HAND TRACKING

by

Çağlayan Dicle

B.S. in CSE.,Yeditepe University,2004

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in System and Control Engineering

Boğaziçi University

2007

ACKNOWLEDGEMENTS

I need to convey my gratitude to numerous people for their help and support during the long period of my thesis. First of all, I should thank to Prof. Bülent Sankur who guided me without giving up on hard times, and motivated me to overcome difficulties by myself. Prof. Lale Akarun showed grate support with her inspiring ideas and appreciation. Two distinguished lab mates Oya Çeliktutan and Neslihan Gerek were incredible friends with their care and attention. The moments I have spent in BUSIM lab would not be that much joyful without Erinç Dikici, Sergül Aydöre and Sıdıka Parlak. I would also like to thank to İpek Şen, Helin Dutağacı and Oya Aran for their invaluable mentoring. I am grateful to Erkin Tekeli for his intensive attention and patience which hold me solid through out my study.

I would especially thank to Ekin Balabanlılar for her incredible motivation that was always with me when I needed. The times would be a lot harder without her.

Finally, I am very thankful to my family for their endless love and trust. I love you.

ABSTRACT

HAND TRACKING

This thesis presents a hand detection and tracking system where hand is initialized using the color clue and tracking is achieved with the integration of color and texture information.

Three nonparametric skin classification schemes; histograms, kernel densities, voronoi tessellations are analyzed on six different colorspaces. The optimal fusion of color features is also investigated for illumination free skin classification.

The texture and color cues are combined to track the hand through the course of action. Texture is defined by Local Binary Patterns (LBP), which is a coarse estimation of joint probability of neighboring pixel values. By combining the color with texture more robust representation of hand is attained and meanshift algorithm is used to locate the hand in this representation space. The results show that texture-color combination can deal with face-hand overlaps and confusions of hand with other skin colored regions.

ÖZET

EL TAKİBİ

Bu tezde bir el sezim ve takip sistemi sunulmaktadır. Sezim işlemi renk temelli olup, takip renk ve doku bilgisinin kaynaştırılmasıyla gerçeklenmiştir.

Üç farklı parametrik olmayan sınıflandırıcı; histogram, kernel ve voroni olasılık kestiricileri, altı farklı renk uzayında karşılaştırılmıştır. Ayrıca aydınlanmadan daha az etkilenen bir renk kaynaştırma sistemi sunulmuştur.

El takibi için renk ve doku bilgileri birleştirilmiş ve dokuyu tanımlamak için aydınlanma farklılıklarından etkilenmeyen bir operatör kullanılmıştır. İki bilginin birleştirilmesiyle daha gürbüz bir el tanımlaması ve takibi elde edilmiştir. Edindiğimiz sonuçlar bu yaklaşımın el-yüz örtüşmesini çözebildiğini ve eli arka plandaki nesnelerden daha iyi ayırabildiğini ortaya koymuştur.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

d	Number of dimensions
h	Bandwidth
N	Total number of points
p	Probability
x	Test data
μ	Mean
σ	Standard deviation

ANN	Artificial Neural Networks
CIE	International Commission on Illumination
EM	Expectation Maximization
FFT	Fast Fourier Transform
GLA	Generalized Lloyd Algorithm
GMM	Gaussian Mixture Models
HCI	Human Computer Interaction
LBP	Local Binary Patterns
LDA	Linear Discriminant Analysis
MLP	Multi Layer Perceptron
ROC	Receiver Operator Characteristics
SL	Sign Language
SNR	Signal-to-Noise Ratio
SOM	Self Organizing Maps
SVT	Supervised Voronoi Tessellations
VT	Voronoi Tessellations

1. INTRODUCTION

Gestures form the majority of our communication. It was believed that gestures are the acts of human being that only accompany the speech. However study of Mahrebian [1] revealed the importance of nonverbal acts. In his research Mahrebian found that only about 7 percent of the emotional meaning of a message is communicated through explicit verbal channels. About 38 percent is communicated by paralanguage, which is basically the use of the voice. About 55 percent conveyed through nonverbal channel. Meadow [2] underlines that gesture can both accompany the speech and substitute speech. She claims that gesture enriches the communication of speakers. Either way, gesture provides another representational channel where the message may be conveyed easier.

Carrying bulk of the message and being a powerful channel, gesture is an alternative way of interaction with computers. The most popular mode of *human-computer interaction* HCI devices, mouse and keyboard, are strictly limited in speed and capability for interaction with computer. Gesture is a more natural and continuous mode in that sense. Not only from *human-computer interaction* point of view but also from automatic analysis of *human-to-human interaction* point of view, gesture is important. As Mahrebian pointed out; understanding the human gesture means understanding the bulk of the message.

It is not a surprise, though, many uses of gesture systems have been envisioned and some of them are already implemented. One scenario is sign-to-text / speech translation system. It is translation of *sign language* (SL) gestures to text or speech of a spoken language such as English. In an ideal system, the SL recognition module would have a large and general vocabulary, and be able to capture and recognize manual information without restricting the user or environment.

Another scenario is bandwidth conserving communication between signers through the use of avatars. Sign input which is recognized at one end of the system can be

translated to the other end to animate an avatar. This is a great substitute for live video transmission.

A different scenario is the improving the game playing experience of the user which has several implemented forms in the industry [3]. The user executes the commands of the game by using gesture actions. For example, a gun is aimed, or the sword of game character is controlled by users' hand movements.

A gesture is composed of four main factors: *hand shape*, *hand position*, *hand orientation* and *hand movement*. *Hand shape* is the general appearance of the hand; the position of palm and fingers. *Hand position* is the location of hand in the scene; the closeness to body, face or an object. *Hand orientation* can be explained as the direction of the hand; however it is clearer when palm is considered. The orientation of the hand is the direction where the palm faces. *Hand movement* is the most important element which shows the direction of motion or the trajectory of hand gesture. A gesture recognition system must have correct and continuous flow of these four elements in order to interpret the gesture right. The backbone of such a system is a *hand tracker* which must be robust to dynamic structure of the hand, in other words, it must be a system which can deal with rapid changes in hand appearance and hand movement.

A gesture recognition system is composed of three main stages: *hand initialization*, *hand tracking*, *hand posture recognition* and *interpretation*. *Hand initialization* is locating the hand in the beginning of a gesture, whereas *hand tracking* is the estimation of the hand location based on the previous information of hand. This information may be in terms of 2D position, velocity or acceleration of hand, or may either be shape, color or texture of hand. A *hand tracker* receiving the previous information, outputs the position and motion estimation of the hand. Although the position or hand trajectory information is adequate for some gesture recognition systems like a boxing game scenario where only the rough estimate of user arm motion is needed, for some other scenarios such as SL recognition the posture information of hand is also expected in addition to trajectory. For such systems an additional *hand posture recognition* module is placed after the *hand tracker* to extract the hand pose. *Interpretation*

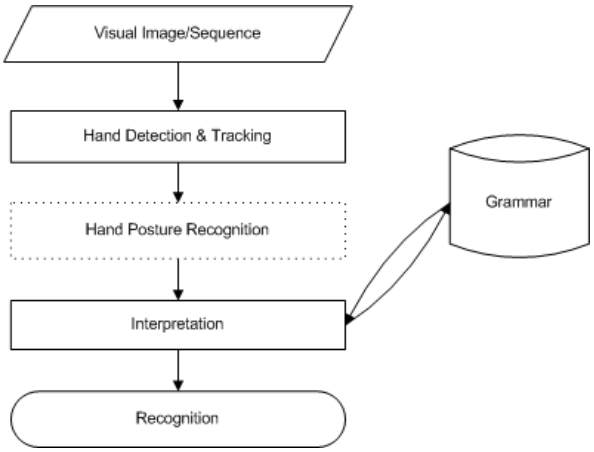


Figure 1.1. Overview of gesture recognition system. The hand position is initialized with a hand detector and hand tracker continuously estimate and locate the hand during the course of action. An optional hand posture recognition may be need for some tasks such as sign language recognition. The recognition of gesture is achieved by interpreting the information supplied by hand tracking and hand posture recognition processes with respect to a grammar where the set of rules lie.

is where recognition of gesture takes place. The *interpretation module* is connected to an external grammar module where a language model or more generally a set of rules lies. Communicating with the grammar module the interpreter distinguishes the gestures from each other.

1.1. Overview of Hand Detection Techniques

A *hand tracker* needs to be initialized to track the hand. The system that realizes this automatic initialization is *hand detector*. The bare task of a *hand detector* is to locate the initial position of the hand. The authors developed numerous approaches (see table 1.1) that can be categorized to four: *marker based*, *body model based*, *appearance based*, *motion based*, *skin color based*.

1.1.1. Marker Based Hand Detection

The earliest efforts to locate the hand in a scene involve colored markers, which are generally colored gloves that are worn by the user. These gloves are single and known colored gloves that can be easily extracted from the scene using the color code of the glove. Starner and Pentland [4] use two different colored gloves for each hand. System initializes seeds of appropriate color, then region growing is applied to segment the hand. Keskin et. al. [5] use a uniform colored glove that is different from background. User moves his/her hand to introduce the system the color of the marker. The system collects the moving pixels and calculates their average hue value which is afterwards used to segment the hand region in the sequence.

Marker based hand detection is a simple solution to locate the hand, yet it is not a natural way of interaction, because the technique requires user to wear a special equipment for interaction.

1.1.2. Body-Model Based Hand Detection

These systems initialize the hand by asking the users to place their hand to known position in the scene or ask them to sit in a predefined location with a predefined pose. Tanibata et. al. [6] define a template of a seated person where the user is required to be seated in a known initial position/pose. The region that coincide the hand location in the template is simply labeled as the hand region. Zieren et. al. [7] distinguish the hand regions from other skin regions using a model that indicates the possible position of hands with respect to each other and with respect to face. Haritaoglu et. al. [8] initiate a cartoon model which represents a person that is standing in an upright pose. It is used to predict locations of body parts such as head, torso, arms and legs. Final location of hand is detected via template matching.

Body models constrain the user considerably, because they require the user be in a known pose. They fail when a reference object, e.g. face, torso, is not present in the scene.

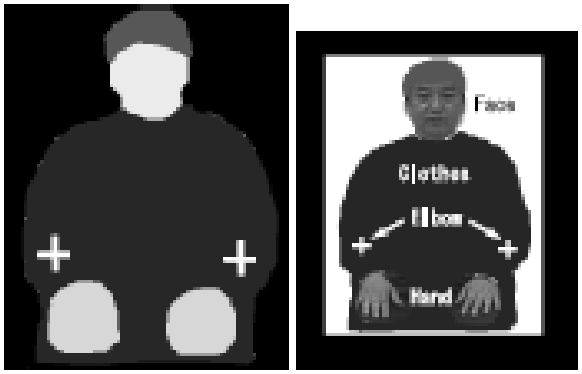


Figure 1.2. Body model based hand detection example. The subject is seated in front of the camera at a known location with a known pose. The hand regions are extracted with respect to body template. The image at the left shows the template and image at the right shows the hand segmentation result.

1.1.3. Appearance Based Hand Detection

Appearance of hand is also preferred by some authors for hand detection. They either, keep a single pose of the hand and ask user to hold that pose, or they train a classifier with several poses and locate the hand by seeking for these poses. Terillon et. al. [9] finds hand location correlating the hand blobs with predefined templates. The authors use phase component of the FFT for correlation process. Their method achieve 82.6% classification rate for 10 different hand postures. Klsch and Turk [10] implement a viola-jones detector [11] for hand localization. The method consists of combination of weak classifiers to obtain one robust classifier. They train their system with 2300 images of 6 hand postures that are taken in different illumination conditions. Their best detector detects and classifies 92.5% correctly. The highest hand detection rates with 99.8% are reported by Ong and Bowden [12] where they also build their detector upon viola-jones methodology. The authors manually form a tree of boosted hand detectors consisting of two layers. The top layer contains general hand detectors and branches specialize in classifying more specific hand shapes.

Although reported rates are high, the appearance based detectors are very sensitive to nuances in hand shape. A minor finger bend may easily fail the detector.

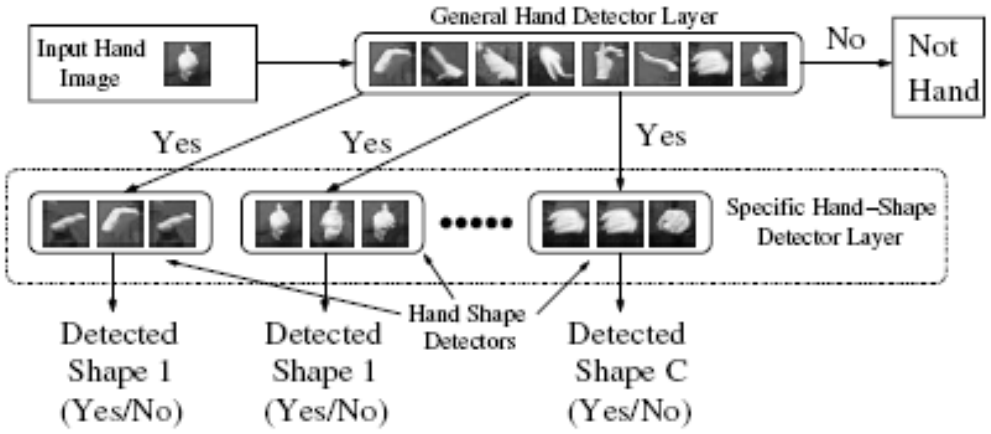


Figure 1.3. Appearance based hand detection example. The Viola-Jones tree is constructed manually where upper levels are trained to detect more general poses of hand and lower levels are trained with more specific poses of hand. This detector detects both location and pose of the hand

Furthermore they are not invariant to hand rotation and pose.

1.1.4. Motion Based Hand Detection

To locate the hand several other authors benefit from motion of the hand. They assume a continuous hand movement and detect the motion or rapid appearance change in the scene to detect the hand. Yuan et. al. [13] rely on the assumption that appearance of the hand changes more frequently than the other objects in the background. They apply block based matching between consecutive frames and label the regions which have high matching error. Akyol et. al. [14] estimate motion probabilities in the scene using motion history images. Final location of hand is detected with watershed segmentation.

The motion based techniques when used alone brings poor performance because the absence of movement means no detection. Motion segmentation results in over-segmented regions which is hard to process afterwards to locate the hand.

Figure 1.4. Motion based hand detection example. The hand regions are detected based on the assumption that hand move and appearance more frequently than other objects. The brighter areas on the left image indicates existence of a hand. The image on the right shows detected hands.

1.1.5. Skin-Color Based Hand Detection

Skin color based hand detection is the most accepted technique in hand detection literature. Either it is used alone or combination with other techniques, it is the most natural way of detection that constrains the user least. This technique relies on the discriminative characteristic of human skin color. Authors usually train a generic skin model and apply a pixel-based skin detection to identify the possible hand regions. In Starner et. al. [15] the system initializes a seed for each hand with respect to a skin model and region growing is carried out to segment hands of the user. Raja et. al. [16] use Gaussian Mixture Models estimate the probability densities of skin color, clothing color and background color. The hand and face locations are identified with these models. The hand is distinguished from the face based on upright posture assumption of the person. The system labels lower blobs as hand regions. Yang et. al. [17] combine skin color cue with geometrical cues. They apply a skin detection to initialize the skin blobs in the images, followed by a region merging process until the blobs have an elliptical shape. They label the biggest blob as head and remaining blobs as hands. Akyol et. al. [14] and Shan et. al. [18] employs skin detection with motion detection to obtain a more robust estimate of hand location.

The human skin color is a strong cue that helps to identify the hand in a scene. It is able to discriminate hand and face from other objects. Moreover it is invariant to scale, rotation, occlusion or pose of the hand. Together with additional information

Table 1.1. The cues used to detect the hand. The plus signs shows the employed cues by the method(s) on the left.

Method	Marker	Body Model	Motion	Skin Color	Appearance/Shape
[4]	+				
[5]	+		+		
[6]		+			
[7]		+	+	+	
[8]		+	+		+
[13]			+		
[14],[19]			+	+	
[15]				+	
[16]		+		+	
[17],[9]				+	+
[10],[12]					+

such as motion, shape, texture etc. the technique has the potential for robust hand detection.

1.2. Overview of Hand Tracking Techniques

After initialization of the hand position by a hand detector, the position and motion information is supplied by a hand tracker. Some authors [10],[12],[15] use detection methodology iteratively to locate the hand separately at every frame. However that is not a real tracking strategy and detectors can not capture dynamics of a hand which limits their tracking capability. For these reasons this section analyzes the tracking mechanisms that estimate the position of the hand based on the previous state of hand.

1.2.1. Appearance Based Hand Tracking

Appearance based tracking methods tracks the hand by matching the portions of hand between two successive frames. Yuan et. al. [13] assume hand undergo rapid

Figure 1.5. Appearance based hand tracking example. The image is segmented to homogenously moving regions and tracking is achieved matching those regions from frame to frame. In the lower row every region is represented with a different gray level value.

appearance changes from frame to frame. They track the hand applying a block based matching between consecutive frames and label the blocks which bring high matching error. Afterwards they apply a temporal filter with respect to a motion model to identify the consistent hand trajectories.Yang et. al. [17] apply a multi-scale motion segmentation to identify homogenously moving regions in each frame. Then these regions are matched between consecutive frames to obtain correspondences. Affine transformations and accordingly motion is estimated from matched region pairs.

Appearance based methods have to rely on the assumption that hand shape doesn't change between consecutive frames. Because otherwise they would not be able to find matches from frame to frame. This implicit assumption limits appearance based methods in hand tracking, because hand dynamics rarely obey this assumption. As a result region matching ambiguities occur leading to false tracking. Furthermore, these methods are easily affected by environmental conditions like illumination and complex background.

1.2.2. Motion Based Hand Tracking

Motion based tracking methods model the motion of the hand using e.g. velocity, accelerationetc. of hand. These methods assume smooth trajectory that the hand follows. They are usually embedded with another method that extracts the exact location of hand. Haritaoglu et. al. [8] employ a second order motion model to estimate

the hands' location in subsequent frames. The prediction from this model is used to estimate bounding box location of hands. Then these bounding boxes are compared to actual hand images to determine the final position of hand. To obtain hand position in the next frame Zieren et. al. [7] apply Kalman prediction and locate the exact position of hand with mean shift algorithm. They run the mean-shift algorithm on a probability map which is combination of skin probabilities and motion probabilities. CONDENSATION is a popular method proposed by Isard and Blake [20]. Their system is based upon sampling the posterior distribution in the previous frame. It propagates these samples iteratively to successive frames. It combines the motion model with visual observations to accomplish robust tracking of hand.

Motion based tracking methods alone is useless, because they only mimic the movement characteristics of hand and they need a feedback to continuously update the model to bring accurate estimates. They make a close estimate of next hand location. On the other hand the inclusion of additional methods that can precisely locate the hand from a motion based estimate can achieve high tracking performances.

1.2.3. Feature Based Hand Tracking

Feature based tracking methods are most popular approaches in hand tracking. They consist of extracting a feature that can represent the hand invariant from scale, rotation and orientation and apply the Mean-shift [21],[22] algorithm on that feature-space. Mean-shift is a gradient based search mechanism which can locate the modes in differentiable surfaces. In the original algorithm Commaniciu et. al. [21] represent the objects with their color probability distributions using a $16x16x16$ RGB histogram and apply mean-shift on this probability feature-space. Polat and Ozden [23] combine Mean-shift tracking with motion detection to deal with background color confusions. They utilize a statistical foreground modeling based on kernel density estimation techniques and detect changes in the scene based on this model. They combine the motion probability estimate with color distribution probability estimate to obtain better object tracking.

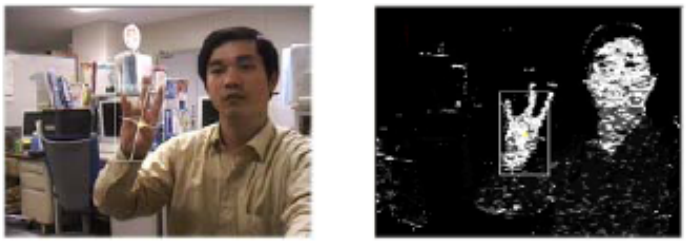


Figure 1.6. Feature based hand tracking example. The skin probability image is calculated with respect to a generic skin model and the tracking is carried out on this feature space.

Mean-shift is a very simple general purpose tracking mechanism. The most important part is the selection of feature that is able to represent the object and distinguish this object from others. Color distribution has been a very powerful representation of objects for many cases which is also robust to orientation, occlusion and scale. However by its own color is not discriminative enough for hand in a gesture recognition scenario. The existence of face and skin-similar colored background objects arise the need of an additional representation.

1.2.4. Limitations of Hand Trackers

There are several points that majority of the hand tracker lacks. If the skin color is used, the performer have to wear long-sleeve clothing and background should be free of skin colored objects. If motion cues are used, hand is assumed to be only moving object on a stationary background. The constant movement of hand is also mandatory for motion-based trackers. Shape based approaches limit the user clothing and background as well. They fail when hand experiences a partial occlusion, or an overlap with the face. Moreover shape-based trackers suffer from dynamic appearance of hand.

Table 1.2. Restrictions and constraints used in hand tracking.

Constraint	Method
long-sleeved clothing	[17],[7],[21],[23]
uniform background	[13],[17],[7]
stationary background	[13],[17],[8],[7], [20],[23]
stationary or minor head movements	[13],[23]
constant hand movement	[13],[23]
fixed body location and pose	[8],[7]
unnatural signing to avoid face overlap	[17],[21]

1.3. Contribution of the thesis

In this thesis skin color detection method is selected to detect the hand position and initialize the tracker. The state-of-art approach [24] *histograms* which belongs to nonparametric density estimation family was systematically compared with two more nonparametric density estimation methods: *kernel density estimates* and *voronoi tessellations*. We propose a general purpose, parameter free density estimation and classification scheme which is shown to be robust on skin classification problem. The effect of colorspace transforms on skin detection problem is also carefully investigated. To improve the performance of a classifier an optimal feature fusion and selection algorithm is illustrated and tested on skin detection.

A new and general purpose tracking methodology is proposed which is based on dynamic texture representation and mean-shift tracking. This feature representation scheme allows keeping the spatial relationships of object appearance still being invariant to rotation and scale.

2. PIXEL-BASED SKIN DETECTION

The human eye may not distinguish an individual pixel whether it is a skin or not, because human cognition needs higher level information that indicates the spatial relationships. However the human skin has a characteristic color which is distinct from many other objects in the nature. In other words, color of human skin, only by itself, has a discriminative power which can be used to identify the hand regions. Thus a skin detector may eliminate non-hand regions and increase the robustness of hand detection.

It is claimed that the skin color has a consistent hue [25], that every human being has the same hue and different saturation change the appearance of skin. However this is not completely true that skin color appearance has a nonlinear relationship [26] with illumination. It is also affected from environmental colors, because of the refractions from these colored surfaces deviate the skin color appearance. Moreover the capture media e.g. camera, scanner, television influence the skin color appearance in a non-standard way.

Bulk of the pixel-based skin detection studies concentrates on statistically representing the skin color distribution and identifying the skin color region using one of *piecewise-linear decision boundaries*, *density estimation* combined with Bayesian decision or *artificial neural networks*. Other color representations than *RGB*, ranging from standard color spaces *YCrCb*, *HSV*, *CIELab* to pseudo color spaces [26], [27],has been examined for better skin color segmentation. In short a skin detection task needs consideration two elements which are representation of skin colors in a colorspace and a classifier that can draw the skin region boundaries clearly.

2.1. Color Representation

The color representation is the main issue that should be considered in a skin detection task. The key reason of color transformation is to decrease or to remove the effect of illumination and environmental factors. A second reason is to increase

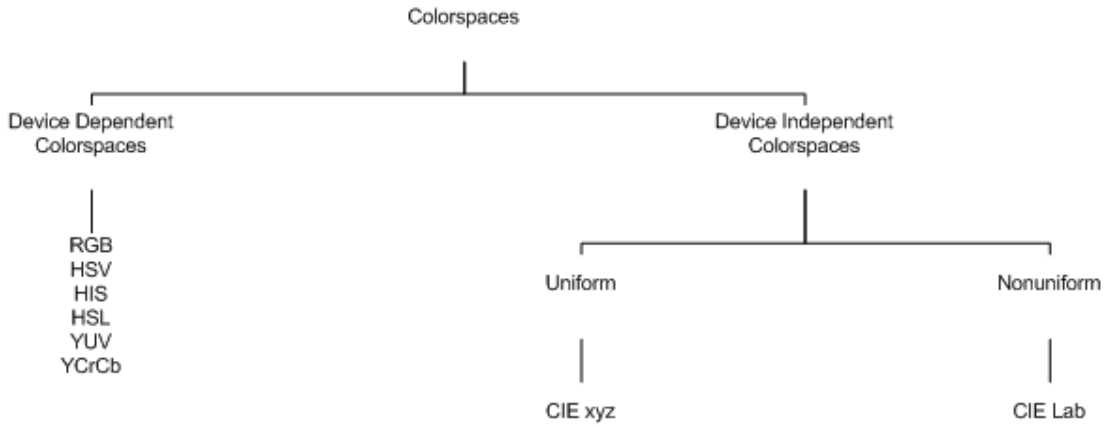


Figure 2.1. Popular colorspaces used in color based studies.

the compactness of skin region that will also increase the discrimination of human skin color from others. The most preferred color representations are *RGB*, *normalized RGB*, *CIELab*, *CIExyz*, *HSV*, *YCbCr*, *YIQ* and *YUV*.

2.1.1. RGB

RGB is a colorspace originated from CRT (or similar) display applications, when it was convenient to describe color as a combination of three colored rays (red, green, blue). It is one of the most widely used colorspaces for processing and storing of digital image data. However, high correlation between channels, significant perceptual non-uniformity mixing of chrominance and luminance data make RGB not a very favorable choice for color based analysis and recognition algorithms.

2.1.2. Normalized RGB

Normalized RGB is a representation that is easily obtained from the RGB values by a simple normalization procedure:

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

where R, G, B are Red, Green, Blue values respectively and r, g, b are normalized versions of them. Because the sum of three normalized components is known ($r+g+b = 1$), the third component does not hold any significant information and can be omitted, reducing the space dimensionality. The remaining components are often called pure colors, yet the dependence of r and g on the brightness of the source RGB color is diminished by the normalization. A remarkable property of this representation is that matte surfaces are invariant (under certain assumptions) to relative changes of surface orientation to the light source.

2.1.3. HSI, HSV, HSL - Hue Saturation Intensity, Value, Lightness

Hue-saturation based colorspaces are outcome of numerical representation need of color properties. They describe the color with intuitive values, based on the artist's idea of tint, saturation and tone. Hue defines the dominant color, such as red, green, purple and yellow, of and area, saturation measures the colorfulness of an area in proportion to its brightness. The intensity, lightness or value is related to the color luminance. The intuitiveness of the colorspace components and explicit discrimination between luminance and chrominance properties make these colorspaces popular in skin color segmentation studies.

2.1.4. CIExyz

The *International Commission on Illumination* CIE has defined a human "Standard Observer", based on measurements of the color-matching abilities of the average human eye. Using data from measurements made in 1931, a system of three primaries, XYZ, was developed in which all visible colors can be represented using only positive values of X, Y and Z. The Y primary is identical to Luminance, X and Z give coloring information. This forms the basis of the CIE 1931 XYZ color space, which is fundamental to all colorimetry. Values are normally assumed to lie in the range [0;1]. Colors are rarely specified in XYZ terms, it is far more common to use "chromaticity coordinates" which are independent of the Luminance (Y). The main advantage of CIE XYZ, and any color space or color definition based on it, is that it is completely

device independent. The main disadvantage with CIE based spaces is the complexity of implementing them; in addition some are not user intuitive.

2.1.5. CIELab

CIELab is based directly on CIExyz and is an attempt to linearize the perceptibility of color differences. The non-linear relations for L, a, and b are intended to mimic the logarithmic response of the eye. Coloring information is referred to the color of the white point of the system.

2.1.6. YCrCb, YUV, YIQ

YUV and YIQ are standard color spaces used for analogue television transmission. YUV is used in European TVs (PAL) and YIQ in North American TVs (NTSC). YCbCr is usually used for image processing work. These colors spaces are device-dependent, like RGB, but they are calibrated. This is because the primaries used in these television systems are specified by the relative standards authorities. Y is the luminance component and is usually referred to as the luma component, U,V or I,Q are the chrominance components (i.e. the color signals).

2.2. Classifier

2.2.1. Piecewise-Linear Decision Boundaries

This model implies drawing the skin cluster with some number of rules. Single or multiple ranges of threshold values for each colorspace component is defined and image pixels that fall into this defined cluster is labeled as skin pixels.

Dai and Nakano [28] use I component of YIQ colorspace which holds the colors from orange band to cyan band. The pixel values that are in the range $R_I = [0, 50]$ are identified as skin pixels. Sobottka and Pitas [29] define the skin region Hue-Saturation colorspace within the ranges $R_H = [0, 50]$ and $R_S = [0.23, 0.68]$. Chai and Ngan [30]

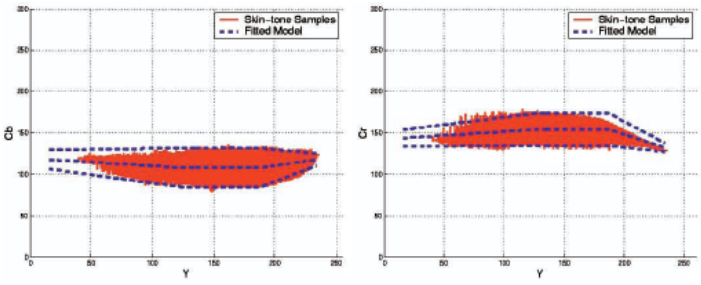


Figure 2.2. Piecewise linear decision boundary based skin detection example. The skin region is defined by a set of lines or curves. A pixel falls into that region is classified as a skin pixel.

applied their face detector in the CbCr colorspace and identify skin pixels that fall into $R_{Cb} = [77, 127]$ and $R_{Cr} = [133, 173]$. Hsu et. al. [26] defined an elliptical decision boundary on a pseudo colorspace based YCbCr.

2.2.2. Density Estimates

In this method the color distribution of skin tone in a colorspace is represented via a density estimation technique. Nonskin pixels may further be modeled with same technique. If only skin tone estimate is present a thresholding is applied that classifies the pixels as skin pixels which have a skinness probability above that threshold. On the other hand, if an estimate of nonskin tones is also present, the final decision is taken by comparing the likelihood probabilities or using the Bayesian rule. The most important issue while using this technique is to obtain an accurate density estimates of skin and nonskin tones.

2.2.2.1. Histogram Models. In this method a 2D or 3D histogram is used to represent the skin tones in some colorspace. The histogram is quantized into a number of histogram bins where each histogram bin stores the count associated with the occurrence of the bin color in the training dataset. The histogram bin counts are converted into probability distribution, $P(c)$ as follows:

$$P(c) = \frac{count(c)}{T} \tag{2.2}$$

where $count(c)$ gives the count in the histogram bin associated with color c and T is the total count of all histogram bins. These values correspond to the likelihood that a color belongs to a skin or nonskin class.

Zarit et. al. [31], Yoo and Oh [32] classify pixels as skin that has a skinness likelihood greater than a predefined threshold. Jones and Rehg [24] examine a 3D RGB histogram model with two billion pixels collected from 18,696 web images. They compute two different histograms for skin and nonskin colors. They compute class conditional probabilities:

$$P(c|skin) = \frac{s(c)}{T_s},\ P(c|nonskin) = \frac{n(c)}{T_n}, \tag{2.3}$$

where $s(c)$ is the count in c-bin of skin histogram. $n(c)$ is the count in the color c-bin of the nonskin histogram. T_s and T_n represents the total counts in the skin and nonskin histogram bins respectively. Jones and Rehg reach to a decision using the rule:

$$\frac{P(c|skin)}{P(c|nonskin)} \geq \theta \tag{2.4}$$

where $0 \leq \theta \leq 1$ is a threshold value which can be adjusted to trade-off between false positives and true positives.

The histogram method is easy and simple table lookup is needed to compute likelihood probabilities. Thus it is also used by Albiol et. al. [33], Shin et. al.[34] , Fleck et. al. [35], Gomez et. al.[27], Zarit et. al. [31], Sigal et. al. [36].

2.2.2.2. Gaussian Mixture Models. Many representative works of skin color modeling have used Gaussian Mixture Models (GMM). The advantage of this parametric model is it's generalization ability with less training data and also have less storage requirements with respect to histograms. The technique relies on weighted combinations multivariate

Gaussians:

$$p(c) = \sum_{i=1}^{N} w_i \frac{1}{(2\pi)^{1/2}|S_i|^{1/2}} \; exp\left[-\frac{1}{2}(c-\mu_i)^T S_i^{-1}(c-\mu_i)\right] \tag{2.5}$$

where c is the color vector and μ_i and S_i are the mean and covariance matrix of ith Gaussian. N is the total number of Gaussians and w_i corresponds to weight of ith Gaussian. The parameters of GMM are estimated from data through the iterative expectation-maximization (EM) technique. The initialization of training parameters are obtained using k-means algorithm. Because there is not any commonly used and sound technique to choose the number of Gaussians, the authors apply their own evaluation scenarios (i.e. visual inspection).

The classification is done as in histogram case. Either a single skin likelihood is thresholded, or skin and nonskin likelihoods are compared using equation (2.4). Terrillon et. al. [37] apply GMM to nine color spaces and compare their skin classification performances with GMM. Jones and Rehg [24] train two GMM for skin and nonskin distributions with 16 Gaussians. On the other hand Lee and Yoo [38] used 6 Gaussians to train GMM on the same dataset.

2.2.3. Artificial Neural Networks

Artificial Neural Networks (ANN) have been successfully applied in pattern recognition tasks due to their ability to learn complex relationships between input and output and their ability to generalize the given data. Chen and Chiang [39] used a Multi Layer Perceptron (MLP) in CIE- xy space for skin classification whereas Phung et. al. [40] used MLP in CbCr space.

Another ANN skin detection methodology is proposed by Brown et. al. [41] based on Self Organizing Maps (SOM). The SOM algorithm is based on unsupervised, competitive learning. The SOM constructs a topology preserving mapping from the high-dimensional space onto map units in such a way that relative distances between data points are preserved. The map units, or neurons, usually form a two dimensional

Table 2.1. Performances of pixel-based skin detection methods

Paper	Colorspace	Classifier	Test Dataset	TD	FD
Jones and Rehg,[24]	RGB	Histogram	Compaq	**90**	**14.2**
	RGB	GMM	Compaq	**90**	**15.5**
Zarit et. al.,[31]	nRGB	Histogram	128 images	**90**	**19**
	YCrCb	Histogram	128 images	**89**	**14.1**
Gomez,[27]	Pseudo	Histogram	31,450 images	**97**	**5.1**
Brown et. al.,[41]	TSL	SOM	Compaq	78	32
Phung et. al.,[40]	YCrCb	Piecewise Linear	ECU	82	18
	RGB	MLP	ECU	88.5	10
Hsu et. al.,[26]	YCrCb	Piecewise Linear	HHI	**96.6**	n/a

regular lattice where the location of a map unit carries semantic information. The SOM can thus serve as a clustering tool of high-dimensional data. Brown et al. [41] train two separate SOMs with 500 images to learn distributions of skin and nonskin color tones. They use different number of mapping units and compared their setups with GMMs. The SOMs perform consistently better than GMMs.

2.3. Discussion

The piecewise linear decision boundaries are fixed and are determined empirically from the visual inspection of skin color distribution. This method has the advantage of being simple and fast. However, it has many limitations. The skin region differs from one illumination source to another and from one capture source to another. It is not possible to represent these skin color deviations using fixed ranges.

High recognition rates, 90% with 14.2% false detection [24] has been reported with histogram technique. Histogram classification performance is usually above the GMMs and ANNs. However due to its inability to interpolate and generalize the data, the histogram method needs a very large training dataset to get a good classification rate. A 3D histogram with 256 bins per channel has 256^3 bins. Due to the sparse distribution of skin points, the number bins is reduced. This helps in creating more

compact and true estimation of the skin probability densities. However, the number of bins that gives the best performance varies with colorspace representation and the size of the training dataset.

GMMs have been a popular choice for skin-color segmentation as they can generalize very well with less training data. The mixture models approximate the skin-color distribution effectively, when the initialization of parameters, number of Gaussians and their centers can be set correctly. However there is no concrete methodology to set these parameters, so the task is done empirically.

ANN methods achieve comparable performances with small number of data and they usually need less storage space. SOM is a good architecture with only one parameter that is number of mapping units. On the other hand, an MLP needs to be set the number of layers, number of the hidden units and the hidden unit function.

Even though several authors concluded that skin cluster has a Gaussian shape [42],[38],[36], it is not completely true. Furthermore, the number of Gaussians that form this distribution is not clear. The best way to estimate the skin distribution is to apply a nonparametric approach and estimate the density without relying on any assumption. The nonparametric methods will not drastically be affected with colorspace transformations, because they do not assume an overall shape about the distribution. The next chapter develops these ideas applying the nonparametric methods; *histograms*, *kernel density estimators* and *voronoi tessellations* on six colorspaces.

3. NONPARAMETRIC SKIN CLASSIFICATION

When there are concrete cues or expertise about a domain, the assumptions may lead to a robust classifier. The assumptions may fill the lacks of training data with the expertise and bring accurate estimations. In the opposite case, when the assumptions do not overlay with real data distribution the results are dramatic. In skin classification case, several authors claim that the skin color distribution resembles the normal distribution which is true for a single person data. However for more general cases the skin color distribution is somehow an arbitrary distribution which can not be explained with a single gaussian. It is most appropriate not to assume anything about the skin distribution and apply a nonparametric model. Nonparametric models are able to imitate the probability densities quite accurately. Consequently, nonparametric models perform equally well in any representation of skin color, because they do not rely on any overall assumption about data distribution.

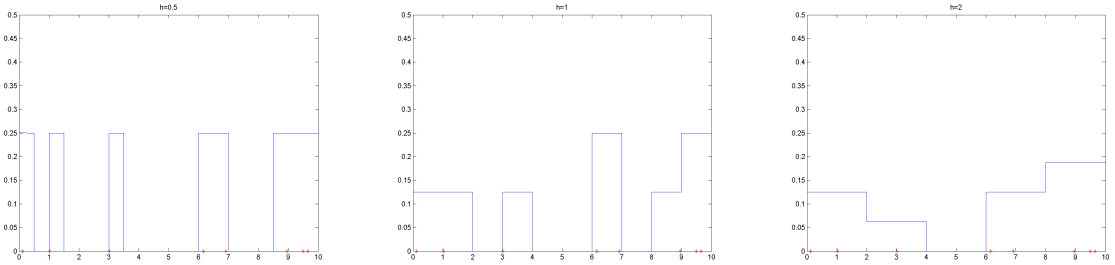
This chapter analyzes histograms, kernel density estimator and voronoi tessellations; compare their performances on skin classification problem. The models are used to form the closest density estimates of skin and nonskin colors, the classification is achieved by comparing the likelihoods of given pixels.

3.1. Histogram

Histogram is the set of non-overlapping bins, where frequency is the number of data points that fall into those defined intervals (bins). It is the simple outcome of "collecting together similar events" logic. Keeping in mind bins may have different widths, in practice the same width is chosen for all bins and histogram is defined with this single parameter. The best bin width value is estimated empirically.

The histogram estimate is defined as,

$$\hat{p}(x) = \frac{\sharp\{x_i \; in \; the \; same \; bin \; as \; x\}}{Nh} \tag{3.1}$$



(a) h=0.5 (b) h=1.0 (c) h=2.0

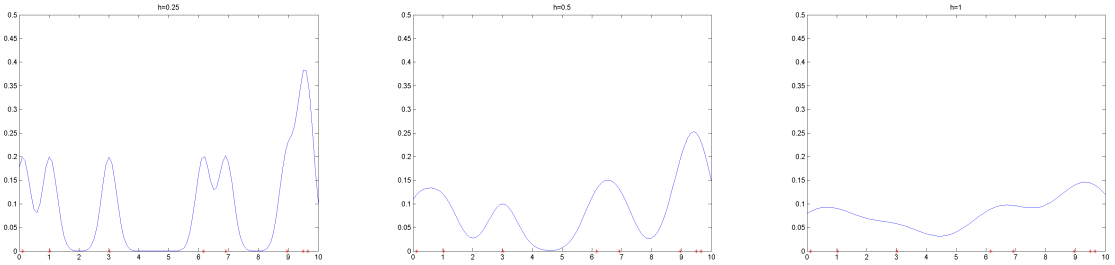
Figure 3.1. Histogram density estimation with different h values. Increasing h means smoother and less sparse estimates whereas local information is lost.

where $\sharp\{\}$ indicates the number of instances that fall into the same bin as x, N is the total number of instances x_i and h is the width of the histogram bins.

3.2. Kernel Density Estimators

Histograms are not always powerful enough to explain a density. Histogram based density estimations are sharp estimations because of the edgy structure of histogram bins. The bins have strict boundaries, so the transitions between similar instances are not smooth. However this is not the case for real world instances; that is, the data that comes from a real problem tend to have smooth transitions. Thus histograms fail to imitate continuity of densities.

Kernel estimators, also known as Parzen windows, can be defined as local estimators, which arrive to a final estimation from weighted summation of local decisions of every individual data point. Like a k-nearest neighborhood classifier, kernel estimators collect the vote of, not only k many points, but every point in the training set and evaluates the result. The effect of a vote changes with respect to a kernel that is placed onto the voting data point. The kernel determines how much contribution a vote will make to a test point that is being questioned.



(a) h=0.25 (b) h=0.5 (c) h=1.0

Figure 3.2. Kernel density estimation with different h values. The influence of a data point is determined by the shape of the kernel. High h increases the range of influence and smoothness while decreasing local effects.

The basic formulation for a kernel estimator can be stated as,

$$\hat{p}(x) = \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right) \tag{3.2}$$

where N is the total number of training instances, K is the kernel function (i.e gaussian) and h is the bandwidth of selected kernel.

There are several ways to analytically calculate an optimal h value in the literature, yet Scott's [43] bandwidth selection rule is a very shorthand computation,

$$h_j = \sigma_j N^{(-1/(d+4))} \tag{3.3}$$

Here j is the index of a dimension, d is the total number of dimensions and σ_j is the standard deviation of jth dimension. Scott's rule distributes the standard deviation to all instances equally to obtain the best approximation and smoothness in the final estimate.

3.3. Voronoi Tessellations

Voronoi Tessellations (VT) is the special case of $k-nn$ classifier where $k = 1$ that every input is associated with nearest sample data. Every sample data is considered

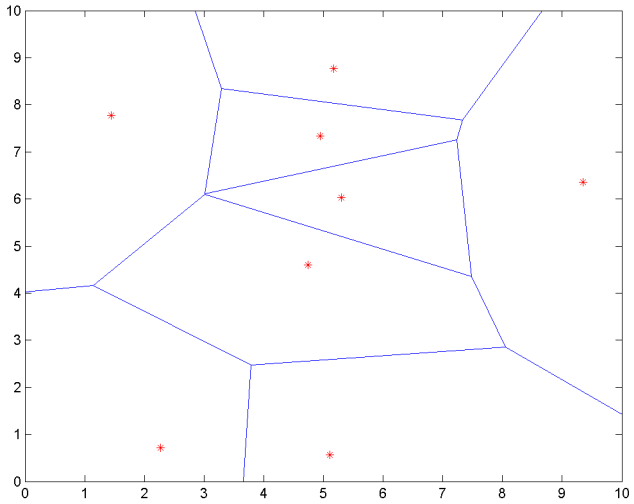


Figure 3.3. Voronoi Tessellations Example. The dots represents the cluster centers and lines represents the boundaries of these clusters. A data that fall into a cluster is associated with the center of that cluster.

as a separate class so the associated input becomes member of that class. Accordingly class conditional probabilities can be computed as,

$$\hat{p}(x|C_k) = \frac{1}{N}\sum_{i=1}^{N} r_i^k \tag{3.4}$$

where x is the data point being tested and N is the normalization constant which is the number of all instances. Here r_i^k is 1 if $x_i \in C_k$ and 0 otherwise.

3.4. Supervised Voronoi Tessellations

This method is the slightly modified version of the standard Voronoi scheme. The difference is the tessellations contain more than one class, because the tessellations map is constructed from every training instance. Then the class conditional probability of a test instance x is computed by,

$$\hat{p}_j(x|C_k) = \frac{1}{N_j}\sum_{i=1}^{N} r_i^{k_j} \tag{3.5}$$

where $\hat{p}_j(x|C_k)$ is the probability that x belongs the class j(e.g. skin, nonskin), and N_j is the total number of instances in all tessellations that belong to class j where N is the number of all instances. $r_i^{k_j}$ is 1 if the $x_i \in C_k$ and $x_i \in j$.

3.5. Classification Principle

The classification was based on comparison of class conditional probability. The skin and nonskin probabilities of every test instance x were computed according to the model being used and x was labeled as a skin pixel if,

$$\frac{P(x|skin)}{P(x|nonskin)} \geq \theta \tag{3.6}$$

where θ is the threshold that trades-off the true positive and false positive ratios.

3.6. Experimental Results

The experiments were carried out on two separate skin datasets [44],[27]. The main purpose to perform the test on two dataset is to measure the accuracy of the classifier schemes. Because the sources of datasets are different from each other, the consistency of the performance can be examined. Another point is, most of the skin detection systems work with the hardware setups that are consistent in capture behavior. The images or the video sequence do not come from various sources. Thus the second dataset [27] would tell more about the accuracy of nonparametric classification on such systems.

The first comparison was carried out with different histogram setups to identify the best bin width. For this purpose four different h values were tested. The dimensions of each color channel was quantized equally so every histogram had $256/h$ bins for every channel. The volume of the histograms were then $(256/h)x(256/h)x(256/h)$. Half of the dataset was used to train the histograms and the other half was used to test the classification accuracy.

Equal detection rates together with Receiver Operating Characteristics (ROC) curves were calculated to have a better view of performance. The ROC curves have been used pretty much in the literature, because they reveal the performance behavior of the system better than equal error rates. Moreover they supply a simple visual comparison without much burden. In order to obtain the ROC curves, different θ values in equation 3.6 were examined. Altering θ resulted in different true and false detection rates, so these values were used to plot the ROC curves.

The training and test routines were repeated ten times, and true detection and false detection rates were calculated for every setup. Afterwards, the true detection and false detection rates were averaged for every threshold value. The resultant ROC curves are shown in figure 3.4 and 3.5).

The ROC curves reveal that the $h = 8$ and $h = 16$ setups are usually better on majority of the colorspaces which is parallel to results of [24]. Very large bin widths return rough estimates where the local information is lost and probability decisions are less adaptive. On the opposite case, very small bin widths result in spiky estimates where the generality is lost. The values $h = 8$ and $h = 16$ bring a good trade-off between generality and locality.

The ROC curves also reveal the effect of colorspace transformations on classification performance. The transformations influence the classification characteristics of histogram setup. The main reason of this performance difference is the difference in skin distribution on every colorspace. Because of the characteristics of the colorspace, the transformations shape the skin region and distributions uniquely. As a result, the variance and location of the skin colors differentiate from one colorspace to another. The same bin width for every color channel setup is not capable to capture enough these nuances.

Kernel density estimation brought the highest results and the best ROC curves on most of the colorspaces. The main reason is the structuring principle of kernel estimates. Because of the influence introduced by kernel function, the estimates tend

to be smooth and continuous. Another consideration point is the bandwidth of the kernel. Results prove the suitability of Scott's rule [43] for bandwidth selection. It trades off generality and locality of the information intelligently by distributing the overall variance to individual instances.

Voronoi performed one step below the histogram. The estimation performance was smooth and accurate on many colorspaces. The incremental tessellation was carried out until the average distortion drop 5 percent, so the Voronoi scheme located majority of the tessellation centers to erroneous regions. The average number of tessellations was around 300 which is significantly less than the size of any histogram. Thus incremental tessellation is a substitute for histogram estimation, where the data distribution is not known and nonparametric; because voronoi tessellation has only one parameter to tune where a histogram may need a bandwidth for every dimension of the data.

Supervised voronoi scheme performed excellent through out all test runs. Both classification and density estimation results are almost equal to kernel densities. Pay attention that the supervised voronoi tessellation algorithm is the only algorithm that minimizes the classification error. The other schemes are density estimation methodologies which are more concentrated on minimizing the probability density estimation error. Of course, a good estimation means good classification however they spend significant memory or computational source for this purpose while supervised voronoi algorithm achieves the same goal with a lot fewer.

3.7. Discussion

Nonparametric schemes are best choice when a concrete knowledge about data is not present. The results show that they all perform in acceptable regions. However their adaptiveness on different transformations is not same. Histograms are very easy to implement and employ, but they need considerable number of instances to obtain a good estimate. Kernel density estimates are the most accurate among all however their applicability is constrained because of massive computational demand. Voronoi tessellations are good substitutes for both, because they balance the memory needs and

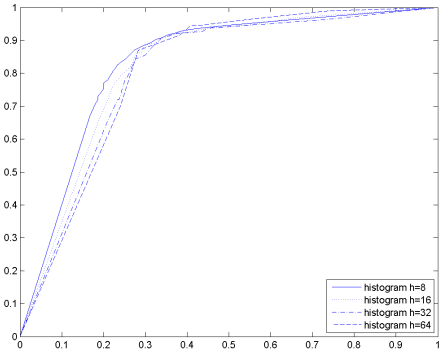
Table 3.1. Equal detection rates of classifiers on first dataset. The table shows the points where true detection and true rejection rates are equal.

Colorspace	$H_{h=8}$	$H_{h=16}$	$H_{h=32}$	$H_{h=64}$	K	V	SV
CIELab	78.5	77.3	76.6	73.0	**79.6**	70.3	77.8
CIExyz	**77.6**	76.3	69.1	67.8	77.0	73.3	75.0
HSV	76.8	81.5	79.1	75.3	**82.3**	77.1	79.5
normalized RGB	**81.3**	77.8	75.3	66.5	77.5	75.6	79.5
RGB	79.0	81.1	78.6	76.5	**82.3**	77.3	78.8
YCrCb	80.8	80.0	72.1	74.3	**81.8**	78.0	80.8

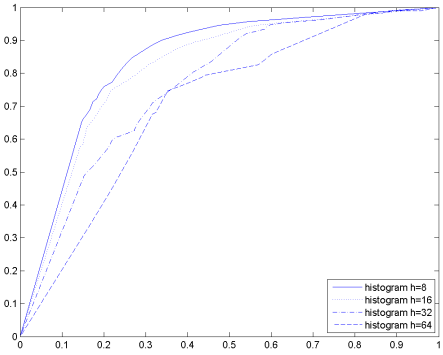
Table 3.2. Equal detection rates of classifiers on second dataset. The table shows the points where true detection and true rejection rates are equal.

Colorspace	$H_{h=8}$	$H_{h=16}$	$H_{h=32}$	$H_{h=64}$	K	V	SV
CIELab	86.3	84.5	84.1	82.0	88.3	81.3	**89.3**
CIExyz	**90.3**	86.1	81.1	72.1	77.5	81.6	79.5
HSV	92.0	**92.6**	87.1	83.3	88.0	85.6	88.3
normalized RGB	91.8	88.5	88.8	56.0	78.1	88.0	**92.8**
RGB	**92.8**	92.0	87.3	81.8	91.8	88.8	90.8
YCrCb	**93.0**	90.1	81.2	78.9	91.5	88.1	89.5

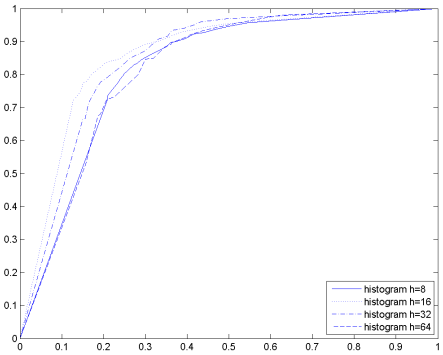
computational needs while performing in the acceptable range. Supervised schemes are the best choice for the classification problem, because they allocate the significant amount of source on critical regions and they iterate directly to reduce the classification error.



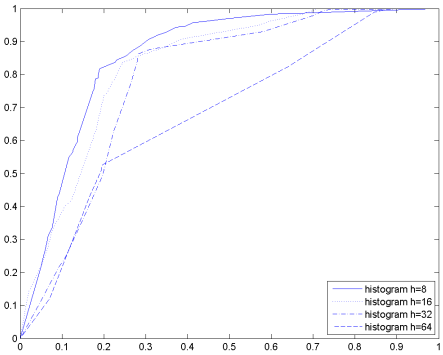
(a) CIELab



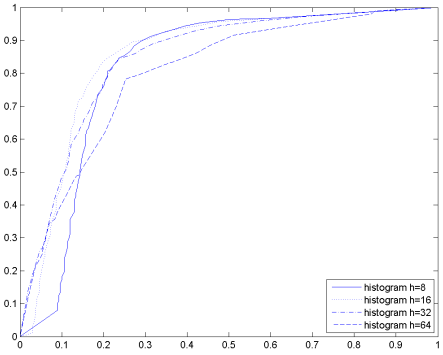
(b) CIExyz



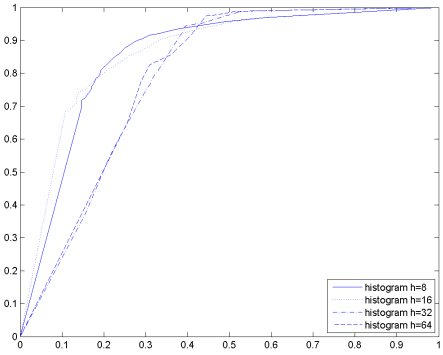
(c) HSV



(d) normalized RGB

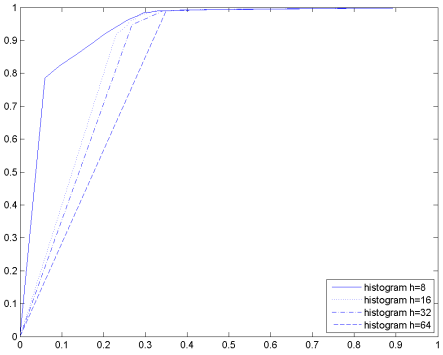


(e) RGB

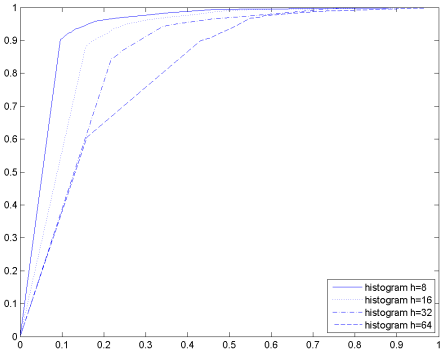


(f) YCrCb

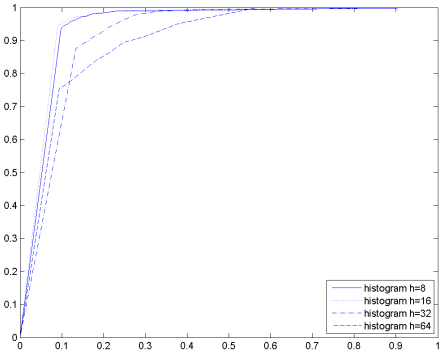
Figure 3.4. ROC curves of histogram based skin detection on first dataset. The figures show ROC curves of histogram based skin detection technique with different bin width h values on six color spaces. $h = 8$ and $h = 16$ perform most consistent on every colorspace.



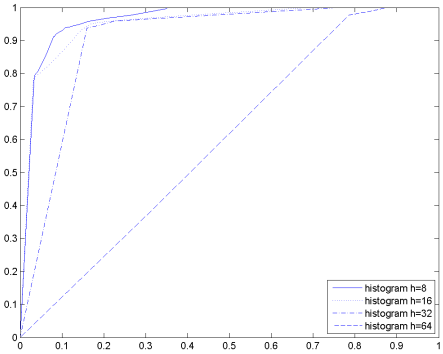
(a) CIELab



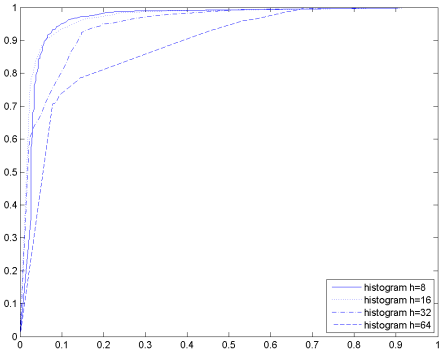
(b) CIExyz



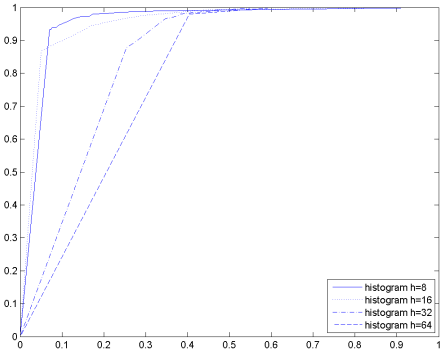
(c) HSV



(d) normalized RGB

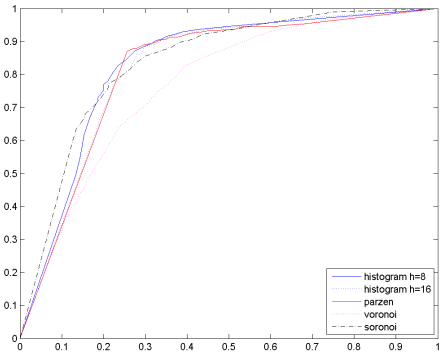


(e) RGB

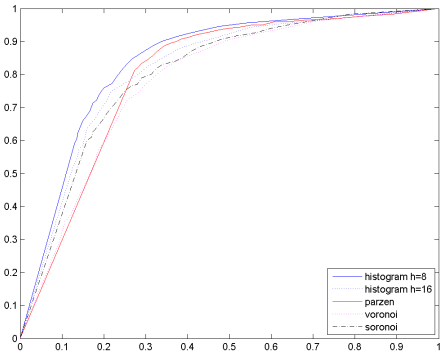


(f) YCrCb

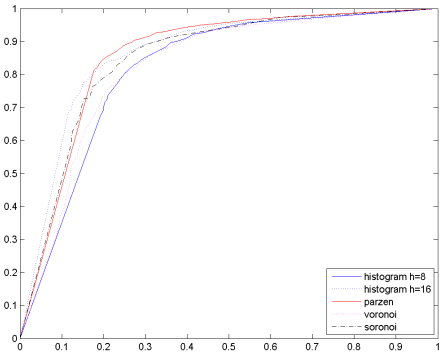
Figure 3.5. ROC curves of histogram based skin detection on second dataset. The figures show ROC curves of histogram based skin detection technique with different bin width h values on six color spaces. $h = 8$ and $h = 16$ perform most consistent on every colorspace.



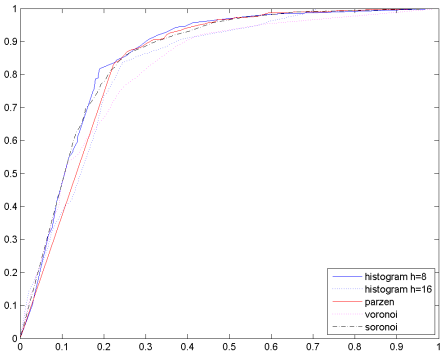
(a) CIELab



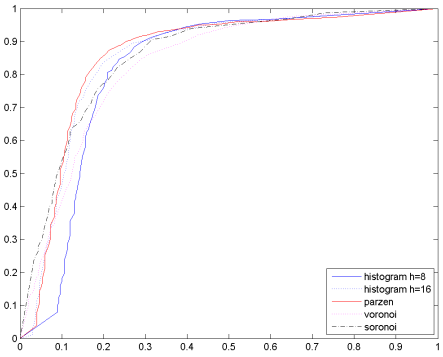
(b) CIExyz



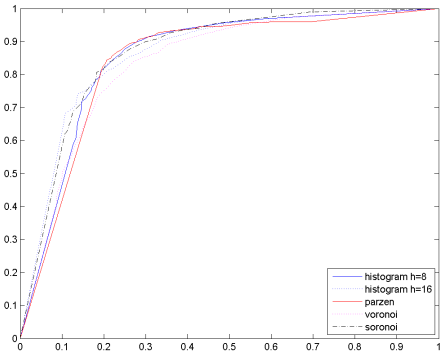
(c) HSV



(d) normalized RGB

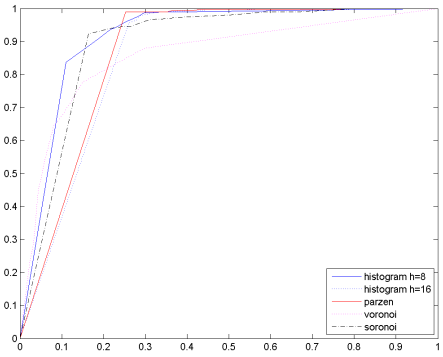


(e) RGB

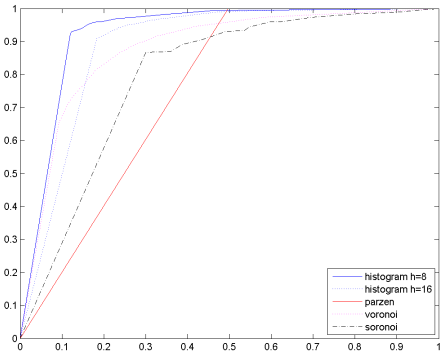


(f) YCrCb

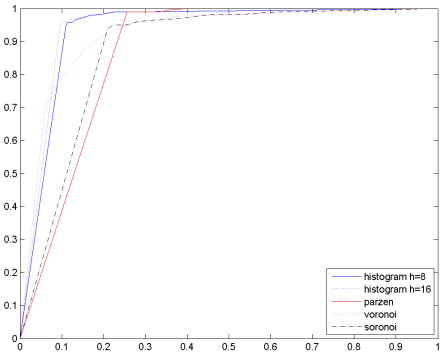
Figure 3.6. ROC curves of all skin detection techniques. The figures show ROC curves of histogram, parzen, voronoi and supervised voronoi techniques on six color spaces.



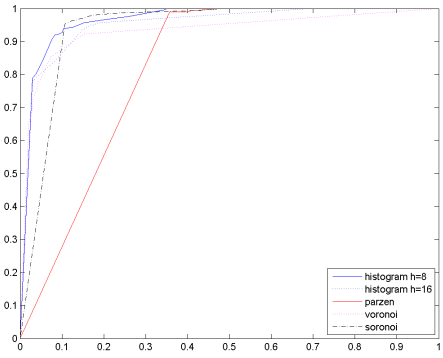
(a) CIELab



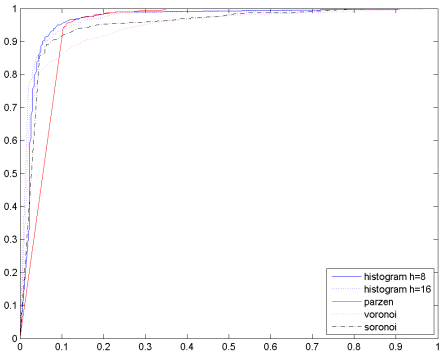
(b) CIExyz



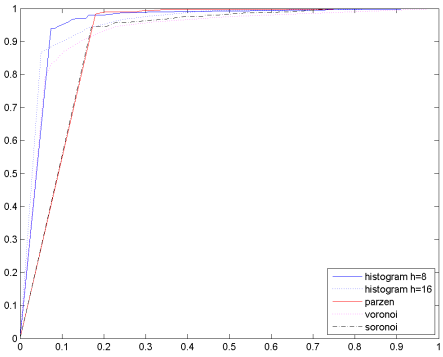
(c) HSV



(d) normalized RGB



(e) RGB



(f) YCrCb

Figure 3.7. ROC curves of all skin detection techniques. The figures show ROC curves of histogram, parzen, voronoi and supervised voronoi techniques on six color spaces.

4. OPTIMAL FEATURE FUSION

The results in previous chapter show that there is not a single colorspace that has the highest discrimination of skin and non-skin pixels. One reason is, these colorspaces can not separate the illumination channel efficiently from the chrominance, so inherit illumination still deviates the skin color representation. However a clever integration of these measurements may have a better separation, that is, an optimally weighted combination of these color spaces may lead to less illumination sensitive skin detection. The question here is how to find this optimal weight set that leads to the optimal color space. In this chapter a fusion method [45] is analyzed that uses the non-perfect correlations between colorspaces to obtain the best weight set.

4.1. Problem Definition

The problem can be redefined as in [45]: "Given the set of observations of the same quantity which are expressed in the same unit but which are obtained by N different methods how are these methods to be combined in order to obtain most accurate measurement of the process?". It is only assumed that the distributions of the observations are unimodal, so an observation may be stated as:

$$u = E(u) \mp \sigma_u \tag{4.1}$$

where $E(u)$ is the average value for the observation u and σ_u is the standard deviation.

The final observation is obtained from weighted combination of particular observations using the standard scheme:

$$E(u) = \sum_{i=1}^{N} w_i E(u_i) \tag{4.2}$$

where $E(u_i)$ is the estimate of particular method i as in 4.1 and w_i is the weight associated with method i. Now, the problem is to find w_i values.

In order to find the w_i values the non-perfect correlations are analyzed between observation methods. Following the idea of [46] the variance of combined observations can be written as

$$\sigma_u^2 = \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j cov(u_i, u_j) \tag{4.3}$$

or equivalently

$$\sigma_u^2 = \sum_{i=1}^{N} w_i^2 \sigma_{u_i}^2 + \sum_{i=1}^{N} \sum_{j=1, i \neq j}^{N} w_i w_j cov(u_i, u_j) \tag{4.4}$$

where u_i is average output of observation method i and σ_{u_i} is the standard deviation of u_i and $cov(u_i, u_j)$ is the covariance between observation method i and j.

The weight set problem with respect to [46] is

$$minimize\ \sigma_u \tag{4.5}$$

such that

$$\sum_{i=1}^{N} w_i = 1 \tag{4.6}$$

$$-1 \leq w_i \leq 1\ , i = 1, \ldots, N \tag{4.7}$$

constraint 4.6 ensures that at least one method will be other than zero, or selected for final decision, and constraint 4.7 limits the search space that w_i will be in.

The objective function 4.5 is quadratic with linear constraints so it can be solved by linear programming like simplex algorithm. Simplex will give a set of algorithm mean-standard deviation pairs which will form a boundary line called frontier. The frontier points are points of maximum mean for a given standard deviation or conversely

minimum standard deviation for a given mean. That's why frontier represents the optimal mean-standard deviation set. Any point on the frontier, accordingly any weight set can be selected yet they are all optimal. However the most appropriate selection depends more on the application.

4.2. Application to Skin Detection

The weighting scheme is applied to twelve color planes that are RGB, rg, RG, YB, L^*, a^*, b^*, S, V. These planes correspond to observations u_i, the contribution weights are w_i and standard deviation of each particular color plane is σ_u. The covariance between color plane i and j is denoted by $cov(u_i, u_j)$.

4.2.1. Training

The skin regions from a set of training images are transformed to twelve color spaces and for all of this color spaces the expected mean, standard deviation and the covariance matrix is calculated. Afterwards the optimal set of weights are calculated using simplex algorithm. The weight set selection is depends on the application, however in skin detection application a weight set that has $\sigma_u \approx 1$ gives good results. After selecting the weight set an expected mean-standard deviation is obtained. Finally an expected $SNR_{Training}$ is calculated from the ratio of this expected mean-standard deviation pair. In the testing phase this $SNR_{Training}$ will be compared to classify the given pixel.

4.2.2. Testing

The weights are obtained in the training stage and combine color planes to a single plane. In order to classify a single pixel the local mean and the local standard deviation pairs are computed using a $3x3$ window. Afterwards the ratio of these local mean and local standard pair is calculated which gives the SNR_{Test} value of the pixel

Table 4.1. Correlation coefficients for skin tone.

	R	G	B	r	g	RG	YB	L	a	b	S	V
R	1.00											
G	0.96	1.00										
B	0.94	0.97	1.00									
r	-0.28	-0.51	-0.55	1.00								
g	0.19	0.42	0.26	-0.62	1.00							
RG	0.65	0.40	0.44	0.45	-0.50	1.00						
YB	0.87	0.96	0.98	-0.69	0.39	0.25	1.00					
L	0.98	0.99	0.97	-0.43	0.33	0.51	0.93	1.00				
a	0.45	0.18	0.26	0.56	-0.71	0.96	0.06	0.29	1.00			
b	0.45	0.39	0.19	0.33	0.36	0.40	0.11	0.41	0.20	1.00		
S	1.00	0.96	0.94	-0.28	0.19	0.65	0.87	0.98	0.45	0.45	1.00	
V	0.73	0.55	0.46	0.39	-0.10	0.87	0.31	0.62	0.72	0.78	0.73	1.00

under test. Finally the skin/nonskin decision is obtained comparing the SNR values,

$$D(x,y) = \begin{cases} 1 & if \quad |SNR_{Test}(x,y) - SNR_{Training}| \leq \theta \\ 0 & otherwise \end{cases} \tag{4.8}$$

where $SNR_{Test}(x,y)$ is the local SNR of (x,y) pixel, $SNR_{Training}$ is the expected SNR value and θ is the threshold value.

4.3. Experimental Results

The first set of experiments was employed to the Solar et. al. dataset [44]. The system achieved %70 classification rate which is not better than nonparametric methods. This poor performance is the result of compression algorithms which is used to compress the dataset images. The compression algorithms apply blocking, and quantization procedures to reduce the image size, but these applications disturb the tone and spatial relationships of pixels that prevents the true measurement and detection. Smoothing the input image reduces the blocking effect, but it also reduces

Table 4.2. Selected set of weights for skin.

R	G	B	r	g	RG	YB	L	a	b	S	V
1.00	-1.00	-0.07	1.00	0.24	-1.00	-1.00	1.00	1.00	1.00	-0.16	-1.00

the local variance and results in false rejections. Thus the performance of the system is demonstrated on a gesture video where hand illumination changes drastically from frame to frame.

The method was able to locate the skin regions not completely but fairly independent from illumination. The local standard deviation is usually smaller than the expected standard deviation, so it results a higher SNR value. Setting a higher threshold value is a solution for this problem, yet a higher threshold allows more false positives. Hence, the threshold is the parameter here to trade off true and false positives.

4.4. Discussion

Overall, the system combines the observations of several independent sources to obtain single solid measurement. Using the non-perfect correlations, the algorithm finds an optimal weight set that gives a minimum standard deviation. The method benefits from local relationships of the skin texture and eliminates the nonskin regions more effectively. The method is less prone to illumination differences than other methods. Although not perfect the method can detect extreme bright and extreme dark regions more consistently.

4.5. Conclusion

All in all skin color is a very strong clue to determine the candidate hand regions. It eliminates significant amount of non-hand regions. The remaining regions may then be eliminated by employing extra information. In this thesis the motion and size information are employed to identify the hand region. It is assumed that the face occupies a larger skin blob than hand and also it is requested from user to wave his/her

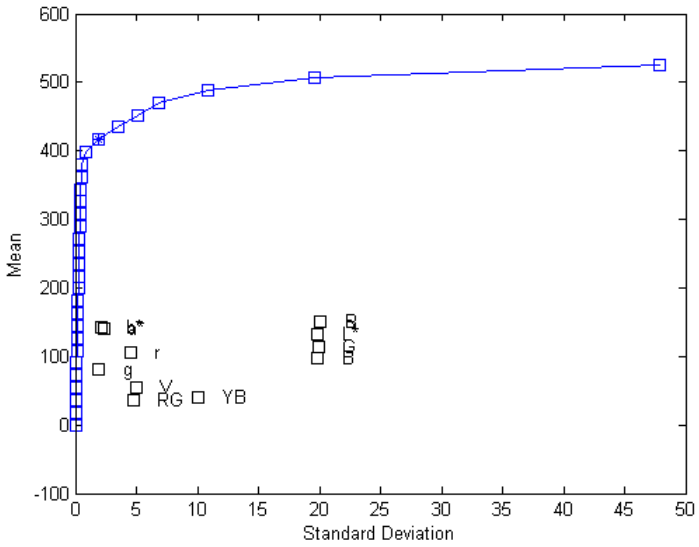


Figure 4.1. Mean-Standard deviation space for skin tone. The curved line is the optimal frontier where optimal mean-standard deviation pairs lie. The selected weight set is labeled with star.

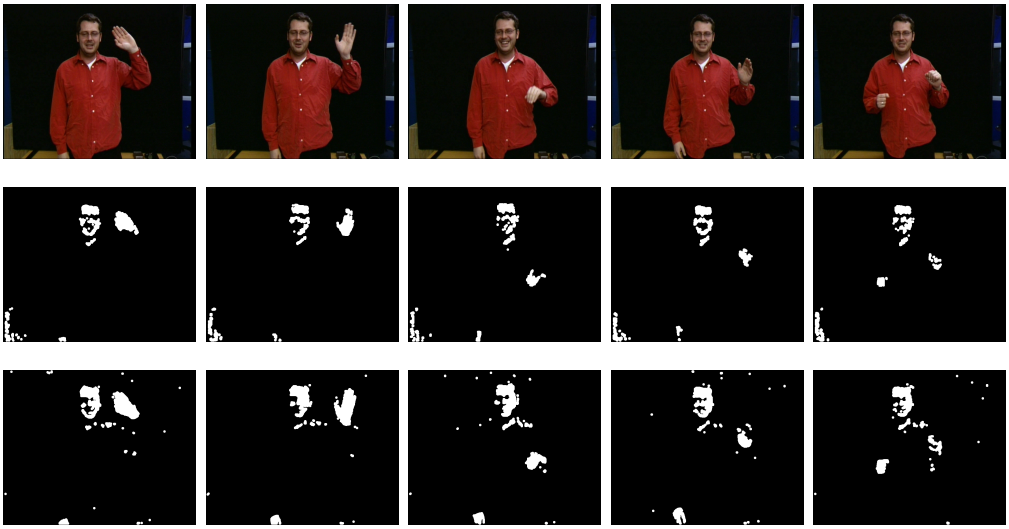


Figure 4.2. Skin detection results of feature fusion. The above figure shows the skin detection results of frame 32,58,74,93,101 from a gesture sequence. The second row shows the detection results with histogram classification and the third row shows the detection results with fusion method. The fusion method is less sensitive to lightning and it can eliminate nonskin regions more robustly using the spatial information.

hand to capture the motion and initial hand location more robustly.

5. MEAN-SHIFT HAND TRACKING

Object tracking is the estimation of object's next location based on the previous measurements. In the simplest case, the template of the object is used to find the next location by exhaustively matching the template in a predefined region [47]. This approach is highly error prone that it assumes object appearance almost never changes during the course of tracking. This is not a true assumption for many real world cases because of the e.g. rotation, lighting change, partial occlusion factors. The object hardly preserves it's initial conditions. The exhaustive search, even it's simple, is not a choice for many situations as it demands considerable computational power and returns poor performance.

A state-of-art and a general purpose tracking algorithm is the Mean-Shift tracker [21] which uses color features to represent the object and gradient descent methods to localize it. The algorithm is invariant to translation, rotation and deformations of object while it does run in the linear time. This section summarizes the Mean-Shift algorithm and presents it's utilization on hand tracking problem.

5.1. Object Representation

To represent the target the color *pdf* of the object is chosen as the feature. The density distribution of object's color is estimated by m-bin histogram because of it's application ease. Thus target model $\hat{q}$ and target candidate $\hat{p}(y)$ at location y is defined as,

$$\hat{q} = \{\hat{q}_u\}_{u=1\ldots m} \quad \sum_{u=1}^{m} \hat{q}_u = 1 \tag{5.1}$$

$$\hat{p}(y) = \{\hat{p}_u(y)\}_{u=1\ldots m} \quad \sum_{u=1}^{m} \hat{p}_u = 1 \tag{5.2}$$

The pixels that lie on the borders of the object are more prone to occlusion and interference with background. In order to increase the robustness of the density estimate and reduce the effect of disturbance a kernel is employed to density estimation procedure that assigns smaller weights to the points that are further from the object center. The target representation is turned into,

$$\hat{q}_u = C \sum_{i=1}^{n} k(\| x_i^* \|^2) \delta[b(x_i^*) - u], \tag{5.3}$$

where x_i^* is the normalized pixel locations with the corresponding bin index $b(x_i^*)$,n is total number pixels in the region of interest, $k(x)$ is the kernel profile that assigns the weights, δ is the Kronecker delta function and C is the normalization constant that assures the condition 5.1 so,

$$C = \frac{1}{\sum_{i=1}^{n} k(\| x_i^* \|^2)}. \tag{5.4}$$

Similarly the target candidate representation is modified to,

$$\hat{p}_u(y) = C_h \sum_{i=1}^{n_h} k(\| \frac{y - x_i}{h} \|^2) \delta[b(x_i^*) - u], \tag{5.5}$$

where

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k(\| \frac{y-x_i}{h} \|^2)}. \tag{5.6}$$

Remember that y is the location of candidate target center,n_h is the total number pixels in the region of interest and C_h is the constant that assures condition 5.2.

5.2. Object Similarity Metric

In all object tracking tasks a metric is defined to measure the similarity between target object and target candidate. The distance used in Mean-Shift is,

$$d(y) = \sqrt{1 - \rho[\hat{p}(y), \hat{q}]}, \tag{5.7}$$

where

$$\rho[\hat{p}_u(y), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y), \hat{q}_u}, \tag{5.8}$$

which is known as Bhattacharyya coefficient. Equation 5.8 can intuitively be interpreted as a good similarity metric that similar pdfs will result in higher $\rho[\hat{p}_u(y), \hat{q}]$ and lower $d(y)$.

5.3. Object Localization

The distance function $d(y)$ inherits the kernels characteristics that was embedded in the density estimations 5.3 and 5.5, so the function is differentiable and gradient-based optimization can be used to find a maximum. The idea is following the gradient the iterations will lead the tracker to the point to which maximum similarity is achieved between target and target candidate.

Taking the Taylor expansion of 5.8 it becomes,

$$\rho[\hat{p}_u(y), \hat{q}] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{y}_0)} \hat{q}_u + \frac{1}{2} \sum_{u=1}^{m} \hat{p}_u(y) \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}} \tag{5.9}$$

recalling 5.5,

$$\rho[\hat{p}_u(y), \hat{q}] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{y}_0)} \hat{q}_u + \frac{C_h}{2} \sum_{i=1}^{n_h} w_i k(\| \frac{y - x_i}{h} \|^2), \tag{5.10}$$

where

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}} \delta[b(x_i) - u]. \tag{5.11}$$

The first being independent of y, only the second term in 5.10 is to be maximized. Employing Mean-Shift procedure [48] the new location of the candidate is,

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g(\| \frac{y - x_i}{h} \|^2)}{\sum_{i=1}^{n_h} x_i g(\| \frac{y - x_i}{h} \|^2)} \tag{5.12}$$

where $g(x) = -k'(x)$.

5.4. Application to Hand Tracking

Like original paper [21] kernel with Epanechnikov profile [43]

$$k(x) = \begin{cases} \frac{1}{2} c_d^{-1} (d+2)(1-x) & if\ x \leq 1 \\ 0 & otherwise \end{cases} \tag{5.13}$$

was used. d is the number of dimensions and c_d is the volume of unit d-dimensional sphere. Because the derivative of $k(x)$ is constant and equation 5.12 reduces to

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i}{\sum_{i=1}^{n_h} x_i} \tag{5.14}$$

RGB color space was chosen to represent the color feature of the hand and a $16x16x16$ histogram was employed to estimate the color pdf of hand. The *Sentence* scene, which is a deaf language video, is used for the tracking experiment. The hand was localized initially and it's movement was tracked through out the course of action.

The Mean-Shift tracker was highly robust to dynamic texture and orientation of the hand, because skin color is a unique identity that discriminates the hand from other regions. Also color feature by nature is invariant to scale and orientation. However the

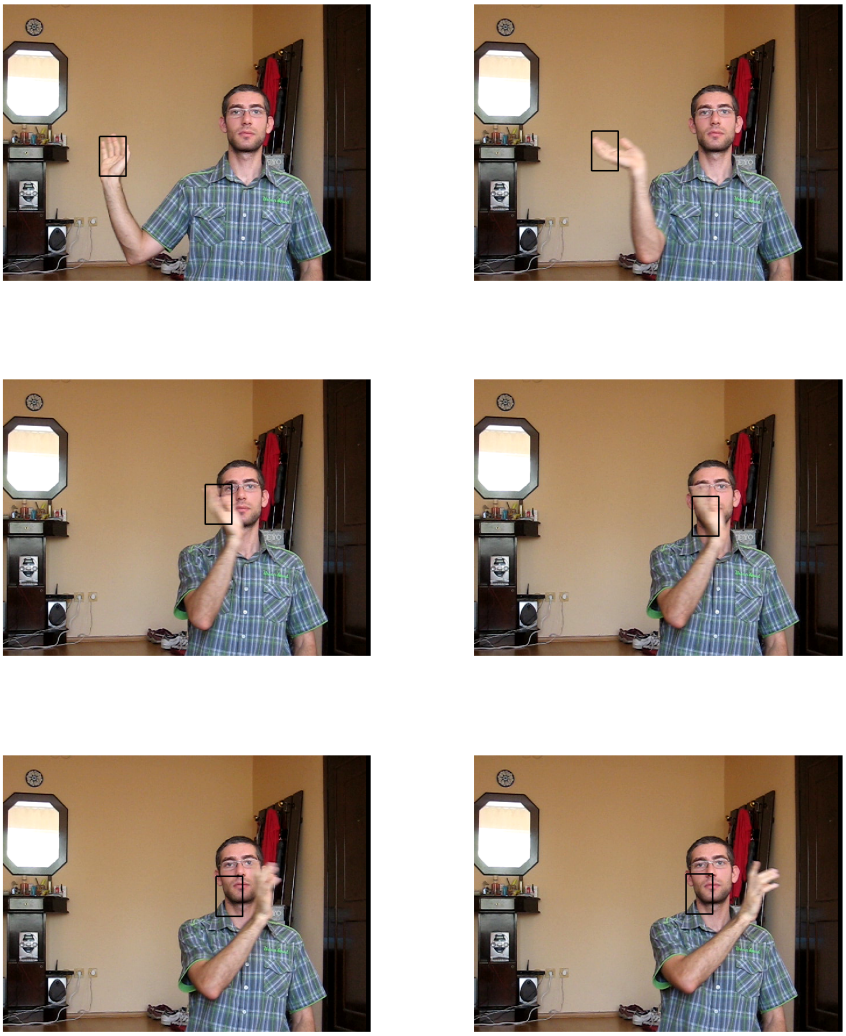


Figure 5.1. Meanshift tracking example. The above figure shows the hand tracking results from frames 21, 32, 38, 39, 41 and 43. Meanshift tracker can not overcome hand and face overlap because face is a bigger attraction field having more skin pixels.

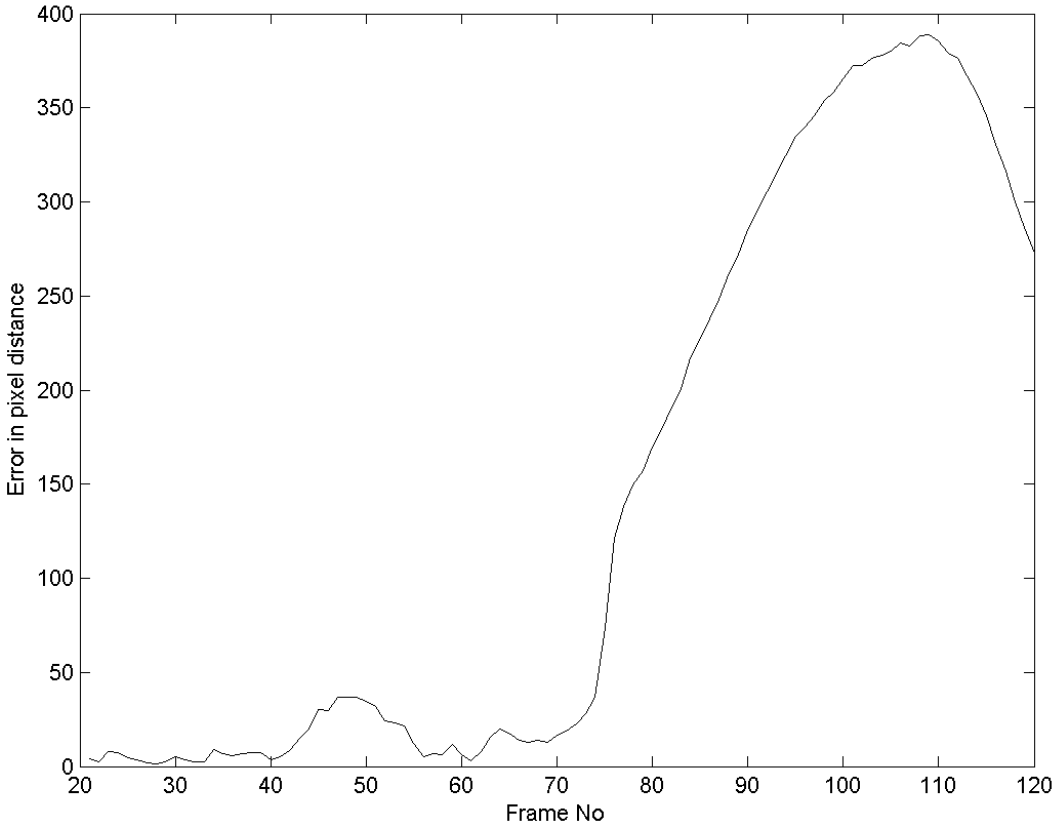


Figure 5.2. Meanshift tracking accuracy. Because meanshift drifts away from face the tracking error increases to high numbers.

color feature does not supply discrimination between hand and face, since they have same color characteristics. Also similar colored background objects are still problem for tracker. Therefore there is a need of extra information to distinguish the hand from face and background. This additional information may be supervised telling the location and boundaries of the face externally or it may be another feature, like texture, that can contribute to the characterization of hand.

The next chapter analyzes the improvement of Mean-Shift tracking utilizing a texture feature together with color for object representation.

6. IMPROVEMENTS ON MEAN-SHIFT

6.1. Introduction

The color feature is not discriminative enough to track the hand robustly and without constraining the user or environment. The skin-like colored objects or other skin regions like arm and face fail the tracker because they have the same color properties. The difference between these regions and the hand may be explained by taking into account the texture of hand. The texture supply a spatial relationship of the pixels so the objects that have similar color representations may be distinguished with texture information. Here we implement the texture representation using LBP operators [49] and develop a more robust tracking scheme by integrating texture representation to color.

The LBP operator characterizes the local variation in gray scale around a pixel, and is invariant to any monotonic transformation of the grey-scale. It follows that the LBP value calculated at each pixel in each color channel will be unchanged by a positive scaling of the channel output. This is a very useful property, since a change in illumination color generally results in an independent scaling of each color channel, thus the LBP value in each color channel is approximately illumination independent.

6.2. Simplest LBP

Simplest Local Binary Pattern operators characterize the pattern around each pixel in an image. To understand the concept of LBP consider a grey-scale image whose intensity I at pixel location (x, y) can be written $I(x, y)$. For a given pixel $p = I(x_0, y_0)$, the 8 neighbors of p can be written as n_i, $i = 0 \ldots, 7$, where $n_0 = I(x_0+1, y_0)$,$n_1 = I(x_0+1, y_0+1), ..., n_7 = I(x_0+1, y_0-1)$. To compute an LBP value, the grey-level value of each neighbor n_i is compared to the value at the central pixel p to determine whether it is greater than or less than p. This amounts to a function

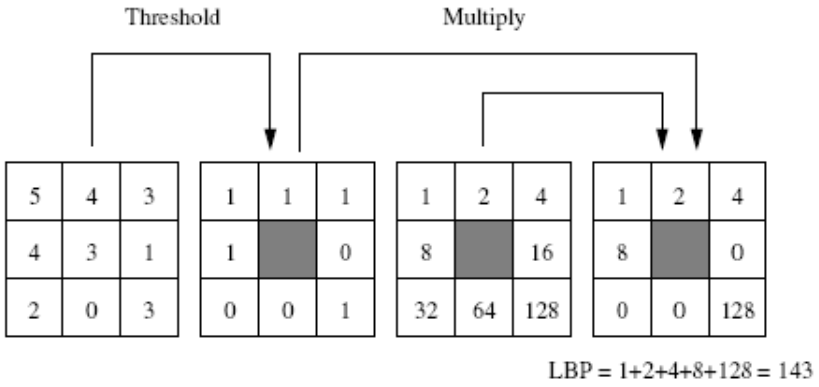


Figure 6.1. Calculation of Simple LBP code [50].

which maps each n_i onto a value b_i as follows:

$$b_i = \begin{cases} 1 & if\ n_i > p \\ 0 & otherwise \end{cases} \tag{6.1}$$

The LBP code of a pixel (x_0, y_0) is derived by choosing an arbitrary starting point and concatenating the 8 binary values b_i into an 8-bit number following the formula,

$$LBP(x_0, y_0) = \sum_{i=0}^{7} b_i 2^i. \tag{6.2}$$

As a result, the local gray level distribution is approximately described with a unique LBP code.

The LBP code of a local texture is able to represent the unique pattern by itself, one step further, higher level textures can be characterized by collecting the statistics of LBP codes in a 2^8 bin histogram. This histogram shows the ingredients of the high level textures by means of low level ones which supply an adequate feature for discrimination.

$$\left.\begin{array}{l} 00111000 \\ 00011100 \\ 00001110 \\ 00000111 \\ 10000011 \\ 11000001 \\ 11100000 \\ 01110000 \end{array}\right\} = 7$$

Figure 6.2. Illustration of ROR operator. The 8-bit number is rotated 7 times and its minimum is attained as the label of that pattern family.

6.3. Rotation Invariant LBP

The texture can rotate due to its nature of motion or with respect to some translation. Under such circumstances, the gray values around the center pixel rotates accordingly, resulting in another LBP code [50]. The situation can be recovered gathering all rotated versions of the same pattern under single label. The formula,

$$LBP^{ri} = \min\{ROR(LBP, i) \mid i = 0, 1, \ldots, 7\} \tag{6.3}$$

performs a circular bitwise right-shift on the 8-bit number LBP 7 times and records the minimum description number, which is assigned as the label of rotated patterns family.

The rotation invariance transform has two major advantages. First, it captures the structure of texture better, since the immunity to orientation of the texture is enhanced. Second, reduction in the number of histogram bins provides less sparse thus more reliable approximations o the distribution.

Not all of the patterns have the same importance in describing the texture. Some

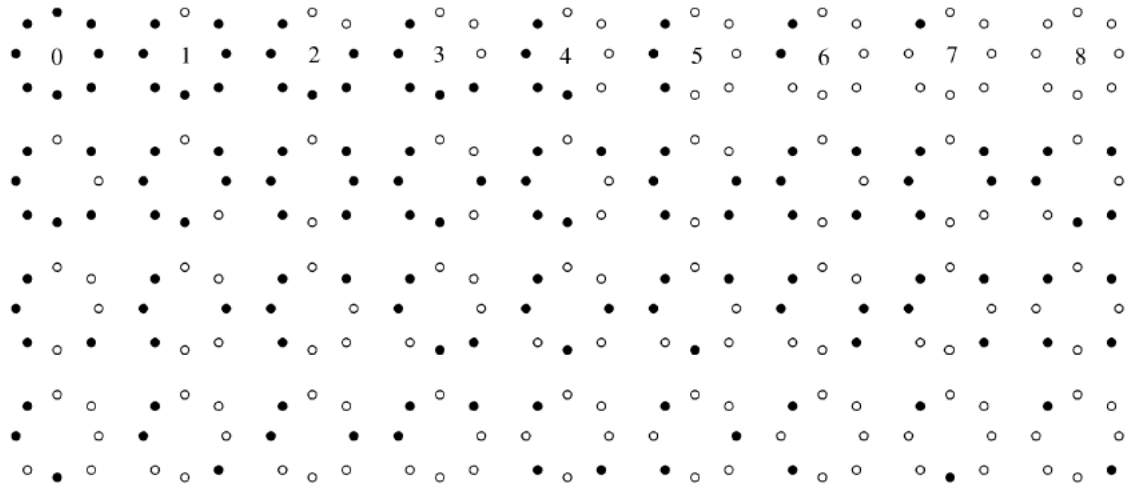


Figure 6.3. Unique rotation invariant LBP patterns for 8 neighbor [51]. The black and white circles correspond to the bit values of 0 and 1. The first row shows the nine uniform patterns which occupy the 90% of textures.

certain patterns hold more significance than others. Proving the observation of [51], it is observed in our work that certain patterns occupy vast majority, above 90% of the texture. Those fundamental patterns have interestingly one thing in common which is having at most two one-to-zero or zero-to-one transitions. In other words, they are formed of either a flat surface or one hill and valley combination. Maenpaa et. al. [51] defines such patterns as "uniform" patterns based on this simple structuring property.

The description power of LBP can be improved by employing the information in hand. The statistical property of "uniform" patterns makes them superior discriminators for texture analysis. One may prefer to use only "uniform" patterns in representation of the texture, alternatively one may benefit from both "uniform" and "nonuniform" patterns collecting all of "nonuniform" patterns under same label. In case of 8 bit representation, 36 rotation invariant patterns will occur where 9 of which is "uniform" and 27 of which is "nonuniform". Taking all "nonuniform" patterns identically the texture can be defined by a $9+1=10$ bin histogram. In short, "nonuniform" patterns statistics reveal that their portion in the texture is so small that they can not be estimated reliably, thus "uniform" patterns are the fundamental structures that advance the dissimilarity texture analysis.

6.4. Application to Hand Tracking

In order to represent hand the joint probability of rotation invariant LBP and hue is used. After locating the hand the detected frame was used to obtain a target model that will be tracked through out the course of action. Basically the hand region is cropped and converted to HSV colorspace. Hue component is quantized with 4 and rotation invariant code of every pixel computed. Afterwards a 2D histogram is constructed with size $256/4(Hue)x10(LBP)$. This histogram served as the representation of target hand.

In the following frames the hue and LBP code of every pixel is extracted and the handness probability is backprojected using the model histogram. The meanshift algorithm is carried out on this probability space to locate the position of the hand.

Experimental results show that the technique can deal with skin colored regions like arm and background. Also it is not disturbed from the hand-face overlap.

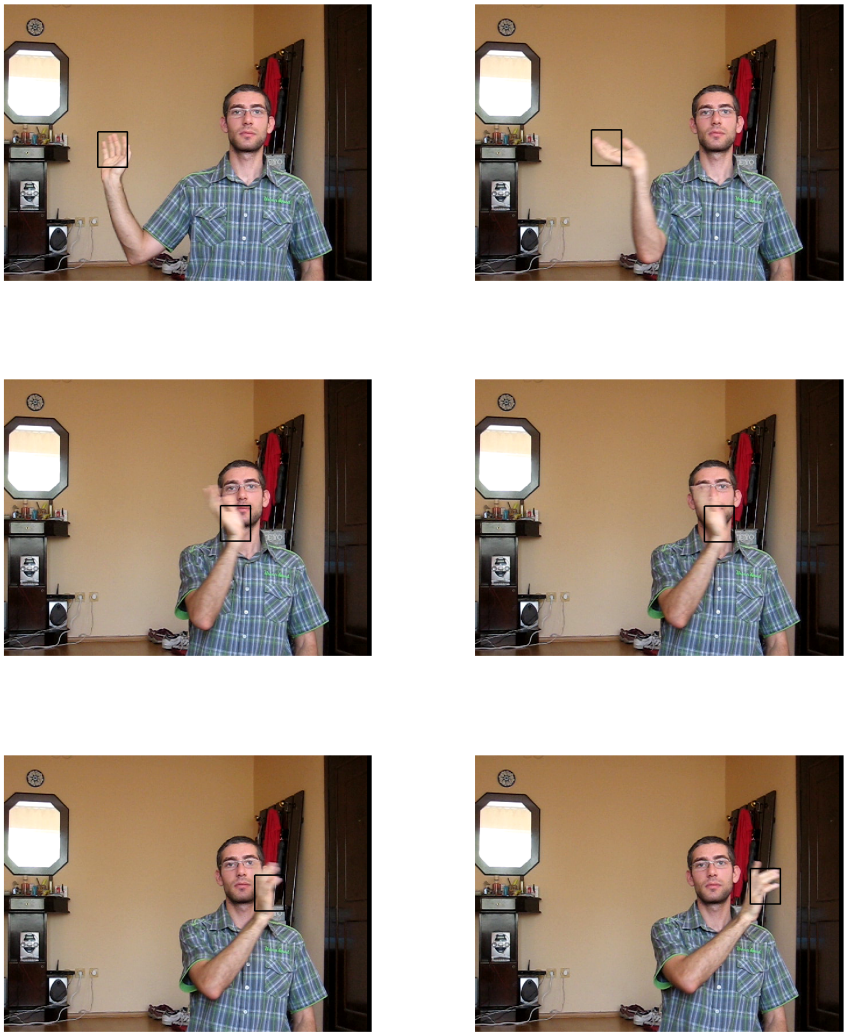


Figure 6.4. Texture integrated meanshift tracking example. The above figure shows the hand tracking results from frames 21, 32, 38, 39, 41 and 43. Integrating texture information brings better discrimination.

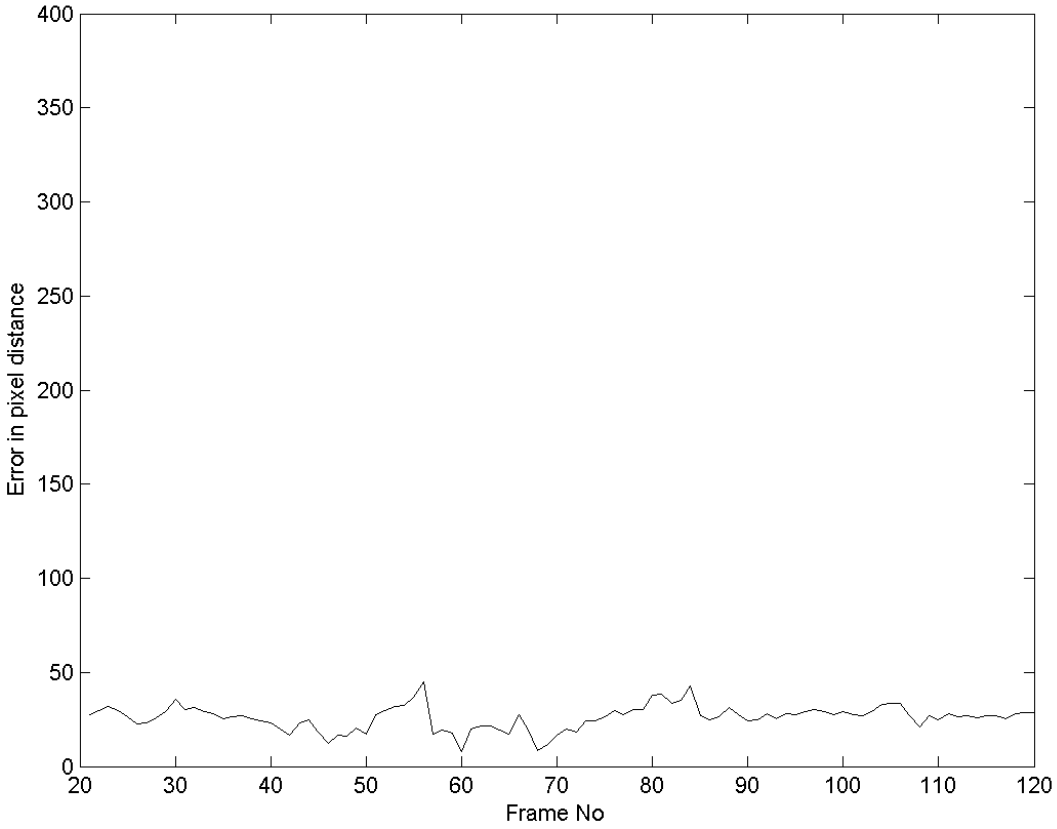


Figure 6.5. LBP based meanshift tracking accuracy. The proposed tracker results in very small error in a $480x640$ image.

7. CONCLUSIONS

In this thesis a hand detection and tracking mechanism was analyzed in detail. For the detection task skin color was used and histogram based skin classification algorithm was compared with two other nonparametric techniques; kernel densities and Voronoi tessellations. Histograms with $h = 8$ $h = 16$ performed well, however they needed high number of training instances. Voronoi was a good alternative to histograms in that sense. Because voronoi scheme grows iteratively to reduce the estimation error, and allocate sources to critical regions. The kernel based densities brought the most consistent density estimation with the help of Scott's [43] bandwidth selection rule. The Scott's rule balanced the generality and locality of a instance to obtain best estimate. Supervised Voronoi tessellation scheme, like unsupervised version was designed to grow iteratively. However it reduced classification error instead of density estimation error. That is why supervised tessellation was like a tailor-made classification tool for the problem.

The colorspace transformations did not change the results significantly. CIELab, CIExyz and normalized RGB brought lower performance rates with respect to others. The colorspace transformations were not successful to separate the illumination channel from chrominance channels. Feature fusion improved this separation and reduced the influence of illumination. The weighted sum of color features resulted in a more compact skin distribution. The algorithm also integrated the spatial information by examining the local variance.

LBP representation of hand together with color information brought robust tracking of hand free from distortion of background and other skin colored regions. The face-hand overlap, the arm or the skin-like regions that are present in the background is not a problem for proposed tracker. Remember that the tracker does not assume any model to achieve tracking.

The further work may be to integrate a Kalman tracker to increase the tracking accuracy and also prevent the tracker from possible drifts. This will also decrease the number of overall meanshift iterations and increase the real-time performance.

APPENDIX A: INCREMENTAL TESSELLATION

A.1. Unsupervised Voronoi Tessellations

The most important step in using Voronoi Tessellations is to identify the sample data points x_i which are the centers, or representatives of x_i. A well-known technique is the Generalized Lloyd Algorithm [52] which works on any arbitrary distribution. This technique initializes given number of centers randomly and iteratively updates their position following an Expectation Maximization procedure. However, there is not any optimum tessellation number for a distribution that will give the best density estimation. Thus conducting the tessellation incrementally is a better idea for practical applications.

Starting with single x_k center, the idea is splitting the center which results in highest error on every iteration. The iterations will continue until the error rate drops under a predefined threshold. It is also possible to stop the iterations when a maximum number of centers is achieved. Either case the resultant solution will be near optimal, because incremental GLA will concentrate on most erroneous regions and will allocate the majority of the sources, that are x_k's, on critical regions. The error in our case is defined as quantization error, the average squared Euclidean distance, which is

$$d_k = \frac{1}{N} \sum_{i=1}^{N} |x_{i_k} - x_k|^2 r_k^i \tag{A.1}$$

where d_k is the average distortion of tessellation center x_k, N_k is the total number of instances that belong to center x_k, N is total number of instances in all tessellations and x_{i_k} is the instances that belong to the tessellation center x_k. The split methodology is designed accordingly to minimize the cluster quantization error, that is, the cluster is divided into two with the plane which minimizes the covariance of the cluster most.

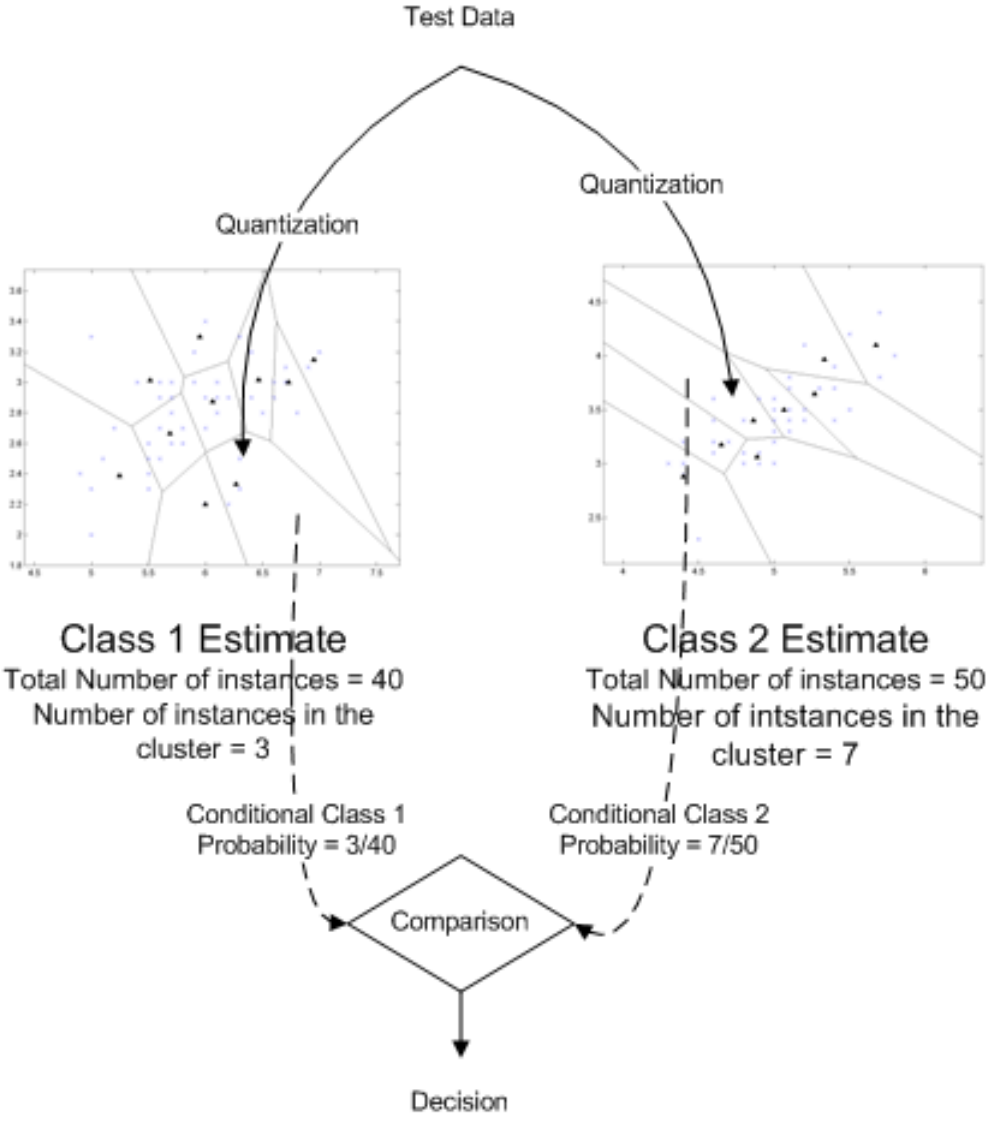


Figure A.1. Voronoi based classification. The data is associated with nearest centers in both tessellation maps. The probability estimate is calculated as if those clusters are histogram bins. The ratio of instances in that cluster with respect to total number of instances from same class gives the conditional probability of input data.

A.2. Supervised Voronoi Tessellations

The incremental voronoi tessellation scheme does not use the class information of data. It blindly quantize the input data to x_k centers to decrease the quantization rate. The decrease in the quantization rate means a better density estimate. However since the final goal is to increase the separation of classes, voronoi tessellation may be trained in this fashion. Thus a voronoi tessellation is structured using the *classification error* as the split criterion. This strategy constructs a single voronoi map where all skin and nonskin pixels lay together, and the center which has the highest classification error rate is split on every iteration. As the split methodology the *linear discriminant analysis*(LDA) is employed because LDA gives the best separating plane for two class discrimination problems.

The classification may be achieved two ways. Remember that every center in the final tessellation may include instances from both classes. That's why every center may be associated with label of the class that is majority, so a test data is first associated with the nearest tessellation center x_k and simply clustered with the label of x_k. The second option is to use this tessellation map again as a density estimate and calculating the class probabilities,

$$\hat{p}_j(x|x_i) = \frac{1}{N_j} \sum_{i=1}^{N} r_{k_j}^i \tag{A.2}$$

where $\hat{p}_j(x|x_i)$ is the probability that x belongs the class j, and N_j is the total number of instances in all tessellations that belong to class j where N is the number of all instances. $r_{k_j}^i$ is 1 if the instance belongs to tessellation center x_k and class j.

Calculating the probabilities instead of class labels allows to trade-off true detection and false detection ratios using the equation (2.4) and obtain the ROC curves.

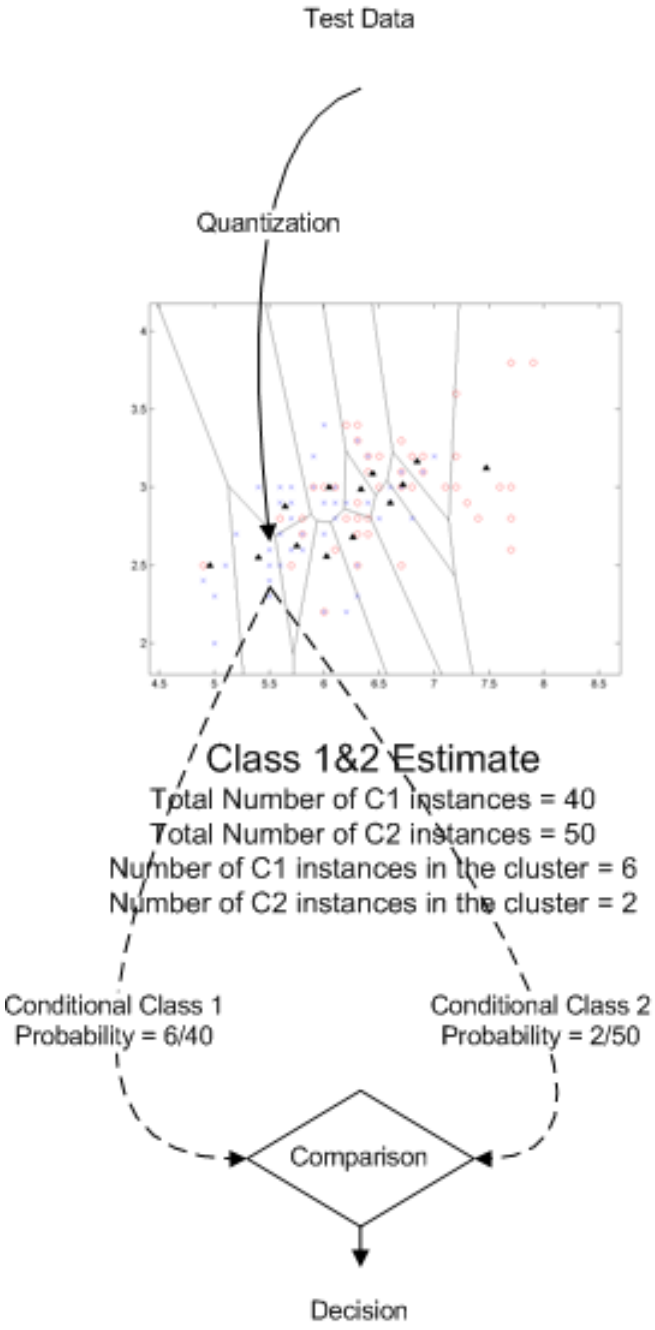


Figure A.2. Supervised Voronoi based classification. The data is associated with nearest centers in the tessellation map. It is whether classified as the class where majority of the instances come or class conditional probabilities are computed to trade off true detection and false detection ratios. The above figure illustrates the calculation of class conditional probabilities.

REFERENCES

1. Mehrabian A. . *Nonverbal Communication*. Aldine-Atherton, 1972.

2. Goldin-Meadow S. . The role of gesture in communication and thinking. *J. Exp. Child Psychol*, 58:88–111.

3. EyeToy . http://www.eyetoy.com.

4. Starner T. and A. Pentland . Real-time american sign language recognition from video using hidden markov models. *International Symposium on Computer Vision*, pages 265–270, 1995.

5. Keskin C. , O. Aran and L. Akarun . Real Time Gestural Interface For Generic Applications. *European Signal Processing Conference*, 2005.

6. Tanibata N. , N. Shimada and Y. Shirai . Extraction of Hand Features for Recognition of Sign Language Words. *The 15th International Conference on Vision Interface May*, pages 27–29, 2002.

7. Zieren J. , N. Unger and S. Akyol . Hands Tracking from Frontal View for Vision-Based Gesture Recognition. *Proc. 24th DAGM Symp*, pages 531–539, 2002.

8. Haritaoglu I. , D. Harwood and L. Davis . W 4: real-time surveillance of people and theiractivities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):809–830, 2000.

9. Terrillon J.C. , A. Pilpré , Y. Niwa and K. Yamamoto . Robust face detection and Japanese sign language hand posture recognition for human-computer interaction in an intelligent room. *15th International Conference on Vision Interface May*, pages 27–29, 2002.

10. Kolsch M. and M. Turk . Robust hand detection. *Proc. IEEE Intl. Conference on*

Automatic Face and Gesture Recognition, 2004.

11. Viola P. and M. Jones . Rapid object detection using a boosted cascade of simple features. *Proc. CVPR*, 1:511–518, 2001.

12. Ong E.J. and R. Bowden . A boosted classifier tree for hand shape detection. *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 889–894, 2004.

13. Yuan Q. , S. Sclaroff and V. Athitsos . Automatic 2D Hand Tracking in Video Sequences. *Proc. WACV*, 2005.

14. Akyol S. and P. Alvarado . Finding Relevant Image Content for mobile Sign Language Recognition. *IASTED International Conference-Signal Processing, Pattern Recognition and Applications (SPPRA), Rhodes*, pages 48–52, 2001.

15. Starner T. , J. Weaver and A. Pentland . A wearable computer-based American sign Language Recogniser. *Personal Technologies*, 1(4):241–250, 1997.

16. Raja Y. , S.J. McKenna and S. Gong . Tracking and segmenting people in varying lighting conditions using colour. *Third International Conference on Automatic Face and Gesture Recognition, Nara, Japan, IEEE Computer Society Press*, pages 228–233, 1998.

17. Yang M.H. , N. Ahuja and M. Tabb . Extraction of 2D motion trajectories and its application to hand gesture recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(8):1061–1074, 2002.

18. Shan C. , Y. Wei , X. Qiu and T. Tan . Gesture recognition using temporal template based trajectories. *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, 3, 2004.

19. Tan C.S.Y.W.T. and F. Ojardias . Real time hand tracking by combining particle filtering and mean shift. *Automatic Face and Gesture Recognition, 2004. Proceed-*

ings. Sixth IEEE International Conference on, pages 669–674, 2004.

20. Isard M. and A. Blake . CONDENSATION - Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

21. Comaniciu D. , V. Ramesh and P. Meer . Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, 2003.

22. Comaniciu D. and P. Meer . Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.

23. Polat E. and M. Ozden . A Nonparametric Adaptive Tracking Algorithm Based on Multiple Feature Distributions. *Multimedia, IEEE Transactions on*, 8(6):1156–1163, 2006.

24. Jones M.J. and J.M. Rehg . Statistical Color Models with Application to Skin Detection. *International Journal of Computer Vision*, 46(1):81–96, 2002.

25. Bradski G.R. . Real time face and object tracking as a component of a perceptual user interface. *IEEE Workshop on Applications of Computer Vision*, pages 214–219, 1998.

26. Hsu R.L. , M. Abdel-Mottaleb and A.K. Jain . Face detection in color images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):696–706, 2002.

27. Gomez G. . On selecting colour components for skin detection. *Proc. of the ICPR*, 2:961–964, 2000.

28. Dai Y. and Y. Nakano . Face-texture model based on SGLD and its application in face detection in a color scene. *Pattern recognition*, 29(6):1007–1018, 1996.

29. Sobottka K. and I. Pitas . Extraction of facial regions and features using color

and shapeinformation. *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, 3, 1996.

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

31. Zarit B.D. , B.J. Super and F.K.H. Quek . Comparison of five color models in skin pixel classification. *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 1999. Proceedings. International Workshop on*, pages 58–63, 1999.

32. Yoo T.W. and I.S. Oh . A fast algorithm for tracking human faces based on chromatic histograms. *Pattern Recognition Letters*, 20(10):967–978, 1999.

33. Albiol A. , L. Torres and E.J. Delp . Optimum color spaces for skin detection. *Image Processing, 2001. Proceedings. 2001 International Conference on*, 1, 2001.

34. Shin M.C. , K.I. Chang and L.V. Tsap . Does colorspace transformation make any difference on skin detection? *Applications of Computer Vision, 2002.(WACV 2002). Proceedings. Sixth IEEE Workshop on*, pages 275–279, 2002.

35. Fleck M.M. , D.A. Forsyth and C. Bregler . Finding naked people. *European Conference on Computer Vision*, 2:592–602, 1996.

36. Sigal L. , S. Sclaroff and V. Athitsos . Skin color-based video segmentation under time-varying illumination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(7):862–877, 2004.

37. Terrillon J.C. , M.N. Shirazi , H. Fukamachi and S. Akamatsu . Comparative performance of different skin chrominance models andchrominance spaces for the automatic detection of human faces in colorimages. *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages

54–61, 2000.

38. Lee J.Y. and S.I. Yoo . An elliptical boundary model for skin color detection. *Proc. of the 2002 International Conference on Imaging Science, Systems, and Technology*, 2002.

39. Chen C. and S.P. Chiang . Detection of human faces in colour images. *Vision, Image and Signal Processing, IEE Proceedings-*, 144(6):384–388, 1997.

40. Phung S.L. , D. Chai and A. Bouzerdoum . A universal and robust human skin color model using neural networks. *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, 4, 2001.

41. Brown D. , I. Craw and J. Lewthwaite . A som based approach to skin detection with application in real time systems. *Proc. of the British Machine Vision Conference, 2001*, 2001.

42. Yang J. , W. Lu and A. Waibel . *Skin-color Modeling and Adaptation*. School of Computer Science, Carnegie Mellon University, 1997.

43. Scott D.W. . *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley-Interscience, 1992.

44. Solar J. and R. Verschae . Skin Detection using Neighborhood Information. *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, 2004.

45. Stokman H. and T. Gevers . Selection and Fusion of Color Models for Image Feature Detection. *IEEE Trans. Pattern Anal. Mach. Intell*, 29:371–381, 2007.

46. Markowitz H. . Portfolio Selection. *The Journal of Finance*, 7(1):77–91, 1952.

47. Wang Y. , Y. Zhang and J. Ostermann . *Video Processing and Communications*. Prentice Hall PTR Upper Saddle River, NJ, USA, 2001.

48. Cheng Y. . Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

49. Ojala T. , M. Pietikainen and T. Maenpaa . Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

50. Pietikäinen M. , T. Ojala and Z. Xu . Rotation-invariant texture classification using feature distributions. *Pattern Recognition*, 33(1):43–52, 2000.

51. Mäenpää T. , T. Ojala , M. Pietikäinen and M. Soriano . Robust texture classification by subsets of Local Binary Patterns. *Proceedings of the 15 thInternational Conference on Pattern Recognition*, 2000.

52. Lloyd S. . Least squares quantization in PCM. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.