test statistic is the ratio of the variance estimate of the difference pixel in the observation window to the variance estimate of the background, the value of the test statistic would be large when the window passes over moving objects. Sign language is characterized by the motion of the mouth, eyes, face, and hands. Since these regions are textured, we would expect the hand and face objects to be marked as foreground in the CDM. The foreground regions cover the face and hand objects reasonably well, with little residual noise (i.e., false alarms) present in the CDMs.

Note that only some parts of the chest area of the subject in the *Silent* sequence are marked as changed. This is due to insufficient texture in the moving regions. It is difficult to detect intensity changes in moving objects if there is insufficient texture, as was illustrated

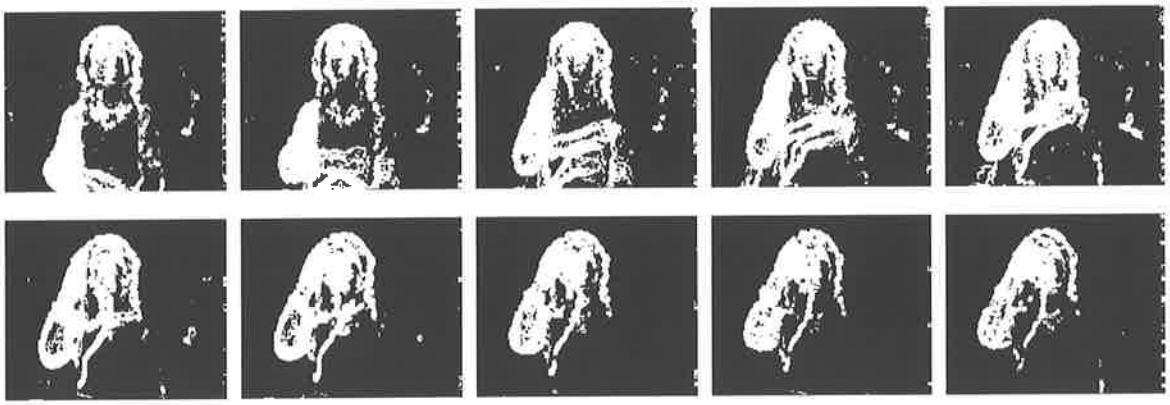
for synthetic frames above. On the other hand, the clothing of the subject in the *Irene* sequence is textured, and consequently the chest area of the subject is marked as changed. The foreground regions on the right hand side of the CDMs (i.e., right side of the moving person) are due to shadow. Shadows can be eliminated from change detection masks using shadow and reflection cancellation strategies devised in [RE95, ZC99], however these strategies have not been employed in this thesis.

The results for the *Salesman* and *Mother & Daughter* sequences are shown in Figures 6.12 and 6.13, respectively. The *Salesman* sequence represents a typical videoconferencing sequence. The salesman is holding an object which he is trying to describe, and in the process, moves the object and turns it around. In the *Mother & Daughter* sequence, the head of the mother has a relatively large motion (mostly small rotations), while her body exhibits little motion. The daughter also exhibits little motion throughout the sequence. The foreground regions in the CDMs cover the moving subjects reasonable well. There is very little residual noise in the CDMs of the *Mother & Daughter* sequence, however there is some residual noise in the CDMs of the *Salesman* sequence.

6.5 Summary

In this chapter, we considered the problem of segmenting video frames into foreground and background regions. A statistical change detection technique based on the F test and block-based motion estimation was proposed. The background pixel population was modeled as a zero-mean normal distribution. The F test compares the sample variance of the difference pixels in W with the sample variance of background pixels. To evaluate the background sample variance, we devised a method based on block-based motion estimation. We observed that the proposed change detection technique detects textured moving objects (e.g., the face and hands) effectively.

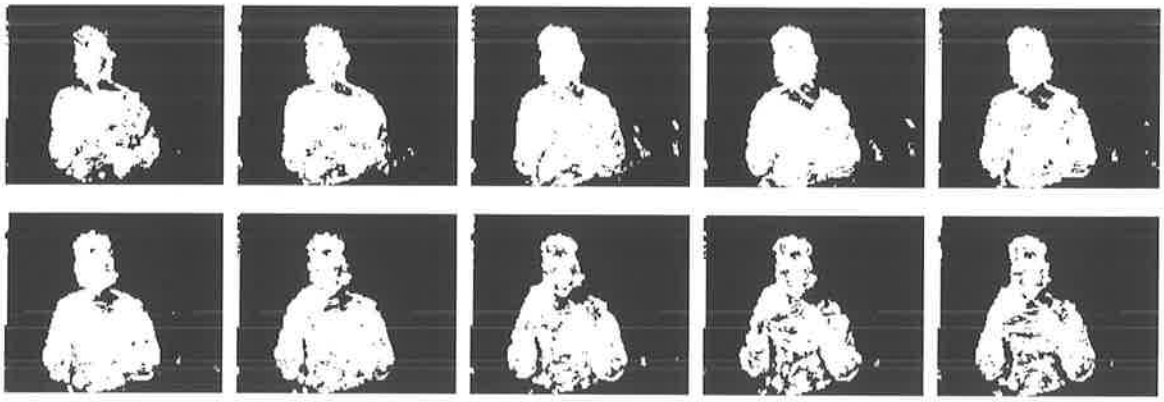
(a)



(b)

Figure 6.10: Change detection masks for 10 consecutive frames of the *Silent* sequence. (a) Original gray-level frames, and (b) CDMs.

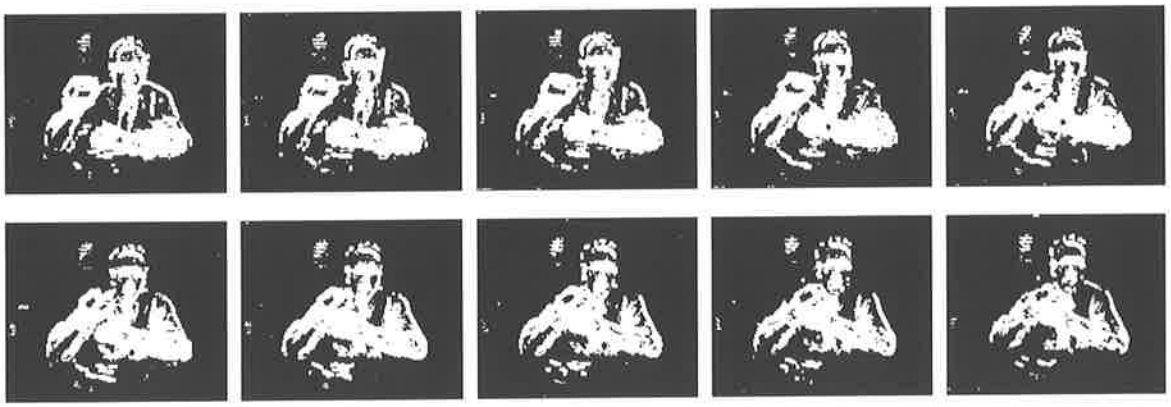
(a)



(b)

Figure 6.11: Change detection masks for 10 consecutive frames of the *Irene* sequence. (a) Original gray-level frames, and (b) CDMs.

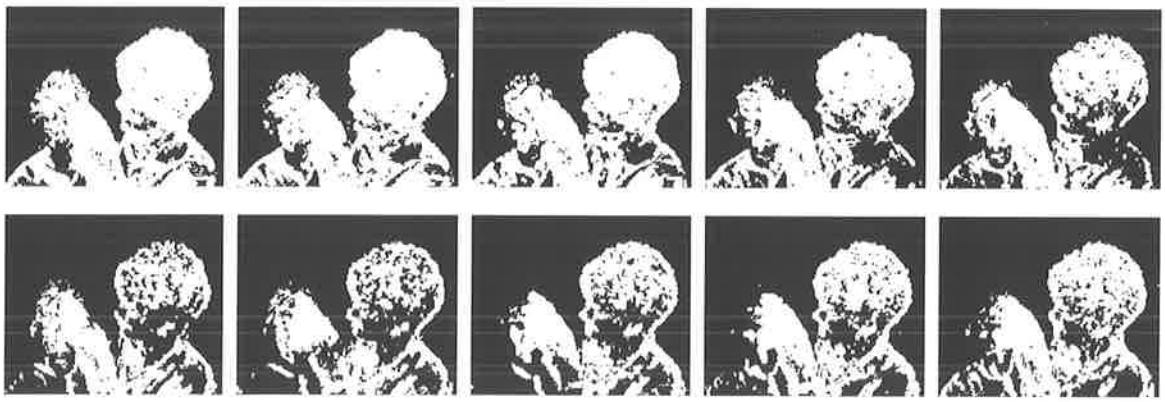
(a)



(b)

Figure 6.12: Change detection masks for 10 consecutive frames of the *Salesman* sequence. (a) Original gray-level frames, and (b) CDMs.

(a)



(b)

Figure 6.13: Change detection masks for five consecutive frames of the *Mother & Daughter* sequence. (a) Original gray-level frames, and (b) CDMs.

Chapter 7

Segmentation and Tracking

"The moving finger writes; and, having writ, moves on..."

- Omar Khayyam (The Rubaiyat)

In the first part of this chapter, the methodology used to generate the face and hand segmentation mask ($FHSM$) is presented. In the second part of this chapter, the techniques used to detect and track the face are described. The $FHSM$ generation method is discussed in Section 7.1, and face detection and tracking is discussed in Section 7.2. Simulations results are presented in Section 7.3, and the chapter is summarized in Section 7.4.

7.1 FHSM Generation

The methodology used to generate the face and hand segmentation mask is described in this section. To generate the $FHSM$, color and motion information from the skin detection mask (SDM) and the change detection mask (CDM) are utilized. We first note that the skin-color regions in a frame are localized in the SDM. Also, as noted previously, sign language is characterized by the motion of the arms, the hands, and the face (including the eyes and the mouth).

Moving objects entail intensity changes between consecutive frames, which are marked as foreground in the CDM. Thus, the CDM can be used to separate the moving skin-color regions from the stationary skin-color regions in the SDM. The $FHSM$ is a binary map where a binary "1" indicates a moving skin-color region, and a binary "0" indicates a background pixel. The $FHSM$ is analogous to the VOP in the MPEG-4 standard (Section 2.2.2).[1] The postprocessing stages are described below.

To generate the $FHSM$, connected components labeling [HS92] is first performed on the SDM to find the connected components (with 8-neighborhood connectivity). If the size of a connected component is less than a certain threshold, we assume that it is a false alarm and eliminated from the SDM. To determine a suitable threshold, we must examine the size of the face and hand objects in the sequence. We note that the size of the face object remains fairly constant throughout a sign language sequence, however the size of the hand objects vary depending on their position. Figure 7.1 shows frame 218 of the *Silent* sequence. The size of the right hand is 243 pixels, and the size of the left hand is 117. After an extensive analysis of different hand positions and their corresponding sizes in both sign language video sequences (i.e., *Silent* and *Irene*), we found that a suitable threshold is 100 pixels. This threshold value was derived empirically. Thus, if the size of a connected component in the SDM is below 100 pixels, it is assumed to be a false alarm and discarded.

To identify the moving skin-color regions, the skin-color regions in the SDM are projected onto the CDM, as shown in Figure 7.2. When the majority of a connected component in the SDM is covered by a foreground region in the CDM, the connected component is declared as a moving skin-color region. We expect the moving skin-color regions to repre-

[1] In this case, the face and hand objects represent a VO.

Figure 7.1: Frame 218 of the *Silent* sequence, indicating the hand objects.

sent either the face or hand objects, however the $FHSM$ may also contain false alarms due to the following reasons:

1. Moving skin-color regions due to clothing or hair.

2. Skin-color regions in the uncovered background. The uncovered background areas are marked as changed in the CDM. To overcome this, the uncovered background areas must be identified, e.g., [MN98a].

3. Shadows produced by moving objects will entail intensity variations that are marked as changed. This may result in false alarms if a skin-color region coincides with a foreground region associated with shadows. To overcome this, shadow cancellation strategies can be employed, e.g., [RE95, ZC99].

The face object may contain holes due to the presence of the eyes, mouth, and eyebrows. In addition, "bright spots" and shadows may also produce holes in the face and hand objects. To fill these holes, we employ the *morphological closing operator* [HS92, Cas96]. Morphological closing has the effect of filling small and thin holes, connecting nearby regions, and generally smoothing the boundaries of regions without significantly changing their areas. Closing is the process of *dilation* followed by *erosion*.

Erosion is defined by [Cas96]:

$$\mathbf{E} = \mathbf{B} \otimes \mathbf{S} = \{x, y | \mathbf{S}_{x,y} \subseteq \mathbf{B}\}, \tag{7.1}$$

where $\mathbf{B}$ and $\mathbf{S}$ denote the binary image and structuring element, respectively, and are defined on a 2D Cartesian grid. The parameter $\mathbf{S}_{x,y}$ denotes the structuring element after it has been

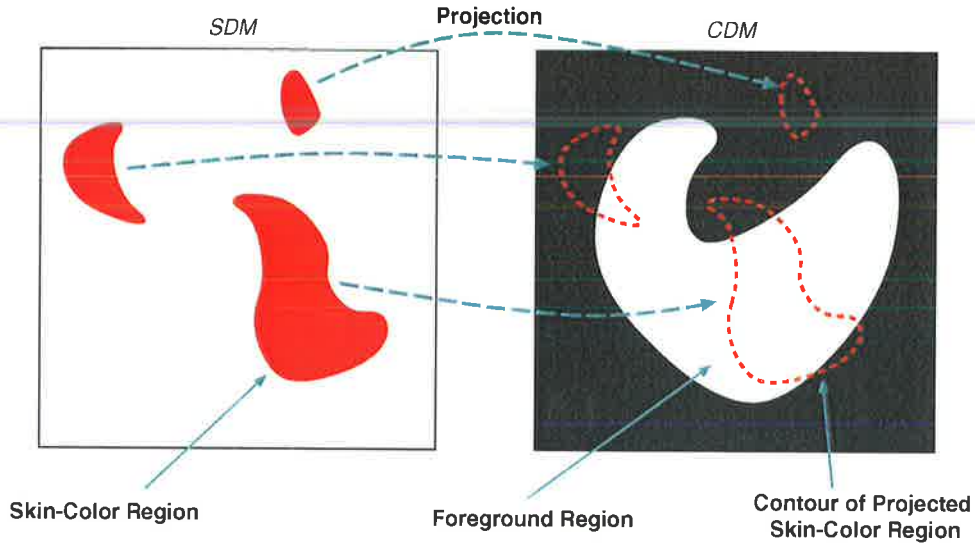


Figure 7.2: SDM projected onto the CDM.

translated so that its origin is located at (x, y). According to (7.1), the binary image $\mathbf{E}$ that results from eroding $\mathbf{B}$ by $\mathbf{S}$ is the set of points (x, y) such that if $\mathbf{S}$ is translated so that its origin is located at (x, y), then it is completely contained within $\mathbf{B}$.

Dilation is defined by [Cas96]:

$$\mathbf{D} = \mathbf{B} \oplus \mathbf{S} = \{x, y | \mathbf{S}_{x,y} \cap \mathbf{B} \neq \emptyset\}. \tag{7.2}$$

That is, the binary image $\mathbf{D}$ that results from dilating $\mathbf{B}$ by $\mathbf{S}$ is the set of points (x, y) such that if $\mathbf{S}$ is translated so that its origin is located at (x, y), then its intersection with $\mathbf{B}$ is not empty. Since closing is the process of dilation followed by erosion, it is defined by:

$$\mathbf{B} \bullet \mathbf{S} = (\mathbf{B} \oplus \mathbf{S}) \otimes \mathbf{S}. \tag{7.3}$$

We found that a large structuring element would merge nearby regions, even though they may represent different objects. Figure 7.3(a) shows frame 16 of the *Silent* sequence, and Figures 7.3(b), (c), and (d) show the result of the morphological closing operator applied to the corresponding $FHSM$ with circular structuring elements of varying diameters. The hand and face objects merge if a structuring element with a diameter of nine pixels or higher is applied. We also found that a small structuring element would not effectively fill holes in some cases. A circular structuring element with a diameter of 7 pixels was found to be most effective in filling holes, while at the same time reducing the chances of nearby regions

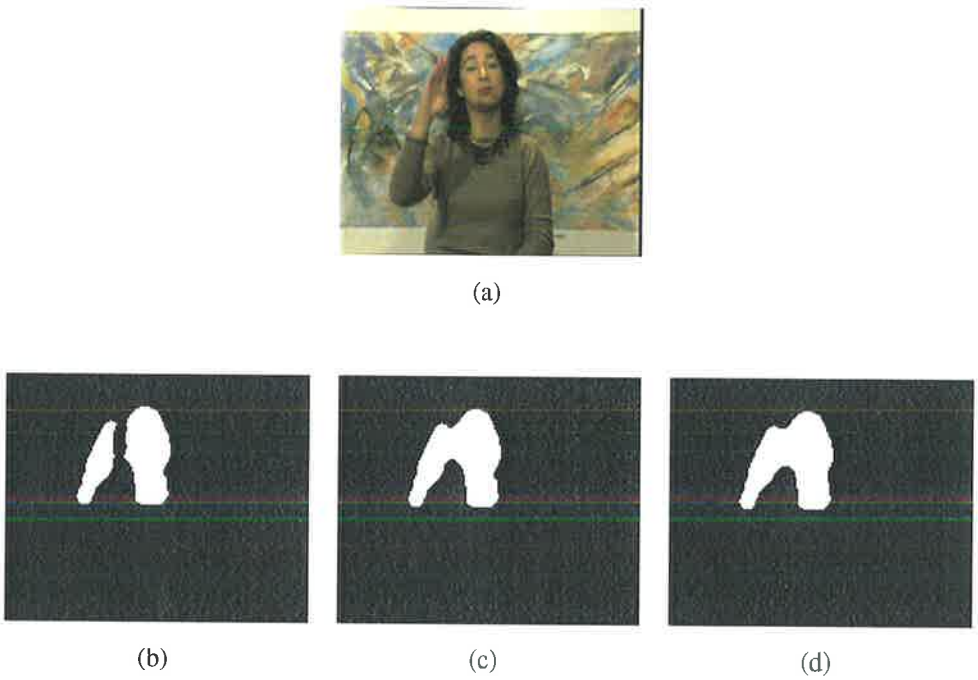
merging. Circular structuring elements tend to promote the formation of smooth and curved object boundaries, which closely resemble those of real objects. The block diagram of the $FHSM$ generation process is depicted in Figure 7.4.

7.2 Face Detection and Tracking

This section presents our face detection and tracking techniques. It may sometimes be necessary to discriminate between the face and the hands in a video sequences. This will allow the face and the hand objects to be coded independently. For example, a sign language video codec may have a "lip-reading" mode [SS01]. In this mode, the face is coded at a higher-bit rate than the hands for a better perceptual quality. People who have become deaf late in life usually find it difficult to communicate via sign language, but can lip-read. By coding the face at a higher bit-rate, people would be able to lip-read more effectively. Consequently, we need a method to detect the face in the video sequence. Once the face is detected, a reference $FHSM$ is formed and is used to track the face in subsequent frames of the sequence.

Face detection has received considerable attention among researchers in recent years. A wide variety of techniques have been proposed, ranging from simple edge-based methods to composite high-level approaches utilizing advanced pattern recognition methods. There are many problems that are closely related to face detection. *Face localization* aims to determine the image position of a single face; this is a simplified detection problem with the assumption that an image contains only one face [MP97]. In *facial feature detection*, the goal is to detect the presence and location of features such as eyes, nose, nostrils, eyebrows, mouth, lips, ears, etc. with the assumption that there is only one face in the image [CTB92]. *Face recognition* or *face identification* compares an input image against a database and reports a match [WFKM97]. The aim of *facial expression recognition* is to identify the mood or state (e.g., happy, sad, angry, etc) of humans [DBH+99]. Surveys on face detection methods can be found in [HL01] and [YKA02].

In this section, we concentrate on the detection of faces in sign language video sequence. The task of face detection in sign language video poses certain unique challenges. These are discussed below.



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

Figure 7.3: The effect of varying the size of the structuring element. (a) Frame 16 of the *Silent* sequence, (b) structuring element with a diameter of 7 pixels, (c) structuring element with a diameter of 9 pixels, and (d) structuring element with a diameter of 11 pixels.

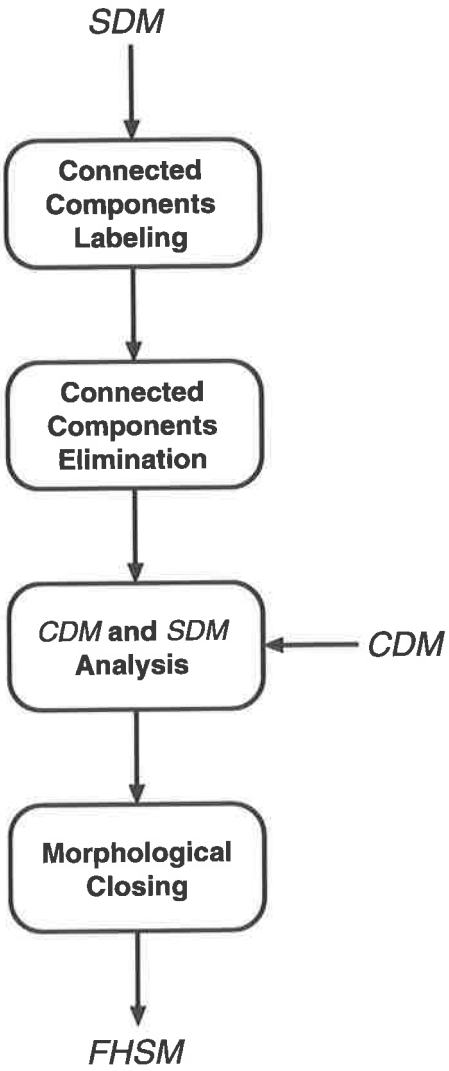


Figure 7.4: Block diagram of the $FHSM$ generation process.

7.2.1 Face Detection

One way to detect the face in a frame is to compare the size of the connected components in the $FHSM$. Intuitively, the face would have the largest size, however if a subject has large parts of an arm exposed, the arm may have a larger size than the face and thus result in inaccurate detection. Also, if the distance between the camera and the hand is significantly shorter than the distance between the camera and the face, the hand may appear disproportionately larger.

In order to avoid the above problems, we will use shape-features to detect the face in a frame. Three tests have been devised to make this differentiation: orientation, aspect ratio, and solidity. In sign language video, the head is typically located in the upper half of a frame. Therefore, the tests are restricted to connected components that have 50% or more of their area in the top half of a frame.

To detect the face in a video frame, Menser and Wien [MW00b] (see Section 5.2) also employed shape-features in their algorithm. The authors considered the following shape-features: aspect ratio, solidity, and compactness. Let D_x and D_y denote the width and height, respectively, of the bounding rectangle of connected component $\mathcal{C}$. The width and height are measured with respect to the x and y axes (Figure 7.5). The aspect ratio, solidity, and compactness are given by:

$$\mathcal{A} = \frac{D_y}{D_x}, \tag{7.4}$$

$$\mathcal{S} = \frac{\sum_{(x,y)\in\mathcal{C}} 1}{D_x D_y}, \tag{7.5}$$

and

$$\mathcal{O} = \frac{\sum_{(x,y)\in\mathcal{C}} 1}{P_{\mathcal{C}}^2}, \tag{7.6}$$

respectively, where $P_{\mathcal{C}}$ is the perimeter of $\mathcal{C}$. Unfortunately, (7.4) and (7.5) do not take into consideration the orientation of $\mathcal{C}$. Since the head can tilt during signing, (7.4) and (7.5) will not produce reliable results. Compactness (also called cicularity) is used to remove connected components with complex contours. Equation (7.6) is minimum for a circle. The connected component that contains the head object may also contain parts of the hair and neck, and therefore possess a complex contour. As a result, (7.6) is not suitable for detecting faces in a frame. Our orientation, aspect ratio, and solidity tests take into consideration

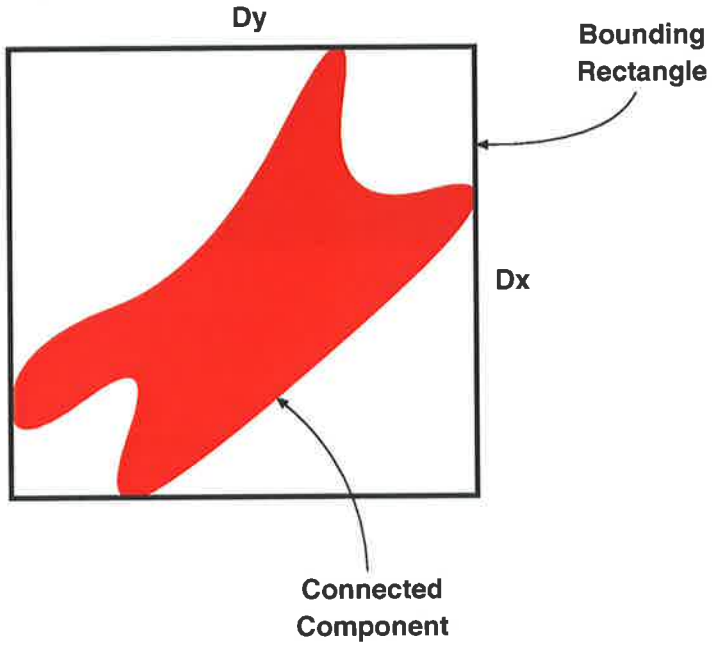
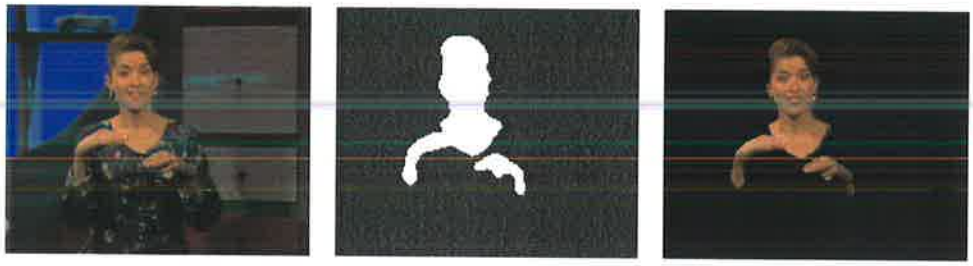


Figure 7.5: The bounding rectangle of a connected component as proposed by Menser and Wien (2000).

the orientation of C and are therefore better suited to face detection in sign language video sequences.

After detecting the face object in a $FHSM$, a reference $FHSM$ is formed. The reference $FHSM$ is then tracked in subsequent frames of the video sequence (Section 7.2.2). There are two reasons why face tracking is necessary. First, it is computationally expensive to perform all three tests for every frame in the sequence. Second, if the face and a hand overlap during signing, they will form one connected component in the $FHSM$ (e.g., Figure 7.6). This occurs frequently in sign language video sequences. The three tests are designed to detect the face object, and are not reliable when the face and hand objects form one connected component. In order to detect the face object, the connected component that contains the face object must be detected, and to do this, face tracking must be employed. The objective of face tracking is to detect the face (or the connected component containing the face object) in subsequent frames of the video sequence, and to establish a correspondence of the face object between frames. The tests are discussed below.



(a) (b) (c)

Figure 7.6: Face and hand objects forming one connected component. (a) Frame 22 of the *Irene* sequence, (b) $FHSM$, and (c) $FHSM$ showing the identified skin pixels.

The Orientation Test

The first test is the orientation test. The center of gravity of a connected component $\mathcal{C}$ is given by [Jai89]:

$$\bar{x} = \frac{1}{N} \sum_{(x,y)\in\mathcal{C}} x, \tag{7.7}$$

and

$$\bar{y} = \frac{1}{N} \sum_{(x,y)\in\mathcal{C}} y, \tag{7.8}$$

where N denotes the number of pixels in $\mathcal{C}$. The (p, q) central moments become:

$$\xi_{p,q} = \sum_{(x,y)\in\mathcal{C}} (x - \bar{x})^p (y - \bar{y})^q. \tag{7.9}$$

Orientation is defined as the angle of axis of the least moment of inertia, and is obtained by minimizing with respect to ϕ the sum:

$$I(\phi) = \sum_{(x,y)\in\mathcal{C}} D^2(x, y) = \sum_{(x,y)\in\mathcal{C}} [(y - \bar{y})cos\phi - (x - \bar{x})sin\phi]^2. \tag{7.10}$$

The orientation can be computed by utilizing the central moments $\xi_{p,q}$ of the connected component:

$$\phi = \frac{1}{2}\tan^{-1}\left[\frac{2\xi_{1,1}}{\xi_{2,0} - \xi_{0,2}}\right]. \tag{7.11}$$

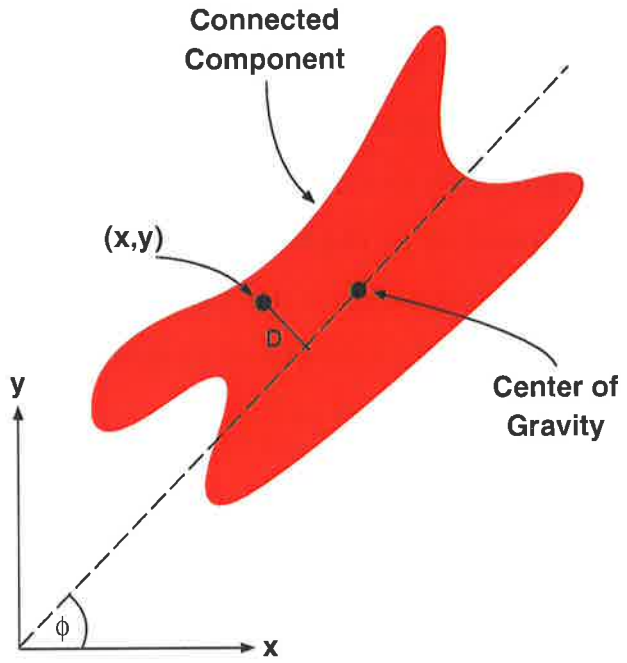


Figure 7.7: Orientation of a connected component.

We have observed that the head can typically tilt in the range $\phi = 70°$ to $110°$ during signing. Therefore, if the orientation of a connected component is not within this range, it cannot be the face. To determine the orientation range, we performed an empirical study of head orientations in the *Silent* and *Irene* video sequences, and in short sequences (usually three frames long) found in "animated" sign language dictionaries [Lap02, Sti97]. In each case, the face objects were manually segmented for various signs, and their orientations computed. An example of a short sign language sequence (from [Lap02]) is shown in Figure 7.8.

Figure 7.8: An example of a short sign language sequence.

The Aspect Ratio Test

The second test deals with the aspect ratio ($\mathcal{A} = a/b$) of $\mathcal{C}$, where a and b denote the lengths of the major and minor axes, respectively, of the best-fit ellipse (Figure 7.9(a)). We have observed that the aspect ratio of the face and any exposed neck, range from 1.6 to 2.6. This range was determined based on an empirical study of more than 50 different faces. Therefore, any connected component outside of this range, cannot represent the face object. The parameters a and b are determined by computing the moments of inertia of $\mathcal{C}$. The least and greatest moments of inertia for an ellipse are

$$I_{min} = \frac{\pi}{4}ab^3, \tag{7.12}$$

and

$$I_{max} = \frac{\pi}{4}a^3b. \tag{7.13}$$

For a given ϕ, the above moments can be calculated as:

$$I'_{min} = \sum_{(x,y)\in\mathcal{C}} \left[(y-\bar{y})cos\phi - (x-\bar{x})sin\phi\right]^2, \tag{7.14}$$

and

$$I'_{max} = \sum_{(x,y)\in\mathcal{C}} \left[(y-\bar{y})sin\phi - (x-\bar{x})cos\phi\right]^2. \tag{7.15}$$

For a best-fit ellipse we want $I_{min} = I'_{min}$ and $I_{max} = I'_{max}$, which gives the lengths of a and b, respectively, as:

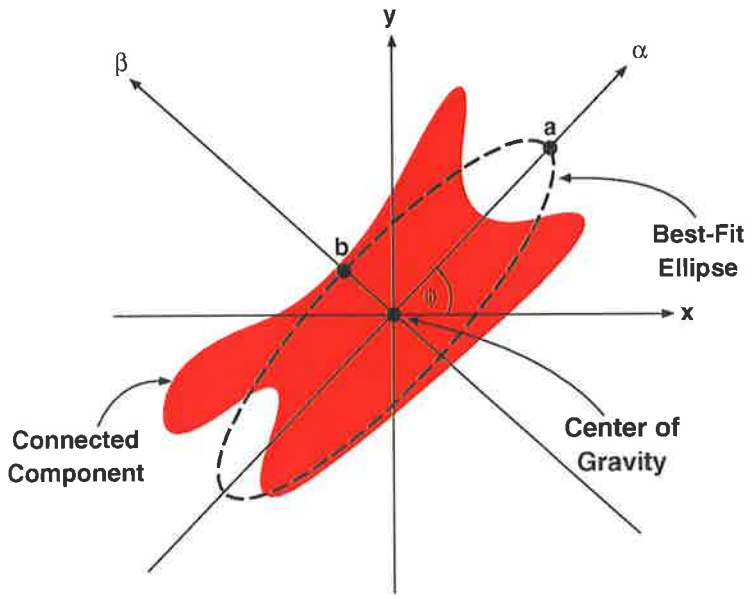
$$a = \left(\frac{4}{\pi}\right)^{\frac{1}{4}} \left[\frac{(I'_{max})^3}{I'_{min}}\right]^{\frac{1}{8}}, \tag{7.16}$$

and

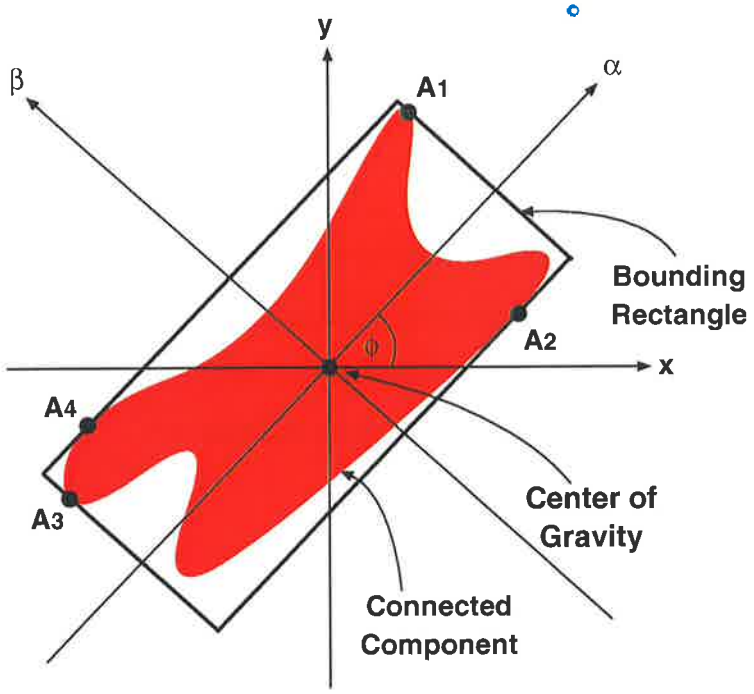
$$b = \left(\frac{4}{\pi}\right)^{\frac{1}{4}} \left[\frac{(I'_{min})^3}{I'_{max}}\right]^{\frac{1}{8}}. \tag{7.17}$$

The Solidity Test

The third test is the solidity test. Solidity is defined as the area of $\mathcal{C}$ divided by the area of the bounding rectangle. The bounding rectangle is the smallest rectangle enclosing the object



(a)



(b)

Figure 7.9: Shaped based features. (a) Best fit ellipse, (b) bounding rectangle.

Test	Acceptable Range
Orientation	$70° \leq \phi \leq 110°$
Aspect ratio	$1.6 \leq \mathcal{A} \leq 2.6$
Solidity	$0.55 \leq \mathcal{S} \leq 0.85$

Table 7.1: Acceptable ranges for the face detection tests.

that is also aligned with its orientation. To find the bounding rectangle the transformation

$$\begin{aligned} \alpha &= x\cos\phi + y\sin\phi \\ \beta &= -x\sin\phi + y\cos\phi \end{aligned} \tag{7.18}$$

is used on the boundary points of C to search for α_{min}, α_{max}, β_{min}, and β_{max}. These give the locations of points A_3, A_1, A_2, and A_4, respectively. Based on the preceding formulations, the length and width of the bounding box are given by $l_b = \alpha_{max} - \alpha_{min}$ and $w_b = \beta_{max} - \beta_{min}$, respectively. Solidity is given by the following ratio:

$$\mathcal{S} = \frac{\sum_{(x,y)\in C} 1}{l_b w_b}. \tag{7.19}$$

Face objects normally maintain a solidity in the range of 0.55 to 0.85. Again, this range was determined based on an empirical study of more than 50 different faces. If the fingers of a hand are spread for example, its solidity will be low. The acceptable ranges for the orientation, aspect ratio, and solidity tests are given in Table 7.1.

7.2.2 Face Tracking

After the reference $FHSM$ is formed, the face object is tracked in subsequent frames. In sign language video, the head tends to maintain a fairly constant position throughout a sequence, similar to typical head and shoulder sequences. Thus, the same reference $FHSM$ can be employed to track the face. However, if in certain situations the position of the head does change significantly, a new $FHSM$ may need to be formed after a certain number of frames. We have experimented with two tracking techniques, namely *region projection*

and *Euclidean distances*. In its present state, our tracking technique is not able to separate overlapping face and hand objects.

Region Projection

Let $\mathcal{C}_F$ denote the face object in the reference $FHSM$, and $\mathcal{C}_{i,k}$, $i = 1, ..., C$, denote the connected components in the current $FHSM$ ($FHSM_k$). In the region projection technique, the reference $FHSM$ is projected onto $FHSM_k$. The projected $\mathcal{C}_F$ provides an estimate of the face object in $FHSM_k$. Let $N_{\mathcal{C}_F \cap \mathcal{C}_{i,k}}$ be the number of pixels in the union $\mathcal{C}_F \cap \mathcal{C}_{i,k}$ of the two connected components $\mathcal{C}_F$ and $\mathcal{C}_{i,k}$. The connected component $\mathcal{C}_{i,k}$ that gives the highest $N_{\mathcal{C}_F \cap \mathcal{C}_{i,k}}$ is then designated as the face object or the connected component that contains the face object in $FHSM_k$. Figure 7.10(a) shows the contour of $\mathcal{C}_F$ projected onto $FHSM_k$. Note that the contour of $\mathcal{C}_F$ coincides with the face object in $FHSM_k$.

Euclidean Distances

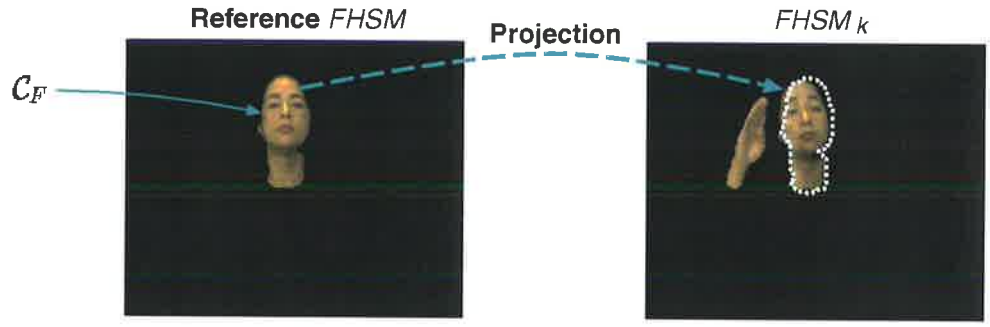
In the second method, we consider the Euclidean distance between $\mathcal{C}_F$ and $\mathcal{C}_{i,k}$, $i = 1, ..., C$. The Euclidean distance between $\mathcal{C}_F$ and $\mathcal{C}_{i,k}$, $i = 1, ..., C$, is given by:

$$r(\mathcal{C}_F, \mathcal{C}_{i,k}) = \sqrt{(\bar{x}_{\mathcal{C}_F} - \bar{x}_{\mathcal{C}_{i,k}})^2 - (\bar{y}_{\mathcal{C}_F} - \bar{y}_{\mathcal{C}_{i,k}})^2}. \tag{7.20}$$

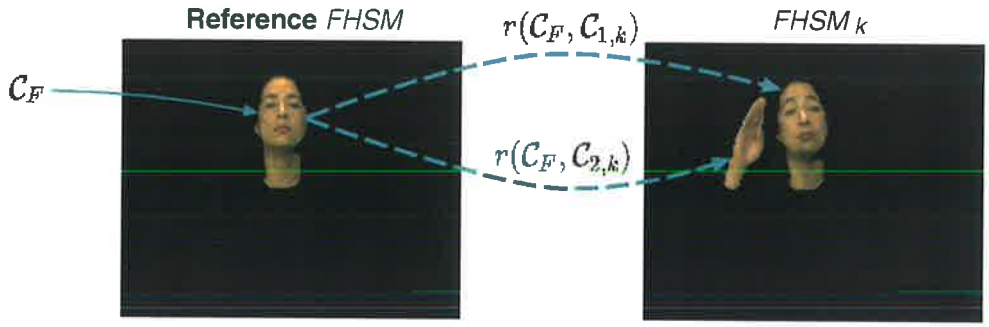
Since the face maintains a fairly constant position throughout the video sequence, the connected component in $FHSM_k$ that minimizes (7.20) is the face object or the connected component that contains the face object. Tracking based on Euclidean distances is more computationally expensive than projection-based tracking, since the center of gravity $(\bar{x}, \bar{y})$ must be computed for each connected component $\mathcal{C}_{i,k}$ in order to calculate the Euclidean distance. Figure 7.10(b) depicts the Euclidean distances between $\mathcal{C}_F$ and $\mathcal{C}_{i,k}$.

7.3 Simulation Results and Discussions

The simulation results are presented in this section. Section 7.3.1 presents the simulation results for $FHSM$ generation, and Section 7.3.2 presents the simulation results for face detection and tracking.



(a)



(b)

Figure 7.10: Face tracking. (a) Contour of C_F projected onto $FHSM_k$, and (b) the Euclidean distances between C_F and $C_{i,k}$.

7.3.1 FHSM Generation

The change detection masks for frames 218 to 223 of the *Silent* sequence, and frames 210 to 215 of the *Irene* sequence are shown in Figures 7.11 and 7.13, respectively. Note that the face and hand objects are well covered by the foreground regions, and there is little residual noise in the CDMs.

Figures 7.12 and 7.14 show the SDMs and the $FHSM$s for six consecutive frames (same as the ones for the CDMs) of the *Silent* and *Irene* sequences, respectively. The SDMs for both the *Silent* (second column, Figure 7.12) and *Irene* sequences (second column, Figure 7.14) contain false alarms. This is due to some background areas with a similar color to that of skin. For the *Silent* sequence, most of the false alarms have been successfully discarded (third column, Figure 7.12), since their size is less than 100 pixels. For the *Irene* sequence, there are two false alarm regions that still remain after connected components analysis. This is because their size is larger than 100 pixels. However, these regions reside in stationary background, and can therefore be eliminated by the use of motion information. We observe that the hair of the subject in the *Irene* sequence has been detected as skin. We do not expect this to significantly affect the performance of a content-based video coder. Moreover, we are not aware of any skin-segmentation technique that can successfully eliminate skin-colored hair that has formed one connected component with the face. The $FHSM$s for the sequences are shown in the fourth column of Figures 7.12 and 7.14. The false alarm regions in the stationary background have been successfully eliminated.

Figures 7.15(a) and (b) show the false alarm and miss rates for 60 consecutive frames (same as the ones tested in Chapter 5) of the *Silent* and *Irene* sequences, respectively. The average false alarm and miss rates are given in Table 7.2. Compared to the rates in Table 5.3, the average false alarm and miss rates for both sequences are lower after postprocessing. This is understandable since the use of motion information and connected components labeling has effectively eliminated most of the false alarm regions, and the morphological closing operator has filled the holes in the face and hand objects.

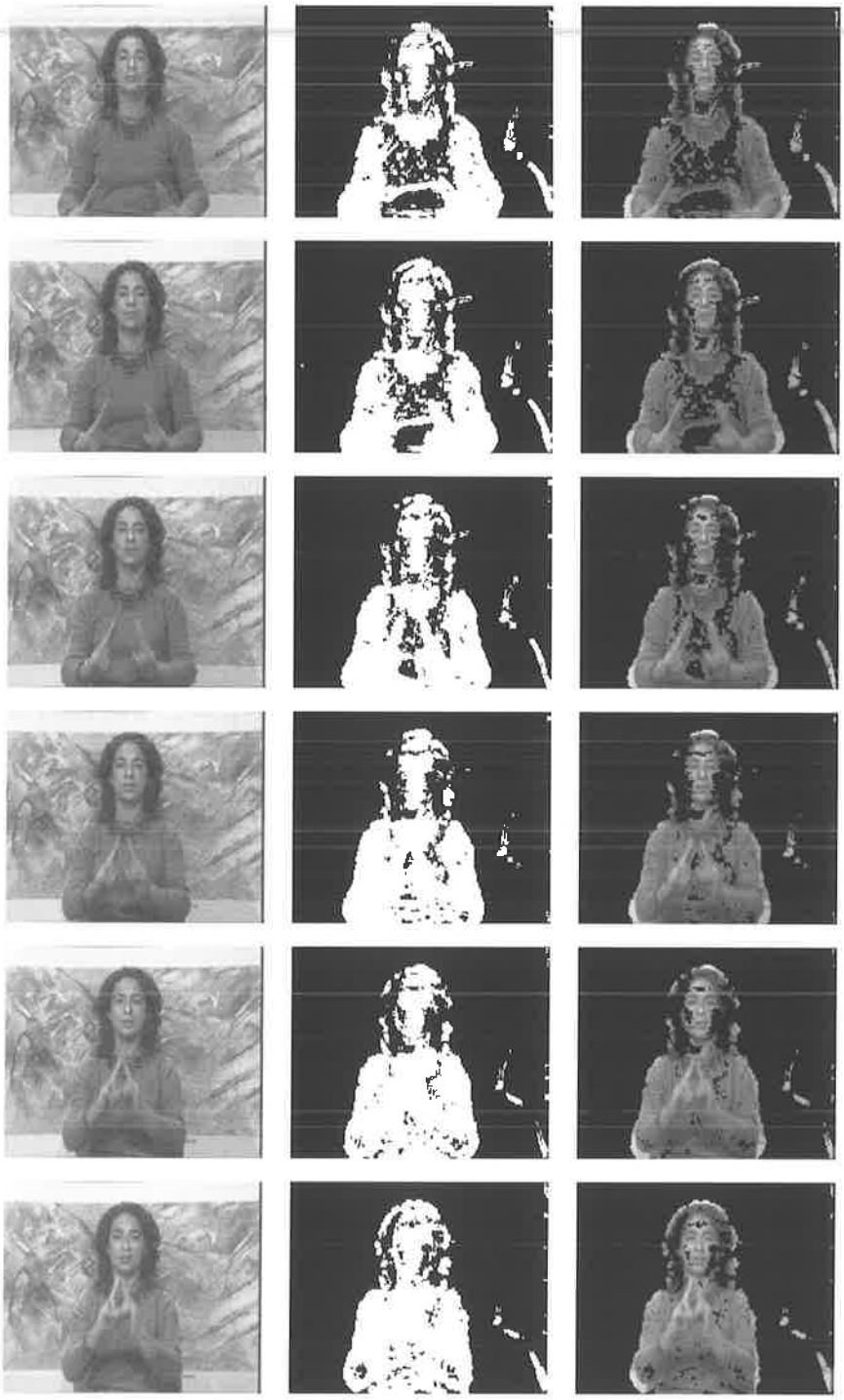


Figure 7.11: Change detection masks: *Silent* sequence. First column: Original gray-level frames. Second column: CDMs. Third column: Identified foreground pixels.

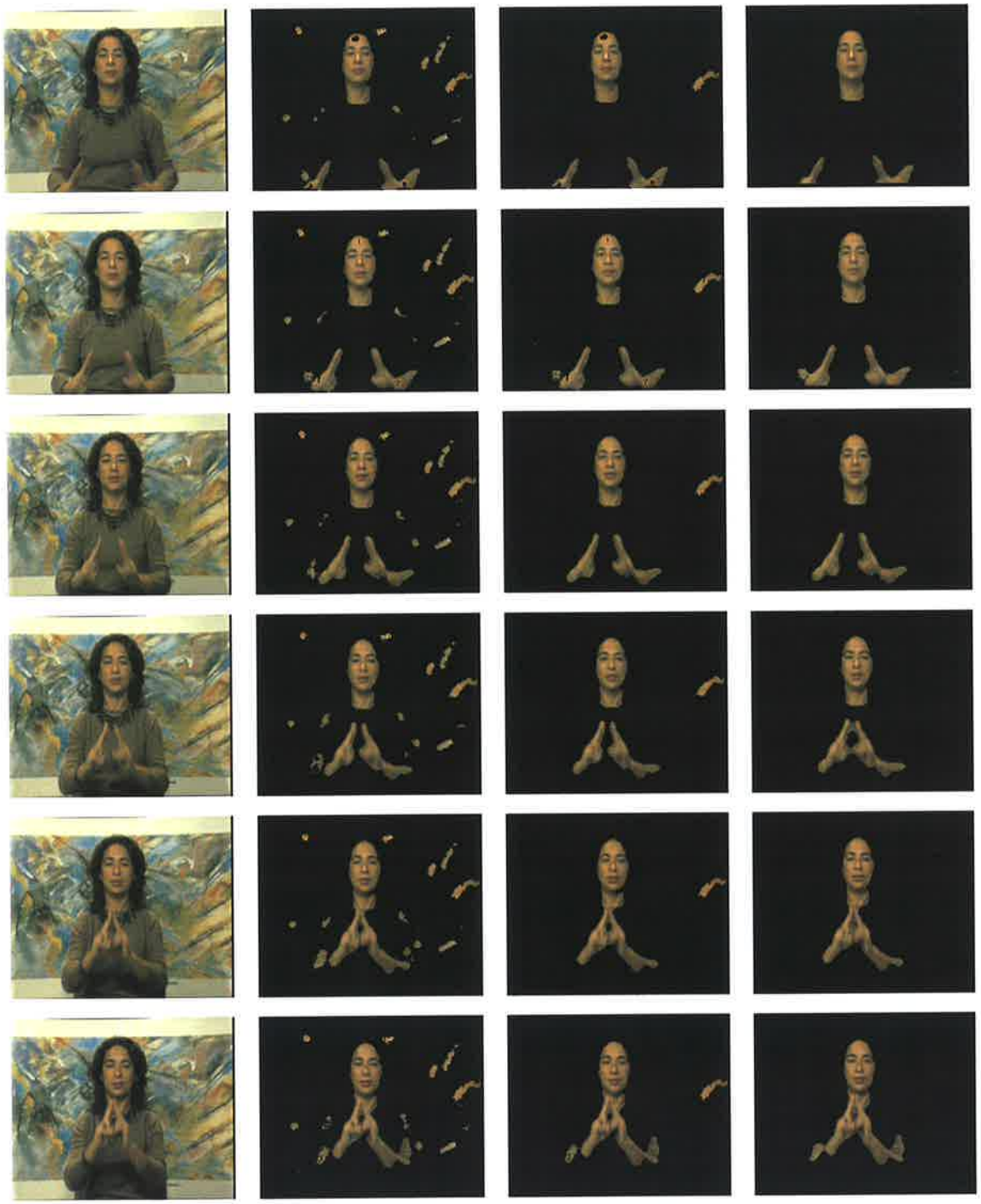


Figure 7.12: Skin and hand segmentation masks showing the identified skin pixels: *Silent* sequence. First column: Original frames. Second column: SDMs. Third column: SDMs after connected components labeling. Fourth column: $FHSM$s.

Figure 7.13: Change detection masks: *Irene* sequence. First column: Original frames. Second column: CDMs. Third column: Identified foreground pixels.

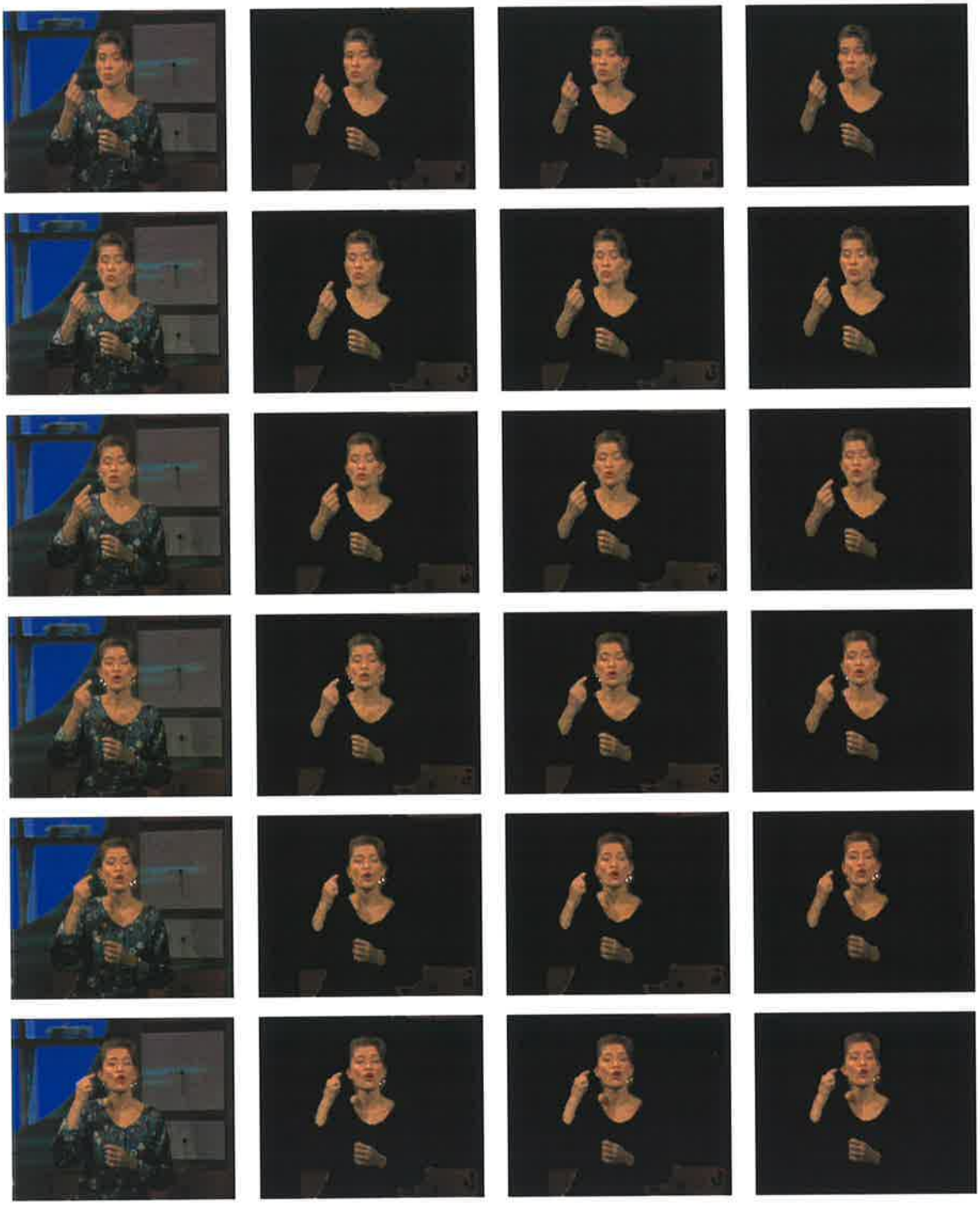
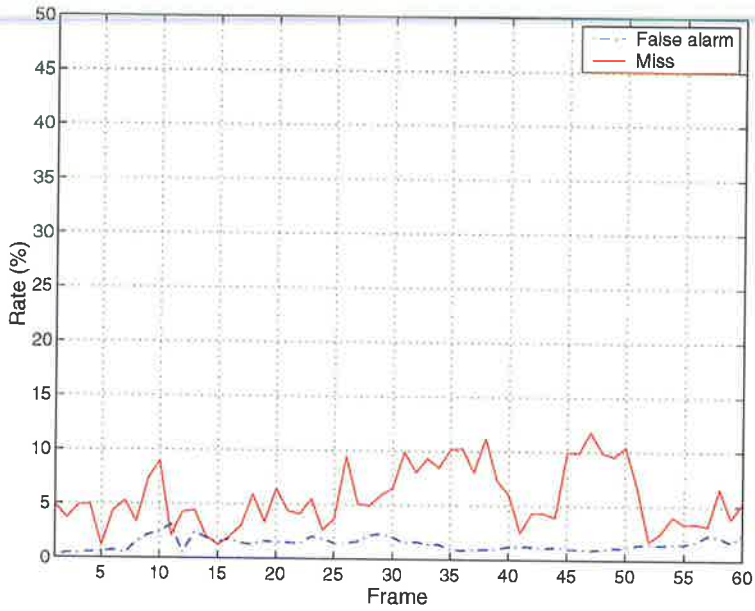
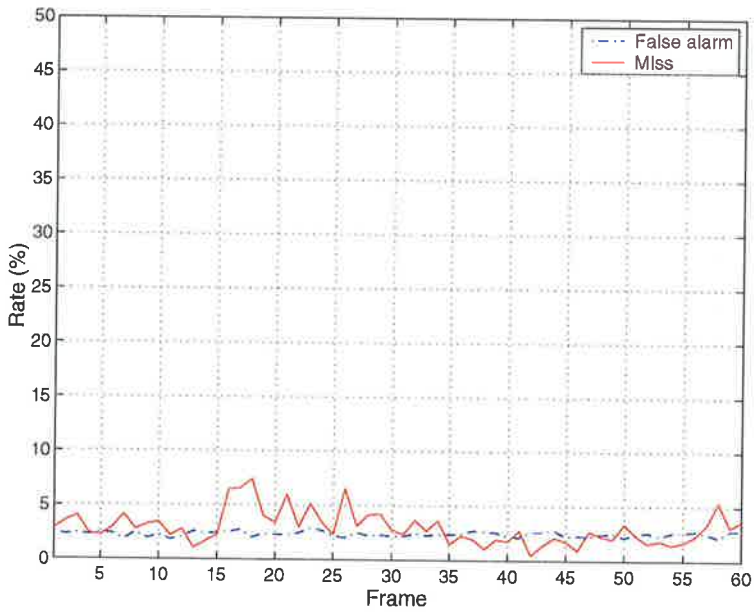


Figure 7.14: Skin and hand segmentation masks showing the identified skin pixels: *Irene* sequence. First column: Original frames. Second column: $SDMs$. Third column: $SDMs$ after connected components labeling. Fourth column: $FHSMs$.



(a)



(b)

Figure 7.15: Miss and false alarm rates for 60 consecutive frames of the (a) *Silent* sequence, and (b) *Irene* sequence.

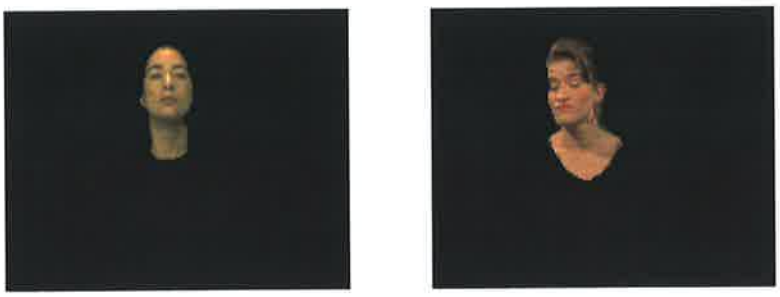
Sequence	Average R_F (%)	Average R_M (%)
Silent	1.4	5.7
Irene	2.4	3.0

Table 7.2: Average miss and false alarm rates for 60 consecutive frames of the *Silent*, and *Irene* sequences after postprocessing.

7.3.2 Face Detection and Tracking

As already noted, the face detection tests may fail if the face and hand objects overlap. For a connected component to be identified as a face object, it must comply with the specifications listed in Table 7.1. The test results for the face and hand connected component in Figure 7.6(b) are given in Table 7.3. The connected component has a solidity of only 0.4, which is outside the acceptable solidity range. Note that the test values for the orientation and aspect ratio are in the desired range.

A reference $FHSM$ was formed at the start of each sequence. To form a reference $FHSM$, all connected components in the top half of the current $FHSM$ were tested. If face detection failed (i.e., none or more than one of the connected components complied with the specifications in Table 7.1), the next frame in the sequence was tested. The face was then tracked in subsequent frames. The reference $FHSM$s are shown in Figure 7.16.



(a) (b)

Figure 7.16: Reference $FHSM$s. (a) *Silent* sequence, and (b) *Irene* sequence.

Both tracking techniques proved to be effective, and produced the same results. Tracking results (using the region projection technique) for six consecutive frames of the *Silent* and

Test	Value
Orientation	74.7°
Aspect ratio	2.0
Solidity	0.41

Table 7.3: Face detection results for the face and hand connected component in Figure 7.6(b).

Irene sequences are shown in Figure 7.17. The contours indicate the outline of the connected components containing the face object. However, since the computational cost of region projection is lower than Euclidean distances, the region projection technique is recommended.

7.4 Summary

This chapter presented a method for the generation of the $FHSM$. Connected components labeling was performed on the SDMs to remove connected components of 100 or less pixels, respectively. These connected components can be generally attributed to false alarms. After connected components labeling, the SDM was then transposed onto the CDM and the moving skin-color regions detected. The morphological closing operator was applied to the $FHSM$ to fill any holes present in the connected components.

In the second part of this chapter, we presented techniques for the detection and tracking of the face. Shape features were used to detect the face object in a $FHSM$. Tracking methods based on region projection or Euclidean distances were then employed to track the face in subsequent frames.

(a)

(b)

Figure 7.17: Face tracking results for six consecutive frames of the (a) *Silent* sequence, and (b) *Irene* sequence.

Chapter 8

Conclusions and Future Work

"So Long, and Thanks for all the Fish."

- The title of a book by Douglas Adams

We summarize the major findings presented in this thesis, and present ideas for feature research.

8.1 Conclusions

It was observed by Schumeyer [Sch98] that for effective transmission of sign language video over low bit-rate channels, content-based coding strategies are required. This is mainly due to the presence of rapid hand and arm motion in sign language video, and the necessity of smooth motion perception. Content-based coding requires the segmentation of the video sequence into different objects, which are then independently coded and transmitted. In this way, more resources can be allocated to the perceptually important objects. Besides improving coding performance, the content-based manipulation of video would also enable other functionalities, such as improved error-robustness, and scalability. In sign language video, the perceptually important objects are the face and hands. Therefore, the goal of this thesis was to segment the face and hands in sign language video sequences. The face object was then detected in a reference $FHSM$, and then tracked throughout the sequence. This would have applications in lip-reading, where more resources must be allocated to the face object.

8.1.1 Skin-Color Segmentation

The skin-color segmentation algorithm was presented in Chapter 5. We observed that the YCbCr color space provides an effective use of chrominance information for modeling the human skin-color. Another useful aspect of the YCbCr color space is that it is employed in digital video. We observed that the skin pixels of people from different descent occupy similar regions in the CbCr plane, and therefore the same skin-color model can be applied to segment various skin types. To generate a skin-color model, training images were manually segmented into skin and non-skin classes. The skin-color class was modeled as a bivariate normal distribution in the CbCr plane. Image pixels were classified as skin or non-skin based on their Mahalanobis distance. The classification threshold was derived by considering the probability of classification error. If the *a priori* probabilities of each class are known, the segmentation threshold is derived by minimizing the probability of classification error. However, if the *a priori* probabilities cannot be estimated with any certainty, then the minimax test can be used to derive the segmentation threshold. The skin detection mask (SDM) is a binary map that indicates skin and non-skin color regions.

The proposed algorithm was tested on both still images, and video sequences. The results were evaluated both qualitatively and quantitatively. For a quantitative assessment of the results, we manually segmented each training image into skin and non-skin classes, and compared the manually segmented images with the automatically segmented images. The false alarm and miss rates were evaluated for each frame. The simulation results demonstrated the effectiveness of the proposed algorithm in segmenting skin-color regions in images of different subjects, body poses, lighting conditions, and background complexities.

8.1.2 Statistical Change Detection

In Chapter 6 we proposed a change detection method based on the F test and block-based motion estimation. The proposed technique extended the change detection method proposed by Kim *et al.* [KCK+99]. Change detection is employed for segmenting video frames into "changed" (foreground) and "unchanged" (background) regions with respect to the previous frame. The unchanged regions denote the stationary background, while the changed regions denote the moving and occlusion regions. The foreground and background regions are represented in a change detection mask CDM.

The background difference population was modeled as a zero-mean normal distribution. The F test compares the sample variance of the difference pixels in the observation window with the sample variance of background pixels. To evaluate the background sample variance, we employed a technique based on full search block-based motion estimation. The simulation results were presented in Section 6.4. The simulation results demonstrate that the proposed method can detect hand and face motion in sign language sequences quite effectively.

8.1.3 FHSM Generation, Face Detection, and Tracking

Chapter 7 developed methods for $FHSM$ generation, and face detection and tracking. The $FHSM$ was generated based on information from the skin detection mask (SDM) and the change detection mask (CDM). Connected components labeling was first performed on SDM to remove connected components of 100 pixels (with 8-neighborhood connectivity). Experimental data suggests that these regions can generally be attributed to false alarms. The

$FHSM$ was then generated by comparing the CDM and the SDM. The morphological closing operator was applied to the $FHSM$ to fill any holes present in the face and hand objects. Simulation results demonstrate the effectiveness of the $FHSM$ generation process.

We then turned our attention to face detection and tracking. The face object was detected using shape features, and a reference $FHSM$ formed. The reference $FHSM$ was then used to track the face object in subsequent frames. It was observed that during sign language, the position of the head does not vary much, thus it was possible to use the same reference $FHSM$ throughout the sequence. Tracking techniques based on region projection and Euclidean distances were investigated. It was discovered that when a hand touches or partially covers a part of the face, the face region cannot be detected successfully. Both tracking methods were successful in tracking the head object.

8.2 Future Work

Unlike the face and hand segmentation scheme proposed by Schumeyer [SHB97, SB98, Sch98], our algorithm does not require a separate face detection algorithm to generate a skin-color model. The face detection algorithm imposes an extra overhead on the overall algorithm, and does not guarantee that reliable skin training pixels can be obtained. To obtain skin and non-skin training pixels, Schumeyer manually segmented a frame from the video sequence to be segmented and compressed, and then modeled the skin and non-skin color distributions based on the manually segmented training pixels. Obviously, such a method is not suitable for an automatic system, and its only useful purpose is to test the viability of content-based coding of sign language video sequences. Rather, our approach builds a universal skin-color model, taking into consideration different skin-colors and lighting conditions. Also, our algorithm takes advantage of motion information to eliminate false alarms in the background. Of course, we will not declare that our study of face and hand segmentation problem in sign language video is a solved problem. Thus, there are several avenues of investigation that we feel will lead to future improvements in hand and face segmentation:

- Texture can be used as a third cue to provide a more effective segmentation of the face and hands. As eluded to in Section 6.1, facial features such as the eyes and mouth add

texture to the face object. Also, the fingers add texture to the hand object.

- Shadow cancellation strategies can be employed to identify foreground regions in the CDM that are due to shadow and not due to moving objects. Shadows may lead to false alarms if a shadow foreground region covers 50% or more of a skin-color region.

- Identification of the face object when the face and hands form one connected component. This has applications in face tracking.

- As well as tracking the face object, the hands can also be tracked. If there is little hand motion between two consecutive frames (e.g., the subject is waiting for a response from the other party), change may not be detected in the CDM, and the hand objects not segmented.

- Adoption of the methodology within a MPEG-4 framework. It is also worth measuring the coding performance of the algorithm, and whether modifications need to be made to the algorithm for real-time content-based applications.

Appendix A

The Common Intermediate Format

One major problem in defining an international standard for videoconferencing was the fact that two different line and frame rate television standards exist. NTSC, which is mainly used in North America and Japan, uses 525 lines per interlaced frame at 30 frames per second. In interlaced video, each frame is comprised of fields, i.e. a top field and a bottom field. Within a frame of interlaced video, scanlines from the two fields are interleaved. Most other countries use 635 lines per interlaced picture at 25 frames per second (PAL). To eliminate the problem of interoperability among systems with different formats, a new common intermediate format (CIF) was adapted. Both the 625 and the 525 line systems need to include pre- and postprocessing modules to convert to and from CIF.

CIF is a non-interlaced format. In non-interlaced video, there is no notion of a field. A frame begins from the top left corner and continues through successive lines to the bottom of the frame. The interlaced and non-interlaced schemes are shown in Figures A.1(a) and A.1(b), respectively (adapted from [BK95], Figure 6.1). The CIF format is based on 352 pixels per line, and 288 non-interlaced lines per frame at 30 frames per second. These values represent half the active lines of a 625/25 television signal and the picture rate of a 525/30 NTSC signal. Thus, 625/25 systems need only to perform a picture rate conversion, and NTSC systems need to perform only a line-number conversion.

Color pictures are coded using one luminance and two color-difference components in the YCbCr color space, as specified in ITU-R BT.601 [IR98] (see also Section 3.5.4). The Cb and Cr components are subsampled by a factor of two on both the horizontal and vertical

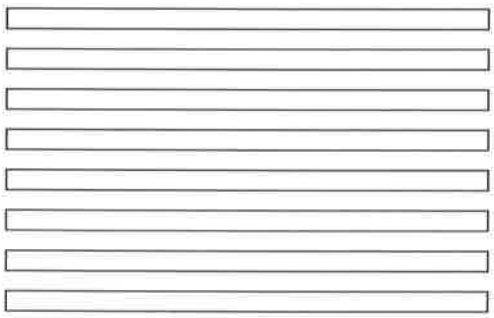
dimensions (known as the 4:2:0 subsampling format) and have 176 pixels per line and 144 lines per frame. This is shown in Figure A.2 (adapted from [BK95], Figure 6.2). The picture area covered by these numbers of pixels and lines has an aspect ratio of 4:3. Table A.1 summarizes the characteristics of a CIF frame.

For low bit-rate applications, in addition to CIF, video coders may also use a quarter-CIF (QCIF) format, which has half the number of pixels and lines required for CIF.

Component	**Size** (pixels × lines)
Y	352 × 288
Cb	176 × 144
Cr	176 × 144

Table A.1: Frame characteristics of the common intermediate format (CIF).

(a)



(b)

Figure A.1: Scanning schemes. (a) Interlaced, and (b) non-interlaced.

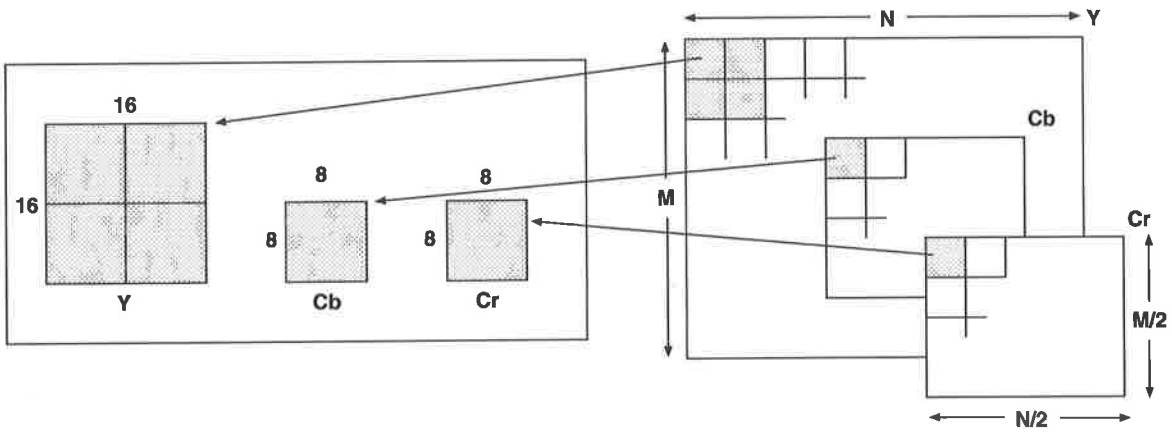


Figure A.2: 4:2:0 subsampling format.

Appendix B

Description of the Video Sequences

The video sequences that were used for segmentation in this thesis were obtained either from the MPEG-4 library of test sequences, or from ITU-T. They were all downloaded over the internet. The sequences are in QCIF format with 4:2:0 subsampling. Two of the sequences, the *Silent* and *Irene*, are sign language sequences. They contain various degrees of background complexity, and are characterized by rapid hand and arm motion, and face and head motion. The other sequences used were the *Carphone*, *Foreman*, *Salesman*, and *Mother & Daughter* test sequences. The *Carphone* sequence contains camera panning. The *Foreman* sequence contains both camera panning and zooming. The *Salesman* and *Mother & Daughter* sequences are typical head and shoulder sequences, and are characterized by smooth movements of the body, head, and facial features such as the eyes and mouth. Note that the *Silent*, *Irene*, *Salesman*, and *Mother & Daughter* test sequences do not contain any global motion. The first frame from each sequence is shown in Figure B.1.

(a) (b)

(c) (d)

(e) (f)

Figure B.1: First frame of the (a) *Silent*, (b) *Irene*, (c) *Carphone*, (d) *Foreman*, (e) *Salesman*, and (f) *Mother & Daughter* sequences.

Appendix C

Additional Simulation Results

This appendix provides additional skin-color segmentation results for still images obtained from the world wide web. These images were part of the test set. Figures C.1, C.2, and C.3 depict results for people with fair skin, Figures C.4, C.5, and C.6 depict results for people of Asian descent, and Figures C.7, C.8, and C.9 depict results for people with dark skin.

Figure C.1: Skin segmentation results for people with fair skin. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

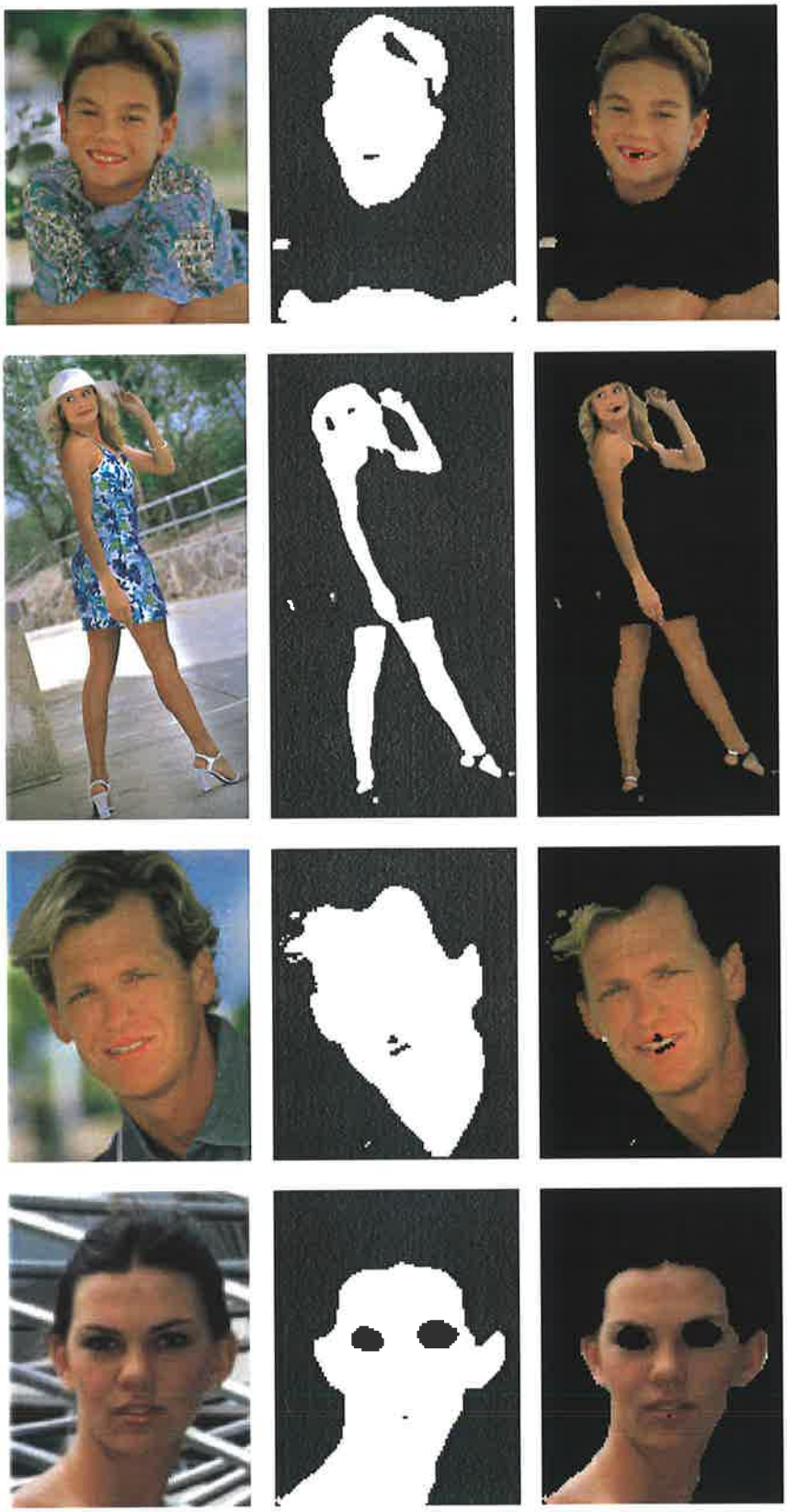


Figure C.2: Skin segmentation results for people with fair skin. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

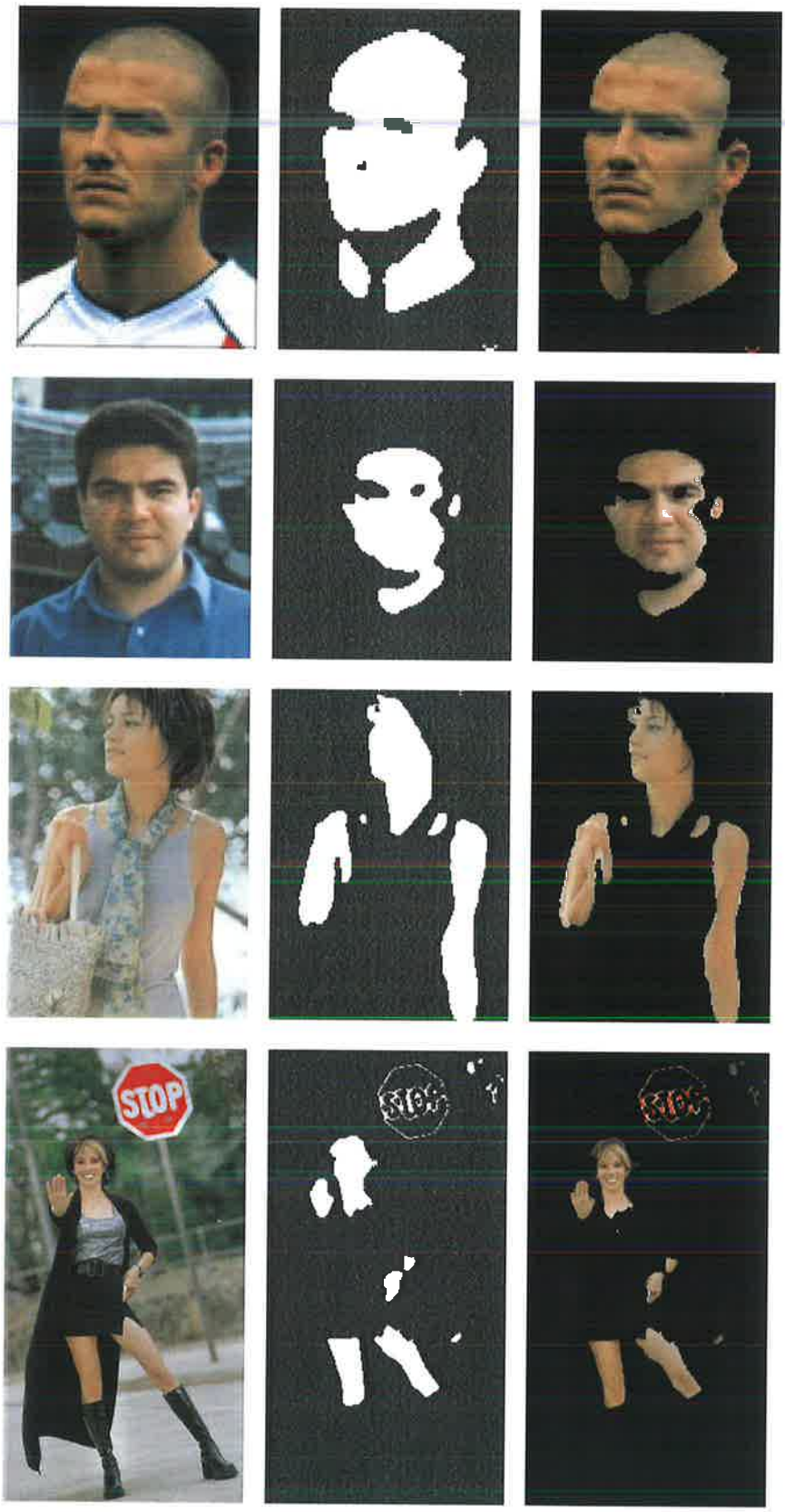


Figure C.3: Skin segmentation results for people with fair skin. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

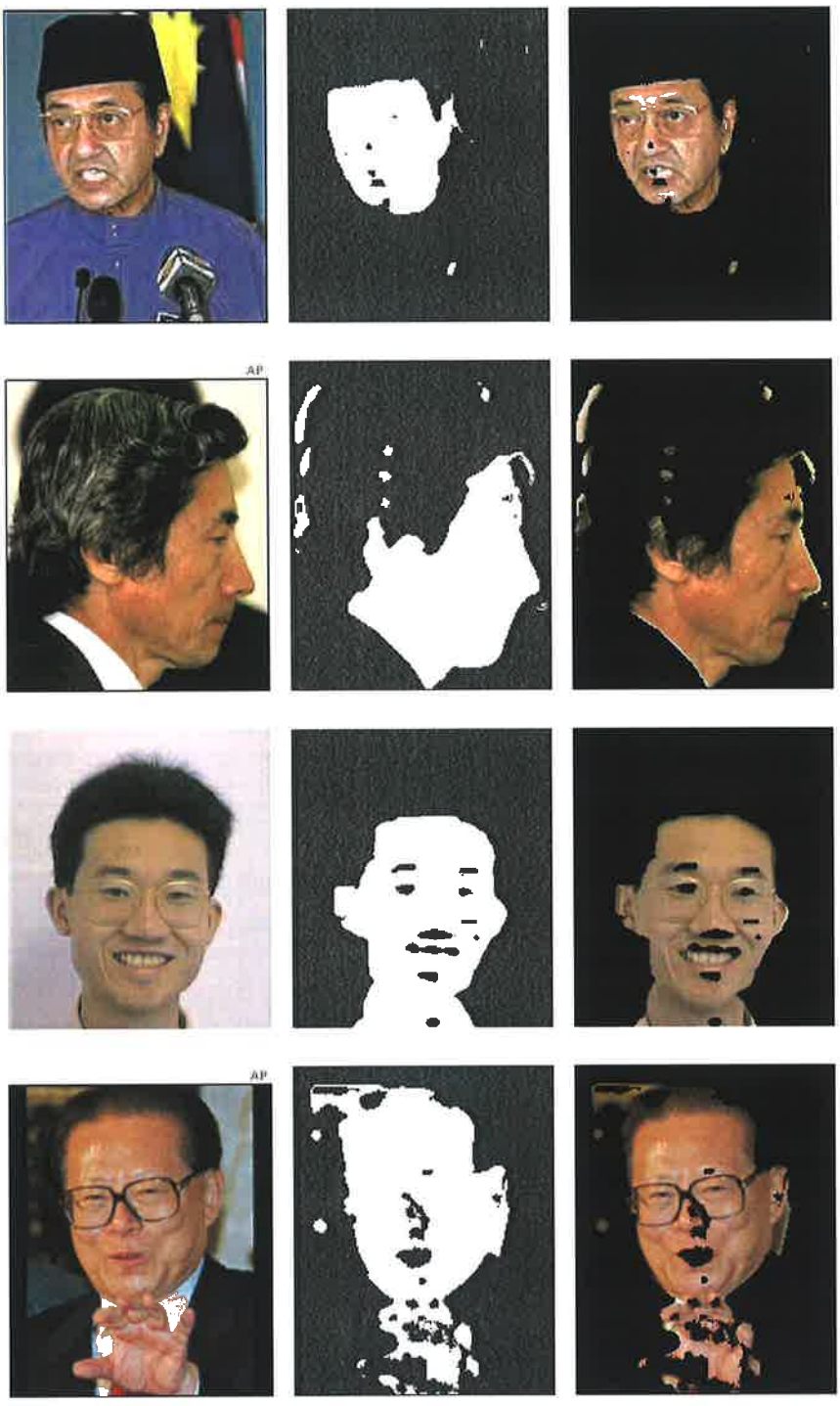


Figure C.4: Skin segmentation results for people of Asian descent. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

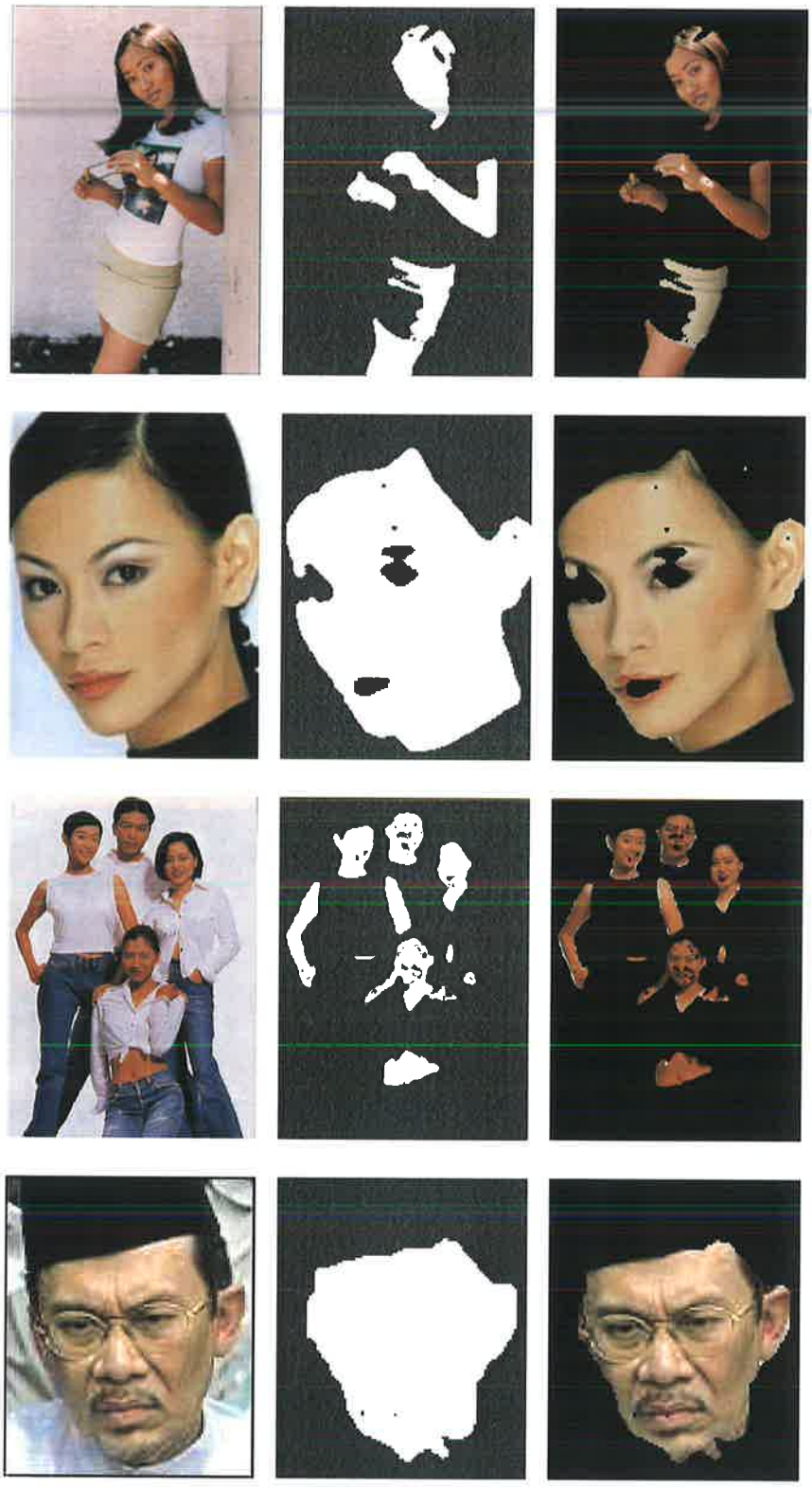


Figure C.5: Skin segmentation results for people of Asian descent. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

Figure C.6: Skin segmentation results for people of Asian descent. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

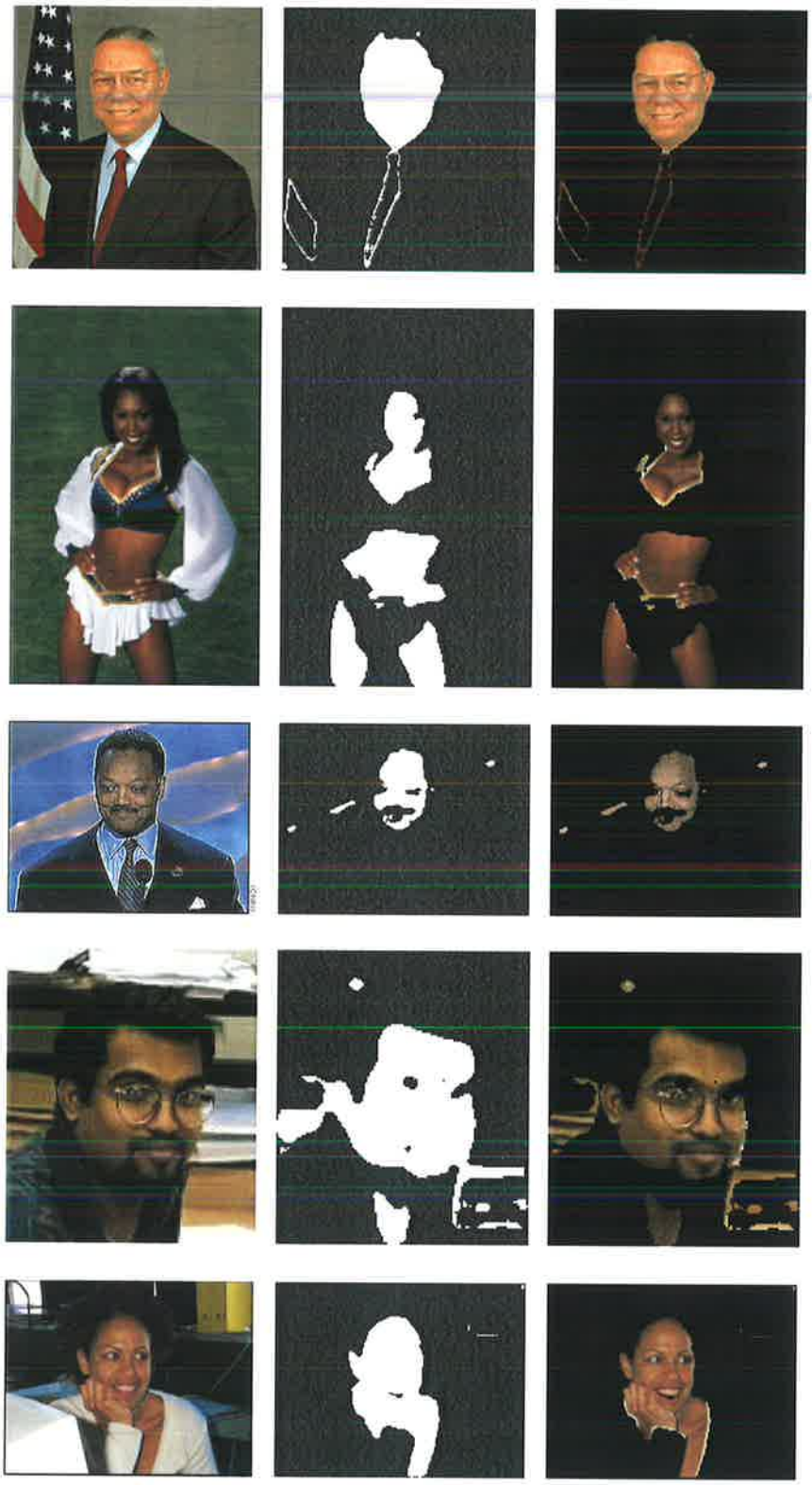


Figure C.7: Skin segmentation results for people with dark skin. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

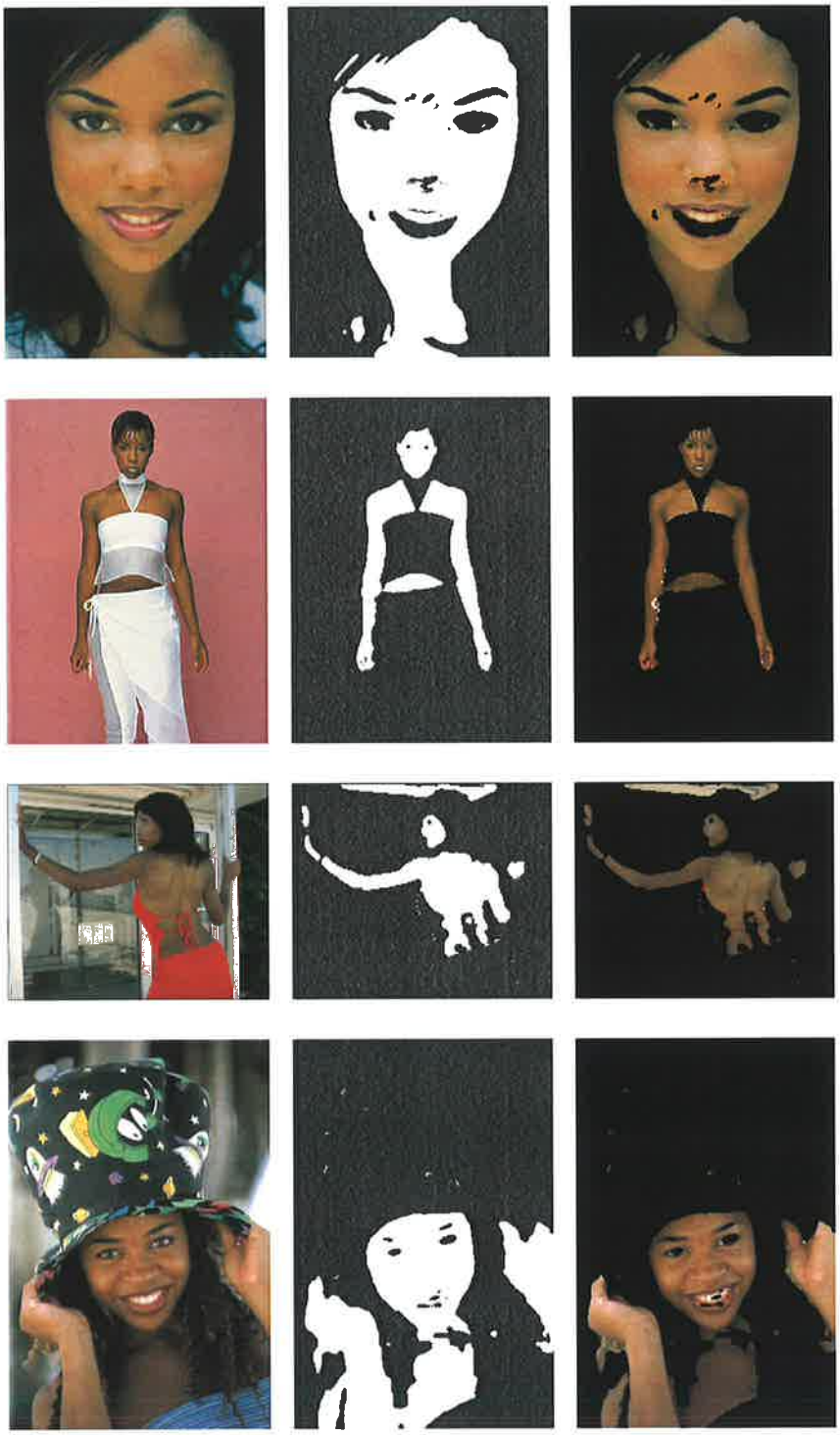


Figure C.8: Skin segmentation results for people with dark skin. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

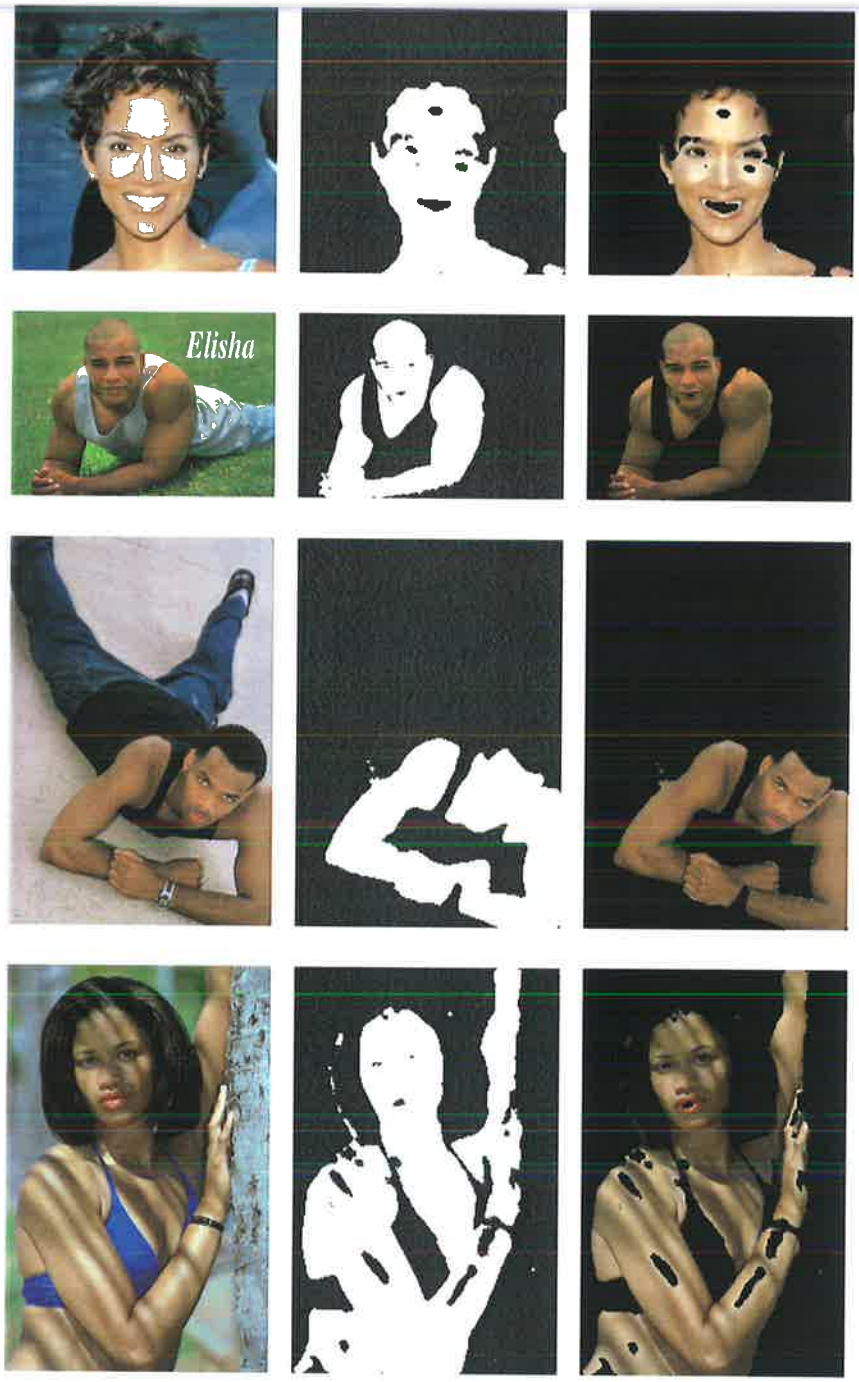


Figure C.9: Skin segmentation results for people with dark skin. Left column: Original image. Center column: SDM. Right column: Identified skin pixels.

Bibliography

[AA01] S. Akyol and P. Alvardo. Finding relevant image content for mobile sign language recognition. In *Proc. International Conference on Signal Processing, Pattern Recognition, and Applications*, pages 48–52, Rhodes, Greece, July 2001.

[AH95] K. Aizawa and T. Huang. Model-based image coding: advanced video coding techniques for very low bit-rate applications. *Proc. IEEE*, 83:259–271, February 1995.

[AKM93] T. Aach, A. Kaup, and R. Mester. Statistical model-based change detection in moving video. *Signal Processing*, 31:165–180, 1993.

[ASW94] D. R. Anderson, D. J. Sweeney, and T. A. Williams. *Introduction to statistics: concepts and applications*. West Publishing Company, St. Paul, Minneapolis, 1994.

[Bez81] J. Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Plenum Press, New York, 1981.

[BK95] V. Bhaskaran and K. Konstantinides. *Image and video compression standards: algorithms and architectures*. Kluwer Academic Publishers, Norwell, Massachusetts, 1995.

[BKKP99] J. Bezdek, J. Keller, R. Krisnapuram, and N. K. Pal. *Fuzzy models and algorithms for pattern recognition and image processing*. Kluwer Academic Publishers, Norwell, Massachusetts, 1999.

[BMG+00] L. M. Bergasa, M. Mazo, A. Gardel, M. A. Sotelo, and L. Boquete. Unsupervised and adaptive Gaussian skin color model. *Image and Vision Computing*, 18:987–1003, 2000.

[Bra99] N. Brady. MPEG-4 standardized methods for the compression of arbitrarily shaped video objects. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(8):1170–1189, December 1999.

[Cas96] K. R. Castleman. *Digital image processing*. Prentice Hall, Upper Saddle River, New Jersey, 1996.

[Cas98] R. Castagno. *Video segmentation based on multiple features for interactive and automatic multimedia applications*. PhD thesis, Swiss Federal Institute of Technology, 1998.

[CB00] D. Chai and A. Bouzerdoum. A Bayesian approach to skin color classification in YCbCr color space. In *Proc. IEEE Region Ten Conference*, volume 2, pages 421–424, Kuala Lumpur, Malaysia, September 2000.

[CCJC91] L. Chen, W. Chen, Y. Jehng, and T. Chiueh. An efficient parallel motion estimation algorithm for digital image processing. *IEEE Trans. on Circuits and Systems for Video Technology*, 1(4):378–385, December 1991.

[CEK98] R. Castagno, T. Ebrahimi, and M. Kunt. Video segmentation based on multiple features for interactive multimedia applications. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(5):562–571, September 1998.

[CHY89] S. Cho, R. Haralick, and S. Yi. Improvement of Kittler and Illingworth's minimum error thresholding. *Pattern Recognition*, 22(5):609–617, 1989.

[CN94] P. Cicconi and H. Nicolas. Efficient region-based motion estimation and symmetry oriented segmentation for image sequence coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 4(3):132–142, June 1994.

[CN99] D. Chai and K. N. Ngan. Face segmentation using skin-color map in videophone applications. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(4):551–564, June 1999.

[Com02] D. Comaniciu. Image segmentation using clustering with saddle point detection. In *Proc. IEEE International Conference on Image Processing*, Rochester, New York, September 2002.

[CTB92] I. Craw, D. Tock, and A. Bennett. Finding face features. In *Proc. Second European Conference on Computer Vision*, pages 92–96, 1992.

[DBH+99] G. Donato, M. S. Bartlett, J. C. Hager, P. Ekman, and T. J. Sejnowski. Classifying facial actions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(10):974–989, October 1999.

[DHS01] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification.* Wiley, New York, 2nd edition, 2001.

[DLR77] A. P. Dempster, N. M. Lairs, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Ser. B*, 39:1–38, 1977.

[eated93] An eigenstructure approach to edge detection. A. H. Tewfik and M. Deriche. *IEEE Trans. on Image Processing*, 2(3):353–368, July 1993.

[EJ95] A. Eleftheriadis and A. Jacquin. Automatic face location detection and tracking for model-assisted coding of video teleconferencing sequences at low bit-rates. *Signal Processing: Image Communication*, 7(3):231–248, September 1995.

[ES98] W. Effelsberg and R. Steinmetz. *Video compression techniques.* dpunkt.verlag, Heidelberg, Germany, 1998.

[EWG00] P. Eisert, T. Wiegand, and B. Girod. Model-aided coding: a new approach to incorporate facial animation into motion-compensated video coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 10(3):344–358, April 2000.

[Fas97] D. Fasulo. Edge detection techniques- an overview. Technical Report 195, Universit de Sherbrooke, 1997.

[Fas99] D. Fasulo. An analysis of recent work on clustering algorithms. Technical Report 01-03-02, University of Washington, April 1999.

[Fuk90] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic Press, Boston, 1990.

[FYEA01] J. Fan, D. K. Y. Yau, A. K. Elmagarmid, and W. G. Aref. Automatic image segmentation by integrating color-edge extraction and seeded region growing. *IEEE Trans. on Image Processing*, 10(10):1454–1466, October 2001.

[gana98] Region growing: a new approach. S. A. Hojjatoleslami and J. Kittler. *IEEE Trans. on Image Processing*, 7(7):1079–1084, July 1998.

[GBL+98] J. D. Gibson, T. Berger, T. Lookabaugh, D. Lindbergh, and R. L. Baker. *Digital compression for multimedia: principles and standards*. Morgan Kaufmann Publishers Inc., San Francisco, Clifornia, 1998.

[GGKV98] B. Girod, R. Gray, J. Kovacevic, and M. Vetterli. Image and video coding. *IEEE Signal Processing Magazine*, 15(2):40–46, March 1998.

[GM90] H. Gharavi and M. Mills. Blockmatching motion estimation algorithms - new results. *IEEE Trans. on Circuits & Systems*, 37:649–651, May 1990.

[Gro98] MPEG Video Group. *Committee draft of MPEG-4*. ISO/IEC JTC1/SC29/WG11 N2202, Tokyo, Japan, May 1998.

[Gro01] MPEG Video Group. *Overview of the MPEG-4 standard V.18*. ISO/IEC JTC1/SC29/WG11 N4030, Singapore, March 2001.

[GSH01] H. Gao, W. C. Siu, and C. H. Hou. Improved techniques for automatic image segmentation. *IEEE Trans. on Circuits and Systems for Video Technology*, 11(12):1273–1280, December 2001.

[GT99] C. Garcia and G. Tziritas. Face detection using quantized skin color regions merging and wavelet packet analysis. *IEEE Trans. on Multimedia*, 1(3):264–277, September 1999.

[GW92] R. C. Gonzalez and R. E. Woods. *Digital image processing*. Addison-Wesley, Reading, Massachusetts, 1992.

[Hay88] W. L. Hays. *Statistics*. Holt, Rinehart and Winston, Inc, New York, fourth edition, 1988.

[HD94] Y. Hu and T. J. Dennis. Textured image segmentation by context enhanced clustering. *IEE Proc. - Vision, Image and Signal Processing*, 141(6):413–421, December 1994.

[Hel97] G. Hellström. Quality measurement on video communication for sign language. In *Proc. 16th International Symposium on Human Factors in Telecommunications*, pages 217–224, Oslo, Norway, May 1997.

[Hel00] G. Hellström. Total conversation, the key to equal status in telecommunication. In *Proc. International Conference on Computers Helping People with Special Needs*, pages 303–312, Karlsruhe, Germany, July 2000.

[HL01] E. Hjelmas and B. K. Low. Face detection: a survey. *Computer Vision and Image Understanding*, 83:236–274, 2001.

[HLM01] N. Habili, C. C. Lim, and A. R. Moini. Hand and face segmentation using motion and color cues in digital image sequences. In *Proc. IEEE International Conference on Multimedia and Expo*, Tokyo, Japan, August 2001.

[HLM02] N. Habili, C. C. Lim, and A. R. Moini. Automatic human skin segmentation based on color information in the YCbCr color space. In *Proc. Information, Decision, and Control*, Adelaide, Australia, February 2002.

[HMB99] N. Habili, A. R. Moini, and N. Burgess. A variable search count block-matching algorithm for video coding. In *Proc. IEEE Region Ten Conference*, pages 108–111, Cheju, Korea, September 1999.

[HMB00a] N. Habili, A. R. Moini, and N. Burgess. Automatic thresholding for change detection in digital video. In *Proc. SPIE Visual Communications and Image Processing*, pages 133–142, Perth, Australia, June 2000.

[HMB00b] N. Habili, A. R. Moini, and N. Burgess. Histogram based temporal object segmentation for VOP extraction in MPEG-4. In *Proc. The Fist IEEE Pacific-Rim Conference on Multimedia*, pages 310–313, Sydney, Australia, December 2000.

[HS81] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[HS92] R. Haralick and L. Shapiro. *Computer and robot vision*, volume 1. Addison-Wesley, Reading, Mass., 1992.

[HSSB96] M. Heath, S. Sarkar, T. Sanocki, and K. Bowyer. Comparison of edge detectors: a methodology and initial study. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 121–130, San Francisco, California, June 1996.

[Inc01] The Western Australian Deaf Society Inc. *Auslan - Australian sign language*. http://www.wadeaf.org.au/auslan.shtml, 2001.

[IR98] ITU-R. *Recommendation BT.601-5: studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*, 1998.

[IR00] ITU-R. *Recommendation BT.709-4: parameter values for the HDTV standards for production and international programme exchange*, 2000.

[IT93] ITU-T. *Recommendation H.261 - video codec for audiovisual services at p x 64 kbit/s*, March 1993.

[IT98] ITU-T. *Recommendation H.263 - video coding for low bit rate communication*, February 1998.

[IT99] ITU-T. *Supplement 1 to series H: application profile - sign language and lip-reading real-time conversation using low bit-rate video communication*, May 1999.

[Ito96] Y. Itoh. An edge-oriented progressive image coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 6(2):132–142, April 1996.

[Jai89] A. Jain. *Fundamentals of digital image processing*. Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[Jas01] Jasc. *Jasc Paint Shop Pro*. http://www.jasc.com/products/psp, 2001.

[JR98] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. Technical Report CRL 98/11, Compaq Cambridge Research Lab., December 1998.

[JR99] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. In *Proc. Computer Vision and Pattern Recognition*, pages 274–280, Ft. Collins, Colorado, June 1999.

[JW82] R. A. Johnson and D. W. Wichern. *Applied multivariate statistical analysis*. Prentice Hall, Englewood Cliffs, New Jersey, 1982.

[JW97] F. Jiulun and X. Winxin. Minimum error thresholding: a note. *Pattern Recognition Letters*, 18:705–709, 1997.

[KC97] S. H. Kwok and A. G. Constantinides. A fast recursive shortest spanning tree for image segmentation and edge detection. *IEEE Trans. on Image Processing*, 6(2):328–332, February 1997.

[KCK+99] M. Kim, J. G. Choi, D. Kim, H. Lee, M. H. Lee, C. Ahn, and Y.-S. Ho. A VOP generation tool: automatic segmentation of moving objects in image sequences based on spatio-temporal information. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(8):1216–1226, December 1999.

[KD00] J. Karlekar and U. B Desai. Finding faces in wavelet domain for content-based coding of color images: two approaches. In *Proc. SPIE Visual Communications and Image Processing*, pages 712–719, Perth, Australia, June 2000.

[KI86] J. Kittler and J. Illingworth. Minimum error thresholding. *Pattern Recognition*, 19(1):41–47, 1986.

[KIH+81] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro. Motion compensated interframe coding for video conferencing. In *Proc. National Telecommunications Conference*, pages C9.6.1–9.6.5, New Orleans, LA, December 1981.

[KJK+01] M. Kim, J. G. Jeon, J. S. Kwak, M. H. Lee, and C. Ahn. Moving object segmentation in video sequences by user interaction and automatic object tracking. *Image and Vision Computing*, 19:245–260, 2001.

[KR90] L. Kaufman and P. J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley, New York, 1990.

[KS01] S. Khan and M. Shah. Object based segmentation of video using color, motion, and spatial information. In *Proc. Computer Vision and Pattern Recognition*, Kauai, Hawaii, December 2001.

[Lap02] J. A. Lapiak. *HandSpeak*. http://www.handspeak.com, 2002.

[LCL+97] M.-C. Lee, W.-G. Chen, C.-L. B. Lin, C. Gu, T. Markoc, S. I. Zabinsky, and R. Szeliski. A layered video object coding system using sprite and affine motion model. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):130–145, February 1997.

[LK81] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130, 1981.

[LZL94] R. Li, B. Zeng, and M. Liou. A new three-step search algorithm for block motion estimation. *IEEE Trans. on Circuits and Systems for Video Technology*, 4(4):438–442, August 1994.

[MB88] G. J. McLachlan and K. E. Basford. *Mixture models: interface and applications to clustering*. Marcel Dekker, New York, 1988.

[MN98a] K. Matthews and N. M. Namazi. A Bayes decision test for detecting uncovered-background and moving pixels in image sequences. *IEEE Trans. on Image Processing*, 7(5):720–728, May 1998.

[MN98b] T. Meier and K. N. Ngan. Automatic segmentation of moving objects for video object plane generation. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(5):525–538, September 1998.

[MN99] T. Meier and K. N. Ngan. Video segmentation for content-based coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(8):1190–1203, December 1999.

[MP97] B. Moghaddam and A. Pentland. Probabilistic visual learning for object recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):696–710, July 1997.

[MPFL96] J. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall. *MPEG video compression standard.* Chapman & Hall, New York, 1996.

[MS97] S. Malassiotis and M. G. Strintzis. Object-based coding of stereo image sequences using three-dimensional models. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(6):892–905, December 1997.

[MW97] R. Mech and M. Wollborn. A noise robust method for segmentation of moving objects in video sequences. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, April 1997.

[MW00a] B. Menser and M. Wien. Automatic face detection and tracking for H.263 compatible region-of-interest coding. In *Proc. SPIE Image & Video Communication & Processing*, pages 882–891, San Jose, California, January 2000.

[MW00b] B. Menser and M. Wien. Segmentation and tracking of facial regions in color image sequences. In *Proc. SPIE Visual Communications and Image Processing*, pages 731–740, Perth, Australia, June 2000.

[MZ01] Y.-F. Ma and H.-J. Zhang. Detecting motion object by spatio-temporal entropy. In *Proc. IEEE International Conference on Multimedia and Expo*, Tokyo, Japan, August 2001.

[NCRT98] A. Neri, S. Colonnese, G. Russo, and P. Talone. Automatic moving object and background separation. *Signal Processing*, 66:219–232, 1998.

[NH95] A. Netravali and B. G. Haskell. *Digital pictures - representation, compression, and standards*. Plenum Press, New York, second edition, 1995.

[NR79] Y. Nakagawa and A. Rosenfeld. Some experiments on variable thresholding. *Pattern Recognition*, 11:191–204, 1979.

[OPR78] R. Ohlander, K. Price, and D. R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8:313–333, 1978.

[oST01] National Institute of Standards and Technology. *Engineering statistics internet handbook*. http://www.itl.nist.gov/div898/handbook/, 2001.

[Pap92] T. Pappas. An adaptive clustering algorithm for Image Segmentation. *IEEE Trans. on Signal Processing*, 40(4):901–914, April 1992.

[PB93] N. R. Pal and D. Bhandari. Image thresholding: some new techniques. *Signal Processing*, 33:139–158, 1993.

[PM96] L. M. Po and W. C. Ma. A novel four-step search algorithm for fast block motion estimation. *IEEE Trans. on Circuits and Systems for Video Technology*, 6(3):313–317, June 1996.

[Poy95a] C. Poynton. *Frequently asked questions about color*. http://www.inforamp.net/~poynton, 1995.

[Poy95b] C. Poynton. *Frequently asked questions about Gamma*. http://www.inforamp.net/~poynton, 1995.

[Poy96] C. A. Poynton. *A technical introduction to digital video*. John Wiley, New York, 1996.

[Poy01] C. Poynton. *YUV and luminance considered harmful: a plea for precise terminology in video*. http://www.inforamp.net/~poynton, 2001.

[PS94] J. Proakis and M. Salehi. *Communication systems engineering*. Prentice Hall, Englewood Cliffs, New Jersey, 1994.

[RE95] P. L. Rosin and T. Ellis. Image difference threshold strategies and shadow detection. In *Proc. 6th British Machine Vision Conference*, pages 347–356, Birmingham, U.K., 1995.

[RL87] P. Rosseeuw and A. Leroy. *Robust regression and outlier detection*. John Wiley, New York, 1987.

[Rom84] H. C. Romesburg. *Cluster analysis for researchers*. Lifetime Learning Publications, Belmont, California, 1984.

[Ros97] P. L. Rosin. Thresholding for change detection. Technical Report ISTR-97-01, Brunel University, UK, June 1997.

[Ros98] P. L. Rosin. Thresholding for change detection. In *Proc. International Conference on Computer Vision*, pages 274–279, Bombay, India, January 1998.

[Ros99] P. L. Rosin. Unimodal thresholding. In *Proc. Scandanavian Conference on Image Analysis*, pages 585–592, Kangerlusuuaq, Greenland, 1999.

[RT99] S. Ray and R. H. Turi. Determination of number of clusters in k-means clustering and application in color image segmentation. In *Proc. 4th International Conference on Advances in Pattern Recognition and Digital Techniques*, pages 137–143, Calcutta, India, December 1999.

[SAG01] M. Störing, H. J. Andersen, and E. Granum. Physics-based modelling of human skin color under mixed illuminants. *Robotics and Autonomous Systems*, 35:131–142, 2001.

[SB98] R. Schumeyer and K. E. Barner. A color-based classifier for region identification in video. In *Proc. SPIE Visual Communications and Image Processing*, pages 189–200, San Jose, California, January 1998.

[Sch98] R. P. Schumeyer. *A video coder based on scene content and visual perception*. PhD thesis, University of Delaware, 1998.

[SGT02] E. Sifakis, I. Grinias, and G. Tziritas. Video segmentation using fast marching and region growing algorithms. *EURASIP Journal on Applied Signal Processing*, 4:379–388, 2002.

[SHB97] R. Schumeyer, E. A. Heredia, and K. E. Barner. Region of interest priority coding for sign language video conferencing. In *Proc. IEEE First Workshop on Multimedia Signal Processing*, pages 531–536, Princeton, New Jersey, June 1997.

[Sik97] T. Sikora. The MPEG-4 video standard verification model. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):19–31, February 1997.

[SM95] T. Sikora and B. Makai. Shape-adaptive DCT for generic coding of video. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):59–62, February 1995.

[SM99] P. Salembier and F. Marqués. Region-based representation of image and video: segmentation tools for multimedia services. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(8):1147–1169, December 1999.

[Sol97] S. J. Solari. *Video and audio compression.* McGraw-Hill, New York, 1997.

[SP96] K. Sobottka and I. Pitas. Segmentation and tracking of faces in color images. In *Proc. Second IEEE International Conference on Automatic Face and Gesture Recognition*, pages 236–241, Killington, Vermont, October 1996.

[SP98] K. Sobottka and I. Pitas. A novel method for automatic face segmentation, facial feature extraction and tracking. *Signal Processing: Image Communication*, 12(3):263–281, June 1998.

[SS01] N. Sarris and M. G. Strintzis. Constructing a videophone for the hearing impaired using MPEG-4 tools. *IEEE Multimedia*, 8(3):56–67, 2001.

[SSW99] R. Sutton-Spence and B. Woll. *The linguistics of British sign language: an introduction.* Cambridge University Press, Cambridge, United Kingdom, 1999.

[ST98] E. Saber and A. M. Tekalp. Frontal-view face detection and facial feature extraction using color, shape and symmetry based cost functions. *Pattern Recognition Letters*, 19:669–680, 1998.

[STB97] E. Saber, A. M. Tekalp, and G. Bozdagi. Fusion of color and edge information for improved segmentation and edge linking. *Image and Vision Computing*, 15(10):769–780, October 1997.

[STEK96] E. Saber, A. M. Tekalp, R. Eschbach, and K. Knox. Automatic image annotation using adaptive color classification. *Graphical Models and Image Processing*, 58(2):115–126, March 1996.

[Sti97] R. Stine. *Animated American sign language dictionary*. http://www.bconnex.net/~randys, 1997.

[TAB97] A. M. Tekalp, Y. Altunbasak, and G. Bozdagi. Two- versus three-dimensional object-based video compression. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(2):391–397, April 1997.

[TDA98] J. C. Terrillon, M. David, and S. Akamatsu. Automatic detection of human faces in natural scene images by use of moments. In *Proc. Third International Conference on Automatic Face and Gesture Recognition*, pages 112–117, Nara, Japan, April 1998.

[Tek95] A. M. Tekalp. *Digital video processing*. Prentice Hall, Upper Saddle River, New Jersey, 1995.

[TG97] J. Theiler and G. Gisler. A contiguity-enhanced k-means clustering algorithm for unsupervised multispectral image segmentation. *Proc. SPIE*, 3159:108–118, 1997.

[Til98] J. C. Tilton. Image segmentation by region growing and spectral clustering with a natural convergence criterion. In *Proc. International Geoscience and Remote Sensing Symposium*, Seattle, Washington, July 1998.

[TOB97] R. Talluri, K. Oehler, and T. Bannon. A robust, scalable, object-based video compression technique for very low bit-rate coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):221–233, February 1997.

[Tre68] H. L. Van Trees. *Detection, estimation and modulation theory*. John Wiley, New York, 1968.

[TRRK98] J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim. A novel unrestricted center-biased diamond search algorithm for block motion estimation. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(4):369–377, August 1998.

[TTE00] C. Toklu, A. M. Tekalp, and A. T. Erdem. Semi-automatic video object segmentation in the presence of occlusion. *IEEE Trans. on Circuits and Systems for Video Technology*, 10(4):624–629, June 2000.

[Tur01] R. H. Turi. *Clustering-based color image segmentation*. PhD thesis, Monash University, 2001.

[Wan98] D. Wang. Unsupervised video segmentation based on watersheds and temporal tracking. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(5):539–546, September 1998.

[WC97] H. Wang and S.-F. Chang. A highly efficient system for automatic face region detection in MPEG video. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(4):615–628, August 1997.

[Wei99] E. Weisstein. *CRC concise encyclopedia of mathematics*. CRC Press, Boca Raton, Florida, 1999.

[WFKM97] L. Wiskott, J. M. Fellous, J. M. Krüger, and C. Von Der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):775–779, July 1997.

[WOZ01] Y. Wang, J. Ostermann, and Y.-Q. Zhang. *Digital video processing and communications*. Prentice Hall, Upper Saddle River, New Jersey, 2001. In print.

[WS82] G. Wyszecki and W. S. Stiles. *Color science: concepts and methods, quantitative data and formulae*. John Wiley & Sons, New York, second edition, 1982.

[YA99] M.-H. Yang and N. Ahuja. Gaussian mixture model for human skin color and its application in image and video databases. In *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases*, San Jose, California, January 1999.

[YKA02] M.-H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: a survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(1):34–58, January 2002.

[You02] T. Young. On the theory of light and colors. *Philosophical Transactions of the Royal Society of London*, 92:20–71, 1802.

[Zar99] B. D. Zarit. Skin detection in video images. Master's thesis, University of Illinois at Chicago, 1999.

[ZC99] F. Ziliani and A. Cavallaro. Image analysis for video surveillance based on spatial regularization of a statistical model-based change detection. In *Proc. 10th International Conference on Image Analysis and Processing*, pages 1108–1110, Venezia, Italy, September 1999.

[Zha98] L. Zhang. Automatic adaptation of a face model using action units for semantic coding of videophone sequences. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(6):781–795, October 1998.

[Zil00] F. Ziliani. *Spatio-temporal image segmentation: a new rule-based approach.* PhD thesis, Swiss Federal Institute of Technology, 2000.

[ZL01] D. Zhang and G. Lu. Segmentation of moving objects in image sequences: a review. *Circuits, Systems, and Signal Processing*, 20(2):143–183, 2001.

[ZYW00] X. Zhu, J. Yang, and A. Waibel. Segmenting hands of arbitrary color. In

Proc. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, Grenoble, France, March 2000.