

HAND GESTURE RECOGNITION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ÇANKAYA UNIVERSITY

BY

AHMET BİRDAL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JUNE 2007

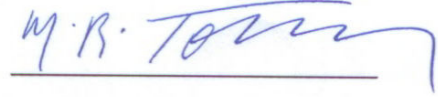
Title of the Thesis : **Hand Gesture Recognition**
Submitted by **Ahmet Birdal**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University



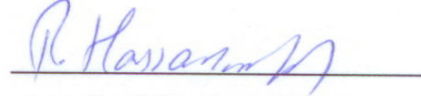
Prof. Dr. Yurdahan Güler
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.



Prof. Dr. Mehmet Reşit Tolun
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.



Asst. Prof. Dr. Reza Hassanpour
Supervisor

Examination Date : 18.06.2007

Examining Committee Members

Prof. Dr. Mehmet Reşit Tolun (Çankaya Univ.)



Asst. Prof. Dr. Reza Hassanpour (Çankaya Univ.)



Asst. Prof. Dr. Hasan Şakir Bilge (Gazi Univ.)



STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : AHMET BİRDAL

Signature :



Date : 18.06.2007

ABSTRACT

HAND GESTURE RECOGNITION

Birdal, Ahmet

M.Sc., Department of Computer Engineering

Supervisor : Asst. Prof. Dr. Reza Hassanpour

June 2007, 93 pages

Evolution of user interfaces shapes the change in the human-computer interaction devices. As three-dimensional (3-D) applications take place, the need for a new type of interaction device arises since traditional devices such as mouse, keyboard, joystick, etc... become inefficient for interaction within these virtual environments. A better interaction in virtual environments requires a natural and suitable device. "Hand Gesture" concept in human computer interaction which has become popular in recent years can be used to develop such an interaction device.

The motivation for this study is to develop a real-time, low cost, vision based hand gesture recognition system that works precisely on a relatively small restricted gesture space for single user robot control and human-computer interaction.

Keywords: Hand Gesture, Gesture Recognition

ÖZ

EL HAREKETİ TANIMA

Birdal, Ahmet

Yükseklisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Yöneticisi : Yrd. Doç. Dr. Reza Hassanpour

Haziran 2007, 93 sayfa

Kullanıcı arayüzlerinin gelişmesi insan-bilgisayar etkileşim cihazlarının değişimini şekillendirmiştir. Üç boyutlu uygulamalar yaygınlaştıkça etkileşim için kullanılan fare, klavye, oyun çubuğu gibi geleneksel cihazlar sanal ortamlarda yetersiz kalmaya başlamış, bu ortamlarda etkileşimi daha rahat sağlayabilecek yeni tip cihazlara ihtiyaç duyulmaya başlanmıştır. Bilgisayar-insan etkileşimi alanında son yıllarda popülerliği gittikçe artan “el hareketi tanıma” konusu bu sorunu aşmak için kullanılabilir çözümler üretebilmektedir.

Bu çalışmada tek kullanıcı, bilgisayar-insan etkileşimini ve robot kontrolünü sağlayan gerçek zamanlı çalışabilecek, düşük maliyetli, göreceli olarak az ve sınırlı sayıda el hareketi üzerinde hassas olarak çalışan, görmeye dayalı bir el hareketi tanıma sistemi geliştirilmesi amaçlanmıştır.

Anahtar Kelimeler: El Hareketi, Hareket Tanıma

ACKNOWLEDGMENTS

I would like to express my deep and sincere gratitude to my supervisor, Asst. Prof. Dr. Reza Hassanpour. His logical way of thinking and wide knowledge have been of great value for me. His understanding, supervision, encouragement, suggestions, personal guidance and trust have provided an excellent basis for the present thesis.

During this work I have collaborated with many colleagues for whom I have great regard, and I wish to extend my warmest thanks to all of my friends working at Meteksan Sistem and Aksa Teknoloji.

Finally, my special gratitude is to my family (Şeref, Gönül, Can Birdal and Tuba Ünlü) for their eternal love, support and trust in me. Without them I would never come up to this stage.

TABLE OF CONTENTS

STATEMENT OF NON PLAGIARISM.....	iii
ABSTRACT.....	iv
ÖZ.....	v
ACKNOWLEDGMENTS	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	xi
LIST OF FIGURES.....	xii
CHAPTERS:	
1. INTRODUCTION	1
1.1 Problem Definition.....	1
1.2 Classification of the Methods	2
1.2.1 Hand Gesture Recognition Systems	2
1.2.2 Vision Based Hand Gesture Recognition Systems.....	3
1.3 Definitions	6
1.3.1 Sign Languages	6
1.3.2 Gestures	7
1.3.3 Input Devices	8
1.4 Scope of the Study	11
2. BACKGROUND REVIEW	13

2.1	Input Devices	13
2.1.1	Wearable Computers	13
2.2	Studies Involving Hand Gesture Recognition	16
2.2.1	High Level Features	19
2.2.2	View-based Approaches	21
2.2.3	Low Level Features	23
2.3	Gesture Classification	26
2.3.1	Rule-Based Approaches.....	27
2.3.2	Learning Based Approaches	28
3.	FUNDAMENTAL ELEMENTS IN VISION BASED HAND GESTURE RECOGNITION SYSTEMS	30
3.1	Color Models	30
3.1.1	The RGB Color Model	30
3.1.2	The CMYK Model	33
3.1.3	YIQ and YUV Models.....	35
3.1.4	HSI, HSV and YCbCr Models.....	37
3.2	Region Labeling.....	40
3.3	Smoothing Linear Filters.....	42
3.4	Morphological Operations.....	43
3.4.1	Binary Erosion and Dilation	43
3.5	Segmentation	45
3.5.1	Difference-Based Segmentation.....	46
3.5.2	Background Subtraction	46
3.5.3	Color-Based Segmentation	49

3.5.4	Marker-Based Segmentation.....	49
3.6	Tracking.....	50
3.6.1	Meanshift	50
4.	PROPOSED METHOD	52
4.1	Application Environment	52
4.2	Hand Features	53
4.2.1	Boundary Rectangles.....	53
4.2.2	Hand Postures.....	55
4.2.3	Hand Positions.....	56
4.3	System Design	60
4.3.1	Image Acquisition	61
4.3.2	Background Modeling	61
4.3.3	Initialization	63
4.3.3.1	Segmentation	65
4.3.3.1.1	Background Subtraction	65
4.3.3.1.2	Skin Color Filtering.....	66
4.3.3.1.3	Skin Color Modeling	66
4.3.3.1.4	Skin Color Segmentation.....	69
4.3.3.1.5	Noise Removal	70
4.3.3.2	Calculating Initial Parameters.....	72
4.3.4	Main Loop.....	73
4.3.4.1	Tracking.....	75

4.3.4.2	Region Checking.....	78
4.3.4.3	Feature Extraction.....	79
4.3.4.4	Classification	80
4.4	The Output.....	82
5.	EXPERIMENTAL RESULTS	84
5.1	Test Application.....	85
6.	CONCLUSIONS AND FUTURE WORK	91
	REFERENCES	R1

LIST OF TABLES

Table 2.1 Action Conditions	27
Table 4.1 State Values	80
Table 5.1 Gesture States	87

LIST OF FIGURES

Figure 1.1 Hand Models	3
Figure 1.2 Anatomic Structure of the Hand.....	4
Figure 1.3 Colorful Gloves	4
Figure 1.4 Hand.....	5
Figure 1.5 Hand with center of hand point.....	5
Figure 1.6 Hand with ellipse.....	5
Figure 1.7 Classification of Gestures	7
Figure 2.1 Wearable Computer-1.....	14
Figure 2.2 Wearable Computer-2.....	14
Figure 2.3 Data Glove	15
Figure 2.4 Tv Control.....	17
Figure 2.5 Anthropometrical Measurements	20
Figure 2.6 Center of gravity and extreme points of the hand	25
Figure 2.7 Action Motion Blobs	28
Figure 3.1 Sample RGB Image.....	31
Figure 3.2 RGB Components of the Sample Image.....	32
Figure 3.3 Cyan.....	34
Figure 3.4 Magenta	34
Figure 3.5 Yellow.....	35

Figure 3.6 Key	35
Figure 3.7 Luminance.....	36
Figure 3.8 Hue	39
Figure 3.9 Neighbor Pixels	41
Figure 3.10.a Background Subtraction.....	47
Figure 3.10.b Background Subtraction.....	48
Figure 3.10.c Background Subtraction.....	48
Figure 4.1 Boundary Rectangle	53
Figure 4.2 Vertical Posture	56
Figure 4.3 Horizontal Posture	56
Figure 4.4 Center point of the bounding rectangle	57
Figure 4.5 Circle-like structure	58
Figure 4.6 Elbow reference.....	58
Figure 4.7 Circle-like structure (marked).....	59
Figure 4.8 Hand Gesture Regions	60
Figure 4.9 Background Modeling	61
Figure 4.10 Background Illustration	62
Figure 4.11 Special Gesture.....	63
Figure 4.12 Initialization	64
Figure 4.13 Manual Skin Segmentation	68
Figure 4.14 Ideal Skin Color Segmentation Output.....	69
Figure 4.15 Background Segmented Image.....	70
Figure 4.16 Skin Segmented Image	71
Figure 4.17 Morphological Operations	71

Figure 4.18 Smoothing Filter.....	72
Figure 4.19 Initialization Output.....	73
Figure 4.20 Main Loop.....	74
Figure 4.21 Tracking Windows	77
Figure 4.22 Hand Gesture Region Locations	78
Figure 4.23 Sample States	81
Figure 5.1 OpenGL Application	85
Figure 5.2 Rotation.....	86
Figure 5.3 Translation	86
Figure 5.4 Axis.....	88
Figure 5.5 Positive Direction	89
Figure 5.6 Negative Direction	89

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

Evolution of user interfaces shapes the change in the human-computer interaction devices. One of the most common human-computer interaction devices is the keyboard which has been the ideal choice for text-based user interfaces. Graphical user interfaces brought mouse into the desktops of the users. As three-dimensional (3-D) applications take place the need for a new type of interaction device arises since traditional devices such as mouse, keyboard, joystick, etc... become inefficient for interaction within these virtual environments. A better interaction in virtual environments requires a natural and suitable device. "Hand Gesture" concept in human computer interaction which has become popular in recent years can be used to develop such an interaction device.

The motivation for this study is to develop a real-time, low cost, vision based hand gesture recognition system that works precisely on a relatively small restricted gesture space for single user robot control and human-computer interaction in such an environment that the lighting is relatively stable and the background is not complex.

1.2 CLASSIFICATION OF THE METHODS

1.2.1 Hand Gesture Recognition Systems

Hand gesture recognition is a relatively new field for the computer science. As it can be easily derived from the name, aim of these systems is to recognize some human hand gestures (sometimes in specific conditions) and use this input for specific purposes.

Applications for hand gesture recognition in machine learning systems have been developed approximately for 20 years. These methods used in these systems can be categorized into two groups. Generally the earlier systems like Liang et. al. [1] used gloves for gesture recognition. That method was unpractical since the user whose gestures will be recognized should wear some special gloves. These gloves were limiting moving abilities of the user. Therefore, recent studies like Malima et. al. [2] concentrated on vision-based systems since they provide relatively cost-effective methods to acquire and interpret human hand gestures while being minimally obtrusive to the participant.

1.2.2 Vision Based Hand Gesture Recognition Systems

A typical vision based hand gesture recognition system consists of a camera(s), a feature extraction module, a gesture classification module and a set of gesture models. In the feature extraction process the necessary features are extracted from the captured frames of the camera(s). This process can be divided into three sub categories:

- High-level features those generally based on three dimensional models,
- The image itself as a feature by view-based approaches,
- Low-level features measured from the image.

High level features can be inferred from the joint angles and pose of the palm.

Figure 1.1 is a classification of the hand models.

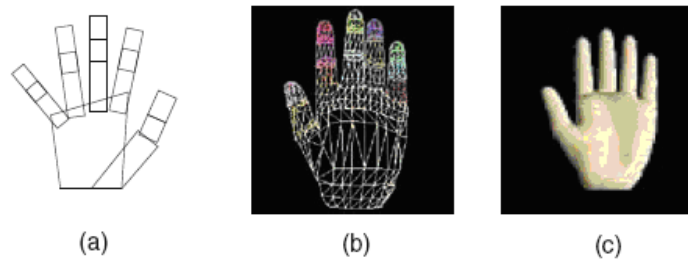


Figure 1.1 : Hand Models (a) Cardboard, (b) Wireframe, and (c) Polygon-mesh
(from [3])

For this feature set, generally the anatomic structure of the hand is used as a reference.

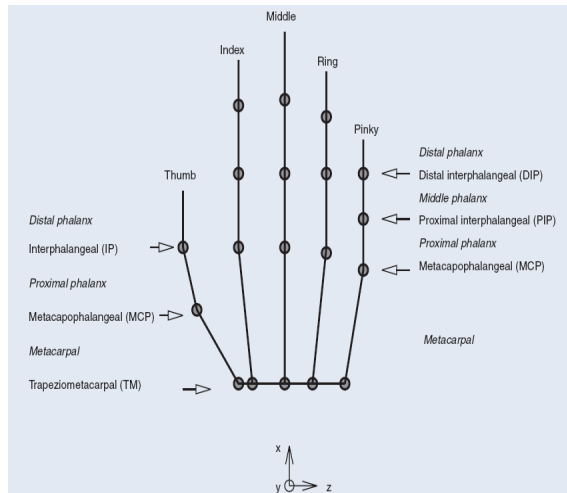


Figure 1.2 Anatomic Structure of the Hand [3]

For precision purposes colorful gloves can be used as illustrated in Figure 1.3 in this approach.

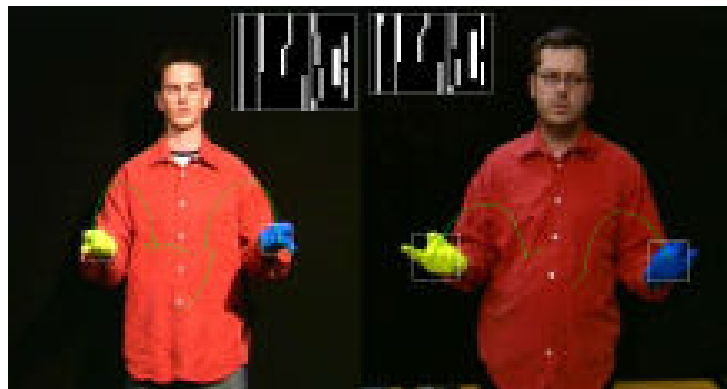


Figure 1.3 Colorful Gloves [4]

View-based approaches are alternatives to the high-level modeling and they model the hands as a set of two dimensional intensity images.

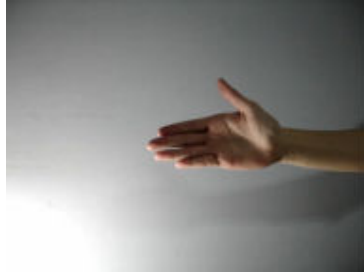


Figure 1.4 Hand

Low level features are based on the thought that the full reconstruction of the hand is not essential for gesture recognition. Therefore only some cues like the centroid of the hand region, the principle axes defining an elliptical bounding region of the hand, the optical flow/affine flow of the hand region in a scene, etc can be chosen as features.

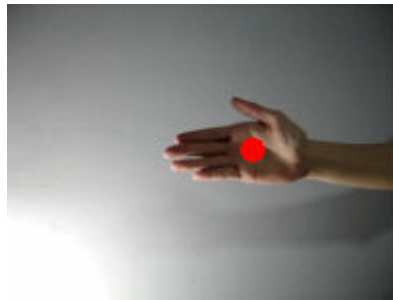


Figure 1.5 Hand with center of hand point

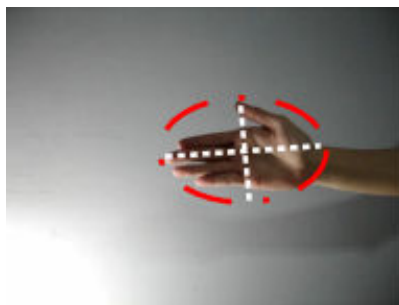


Figure 1.6 Hand with ellipse

Low level features are used in this work since these features are easy and fast to track and recognize unlike the high level features which takes relatively much more processing time due to its 3D nature.

1.3 DEFINITIONS

1.3.1 Sign Languages

Utsumi et. al. [5] defines a sign language as “a language which uses manual communication instead of sound to convey meaning - simultaneously combining hand shapes, orientation and movement of the hands, arms or body, and facial expressions to express fluidly a speaker's thoughts”. Sign languages are generally used in the communities those consist of people who have hearing disabilities or who are hard of hearing.

Sign language differs from one region to another as is the case in spoken language. However, communication via sign language is easier than the spoken language when two people with different languages meet. By this way, sign language can be thought as more common when international communication considered. However, sign language is not universal because there are many distinct sign languages that have evolved independently of each other. [6]

1.3.2 Gestures

Gesture is defined as “A motion of hands or body to emphasize or help to express a thought or feeling” and also as “the use of movements, especially of the hands, to communicate familiar or prearranged signals” [7]. Generally gestures do not replace the speech except for deaf people; instead they assist the speech to clarify the meaning.

Hoang et al [8] classify the gestures as follows:

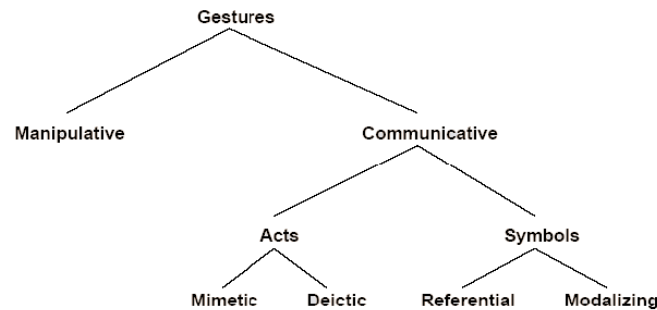


Figure 1.7 Classification of Gestures [8]

Gestures themselves can have two modalities: communicative and manipulative. Manipulative gestures are the ones used to act on objects in an environment (object movement, rotation, etc.). Communicative gestures, on the other hand, have an inherent communicational purpose. In a natural

environment they are usually accompanied by speech. Communicative gestures can be either acts or symbols. Symbols are those gestures that have a linguistic role. They symbolize some referential action (for instance, circular motion of index finger may be a sign for a wheel) or are used as modalizers, often of speech ("Look at that wing!" and a modalizing gesture specifying that the wing is vibrating, for example). In the HCI context, symbols are, so far, one of the most commonly used gestures since they can often be represented by different static hand postures. Finally, acts are gestures that are directly related to the interpretation of the movement itself. Such movements are classified as either mimetic (which imitate some actions) or deictic (pointing acts).

This study is focused on hand gestures. Therefore, within the work gestures are defined as dynamic hand gestures and postures as static hand gestures. Postures are called those not time varying and a gesture is called a sequence of postures over a short time span.

1.3.3 Input Devices

As the technology improves, computers become more involved in daily lives. Face to face communication shifts towards the medium of email, chat, video-conferencing, and this kind of applications of computers and mobile phones. Some daily life processes like cooking; washing dishes and laundry, brushing teeth, controlling the television, etc. become computer-based. Since computers

take such important roles in our lives, interaction with these devices should not be so complicated and unpractical.

Long and Long [9] classifies and defines the traditional input devices as follows:

1) KEYBOARD

2) POINT-AND-DRAW DEVICES

Mouse

Joystick

Trackball

Mouse Pen

Digitizer Tablet and Pen

Trackpoints

Trackpads

The keyboard is the most common device for transferring user input to the computer system, similar to a typewriter. However, keyboard is too cumbersome for some applications, especially those that rely on a graphical user interface or require the user to point and draw. The effectiveness of a GUI depends on the user's ability to make a rapid selection from a screen full of graphic icons or menus. In these instances the mouse can position the pointer over an icon quickly and efficiently. Currently, the mouse remains the most popular point-and-draw device.

The joystick is a vertical stick that moves the graphics cursor in the direction the stick is pushed. The trackball is a ball inset in a small external box or adjacent to and in the same unit as the keyboard. The ball is “rolled” with the fingers to move the graphics cursor. The mouse pen is rolled across any smooth surface, but it is held like pen. The digitizer tablet and pen are a pen and a pressure-sensitive tablet whose X-Y coordinates correspond with those on the computer’s display screen. Track points are usually positioned in or near a laptop’s keyboard. They function like miniature joysticks but are operated with the tip of the finger. The trackpad has no moving parts. Simply move your finger about a touch-sensitive pad to move the graphics cursor.

Apart from the above input devices, webcams are generally used as input devices for the vision based recognition systems.

A webcam is a type of video camera, generally connected to a computer directly, whose current frame is requestable from a web site. A typical webcam consists of a lens, an image sensor plane, and some support electronics. The most commonly used lens for webcams is a plastic lens that can be rotated clockwise and counterclockwise to set the camera’s focus.

There are two types of image sensors: CMOS and CCD. CMOS (Complementary Metal-Oxide-Semiconductor) is an image sensor consisting of an integrated circuit containing an array of pixel sensors, each containing a photodetector and connecting to an active transistor reset and readout circuit. A charge-coupled device (CCD) is an image sensor, consisting of an integrated

circuit containing an array of linked, or coupled, light-sensitive capacitors. This device is also known as a Color-Capture Device. [10]

CCD is still the technology of choice in high-end camcorders and digital still cameras, as well as science and astronomy. This supremacy is challenged by the novel developments in CMOS imagers. The latter are expected to penetrate into existing CCD markets, for example in machine vision, due to the improved performances in terms of random access and speed as well as in the field of security / surveillance due to the higher integration / miniaturization. CMOS image sensors will be the technology of choice for new emerging markets such as portable devices, consumer, biomedical, biometric and automotive, as well as applications that rely on custom sensors and smart pixels to locally extract the relevant information, such as the position, presence, distance or temperature of an object. But for the foreseeable future both technologies will remain to a large extent complementary. [11]

Recently, consumer level webcams have the speed around 25 frames per second and resolution around 1.3 Megapixels.

1.4 SCOPE OF THE STUDY

The scope of this study covers the following issues for hand gesture recognition:

Speed: The system is expected to work real-time.

Cost: A relatively low cost system is designed. The components (like video camera, personal computer, etc) are aimed to be in consumer-market segment.

Basic Method: The system is considered to be vision based.

Gesture Space: Relatively small restricted gesture space is accepted in the system.

Basic Restrictions: Occlusion of the skin parts is not accepted. Skin color should not used for clothing and contained at background. Background should be relatively static and not be complex.

Users: The system is operated by just one user.

Lighting: Controlled lighting is used. Daylight is preferred for the system environment.

CHAPTER 2

BACKGROUND REVIEW

2.1 INPUT DEVICES

2.1.1 Wearable Computers

One of the most popular wearable computers is the type that contains head-attached interface devices. This type of interface devices can contain multiple input and output components like LCD screens, microphones, webcams, earphones, etc. Shimoda et. al. [12] presents a device named HIDE that has flexible integrated functions which consist of speech recognition and view direction detection as hand-free input channels, direct presentation of audio/visual information as output channels, and mobile system configuration. As a sample application software for the HIDE, they developed a World Wide Web browser which can be controlled by the view direction detection and the speech recognition.

Head-attached interface devices are also used together with hand gesture recognition. Kolsch and Hollerer [13] present a real-time hand gesture

recognition system that uses the camera on the human head or object of interest. Preliminary evaluation of their prototype system leads to the conclusion that vision-based interfaces have achieved the maturity necessary to help overcome some limitations of more traditional mobile user interfaces.

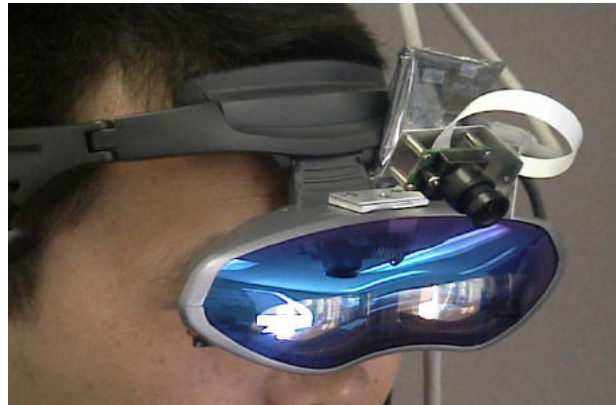


Figure 2.1 Wearable Computer-1 [13]



Figure 2.2 Wearable Computer-2 [13]

Sorrentino et. al. [14] developed “Virtual Office Environment System” that controls a virtual character: “avatar”. A head-mounted wearable device was used in the study.

Another type of components used in wearable computers is glove. Malkawi and Srinivasan [15] studied on an interactive speech and hand gesture recognition-based, immersive Augmented Reality model designed to visualize and interact with buildings and their thermal environments. In the study, they use special gloves named “CyberGlove”. CyberGlove was used to capture global hand motion using trackers attached to the glove and local finger motion as a set of joint angles. Metais and Georganas [16] also used CyberGlove for their glove gesture interface study. Kavaklı and Jayarathna [17] described a virtual environment in which a designer can define the contour of a sketch by controlling a pointer using a pair of data gloves in 3D space.



Figure 2.3 Data Glove [18]

Marschall [19] described the development of a gesture-based interface system for interactive multimedia applications.

Glove is also used as an input device for computer-vision based interaction devices. The special markers on the glove can provide clear location information of the hand in each frame of camera caption. Yachida and Iwai [20] used a glove with twelve different colored regions corresponding to feature areas in their hand model. Maggioni and Kammerer [21] placed eccentric circles on the gloves so that the segmentation is easily performed knowing the color properties of the marker and its geometrical properties yields the location of the marker and hence the hand. For similar purposes Davis and Shah [22] used identical white markers for each fingertip, and a different marker for the thumb tip.

2.2 STUDIES INVOLVING HAND GESTURE RECOGNITION

In computer science, there are many studies for sign language applications. One of the most popular areas is recognition of a local sign language. Liang et. al. [23] worked on Taiwanese sign language, Starner et. al. [24] worked on American Sign Language and Haberdar [25] studied Turkish Sign Language for his thesis work. Similarly Gejgus et. al. [26] worked on finger alphabet. The general purpose of these applications is either helping the deaf-dumb

people for their communication with others or completely translating from a sign language into a normal one.

Another type of application about sign languages is man-computer interaction, in other words, using sign languages as input, the information conveyed by the gestures is transferred to the computer via webcam/webcams. Eisenstein and Davis [27] controlled a display in their application. Bretzner [28] developed a prototype system, where the user can control a TV set and a lamp.



Figure 2.4 Tv Control [28]

Robot control is the aim of the works of Ren et. al. [29], Malima et. al. [2], Ludovic et. al. [1], Agrawal and Chaudhuri [30], Starner et. al. [14], Postigo et. al. [31]. Malima et. al. [2], propose an algorithm for automatically recognizing a limited set of gestures from hand images for a robot control application. The algorithm enables the robot to identify a hand pose sign in the input image, as one of five possible commands. The identified command will then be used as a

controller input for the robot to perform a certain task. They indicate that they used a 4 megapixel camera as well as a simple webcam for the research.

Application of Kolsch and Hollerer [13] helps people with a wearable device. Fujisawa et. al. [32] developed an HID device as an alternative for mouse for physically handicapped persons. Human-Building interaction is considered at Malkawi and Srinivasan [15] . Marschall [19] has an interesting application which provides a visual sculpture. Pedestrian tracking from a moving vehicle is the primary goal of Philomin et. al . [33]. Mantyla et. al. [34] developed a system for mobile device users.

While some of the works mentioned above uses a complete sign language, some of them uses just a part of a sign language or develops an application-specific sign language for human-computer interaction.

As summarized in section 1.2.2 the approaches for the features used in the mentioned and similar applications can be classified as the following:

- i. High Level Features
- ii. View-based Approaches
- iii. Low Level Features

2.2.1 High Level Features

High level features are extracted by model based approaches. A typical model based approach is creating a 3D model of a hand and projecting its edges onto 2D space by using some kinematics parameters.

Ueda et. al. [35] used such a method that estimates all joint angles to manipulate an object in the virtual space. In the method, the hand regions are extracted from multiple images obtained by the multi-viewpoint camera system. By integrating these multi-viewpoint silhouette images, a hand pose is reconstructed as a “voxel model”. Then all joint angles are estimated using three dimensional model fitting between hand model and voxel model. An experiment was performed in which the joint angles were estimated from the silhouette images by the hand-pose simulator.

Utsumi et. al. [5] used multi-viewpoint images to control objects in the virtual world. Eight kinds of commands are recognized based on the shape and movement of the hands.

Bray et. al. [36] proposed a tracker based on ‘Stochastic Meta-Descent’ for optimizations in such high dimensional state spaces. The algorithm is based on a gradient descent approach with adaptive and parameter-specific step sizes. The Stochastic Meta-Descent tracker facilitates the integration of constraints, and combined with a stochastic sampling technique, can get out of spurious

local minima. Furthermore, the integration of a deformable hand model based on linear blend skinning and anthropometrical measurements reinforce the robustness of the tracker.

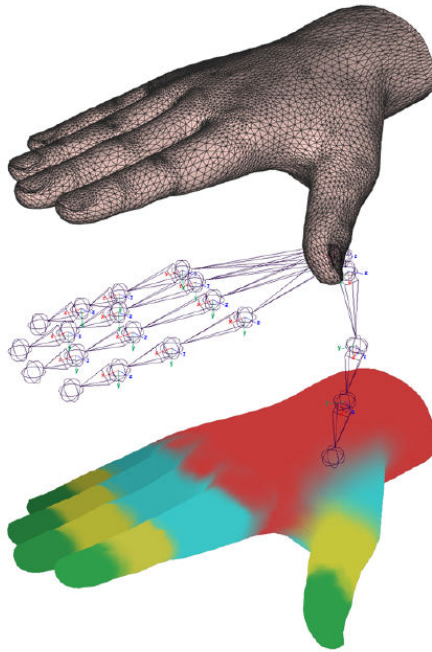


Figure 2.5 Anthropometrical Measurements [36]

Bettio et. al. [37] presented a practical approach for developing interactive environments that allows humans to interact with large complex 3D models without them having to manually operate input devices. The system provides support for scene manipulation based on hand tracking and gesture recognition and for direct 3D interaction with the 3D models in the display space if a suitably registered 3D display is used. Being based on markerless tracking of a

user's two hands, the system does not require users to wear any input or output devices.

In model based approaches the initial parameters have to be close to the solution at each frame and noise is a real problem for fitting process. Another problem is the textureless nature of the human hand difficulties to detect the inner edges of the hand. Davis and Shah [22] , Dorner [38] and Lee and Kunii [39] used a glove with markers in order to make the feature extraction process easier. Similarly manual parameter instantiation or placing user hands in a specific position were also used for the ease of initialization process.

Processes on these features may be relatively slower than the other feature approaches due to the 3D structure complexity of high-level features.

2.2.2 View-based Approaches

These approaches are also called appearance-based based approaches. As described in the section 1.2.2 these approaches model the hand by a collection of 2D intensity images. At the same time, gestures are modeled as a sequence of views.

Eigenspace approaches are used within the view-based approaches. They provide an efficient representation of a large set of high dimensional points

using a small set of orthogonal basis vectors. These basis vectors span a subspace of the training set called the eigenspace and a linear combination of these images can be used to approximately reconstruct any of the training images. These approaches were used in many of the face recognition approaches. Success of them in face recognition made them attractive for other recognition applications like hand gesture recognition. (e.g. Gupta et. al. [40] and Black [41])

Black [41] demonstrated their approach by tracking four hand gestures with 25 basis images and provided three major improvements to the original eigenspace approach formulation:

- i) A large invariance to occlusions
- ii) Some invariance to differences in background from the input images and the training images
- iii) The ability to handle both small and large affine transformations of the input image with respect to the training images

Zahedi et. al. [42] presented how appearance-based features can be used for the recognition of words in American Sign Language from a video stream. The features are extracted without any segmentation or tracking of the hands or head of the signer. Experiments are performed on a database that consists of 10 words in American Sign Language with 110 utterances in total. The video streams of two stationary cameras are used for classification. Hidden Markov Models and the leaving one out method are employed for training and

classification. Using the appearance-based features, they achieved an error rate of 7%. About half of the remaining errors are due to words that are visually different from all other utterances.

Although these approaches may be sufficient for a small set of gestures, with a large gesture space collecting adequate training sets may be problematic. Another problem maybe the loss of compactness in the subspace required for efficient processing.

2.2.3 Low Level Features

Starner et. al. [24] noticed that prior systems could recover relatively detailed models of the hands from video images when given some constraints. However, many of those constraints conflicted with recognizing American Sign Language in a natural context, either by requiring simple, unchanging backgrounds; not allowing occlusion; requiring carefully labeled gloves; or being difficult to run in real time. Therefore they presented such a new and relatively simple feature space that assumes detailed information about hand shape is not necessary for humans to interpret sign language. They found that all human hands have approximately the same hue and saturation, and vary primarily in their brightness. By using this color cue they used the low level features of hand's x and y position, angle of axis of least inertia, and eccentricity of the bounding ellipse. This feature set is one of the first low-level features in the literature for hand gesture concept of computer vision. They

combined the low-level feature set by HMM network and achieved the accuracy of %97 per word on a 40 word lexicon.

Göknaar and Yıldırım [43] presented a hand gesture recognition system using an inexpensive camera with fast computation time. They used skin tone density and eccentricity of the bounding ellipse low level features and Multilayer Perceptron and Radial Basis Function neural networks for classification. They achieved the success of %78.3 on 3 layered structure and %80 for 4 layered structure.

Lee [44] used low level feature, the distance from the centroid of the hand region to the contour boundary. The method obtains the image through subtract one image from another sequential image, measuring the entropy, separating hand region from images, tracking the hand region and recognizing hand gestures. Through entropy measurement, they have got color information that have near distribution in complexion for region that have big value and extracted hand region from input images. They could draw hand region adaptively in change of lighting or individual's difference because entropy offer color information as well as motion information at the same time. Detected contour using chain code for hand region that is extracted, and present centroidal profile method that is improved little more and recognized gesture of hand. In the experimental results for 6 kinds of hand gesture, the recognition rate was found more than %95.

Malima et. al. [2] proposed a fast algorithm for automatically recognizing a limited set of gestures from hand images for a robot control application. They considered a fixed set of manual commands and a reasonably structured environment, and developed a procedure for gesture recognition. The algorithm is invariant to translation, rotation, and scale of the hand. The low-level feature used in the algorithm is the center of the gravity and the distance from the most extreme point in the hand to the center which is the farthest distance from centroid to tip of the longest active finger in the particular gesture.

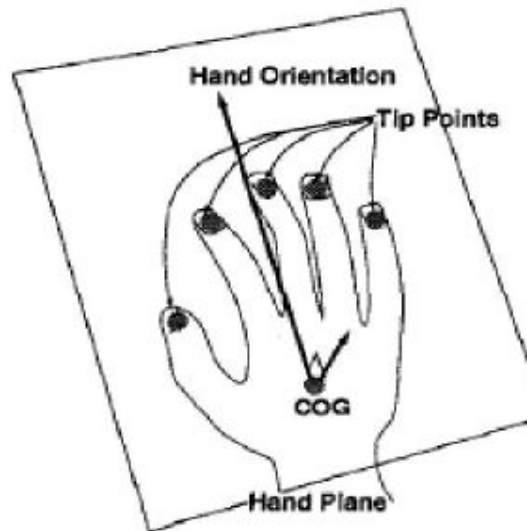


Figure 2.6: Center of gravity and extreme points of the hand [2]

Yang [45] presented an algorithm for extracting and classifying two-dimensional motion in an image sequence based on motion trajectories. First, a multi-scale segmentation is performed to generate homogeneous regions in each frame. Regions between consecutive frames are then matched to obtain

two-view correspondences. Affine transformations are computed from each pair of corresponding regions to define pixel matches. Pixels matches over consecutive image pairs are concatenated to obtain pixel-level motion trajectories across the image sequence. Motion patterns are learned from the extracted trajectories using a time-delay neural network. They applied the proposed method to recognize 40 hand gestures of American Sign Language. They approximated the human head and hand shapes by ellipses.

Roy and Jawahar [46] presented a feature selection method for hand geometry based person authentication system. They used lengths of four fingers and widths at five equidistant points on each finger as raw features.

Since the localization of hands in arbitrary scenes is difficult, one of the major difficulties associated with low level features is that the hand has to be localized before feature extraction.

2.3 GESTURE CLASSIFICATION

The hand gesture classification approaches in the literature can be categorized into two main categories: rule-based approaches, in which the gestures are classified according to manually encoded rules and machine learning based approaches those are using a set of exemplars to infer models of gestures.

2.3.1 Rule-Based Approaches

In these approaches features of the input features are compared to the manually encoded rules. If any of the features or feature sets matches a rule, the related gesture will be given as output. As an example Cutler and Turk [47] used a rule-based technique to identify an action based on a set of conditions in their view-based approach to gesture recognition. They defined six motion rules for corresponding six gestures. When the hands trace a motion path like a predefined rule, corresponding gesture is selected as output. The gestures and the rules used in the work are defined as the following:

Table 2.1: Action Conditions

Action	# of Blobs	Abs. Motion	Relative Motion	Size
waving	1	rot.	NA	0.2
jumping	1	vert.	NA	0.8
clapping	2	horiz.	oppose	0.2
drumming	2	vert.	oppose	0.4
flapping	2	rot.	same	0.8
marching	2	vert.	oppose	0.4

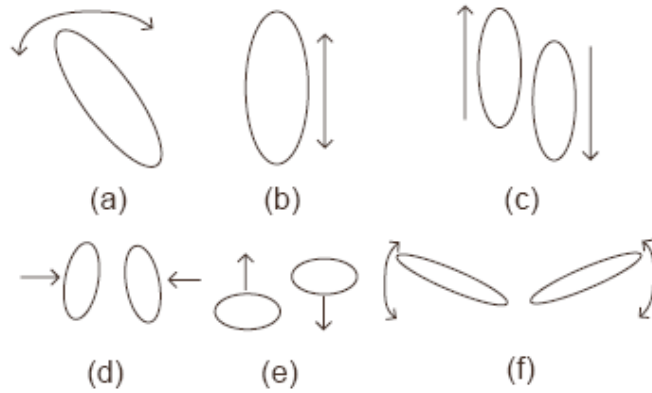


Figure 2.7 : Action Motion Blobs
 a) waving b) jumping c)marching d) clapping e) drumming f) flapping
 [47]

2.3.2 Learning Based Approaches

As indicated in the previous section, the rule-based approaches depend on the ability of humans to find rules to classify the gestures. Learning-based approaches are alternative solutions to this problem when finding rules between features is not applicable. In this approach mappings between high-dimensional feature sets and gestures are done by machine learning algorithms.

The most popular method for this approach is using Hidden Markov Models in which gestures are treated as the output of a stochastic process. Many of the recent works Nair and Clark [48], Starner et. al. [24], Zheng [49] and Marcel [50] focused on Hidden Markov Models for gesture recognition. Russell and Norvig [51] defines the HMM as “a temporal probabilistic model in which the

state of the process is described by a single discrete random variable.” The possible values of the variable are the possible states of the world.

Haberdar [25] used HMM for gesture recognition in his thesis study which is about Turkish Sign Language recognition. In the study 172 signs are used. Model parameters are determined by training G different HMM and using training data. For recognition of a sequence of observations, forward and backward algorithms are used by calculating the probabilities of HMM to generate the related observation.

CHAPTER 3

FUNDAMENTAL ELEMENTS IN VISION BASED HAND GESTURE RECOGNITION SYSTEMS

3.1 COLOR MODELS

Color models are used for specifying the colors in a standard way. In many sources the titles “Color Model”, “Color Space” and “Color System” are interchangeably used.

Selection of an appropriate color model depends on the type of application planning to be developed. Primary color components, luminance, saturation and hue are the most frequently used features to represent a color in a color space.

3.1.1 The RGB Color Model

The abbreviation and the name of the model come from the three primary colors, red, green and blue. In the RGB Model each color can be reproduced by the additive mixing of the three primary color intensity values ranging from 0 (no color) to 255 (full intensity) in the computer science. The sample image

illustrated in Figure 3.1 consists of three component images illustrated in Figure 3.2.



Figure 3.1 : Sample RGB image

When the original sample image is compared with the following RGB components of the image the effect of the related components can be seen easily. For example, the red paint on the rod at the right side of the picture is represented by dark tones in the corresponding region of the green (Figure 3.2, b) and blue (Figure 3.2, c) component images while the same part is nearly white within the same region at the red component image (Figure 3.2,a) . The reason for this difference is that the green and blue components have values near 0 within the region therefore this region has darker tones compared to the red component which has higher values represented by brighter tones within the region.



(a)



(b)



(c)

Figure 3.2 (a) Red Component, (b) Green Component, (c) Blue Component

3.1.2 CMYK Model

CMYK color model is taking its name from and based on mixing pigments of Cyan, Magenta, Yellow, and Key (Black). In CMYK, magenta plus yellow produces red, magenta plus cyan makes blue and cyan plus yellow generates green. Due to its subtractive nature this model is used by most devices that deposit colored pigments on paper, such as color printers and copiers. The mixture of ideal CMY colors is subtractive, that means equal amounts of the pigment primaries, cyan, magenta and yellow produce black, in theory. But, in practice combining these colors for printing produces a muddy-looking black. Therefore, in order to produce true black, the system needs the fourth space which is black. [52]

Black color in CMYK is referred to using the letter K for key ; a shorthand for the printing term key plate . This plate impressed the artistic detail of an image, usually in black ink. This use of the letter K also helped avoid confusion with the letter B as used in the acronym RGB. The amount of black to use to replace amounts of the other ink is variable, and the choice depends on the technology, paper and ink in use. [53]

Converting the model from RGB to CMY is done by 3.1 [52]:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (3.1)$$

Following figures Figure 3.3 , Figure 3.4, Figure 3.5 and Figure 3.6 are the resulting images of the RGB to CMYK conversion process for the sample image at Figure 3.1:



Figure 3.3 Cyan



Figure 3.4 Magenta



Figure 3.5 Yellow



Figure 3.6 Key

3.1.3 YIQ and YUV Models

YIQ or YUV color models are used in video systems in order to minimize the bandwidth since the RGB model uses much more bandwidth when compared to them. In YUV, Y stands for the luminance component (the brightness) and U and V are the chrominance (color) components whereas in YIQ the Y component similarly represents the luminance and I stands for in-phase, while Q stands for quadrature, referring to the components used in quadrature

amplitude modulation. Therefore, in both color models Y component is represented by the grayscale form (brightness) of the original RGB image while other components store the color information.



Figure 3.7 Grayscale (Luminance) Image for Figure 3.1

The relationship between the RGB color space and these two color spaces are formulated as 3.2 and 3.3 [52]:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (3.2)$$

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.437 \\ 0.615 & -0.515 & -0.1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (3.3)$$

3.1.4 HSI, HSV and YCbCr Color Models

The three color models HSI (Hue Saturation Intensity), HSV (Hue Saturation Value) and YCbCr (Luminance, Blue Chroma, Red Chroma) have significant similarities with the human vision perception system. Therefore it is generally practical to use these models for image processing applications.

The HSV color model attempts to characterize colors according to their hue, saturation, and value (brightness) whereas the HSL color model, also called HLS or HSI, attempts to characterize colors according to their hue, saturation, and Luminance. HSL and HSV have the same definition of hue, but the other components differ.

Hue component of both HSV and HSI models can be formulized as 3.4 to 3.7:

$$H = \begin{cases} \theta & , \text{ if } B \leq G \\ 360 - \theta & , \text{ if } B > G \end{cases} \quad (3.4)$$

$$\theta = \cos^{-1} \left\{ \frac{\frac{1}{2} [(R-G) + (R-B)]}{\left[\frac{1}{4} [(R-G)^2 + (R-B)(G-B)] \right]^{1/2}} \right\} \quad (3.5)$$

$$S = 1 - \frac{3}{(R+G+B)} [\min(R,G,B)] \quad (3.6)$$

$$I = \frac{1}{3} (R+G+B) \quad (3.7)$$

Formulation of S and V components of HSV model [54] is shown in 3.8 and 3.9 below:

$$S = \begin{cases} 0, & \text{if } MAX = 0 \\ 1 - \frac{MIN}{MAX}, & \text{otherwise} \end{cases} \quad (3.8)$$

$$V = MAX \quad (3.9)$$

Formulation of S and L components of HSL model is shown in 3.10 and 3.1:

$$S = \begin{cases} 0, & \text{if } L=0 \text{ or } MAX=MIN \\ \frac{MAX-MIN}{MAX+MIN} = \frac{MAX-MIN}{2L}, & \text{if } 0 < L \leq \frac{1}{2} \\ \frac{MAX-MIN}{2-(MAX+MIN)} = \frac{MAX-MIN}{2-2L}, & \text{if } L > \frac{1}{2} \end{cases} \quad (3.10)$$

$$L = \frac{1}{2}(MAX + MIN) \quad (3.11)$$

Following Figure 3.8 is the H component which is the common component of the HSL and HSI models, for the sample image at Figure 3.1



Figure 3.8 Hue

YCbCr is a family of color models used in video systems. Y is the luminance component and Cb and Cr are the blue and red chroma components. It is often confused with the YUV color model, and typically the terms YCbCr and YUV are used interchangeably, leading to confusion. In fact, when referring to signals in digital form, the term "YUV" probably really means "YCbCr" more often than not. YCbCr is also different than YIQ color model. However, Y component for these three color models which is luminance is the same for all.

Due to its nature this model can be for color segmentation part of the digital image processing applications.

3.2 REGION LABELING

Region labeling ,also called blob coloring or connected component identification, is a very useful process for identifying and labeling the objects in a binary image so they can be seperately modified, displayed or manipulated. A typical use of this process, for example, is displaying each object with a different color that identifies the respective object after labeling.

The algorithm simply finds the connected groups of pixels, that is finding pixels with the same binary value, in a binary image. This can be done by scanning the image and searching for the pixels those have the same binary value and are connected. Bovik and Desai [55] describe the algorithm as the following steps by assuming the region labels used are positive integers:

Region Labeling Algorithm

1. Given an $N \times M$ binary image f , initialize an associated $N \times M$ region label array: $r(n)=0$ for all n . Also initialize a region number counter :
 $k=1$.

Then, scanning the image from left to right and top to bottom, for every n do the following:

2. If $f(n) = 0$ then do nothing.

3. If $f(n)=1$ and also $f(n-(1,0)) = f(n-(0,1)) = 0$, then set $r(n) = 0$ and $k=k+1$. In this case the left and upper neighbors of $f(n)$ do not belong to objects. (Figure 3.9, a)

4. If $f(n) = 1$, $f(n-(1,0))=1$, and $f(n-(0,1))=0$ then set $r(n) = r(n-(1,0))$. In this case the upper neighbor $f(n-(1,0))$ belongs to the same object as $f(n)$. (Figure 3.9, b)

5. If $f(n) = 1$, and $f(n-(0,1)) = 0$, and $f(n-(1,0))=1$, then set $r(n)=r(n-(0,1))$. In this case the left neighbor $f(n-(0,1))$ belongs to the same object as $f(n)$. (Figure 3.9, c)

6. If $f(n) = 1$, and $f(n-(1,0)) = f(n-(0,1)) = 1$, then set $r(n) = r(n-(0,1))$. If $r(n-(0,1)) \neq r(n-(1,0))$, then record the labels $r(n-(0,1))$ and $r(n-(1,0))$ as equivalent. In this case both the left and upper neighbors belong to the same object as $f(n)$, although they may have been labeled differently. (Figure 3.9, d)

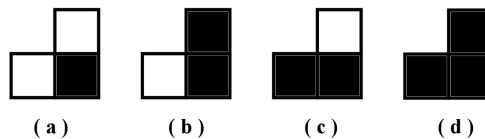


Figure 3.9 Neighbor Pixels

3.3 Smoothing Linear Filters

The output (response) of a smoothing, linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. These filters sometimes are called averaging filters. [52]

The smoothing filter replaces each pixel value of the image with the average of its neighborhood, which is defined by the mask of the filter, gray levels. As a result the whole image is smoothed which means the sharp edges within the images are reduced as well as the random noise contained within the image. Averaging filters are generally used for reduction of the detailed pixel regions that are smaller than the filter mask in an image.

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (3.12)$$

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad (3.13)$$

If all coefficients are equal in an averaging filter (like in 3.9) it is called a box filter. Otherwise, if the coefficients within the mask are different this filter is called weighted average filter. In a weighted average filter, some pixels have greater importance than the others as in 3.10 the pixel at the center of the mask

is multiplied by a higher value than any other, thus this pixel have more importance in the calculation of the average.

3.4 Morphological Operations

Morphological operations consist of a collection of techniques for digital image processing based on mathematical morphology. Some basic operations are: erosion, dilation, opening, closing, shrinking, thinning, thickening, skeletonization, pruning, distance transform. These operations are generally applied on binary or grayscale images. [53]. Most morphological functions operate on 3 x 3 pixel neighborhoods but the size of the mask can be changed. The pixel of interest is located at the center of the mask. In the study, opening and closing morphological operations are used.

3.4.1 Binary Erosion and Dilation

Binary morphological operators such as erosion and dilation combine a local neighborhood of pixels with a pixel mask to achieve the result. 3.14 shows this relationship. The output pixel, 0, is set to either a (1) or a (0) based on the logical AND relationship. Binary erosion uses the following for its mask:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (3.14)$$

This means that every pixel in the neighborhood must be 1 for the output pixel to be 1. Otherwise, the pixel will become 0. No matter what value the neighboring pixels have, if the central pixel is 0 the output pixel is 0. Just a single 0 pixel anywhere within the neighborhood will cause the output pixel to become 0. Erosion can be used to eliminate unwanted white noise pixels from an otherwise black area. The only condition in which a white pixel will remain white in the output image is if all of its neighbors are white. The effect on a binary image is to diminish, or erode, the edges of a white area of pixels.

Dilation is the opposite of erosion. Its mask is:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (3.15)$$

This mask will make white areas grow, or dilate. The same rules that applied to erosion conditions apply to dilation, but the logic is inverted - use the NAND rather than the AND logical operation. Being the opposite of erosion, dilation will allow a black pixel to remain black only if all of its neighbors are black. This operator is useful for removing isolated black pixels from an image. [52]

We can combine dilation and erosion to build two important higher order operations:

$$\text{Opening :} \quad \mathcal{O}_{(A,B)} = A \circ B = D(E_{(A,B)} , B) \quad (3.16)$$

$$\text{Closing :} \quad \mathcal{C}_{(A,B)} = A \bullet B = E(D_{(A,-B)} , -B) \quad (3.17)$$

Opening operation can separate objects that are connected in an image. The closing operation can fill in small holes. Given a “smooth” structuring element the opening and closing operations performs a smoothing effect. The difference between the smoothing effects is the direction of smoothing, that is, opening smoothes from the inside of the object contour while closing smoothes from the outside of the object contour.

3.5 SEGMENTATION

Some image processing applications like Hand Gesture Recognition do not require every pixel in the image to be processed during all stages of the application. Segmentation subdivides an image into its constituent regions or objects [52]. Therefore large proportion of redundant information can be reduced by isolating areas of interest by using segmentation.

In some hand gesture recognition systems the operation environments are constrained so that gesturing limbs have simple and robust isolation. Choosing a homogeneous single colored background is a common approach exemplified by Cipola and Hollinghurst [56]. Other approaches consider static real backgrounds, or dynamically changing backgrounds, like in the studies of Lee [44], and Mittal and Paragios [16] . Due to their nature these approaches are more complicated but they allow practical usage and easy installation.

The segmentation algorithms are specifically designed for the final feature space required. Generally the output of the segmentation stage is a binary image representing the areas of interest.

3.5.1 Difference-Based Segmentation

The key point of difference-based segmentation techniques is separating the areas of interest that is the changing parts of the input images from the background. A typical system configuration for difference-based segmentation consists of a fixed camera, a target object / objects and an environment independent from the object / objects. Such a configuration leads to perceive the static parts of the image as background whereas the moving object / objects as foreground without considering the physical positions of the object / objects and other fixed elements in the field of view.

3.5.2 Background Subtraction

The basic principle of background subtraction procedure is subtracting the current frame from the pre-stored static environment background image to find the difference regions.

Pixel by pixel subtraction is done with the two images and the magnitude of the difference is compared to a threshold.

The foreground image can be described with 3.15 in which C is the current frame, B is the background image and T is the threshold determined before the current process.

$$F(x,y) = \begin{cases} 1 & \text{if } |C(x,y) - B(x,y)| > T \\ 0 & \text{otherwise} \end{cases} \quad (3.18)$$

The background subtraction procedure is demonstrated in the Figure 3.10. Figure 3.10 (a) shows the background image. The purple star in Figure 3.10 (b) represents the foreground, that is the object enters the scene later. And Figure 3.10 (c) shows the result of the background subtraction.



(a)



(b)



(c)

Figure 3.10 (a,b,c) Background Subtraction Steps

Although it is easy to implement and computationally efficient there are some limits and disadvantages of background subtraction. It is expected that the environment will be static. The background image must be prepared before the process. If a proper color model is not chosen the results will not be good because of the lack of robustness to the image illumination.

3.5.3 Color-Based Segmentation

Color-based segmentation uses the color as a cue for separating the areas of interest from the rest of the image. Basically each pixel in the current frame is compared with a predefined threshold value or LUT (Lookup Table) so that the target regions can be easily segmented. Therefore this segmentation technique is very efficient; however as a trade-off it places rigid constraints on the colors of areas of interest and backgrounds. Varying illumination conditions also affect complete color representations, so the color space used in the technique should be carefully chosen to reduce the effect of illumination.

As an example of color-based segmentation, by masking the non-purple colors in Figure 3.10 (b) will give the output pixels in Figure 3.10 (c) indicated as purple.

3.5.4 Marker-Based Segmentation

In order to make the color-based segmentation process more accurate and efficient easily identified markers are used to locate relevant image areas. A black glove with white fingertip markers is used by Davis and Shah [22] to reduce their segmentation process to a simple binary thresholding operation.

Like Davis and Shah [47], Yachida and Iwai [20] also use gloves but the gloves they use are multi-colored, in other words different regions within hands

have different colors. By this way each hand can be easily segmented and the colors provides cues for the pose of the hands.

Maggioni and Kammerer [21] have a different approach for marker-based segmentation. They use a glove with marks on it, too. However, they use geometric features of the markers. The gloves have black and white eccentric circles on them. Geometric properties of the markers provide cues for the feature extraction and segmentation is again a simple binary subtraction.

3.6 TRACKING

3.6.1 Meanshift

The meanshift algorithm deals with the relation of the computed color distribution with the predefined target color distribution. The algorithm tries to find the nearest location where the difference between these two values is lower than the predefined threshold value, that is, the local density maximum.

Basically there are three steps in the meanshift algorithm [57] :

1. Choose an appropriate size for the search window.
2. Locate the search window to the predefined initial coordinates.
3. Loop until the search window center converges. There are two sub steps in the loop:

3.1) Calculate the mean location within the boundaries of the search window.

3.2) Locate the center of the search window at the mean location.

For step 3.1, the mean location within the boundaries of the search window is calculated as follows [58]:

Find the zeroth moment:

$$M_{00} = \sum_x \sum_y I_{(x,y)} \quad (3.19)$$

Find the first moment for x and y:

$$M_{10} = \sum_x \sum_y x I_{(x,y)} \quad (3.20)$$

$$M_{01} = \sum_x \sum_y y I_{(x,y)} \quad (3.21)$$

Finally, calculate the search window location:

$$x_c = \frac{M_{10}}{M_{00}} \quad (3.22)$$

$$y_c = \frac{M_{01}}{M_{00}} \quad (3.23)$$

CHAPTER 4

PROPOSED METHOD

4.1 APPLICATION ENVIRONMENT

The developed system can be used both as a human-computer interaction device for a PC to operate in 3D environment and controller of a robot.

Any pc-based webcam or a camera recorder can be used as a capture device for the system. The constraints about the camera come from the position and viewpoint of the camera. The camera should be placed in front of the user and perpendicular to the ground. The distance between the user and the capture device depends on the focal length of the camera. So, the minimum and the maximum distance from the camera to the user varies with the type of the camera and the camera should be placed to the location where the view of the camera covers the top of the head and the hands of the user when his/her arms and hands are wide open. And there should be nobody near the person whose gestures are considered as input to the system.

The environment has also some restrictions. The background should not contain skin-like colors and should not be relatively complex.

4.2 HAND FEATURES

Two features of the hand, posture and position of the hands, are used in the system. The “boundary rectangle” structure used for finding posture and position of the hands are described at the following section.

4.2.1 Boundary Rectangles

When the hands are segmented, an imaginary rectangular region is drawn outside of each hand. The boundaries are simply calculated by finding the positions of the minimum and maximum skin pixels in both vertical and horizontal. Figure 4.1 illustrates a sample boundary rectangle with a green mark.

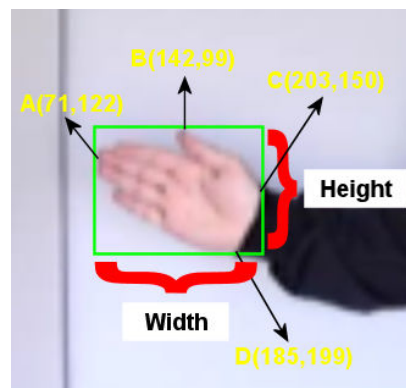


Figure 4.1 Boundary Rectangle

If the origin is at the top left corner of the image:

The leftmost pixel that contains a skin color, which is point A in Figure 4.1, determines the horizontal start point component of the boundary rectangle. In our case it is 71.

The uppermost pixel that contains the skin color, which is point B in Figure 4.1, determines the vertical start point component of the boundary rectangle. In our case it is 99.

The rightmost pixel that contains the skin color, which is point C in Figure 4.1, determines the horizontal finish point component of the boundary rectangle. In our case it is 203.

The pixel at the bottom that contains the skin color, which is point D in Figure 4.1, determines the vertical finish point component of the boundary rectangle. In our case it is 199.

As a result of our case, the starting and finishing points of the boundary rectangle is determined as:

$$\text{Start}(x,y) = (A(x), B(y)) = (71 , 99) \quad (4.1)$$

$$\text{Finish}(x,y) = (C(x), D(y)) = (203 , 199) \quad (4.2)$$

where x and y are the pixel coordinates of the image.

The width of the rectangle is calculated as:

$$\text{width} = \text{Finish}(x) - \text{Start}(x) \quad (4.3)$$

The height of the rectangle is calculated as:

$$\text{height} = \text{Finish}(y) - \text{Start}(y) \quad (4.4)$$

Width and height are illustrated in Figure 4.1. The values of width and height according to the sample values are calculated as follows:

$$\text{width} = \text{Finish}(x) - \text{Start}(x) = 203 - 71 = 132 \quad (4.5)$$

$$\text{height} = \text{Finish}(y) - \text{Start}(y) = 199 - 99 = 100 \quad (4.6)$$

4.2.2 Hand Postures

For the ease of learning system usage and performance purposes two general groups of postures are used in the system.

The longest edge of the boundary rectangle of each hand determines the type of the corresponding posture. If the longest edge is the height as shown in the Figure 4.2, this posture is called a “vertical posture”. Similarly, if the longest

edge is the width, as shown in the Figure 4.3, the posture is called a “horizontal posture”.



Figure 4.2 Vertical Posture

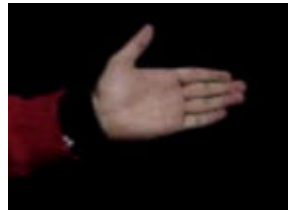


Figure 4.3 Horizontal Posture

The type of the hand shape can be formulized as the following:

$$type = \begin{cases} \text{if } width \geq height & , \text{ horizontal} \\ \text{else} & , \text{ vertical} \end{cases} \quad (4.7)$$

4.2.3 Hand Positions

In the system, the center point of a boundary hand rectangle is accepted as the position of the corresponding hand.



Figure 4.4 : Center point of the rectangle

In Figure 4.5, some positions of each hand are demonstrated and it can be seen that they are relative to the position of elbows those nearly have the same positions as the postures at Figure 4.6. Figure 4.5 is so constructed that many arm regions from different frames collected together without changing the original position of arms.



Figure 4.5 Circle-like structure



Figure 4.6 Elbow reference

When the position of an elbow / the posture at Figure 4.6 is considered as a static reference point, all possible positions of hands draw circular structures around the reference as illustrated in Figure 4.7.

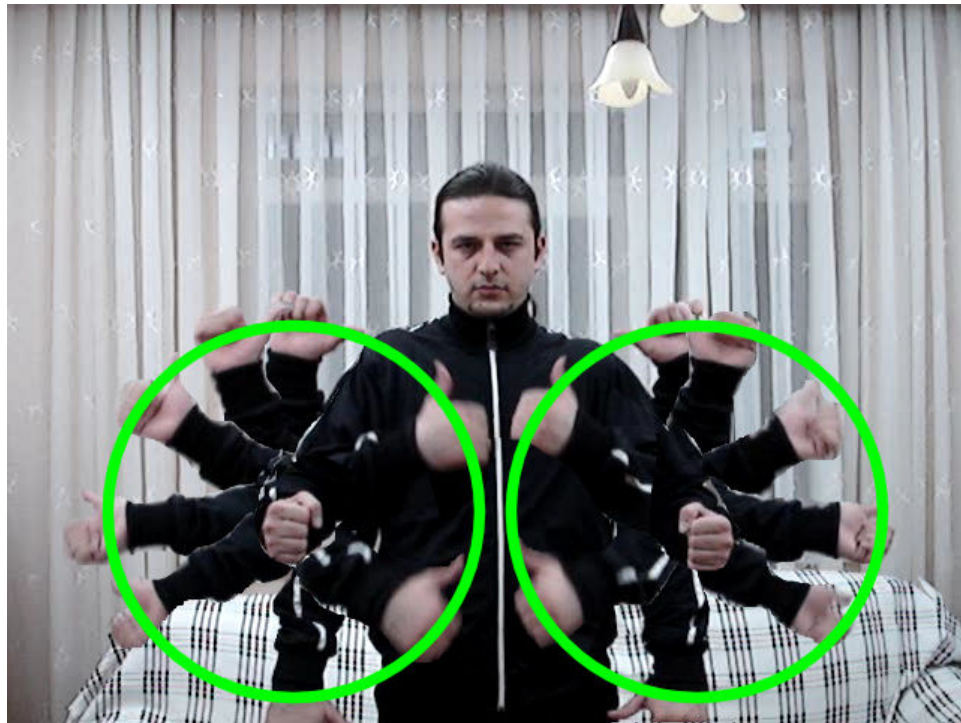


Figure 4.7 Circle-like structure (marked with green)

By using this information we propose a hand gesture system that focuses on the specific positions of hands relative to the elbow position. We are interested in the gestures staying near the regions marked with red circles in Figure 4.8.

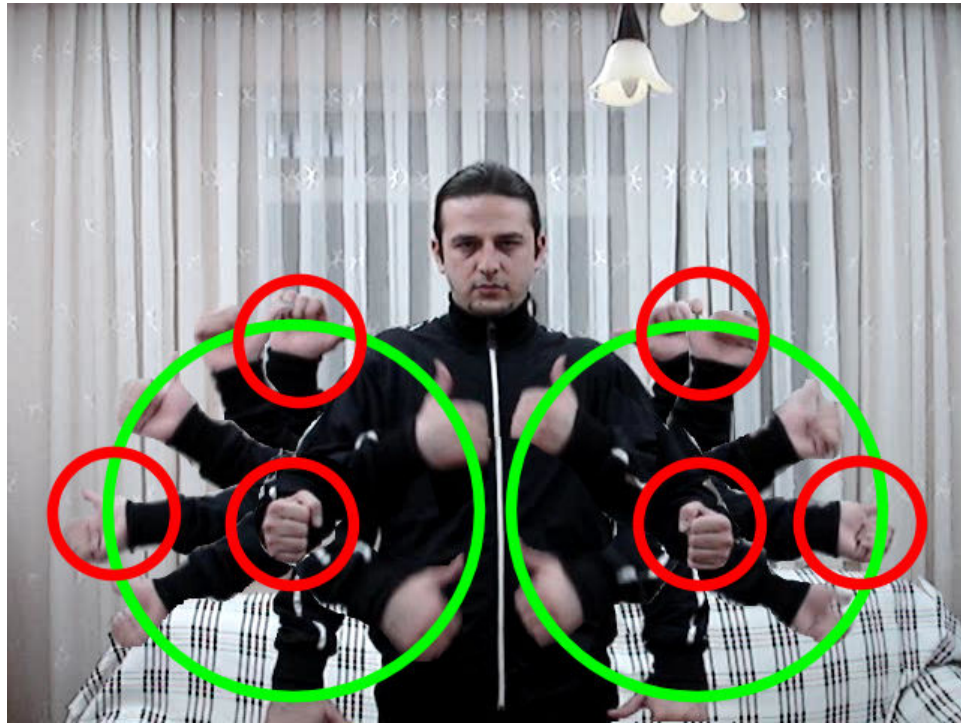


Figure 4.8 : The regions considered in the proposed system

4.3 SYSTEM DESIGN

The system requirements have been established by taking consideration of practical usability and computer vision constraints. General algorithm of the system consists of three parts:

- Background Modeling
- Initialization
- Main Loop

4.3.1 Image Acquisition

The common component among all the three steps in the algorithm is image acquisition.

Image acquisition process is done by consumer-level video input hardware, as it is mentioned in Section 4.1 and operating system support by these devices via OpenCV library. CCD type camera is used within the work.

4.3.2 Background Modeling

The first step for the system to work properly is the background modeling. This model will be used as a reference to subtract the captured frames during the hand gesture recognition process. The general algorithm is shown in the following figure.

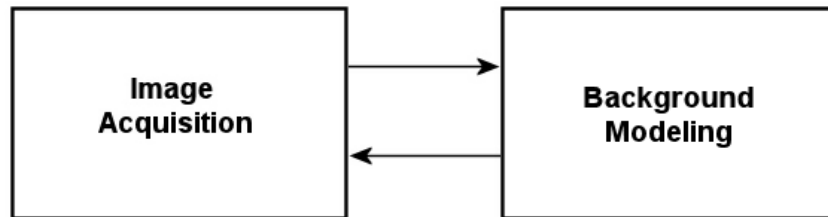


Figure 4.9 Background Modeling

The background is modeled by computing the mean and standard deviation of each pixel from a sequence of images taken while nobody exists in the scene.

The background model works on HSV color model. Therefore, the mean of the squares sum and the square of the mean pixel values of these images are calculated after converting them into HSV color format.

The standard deviation is calculated by using these values as the following:

$$\alpha_{(x,y)} = \sqrt{\frac{\mathit{Squares}_{(x,y)}}{N} - \left(\frac{\mathit{Sums}_{(x,y)}}{N}\right)^2} \quad (4.8)$$

The background model consisting of the mean pixel values of the frames captured during the background modeling step is considered as a reference image. The background model creation is illustrated in the following figure.

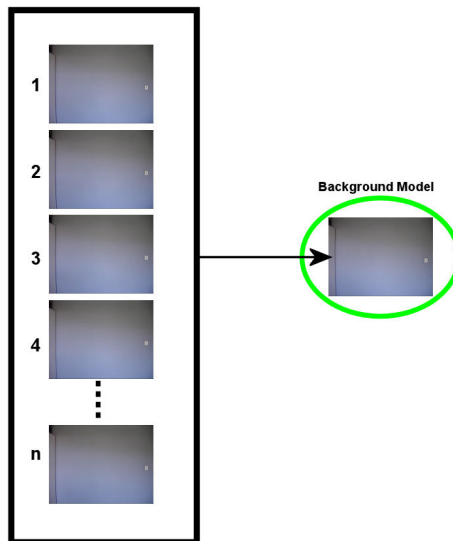


Figure 4.10 Background Illustration

4.3.3 Initialization

After background modeling, the performer should make a predefined special gesture. This gesture is used for initializing parameters for the trackers. Figure 4.11 shows the initializing gesture.



Figure 4.11: Special Gesture

Initialization process consists of three steps:

- Image Acquisition
- Segmentation
- Calculating Initial Parameters

After a successful image acquisition, segmentation process filters out the pixels those are not belonging to the head and hands in the current image. This process is based on skin color and background color information. Segmented

hand and head regions are used for calculating initial parameters for the system.

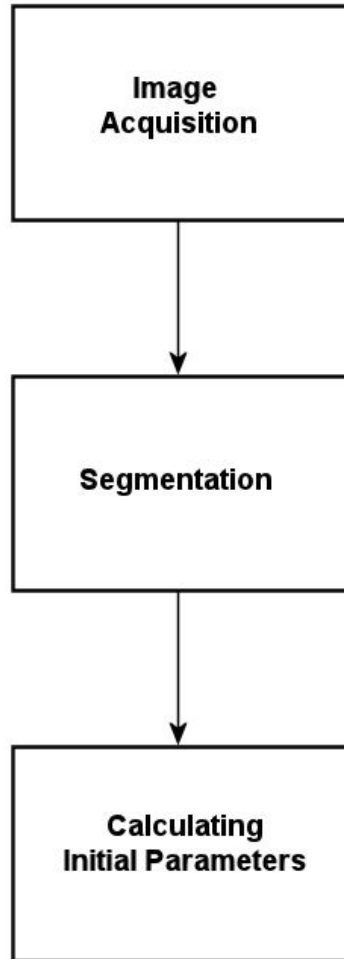


Figure 4.12 Initialization

4.3.3.1 Segmentation

Segmentation process consists of three parts:

- Background Subtraction
- Skin Filtering
- Noise Removal

4.3.3.1.1 Background Subtraction

Once an initial background model is created, the current frame captured by the device is subtracted from the reference image which is created at the background modeling process.

$$\forall x, \forall y \quad D_t(x, y) = \begin{cases} 1 & \text{for } I_{t-1}(x, y) - I(x, y) \geq \text{threshold} \\ 0 & \text{for } I_{t-1}(x, y) - I(x, y) < \text{threshold} \end{cases} \quad (4.9)$$

The threshold value used in the image subtraction is the standard deviation calculated at the background modeling process multiplied by an experimentally found constant.

The ideal output of the background subtraction process is pixels which belong to the objects those are entered the view of the camera lately. This kind of special output contains pixels having value of “0” which is black that represents the background and other pixels those belong to an object which is in front of the background, in our case the person whose hand gestures are

captured. Some noise maybe observed in the resulting image because of the lighting and focusing problems.

4.3.3.1.2 Skin Color Filtering

In skin filtering process, the pixels those do not have a skin color are eliminated. Thus, the resulting filtered image contains the skin pixels. Prior to skin filtering, a skin color model should be created to use in the skin color filtering.

4.3.3.1.3 Skin Color Modeling

Hue and Saturation components of the HSV color model is used for skin color modeling since these components are not affected by the change in illumination so much. Özbay [59] has pointed out the success of HSV color model in terms of performance and false detection rate in her study which is about face detection using skin color segmentation. The skin color model is put into Gaussian model by $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ in which $\boldsymbol{\mu}$ is the mean and $\boldsymbol{\Sigma}$ is the covariance. The response of the model which is also called as skin likelihood function is like stated below :

$$\text{Response} = \exp\left[\frac{-1}{2} \cdot (c - \boldsymbol{\mu})^T \cdot \boldsymbol{\Sigma}^{-1} \cdot (c - \boldsymbol{\mu})\right] \quad (4.10)$$

Here, c represents the color, consisting of hue and saturation components, to be examined. μ is a matrix in which there are Hue and Saturation value arithmetic means of the training skin pixels. This matrix is defined as below :

$$\mu = \left[\mu_{HUE} \quad , \quad \mu_{SAT} \right] \quad (4.11)$$

In the response function, Σ represents the covariance matrix, which is calculated as:

$$\Sigma = \begin{bmatrix} \sigma_{HUE}^2 & \frac{\sigma_{HUE} \cdot \sigma_{SAT}}{\rho} \\ \frac{\sigma_{HUE} \cdot \sigma_{SAT}}{\rho} & \sigma_{SAT}^2 \end{bmatrix} \quad (4.12)$$

ρ is the correlation of hue and saturation values of the training skin pixels and calculated as:

$$\rho = \frac{\sum_{i=1}^n (HUE_{(i)} - \mu_{HUE}) \cdot (SAT_{(i)} - \mu_{SAT})}{\sqrt{\sum_{i=1}^n (HUE_{(i)} - \mu_{HUE})^2} \cdot \sqrt{\sum_{i=1}^n (SAT_{(i)} - \mu_{SAT})^2}} \quad (4.13)$$

σ : Variances of hue and saturation components

$$\sigma_{HUE} = \frac{\sum_{i=1}^n (HUE_{(i)} - \mu_{HUE})}{n-1} \quad (4.14)$$

$$\sigma_{SAT} = \frac{\sum_{i=1}^n (SAT_{(i)} - \mu_{SAT})}{n-1} \quad (4.15)$$

We have classified the human skin model into three groups from white to darker : white, medium, dark. In order to create the model 33 images are collected for each group, and then skin parts in these images are manually segmented by marking the nonskin parts with black pixel as shown in the Figure 4.13 manual skin segmentation after the noise in the images is filtered out by low-pass filter.

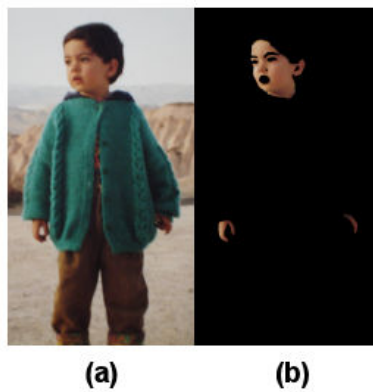


Figure 4.13 Manual Skin Segmentation

So, there will be three responses from each model. When examining a pixel whether it is a skin color or not, the three responses are taken for the examined pixel value. If any of the responses is greater than the predefined threshold (which is experimentally found as 0.6), the examined pixel will be considered as skin pixel. If none of the three responses is above the threshold value, that examined pixel will be considered as nonskin pixel.

4.3.3.1.4 Skin Color Segmentation

By using the skin color model created before the application, each pixel value of the background filtered image is examined by the color model and it is determined whether the pixel is a skin color or not. The skin color segmentation input is Figure 4.8 and the ideal output is shown in Figure 4.9.

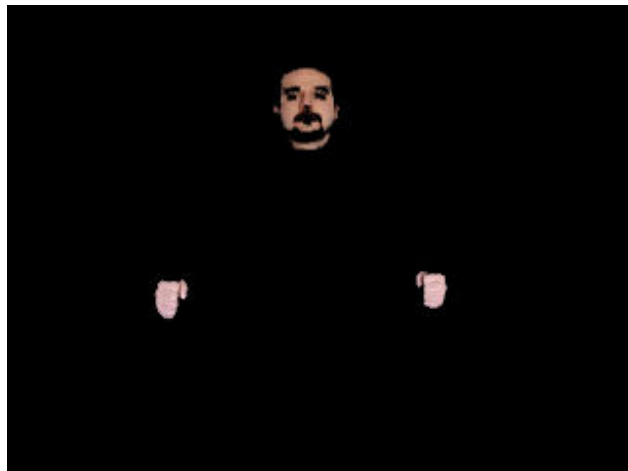


Figure 4.14 : Ideal output for skin color segmentation

4.3.3.1.5 Noise Removal

It may not always be possible to perfectly segment a given image. Because of some problems due to the camera lens, lighting and skin-like colors contained in the background or foreground object (that is the person in front of the camera in our case) sometimes undesired noise pixels may appear at the segmentation results. The purpose of this process is filtering out the undesired noise pixels. In order to filter out the noise pixels, opening and closing morphological operations described at Section 3.4 are applied to the image.

Apart from the opening and closing morphological operations, a median smoothing filter is applied to the image to prepare the image for the following processes.

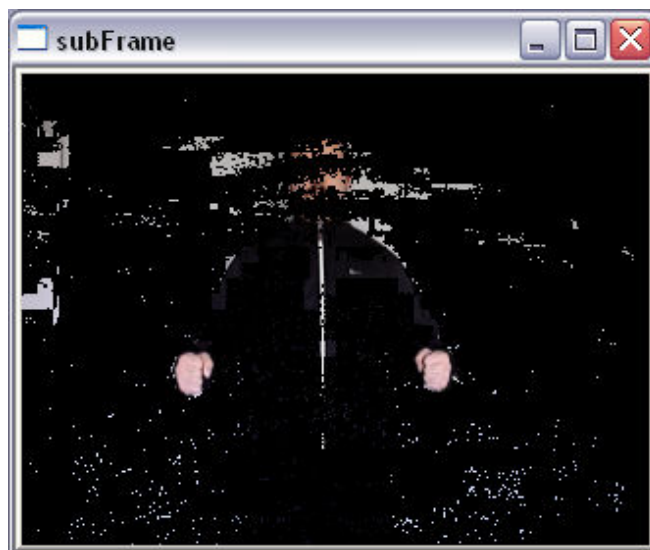


Figure 4.15 Background Segmented Image



Figure 4.16 Skin Segmented Image

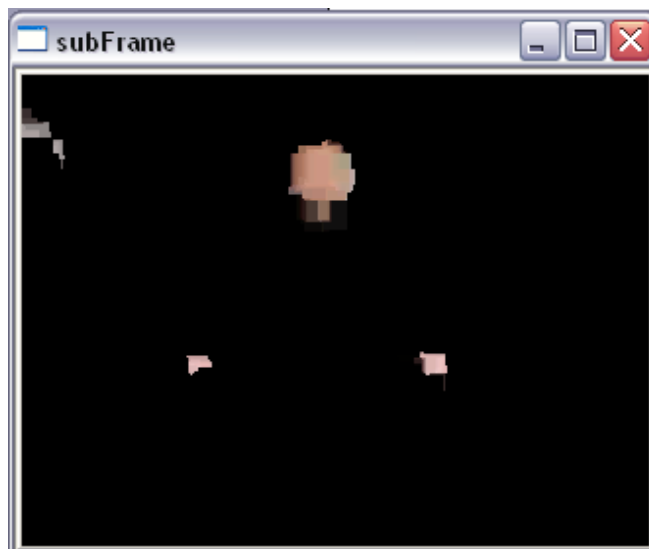


Figure 4.17 Morphological Operations

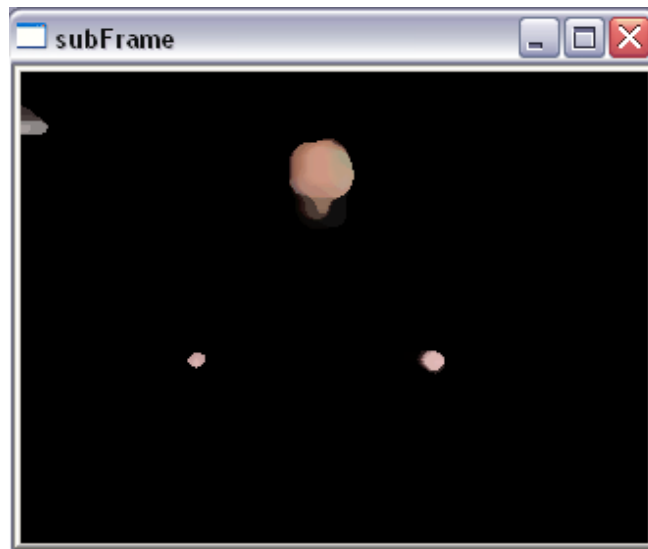


Figure 4.18 Smoothing Filter

4.3.3.2 Calculating Initial Parameters

This process simply consists of locating the head and two hands and drawing boundary rectangles around these parts. Once a skin pixel is found region labeling algorithm described in the section 3.2 is used to connect the pixel to the neighbor skin pixels so that the skin pixels are grouped together. The head is assumed to be near the center of the image and the hands are expected to be below the head. Following figure is a sample result for this process.

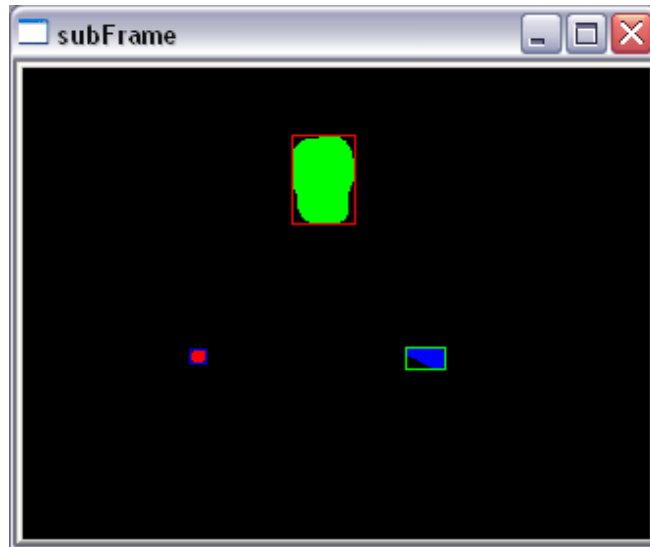


Figure 4.19 Initialization Output

The parameters those are output of this process and also the entire initialization step are the center points of the head's and both hands' boundary rectangles and also the height and width of the head boundary rectangle.

4.3.4 Main Loop

The basic system model specific to our work is given in Figure 4.7.

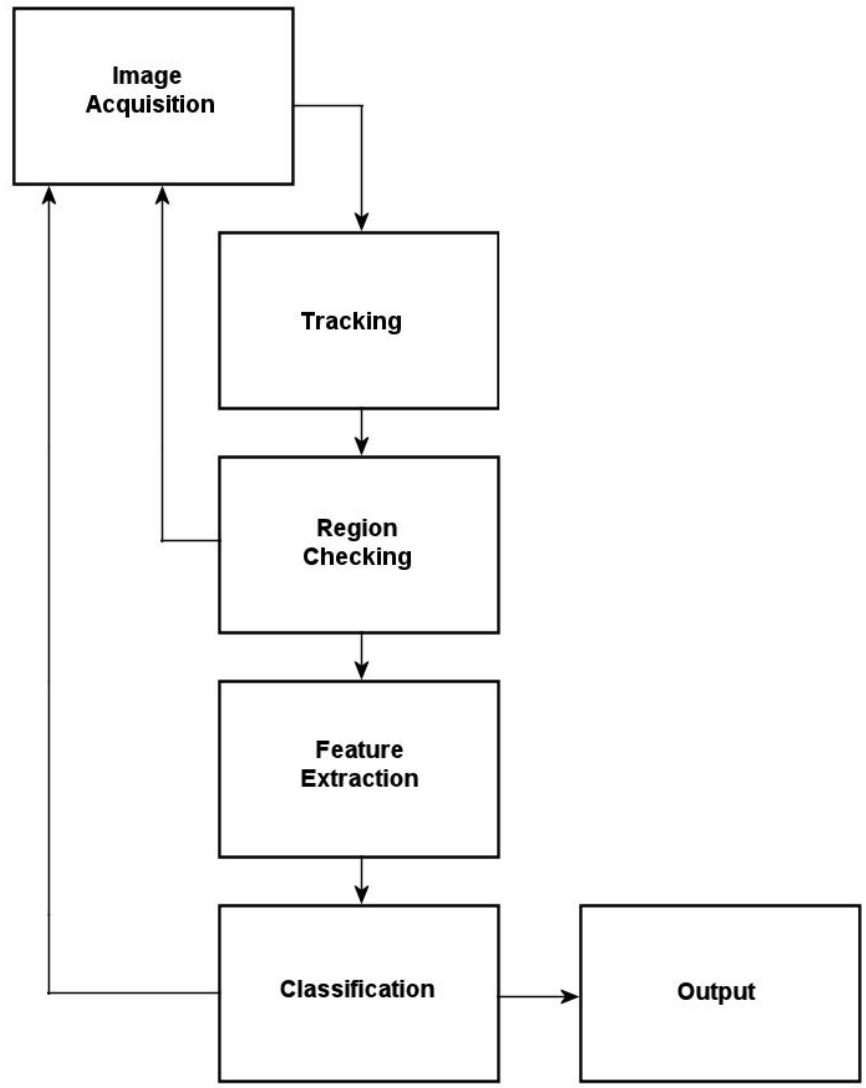


Figure 4.20 Main Loop

4.3.4.1 Tracking

Tracking process is basically a meanshift algorithm application. The initial parameters coming from the section 4.3.3 are used for determining the initial locations of the meanshift search windows. The size of the initial tracking windows of the hand regions is the same as the boundary rectangle of the head. The head boundary rectangle is also used for the meanshift tracker window without change.

Inputs coming from the initialization step are:

- HBRC (Head Boundary Rectangle Center): Location of the center of head boundary rectangle.
- HH (Head Height): Height of the head boundary rectangle.
- HW (Head Width): Width of the head boundary rectangle.
- LHBRC (Left Hand Boundary Rectangle Center): Location of the center of left hand boundary rectangle.
- RHBRC (Right Hand Boundary Rectangle Center): Location of the center of right hand boundary rectangle.

These input parameters are used in tracking step as follows:

$$\text{HTWL (Head Tracking Window Location)} = \text{HBRC} \quad (4.16)$$

$$\text{HTWH (Head Tracking Window Height)} = \text{HH} \quad (4.17)$$

$$\text{HTWW (Head Tracking Window Width)} = \text{HW} \quad (4.18)$$

$$\text{LHTWL (Left Hand Tracking Window Location)} = \text{LHBRC} \quad (4.19)$$

$$\text{LHTWH (Left Hand Tracking Window Height)} = \text{HH} \quad (4.20)$$

$$\text{LHTWW (Left Hand Tracking Window Width)} = \text{HW} \quad (4.21)$$

$$\text{RHTWL (Right Hand Tracking Window Location)} = \text{RHBRC} \quad (4.22)$$

$$\text{RHTWH (Right Hand Tracking Window Height)} = \text{HH} \quad (4.23)$$

$$\text{RHTWW (Right Hand Tracking Window Width)} = \text{HW} \quad (4.24)$$

The segmentation step of the initialization process is applied here to have clear results from the meanshift tracking algorithm.

Some frames from application in which the tracking windows demonstrated with green rectangles are shown in the Figure 4.21.

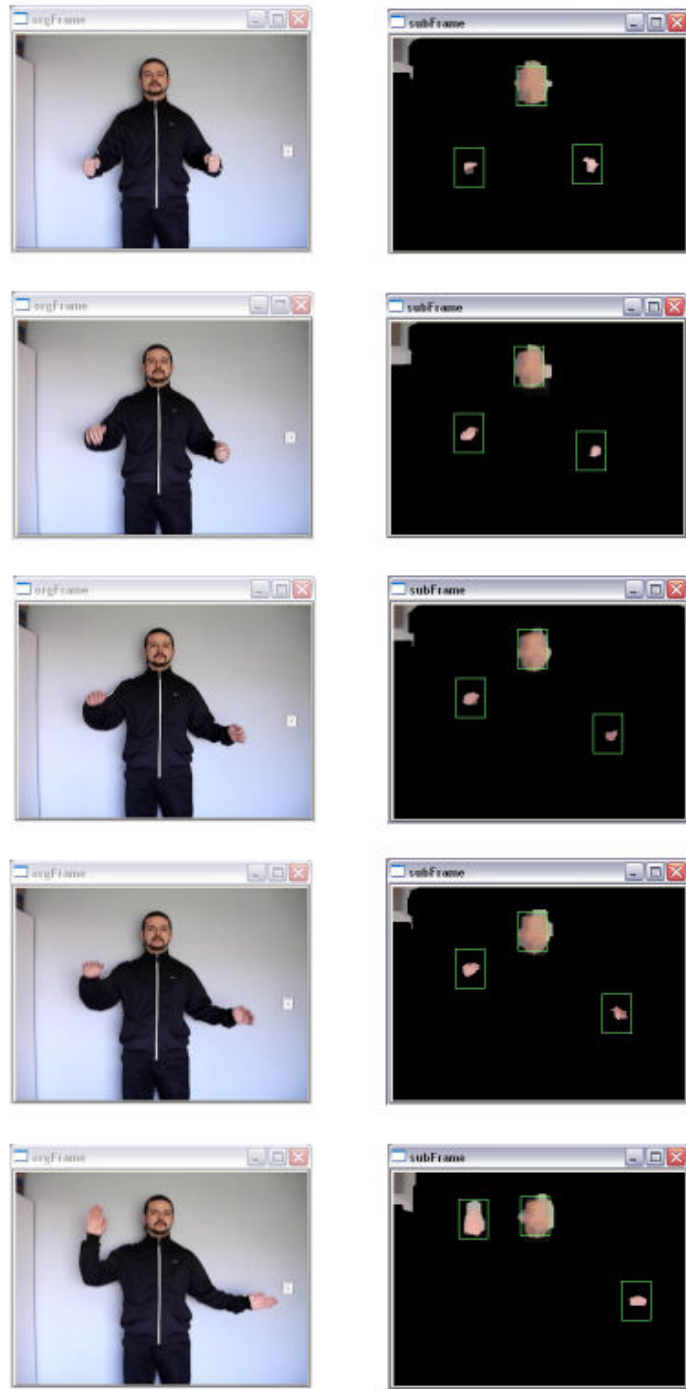


Figure 4.21 Tracking Windows

4.3.4.2 Region Checking

In this step the center of the tracking windows are checked whether they are in the predefined hand gesture regions. If any of the hands' center of tracking windows are within any hand gesture regions, the region is reported to the Feature Extraction step, otherwise the system turns back to the first step of the main loop.

The hand gesture region locations and sizes are calculated with the outputs of the initialization step. The shape of the hand gesture regions is square and the edges of these squares have the length of face rectangular region height. For practical purposes, these regions are named as in Figure 4.22.

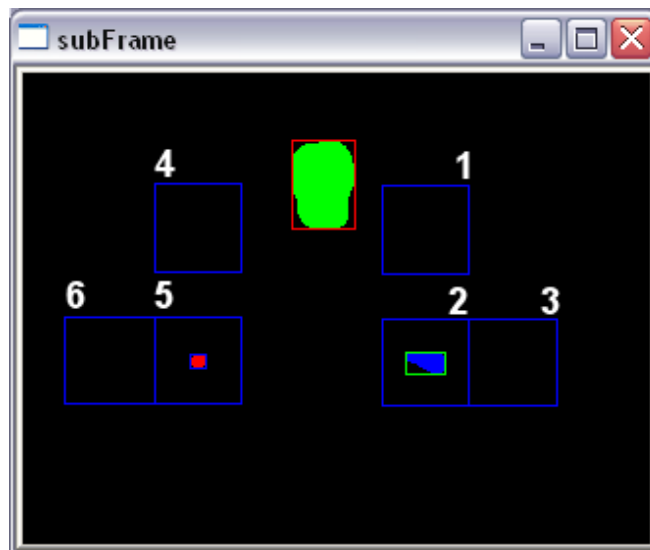


Figure 4.22 Hand Gesture Region Locations

- The center position of the hand gesture region 2 is same as the center of the left hand boundary rectangular.
- Hand gesture region 3 center position is one head boundary rectangle height long away from the hand gesture region 2 to the right.
- Hand gesture region 1 center position is one and half head boundary rectangle height long away from the hand gesture region 2 to the up.
- The center position of the hand gesture region 5 is same as the center of the right hand boundary rectangular.
- Hand gesture region 6 center position is one head boundary rectangle height long away from the hand gesture region 5 to the left.
- Hand gesture region 4 center position is one and half head boundary rectangle height long away from the hand gesture region 5 to the up.

4.3.4.3 Feature Extraction

As it is indicated in section 4.2, there are two types of features used in the system; hand positions and hand postures. In section 4.2.3 the geometric possibilities of hand positions in terms of the elbow positions are mentioned. For determining hand posture type a boundary rectangle for the related hand is drawn and the width and height properties of this rectangle are used as described in the hand postures section 4.2.2.

4.3.4.4 Classification

A rule-based approach, as mentioned in section 2.3.1 is used in the system. The combinations of position and posture data of two hands give the value of the current state. Table 4.1 lists the state values.

Table 4.1 State Values

STATE	HEAD	LEFT HAND		RIGHT HAND	
	Exists	Position	Posture	Position	Posture
S0	0	x	X	x	x
S1	1	1	V	4	V
S2	1	1	V	4	H
S3	1	1	V	5	V
S4	1	1	V	5	H
S5	1	1	V	6	V
S6	1	1	V	6	H
S7	1	1	H	4	V
S8	1	1	H	4	H
S9	1	1	H	5	V
S10	1	1	H	5	H
S11	1	1	H	6	V
S12	1	1	H	6	H
S13	1	2	V	4	V
S14	1	2	V	4	H
S15	1	2	V	5	V
S16	1	2	V	5	H
S17	1	2	V	6	V
S18	1	2	V	6	H
S19	1	2	H	4	V
S20	1	2	H	4	H
S21	1	2	H	5	V
S22	1	2	H	5	H
S23	1	2	H	6	V
S24	1	2	H	6	H
S25	1	3	V	4	V
S26	1	3	V	4	H
S27	1	3	V	5	V
S28	1	3	V	5	H
S29	1	3	V	6	V
S30	1	3	V	6	H

S31	1	3	H	4	V
S32	1	3	H	4	H
S33	1	3	H	5	V
S34	1	3	H	5	H
S35	1	3	H	6	V
S36	1	3	H	6	H
S37	1	0	X	x	x
S38	1	x	X	0	X

State S1 to S36 represents all the hand gestures within the hand gesture regions defined previously. An action classification only occurs when the left hand is inside any of the regions 1,2,3 and at the same time the right hand is inside any of the regions 4,5,6 and head is found. Some of the other cases such as one hand found in a hand gesture region may be considered as an application related command such as “stop the application”, “start the application”, “undo the last operation”, etc...

Some sample states are illustrated in the following figure:

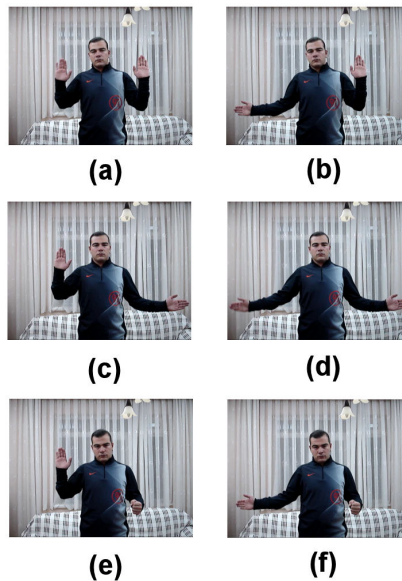


Figure 4.23 Sample States

- Figure 4.23 (a) represents State 1
- Figure 4.23 (b) represents State 6
- Figure 4.23 (c) represents State 31
- Figure 4.23 (d) represents State 36
- Figure 4.23 (e) represents State 13
- Figure 4.23 (f) represents State 18

There are three exception cases in Table 4.1. State “S0” indicates that the head is not properly found in the view, so general approach to this case is omitting the case or giving error.

Similarly states “S37” and “S38” indicates that at least one of the hands in the view has not found or an error occurred.

Each state value is processed till a predefined pattern is met or a special finishing state value is given by the user by the corresponding hand gesture.

4.4 THE OUTPUT

The output of the system is the gesture string consisting of the states described in the section 4.3.4.4. However, since the primary goal of the system is human-computer interaction and robot control, the output is observed from the application bound to the end of the system. If the application is specifically

designed so that the input gestures of the application is selected by considering the grouped structure of the system, the application will be so practical since learning of the system usage will be easy and the system will give accurately classified gestures due to the system design.

CHAPTER 5

EXPERIMENTAL RESULTS

The experimental results for our methods and the test environment are presented in this chapter.

Our system is designed to recognize some specific hand gestures by using consumer-level hardware and video devices. The system has been tested on AMD 3200+ processor as well as various Intel Pentium-IV based systems. The main input of the system is the system user's hand gestures captured via a webcam/camera. Based on the inputs, an action is provided by the system. For demonstration purposes a 3D shape within a 3D space is controlled by user's hand gestures via the system. In the terminology of vision-based hand recognition systems presented previously, our system is a low-level feature based system.

OpenCV computer vision library provided by Intel has been used for development of the system. The development environment of the system is

Microsoft Visual Studio .Net and the programming language used for coding the system is C++.

5.1 TEST APPLICATION

In order to present the ease of use and efficiency of our system, we implemented a 3D application. The application consists of controlling a teapot shown in a window and it is developed by using Microsoft Visual Studio .Net , C++ with OpenGL library.

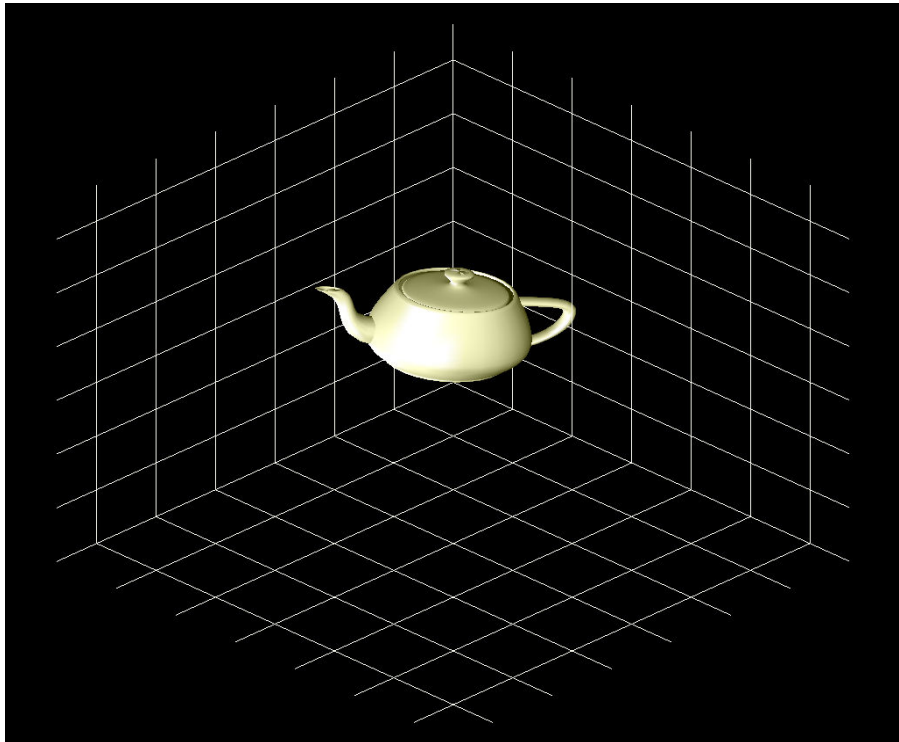


Figure 5.1 OpenGL Application

The teapot can be translated along any X, Y, Z axis in both directions or rotated around any of these axes by hand gestures of a user.

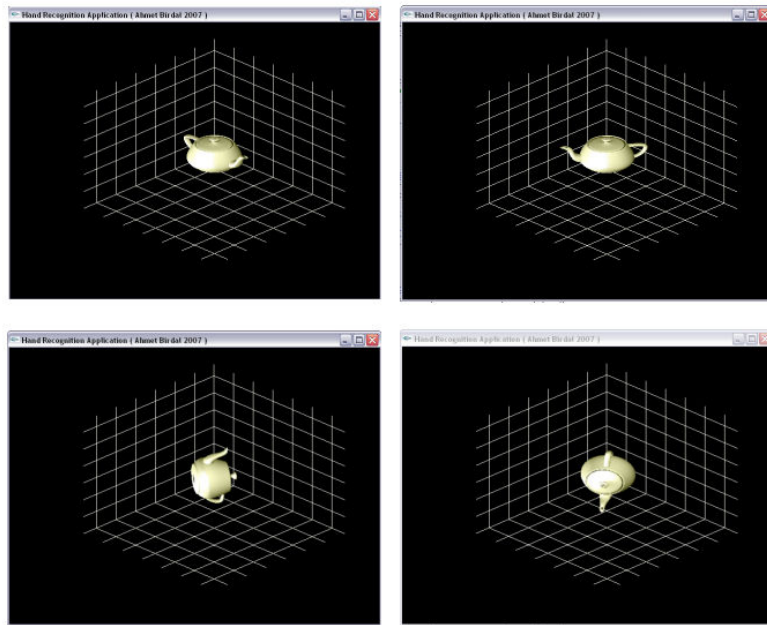


Figure 5.2 Rotation

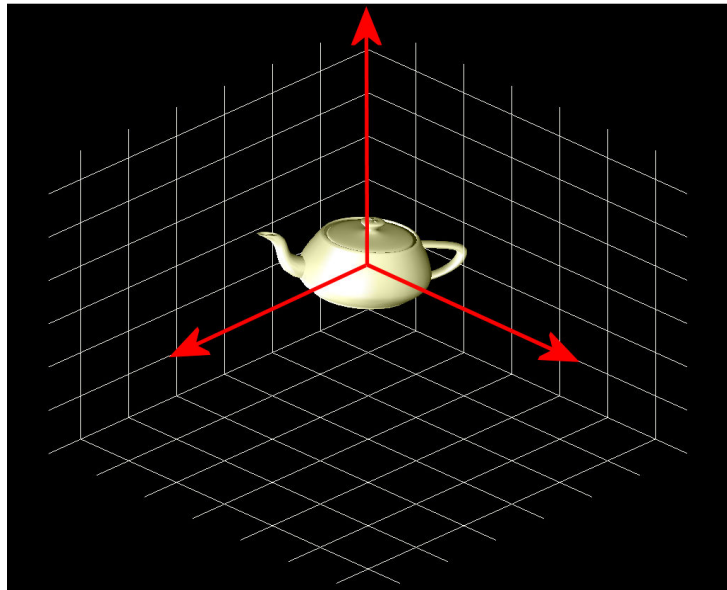


Figure 5.3 Translation

The actions taken in the application, corresponding to the hand gesture states, defined in Table 4.1 is shown in Table 5.1.

Table 5.1 The Actions Taken In The Application

STATE	ACTION	AXIS	DIRECTION
S0	Give Error		
S1	TRANSLATE	Y	+
S2	TRANSLATE	Y	+
S3			
S4			
S5	ROTATE	Y	+
S6	ROTATE	Y	+
S7	TRANSLATE	Y	-
S8	TRANSLATE	Y	-
S9			
S10			
S11	ROTATE	Y	-
S12	ROTATE	Y	-
S13	TRANSLATE	Z	+
S14	TRANSLATE	Z	+
S15			
S16			
S17	ROTATE	Z	+
S18	ROTATE	Z	+
S19	TRANSLATE	Z	-
S20	TRANSLATE	Z	-
S21			
S22			
S23	ROTATE	Z	-
S24	ROTATE	Z	-
S25	TRANSLATE	X	+
S26	TRANSLATE	X	+
S27			
S28			
S29	ROTATE	X	+
S30	ROTATE	X	+
S31	TRANSLATE	X	-
S32	TRANSLATE	X	-
S33			
S34			
S35	ROTATE	X	-
S36	ROTATE	X	-
S37	No Action		
S38	No Action		

The left hand of the user determines the axis and direction of the action. The position of the postures is so selected that the axis schema used in the system is similar to the natural axis schema as illustrated in the Figure 5.1.

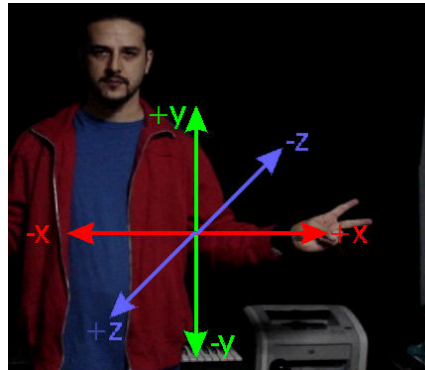


Figure 5.4 Axis

The posture of the left hand determines the direction for the action. The postures and the corresponding directions for the left hand in X axis are shown in the following figure. (The directions are given relative to the user. This means that when user points the +x direction, this will mean the system as $-x$ since the user is opposite to the camera)



Figure 5.5 Positive Direction

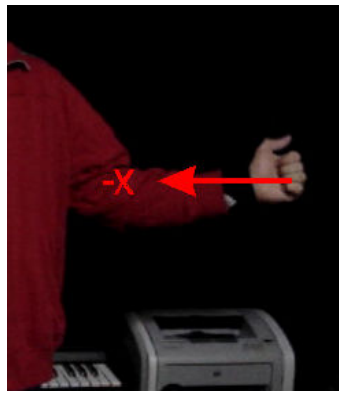


Figure 5.6 Negative Direction

The right hand determines the type of the action that will be taken. Either a translation or rotation action can be taken at a time.

To measure the success of the proposed method we randomly chose 6 frames per action listed in the Table 5.1, that is 72 frames in total. 63 of the gestures are recognized successfully. Therefore, the success of the system is calculated as %87,5 . The failure of the system to recognize the gesture is due to the unstable lighting conditions, performer's failure to move the hand to the proper

region and initialization problems due to the skin-like colors present at the background.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

The system here described, offers a solution to some basic problems of hand gesture recognition systems: transition between gestures, precision, unintended gesture prevention, learning the system usage, feedback.

Many of the studies like Haberdar [25], Liang et. al. [23], and Starner et. al. [24] proposed successful systems for recognition of sign languages. However, learning to use these sign languages is not so easy. Even the researchers may not know the entire sign language the system uses. Therefore, using these systems for human-computer interaction purposes does not seem practical in terms of learning of usage. Proposed system is designed to operate especially for three dimensional virtual space and due to the nature of the method it is very easy to learn the system in a short amount of time. Comparison of hand recognition system usage learning periods with different systems may be a future work for this system.

Generally traditional hand gesture recognition systems do not provide an effective solution to determine when to include a posture as a part of the gesture during the application. Waiting some short amount of time for a posture to stay freeze can be a simple solution to the problem. In contrast, proposed system uses such kind of waiting time periods as the part of the gesture. In other words, time is an element of hand gestures in the method. So, there are two main advantages of this method for transition problems. First, since there are only three main hand gesture locations for one hand and each region at most contains (for the sample system) only two postures there are only six gestures that one hand can perform. Therefore, precision is increased by decreasing the number of transition spaces. And secondly, as described above, each captured frame (not a sequence) that contains two postures within the predefined hand recognition regions describes an action to the system. So, the speed of the system usage is also increased.

The application for the proposed system is interactive. So, the user of the system will be aware of whether the system is recognized the input gesture correctly. If something is wrong with the output, user can undo the last operation or do the opposite action manually in order to undo the mistaken operation since inverting the last operation so easy that one transition for the correct hand will be enough to do it.

Although the system is expected to be satisfactory in an average environment the restrictions may be found as a handicap to use the system. For instance, the

gesture space can be extended to a greater number by increasing the number of rectangles used for determining the scan regions for the hands.

We have noticed that images taken under insufficient light have led to the incorrect results. Under the ordinary room lighting conditions the CCD camera output is found to be relatively unsuccessful. This observation is done in many studies like Malima et. al. [2]. The proposed algorithm successfully works when the background does not contain skin-like colors and the lighting is relatively stable.

The initial localization of the head in the first scene is found quite difficult. Some other segmentation algorithms or methods may be tested and integrated to the system to improve the initialization part.

During the work, an article titled “*Real Time Hand-Face Tracking using Skin Color and Motion*“ , was presented at the 9th World Multiconference on Systemics, Cybernetics and Informatics (2005).

REFERENCES

- [1] **LUDOVIC B.** et. al.(2004). *Face Tracking and Hand Gesture Recognition for Human Robot Interaction*, In “International Conference on Robotics and Automation”, New Orleans.
- [2] **MALIMA A.** et.al.(2006).*A Fast Algorithm for Vision-based Hand Gesture Recognition for Robot Control*, SIU06, Antalya, Turkey
- [3] **WU Y., HUANG, T.S.**, *Hand Modeling, Analysis, and Recognition for Vision-based Human Computer Interaction*, IEEE Signal Processing Magazine, 18(3).
- [4] **BOWDEN R.** et. al.(2004), *A Linguistic Feature Vector for the Visual Interpretation of Sign Language*. ECCV (1) 2004: 390-401.
- [5] **UTSUMI A.** et. al.(1999) *Multiple-Hand-Gesture Tracking using Multiple Cameras*, In Proc. of International Conference on Computer Vision and Pattern Recognition, pp.473–478.
- [6] **EMMOREY K.** (2002). *Language, Cognition, and the Brain: Insights from Sign Language Research*. Lawrence Erlbaum Associates
- [7] **DAVIS L.** et.al.(2000), *Tracking Humans from a Moving Platform*, 15th Conference on Pattern Recognition.
- [8] **M. A. HOANG, J. M. GEUSEBROEK**(2002). *Measurement of Color Texture*, In M. Chantler, Editor, Proceedings of the 2nd International Workshop on Texture Analysis and Synthesis (Texture 2002), pages 73-76. Heriot-Watt University.

- [9] **LONG N., LONG L.**(2000). *Computers*, 8th Edition Prentice Hall, New Jersey.
- [10] **HEISELE B.** et.al.(2000). *Face Detection in Still Gray Images*, A.I. Memo No. 1687, C.B.C.L. Paper No. 187 Center for Biological and Computational Learning, M.I.T., Cambridge MA.
- [11] **BLANC Z.** (2001), *CCD versus CMOS – Has CCD Imaging Come to an End?*, Photogrammetric, Week1, pp131-137, Zurich
- [12] **SHIMODA H.** et.al.(1998), *Development of Head-attached Interface Device (HIDE) and its Application Experiments*, IEEE International Conference.
- [13] **KOLSCH M., HOLLERER T.T.**(2004). *Vision-Based Interfaces for Mobility*, In Proc. MobiQuitous '04 (1st IEEE Intl. Conf. on Mobile and Ubiquitous Systems: Networking and Services), pages 86-94, Boston, MA.
- [14] **SORRENTINO A.** et. al.(1997). *Using Hidden Markov Models and Dynamic Size Functions for Gesture Recognition*, BMVC.
- [15] **MALKAWI AM and SRINIVASAN RS** (2005) *A New Paradigm for Human-Building Interaction: The Use of CFD and Augmented Reality Automation in Construction* .Journal 14 (1): 71-84.
- [16] **MITTAL A., PARAGIOS N.**(2004). *Motion-Based Background Subtraction using Adaptive Kernel Density Estimation*, Proc.CVPR, pp. 302-309.
- [17] **KAVAKLI M., JAYARATHNA D.** (2005). *Virtual Hand: An Interface for Interactive Sketching in Virtual Reality*, CIMCA/IAWTIC : 613-618
- [18] <http://www.5dt.com/products.html>, 1999

- [19] **MARSCHALL M.**, *Virtual Sculpture - Gesture-Controlled System for Artistic Expression*, ConGAS Symposium on Gesture Interfaces for Multimedia Systems, AISB2004, Leeds, UK.
- [20] **M. YACHIDA and Y. IWAI** (1998). *Looking at Gestures*, Ed. by R. Cipolla and A. Pentland, Cambridge University Press.
- [21] **MAGGIONI C., KAMMERER B.**,(1998). *Gesture Computer- History, Design and Applications*, Computer Vision for Human-Machine Interaction. Cambridge Univ. Press.
- [22] **DAVIS J. and SHAH M.** (994). *Visual Gesture Recognition*, Vision, Image and Signal Processing, 141(2):101–106.
- [23] **LIANG, R.H.** et.al. (1996), *A Sign Language Recognition System Using Hidden Markov Model and Context Sensitive Search*, HongKong.
- [24] **STARNER T.** et.al.(1997). *Wearable Computing Meets Ubiquitous Computing: Reaping the Best of Both Worlds*. ISWC 1999: 141-149
- [25] **HABERDAR H.**(2005). *Real Time Isolated Turkish Sign Language Recognition from Video Using Hidden Markov Models with Global Features*, MSc. Thesis., Istanbul: Computer Engineering, Yıldız Teknik University.
- [26] **GEJGUS P.** et.al.(2004). *Skin Color Segmentation Method Based on Mixture of Gaussians and Its Application in Learning System for Finger Alphabet*, Proceedings:CompSysTech'2004, ACM-acmbul&UIA, Rouse, 2004, pp.IIIA. 1-1 – IIIA.-1-6.
- [27] **EISENSTEIN J.,DAVIS R.**,(2003) *Natural Gesture in Descriptive Monologues*, Proc. ACM Symp. User Interface Software and Technology (UIST 2003), ACM Press, , pp. 69–70.
- [28] **BRETZNER L.**, et.al.(2001). *A Prototype System for Computer vision based Human Computer Interaction*, Technical report, ISRN KTH/NA/P-01/09-SE, Stockholm, Sweden.

- [29] **REN X.** Et.al.(2005). *Recovering Human Body Configurations using Pairwise Constraints Between Parts*, In ICCV, volume 1, pages 824–831.
- [30] **AGRAWAL, T., CHAUDHURI, S.** (2003). *Gesture Recognition using Position and Appearance Features*, ICIP (2) : 109-112
- [31] **POSTIGO J.F.** et.al.(2000). *Hand Controller for Bilateral Teleoperation of Robots*, ROBOTICA, 18, 2000, pp. 677 – 686, UK
- [32] **FUJISAWA S.** Et.al.(1997). *Fundamental Research on Human Interface Devices for Physically Handicapped Persons*, IECON 97. 23rd International Conference, New Orleans.
- [33] **PHILOMIN V.** et.al (2000). *Pedestrian Tracking from a Moving Vehicle*, IEEE Intelligent Vehicles Symposium.
- [34] **MANTYLA V.M.** et.al.(2000) *Hand Gesture Recognition of a Mobile Device User*, in IEEE International Conference on Multimedia and Expo, 2000. ICME 2000., vol. 1, pp. 281 – 284, NewYork.
- [35] **UEDA E.** et.al.(2003) *A Hand-Pose Estimation for Vision-Based Human Interfaces*, IEEE Transactions on Industrial Elec-tronics, Vol. 50, No. 4, pp.676–684.
- [36] **BRAY M.** et.al (2004), *3d Hand Tracking By Rapid Stochastic Gradient Descent Using A Skinning Model*, Visual Media Production,. (1st European Conference publication) 59- 68.
- [37] **BETTIO F.** et. al.(2007). *A Practical Vision Based Approach to Unencumbered Direct Spatial Manipulation in Virtual Worlds*, In Eurographics Italian Chapter Conference. Eurographics Association.
- [38] **DORNER B.**(1994). *Chasing the Colour Glove*, MSc. Thesis. Burnaby, BC, Canada: School of Computer Science, Simon Fraser University
- [39] **LEE J., KUNII T.**(1996). *Model-Based Analysis of Hand Posture*, IEEE Computer Graphics and Applications, pp.77-86.

- [40] **GUPTA N.** et.al. (2002). *Developing a Gesture-based Interface*, IETE, Journal of Research: Special Issue on Visual Media Processing.
- [41] **BLACK M. J.** (1996). *EigenTracking : Robust Matching and Tracking of Articulated Objects Using a View-Based Representation*, International Journal of Computer Vision, 26(1), pp. 63-84, 1998. also Xerox PARC, Technical Report P95-000515.
- [42] **ZAHEDI M.**et.al.(2005). *Appearance-Based Recognition of Words in American Sign Language*, In IbPRIA 2005, (2nd Iberian Conference on Pattern Recognition and Image Analysis), LNCS volume 3522, pp511-519, Estoril, Portugal.
- [43] **GÖKNAR, G.; YILDIRIM, T.** (2005) *Hand Gesture Recognition Using Artificial Neural Networks*, Signal Processing and Communications Applications Conference, 2005. (Proceedings of the IEEE)13th Volume , Issue , 2005 Page(s): 210 - 213
- [44] **LEE J.S.**(2004). *Hand Region Extraction and Gesture Recognition from Video Stream with Complex Background Through Entropy Analysis Engineering in Medicine and Biology Society, IEMBS '04.* (26th Annual International Conference of the IEEE Publication) ,pp1513- 1516 Vol.2
- [45] **YANG M.H.** *Extraction of 2D Motion Trajectories and Its Application to Hand Gesture Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence archive.
- [46] **ROY V. and JAWAHAR C.V.**(2005). *Feature Selection for Hand-Geometry based Person Authentication*, in proceedings of International conference on advanced computing and communication, coimbatore, India.
- [47] **CUTLER R., TURK M.** *View-based Interpretation of Real-time Optical Flow for Gesture Recognition*, Third IEEE Conference on Face and Gesture Recognition, Nara, Japan, April 1998."
- [48] **NAIR V., CLARK J. J.**(2002) *Automated Visual Surveillance Using Hidden Markov Models*, In VI02, pp 88.

- [49] **ZHENG B. J. H.**(2002). *Maximum Entropy Models for Skin Detection*, Technical Report publication IRMA, Volume 57, number XIII, Universite des Sciences et Technologies de Lille, France, 2002.
- [50] **MARCEL S.** (2000). *Hand Gesture Recognition using Input–Output Hidden Markov Models*, Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition,pp 456.
- [51] **RUSSELL S., NORVIG P** (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ.
- [52] **GONZALEZ R. C., WOODS R.E.***Digital Image Processing*, Prentice Hall, ISBN 0-201,18075-8.
- [53] <http://en.wikipedia.org>
- [54] **TRAVIS D.**, Effective Color Displays. Theory and Practice, Prentice Hall, 1991, ISBN 0-201, 18075-8.
- [55] **BOVIK C. AND DESAI M.D.**(2000). *Basic Binary Image Processing*,in *Handbook of Image and Video Processing*, , pp. 37–52, Academic Press, San Diego, Calif, USA.
- [56] **CIPOLLA R. AND HOLLINGHURST N.J.**(1998). *A Human-Robot Interface Using Pointing with Uncalibrated Stereo Vision*, in *Computer Vision for Human-Machine Interaction*. Ed. R. Cipolla and A. Pentland. Cambridge University Press.
- [57] **HASSANPOUR R. ,BIRDAL A.** (2005). *Real Time Hand-Face Tracking using Skin Color and Motion* , The 9th World Multiconference on Systemics, Cybernetics and Informatics
- [58] **INTEL Corp.,** (2000) , *Open Source Computer Vision Library, Reference Manual*, Order Number : 123456 – 001

- [59] **ÖZBAY E.** (2005). *Model Based Human Face Detection Using Skin Color Segmentation*, MSc. Thesis., Ankara: Computer Engineering, Çankaya University.