



Real Time Zone Based Hand Gesture Recognition System

Acc. No. (E.M.S) T-1411



Developed by
Fatimah Mohammad
217-CS/MS/2004

Supervised by
Dr. Syed Afaq Husain



Department of Computer Science
Faculty of Applied Sciences
International Islamic University, Islamabad
2006

**A dissertation submitted to the
Department of Computer Science,
International Islamic University, Islamabad
as a partial fulfillment of the requirements
for the award of the degree of
MS in Computer Science**

**Department of Computer Sciences
International Islamic University, Islamabad**

Date: August 12, 2006

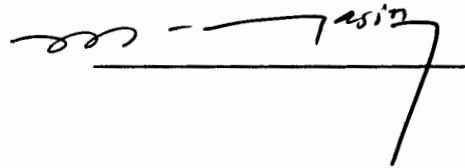
Final Approval

It is certified that we have read the thesis submitted by Fatimah Mohammad, registration number 217-CS/MS/04, and it is our judgment that this thesis is of sufficient standard to warrant its acceptance by the International Islamic University, Islamabad for the MS Degree in Computer Science.

Committee:

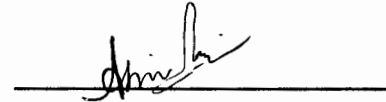
External Examiner

Prof. Dr. Mehboob Yasin
Department of Computer Sciences, COMSATS
Islamabad



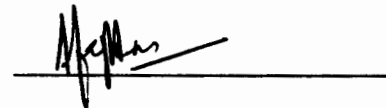
Internal Examiner

Mr. Asim Munir
Assistant Professor
Faculty of Applied Sciences
International Islamic University,
Islamabad



Supervisor

Dr. Syed Afaq Husain
Head, Departments of Telecom & Computer Engineering
International Islamic University,
Islamabad



"If people knew how hard I worked to achieve my mastery, it wouldn't seem so wonderful at all."

- Michelangelo Buonarroti (1475-1564 Italian Renaissance sculptor, painter, architect, and poet)

**I dedicate this work to my parents,
My brothers and sisters,
My nieces and nephew,**

**I dedicate it to my teachers,
And to my friends.**

**Each one has contributed,
In some way,
To making me the person,
I am today...**

Declaration

I, Fatimah Mohammad, D/O Dr. Sultan Mohammad, hereby declare and affirm that this software neither as a whole nor as a part thereof has been copied out from any source. It is further declared that I have developed this software and accompanied thesis entirely on the basis of my personal efforts, made under the sincere guidance of my teachers. If any part of this project is proved to be copied out or found to be a reproduction of some other project, I shall stand by the consequences. No portion of the work presented in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or institute of learning.

Fatimah Mohammad
217-CS/MS/04

Acknowledgement

“I have been complimented many times and they always embarrass me; I feel that they have not said enough”

-Mark Twain (1835-1910 American writer, journalist, and humorist)

After spending seven months of tiresome and uncertain hours in front of my computer system, I finally get to use this time and page to thank those people who helped me not only during this project semester, but also throughout the entire six years that I've been here.

It is with genuine gratitude that I thank my supervisor, Dr. Syed Afaq Husain, the Head of the Telecommunications and Computer Engineering Department. He taught me many moral lessons, both academic and nonacademic, that I know will help me a great deal throughout life. Had it not been for his constant supply of insight and encouragement, I am sure this project would not have been completed in good spirits nor would it have been completed in time. It is to the continuous inspiration he instilled in me that I attribute the successful completion of not only my MS coursework but also my research.

Special thanks go to all my teachers, starting from my first semester in BCS up until my last semester in MS, for their help and encouragement. In particular, I would like thank Mr. Asim Munir, Assistant Professor, Department of Computer Science. He has been my teacher for several years, and has helped and guided me time and again with the difficulties I encountered during the course of this research, projects, and my entire stay here.

Finally, I thank my parents, brothers, sisters, and friends for standing by me every step of the way. It is because of their guidance, motivation, support, and of course their prayers that I have attained the position in which I am today.

Fatimah Mohammad
217-CS/MS/04

Project in Brief

Project Title:	Real Time Zone Based Hand Gesture Recognition System
Undertaken by:	Fatimah Mohammad
Supervised by:	Dr. Syed Afaq Husain
Starting Date:	November 2005
Completion Date:	May 24, 2006
Tool Used:	MATLAB 7.0
Operating System:	Microsoft Windows 2000 Professional
System Used:	Pentium IV

Abstract

Human Computer Interaction is the discipline that studies the existing and possible communication interfaces between man and machine. The mouse and keyboard have been the standard mechanism for data input for several years. Other methods include light pens and writing tablets. One common feature shared by these input devices is that they tie the user to the machine, thus limiting the users' freedom of expression and movement.

With the passing of time and enhancement of technology, it is becoming increasingly clear that new and more natural methods of interaction are necessary. Hand gesture recognition is a field that has been rapidly gaining popularity over the last thirty years. The user becomes more integrated with the system and experiences a natural feeling by using hand gestures as a method to input data.

This research work proposes a Human Computer Interface that uses hand gestures to input data while giving a presentation using Microsoft PowerPoint. Hand gestures are acquired in real time. The system attempts to recognize the hand gesture only when the hand designates a *work area*. In this way the user does not face problems of *immersion* i.e. the system does not attempt to recognize, classify, and execute every movement made by user. The work area is divided into three zones and the gesture vocabulary consists of five gestures. The same gesture in different zones is used to execute different commands. In this way the vocabulary is easily limited facilitating the user as he or she does not have to remember a large number of hand poses. The gestures are also natural and can be performed by any right handed or left handed user. The method used consists of counting the number of fingers extended. Upon recognition, the gestures can be used to control a Microsoft PowerPoint presentation.

Table of Contents

Chapter No	Contents	Page No
1.	Introduction	2
	1.1 Conventional Methods for Data Input	2
	1.2 History and Applications of Human Computer Interaction	3
	1.2.1 Text Editors	3
	1.2.2 Hyper Text	4
	1.2.3 CAD and Video Games	4
	1.3 Upcoming Areas of Human Computer Interaction	4
	1.4 Nature of Gesture	5
	1.4.1 Representation of Gesture	6
	1.4.2 Hand Gesture Input	6
	1.4.3 Gesture Recognition	7
	1.4.3.1 Data Gloves	7
	1.4.3.2 Vision Based Gesture Recognition	8
2	LITERATURE SURVEY	10
	2.1 Vision Based Gesture Recognition	11
	2.1.1 Hand Segmentation	11
	2.1.2 Gesture System	11
	2.1.2.1 Pointing Analysis	12
	2.1.2.2 Command Gesture System	12
	2.2 Real Time Hand Gesture System Based on Evolutionary Search	12
	2.2.1 System Overview	12
	2.2.2 Segmentation	13
	2.2.3 Pose Recognition	13
	2.3 Computer Vision Based Gesture Recognition for an Augmented Reality Interface	14
	2.3.1 Counting Fingers	14
	2.4 Accurate Recognition of Large Number of Hand Gestures	14
	2.4.1 Hand Movement Tracking	14
	2.5 CHARADE: Remote Control of Objects Using Free Hand Gestures	15
	2.5.1 Structure of the Model	15
	2.5.2 Detection of Intention	15
	2.5.3 Segmentation of Gesture	15
	2.5.4 Classification	15
	2.6 Interaction via Motion Observation	16
3.	SYSTEM DESIGN	17
	3.1 System Setup	18
	3.2 Gesture Vocabulary	19

3.3	Image Acquisition	20
3.4	Frame Averaging	20
3.5	Hand Segmentation	21
3.6	Image Preprocessing	21
3.6.1	Image Cleaning	21
3.6.2	Smoothing Using Mode	21
3.6.3	Image Erosion	21
3.7	Connected Component Labeling	22
3.8	Feature Extraction	22
3.8.1	Area of Object	22
3.8.2	Minimum Enclosing Rectangle	22
3.9	Skeletonization	23
3.10	Zone Determination	24
3.11	Gesture Recognition	24
3.11.1	Determination of the Number of Fingers	25
3.11.1.1	Finding Finger End Points	25
3.11.1.2	Calculation of Finger Length	26
3.12	Interfacing With PowerPoint	26
3.13	Application Control	26
3.13.1	Training	26
3.13.2	Running the Presentation	27
4.	IMPLEMENTATION	28
4.1	MATLAB 7.0	29
4.2	Image Acquisition	29
4.3	Frame Averaging	30
4.4	Image Preprocessing	30
4.5	Region Labeling	31
4.6	Area Calculation	31
4.7	Minimum Enclosing Rectangle	31
4.8	Thinning	32
4.9	Zone Determination	32
4.10	Gesture Recognition	33
4.10.1	Determination of Finger End Points	33
4.10.2	Calculation of Finger Length	33
5	RESULTS & CONCLUSION	34
5.1	Image Preprocessing	35
5.1.1	Image Cleaning and Segmentation Results	35
5.2	Thinning	36
5.3	Calculating Finger Length	37
5.4	Thresholding Lengths	38
5.5	Gesture Recognition Results	38
5.5.1	Recognition of Previous Slide	40
5.5.2	Recognition of Next Slide	41
5.5.3	First Slide	42

5.5.4	Last Slide	43
5.5.5	GOTO Slide 1	44
5.5.6	GOTO Slide 2	45
5.5.7	GOTO Slide 3	46
5.5.8	GOTO Slide 4	47
5.5.9	Exit Slideshow	48
5.5.10	Begin Presentation	49
5.6	Analysis	50
5.7	Conclusion	51
5.7.1	Elimination of Immersion Syndrome	51
5.7.2	Natural Gestures to Allow Ease of Use	51
5.7.3	Open to Left or Right Handed Users	51
5.7.4	Same Gestures in Different Zones	52
5.7.5	Real Time Processing	52
5.7.6	Future Enhancements	52
Appendix	Appendix A	53
	A.1 Hardware and Software Installation Notes	54
	A.2 Frame Averaging	55
	A.3 Image Preprocessing	55
	A.4 Region Labeling	55
	A.5 Thinning	56
	A.6 Zone Determination	56
	A.7 Gesture Recognition	56
	A.7.1 Determination of Finger Endpoints	56
	A.7.2 Calculation of Finger Length	57
References		59

CHAPTER 1

Introduction

1. Introduction

Human Computer Interaction (HCI) is the discipline that deals with the mechanisms by which the human user or users can communicate with one or more computational machines. As HCI studies a human and a machine in communication, it uses knowledge drawn from both the human and machine. On the human side, communication mechanisms, graphical design disciplines, language, psychology, and human performance are relevant. On the machine side, techniques in computer graphics, operating systems, programming languages, and development environments are relevant.

For several decades, the customary method of communicating with computers has been through the use of the standard keyboard and mouse. Besides the addition of a few keys and buttons, there has not been much technological advancement in the design and functionality of these devices since they first appeared in the market many, many years ago.

1.1 Conventional Methods for Data Input

With the advent of time and enhancement in technology, there is an increase in the need to improve the interface between the user and the computer. Certain advancements with respect to interfacing include light pens, and writing tablets. As shown in Figure 1.1, a light pen is a computer input device in the form of a light-sensitive wand used in conjunction with the computer's CRT monitor. It allows the user to point to displayed objects, or draw on the screen, in a similar way to a touch screen but with greater positional accuracy. The light pen became moderately popular during the early 1980's.

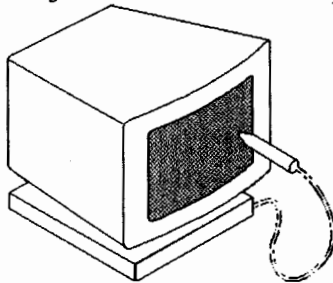


Figure 1.1 Light Pen

However, due to the fact that the user was required to hold his or her arm in front of the screen for long periods of time, the light pen fell out of use as a general purpose input device.

Virtual Reality aims at addressing as many human senses as possible. Generally speaking, the term Virtual Reality (VR) describes a computer generated scenario of objects (virtual world) the user can interact with. In contrast to conventional computer interfaces, the interaction is designed in three dimensions rather than two and the user is completely immersed in an artificial 3-D world completely generated by a computer. The combination of three-dimensional computer graphics; special display techniques; and specific input devices allows intuitive manipulation of objects in the virtual scenario thus giving users the impression of being part of the picture.

In virtual reality systems, a common input device is the Head Mounted Display (HMD). The HMD was the first device providing its wearer with an immersive experience. As shown in Figure 1.2, a typical HMD houses two miniature display screens and an optical system that channels the images from the screens to the eyes, thereby, presenting a stereo view of a virtual world. A motion tracker continuously measures the position and orientation of the user's head and allows the image generating computer to adjust the scene representation to the current view. As a result, the viewer can look around and walk through the surrounding virtual environment. A variety of input devices like data gloves, joysticks, and hand-held wands allow the user to navigate through a virtual environment and to interact with virtual objects.



Figure 1.2 Head Mounted Display

Directional sound, tactile and force feedback devices, voice recognition and other technologies are being employed to enrich the immersive experience and to create more "sensualized" interfaces.

The ubiquitous mouse and keyboard, along with the afore-mentioned light pens and other similar input devices all share the property of being tied to the computer, thus binding the user to the machine and greatly limiting the users' motion and expression. By allowing the user to become more integrated with the computer environment, more natural means of interaction can be possible thus allowing the user to control and manipulate the computer system using a more human-like approach.

1.2 History and Applications of Human Computer Interaction

Research in HCI has been quite successful. In 1963, Ivan Sutherland demonstrated in his MIT PhD thesis SketchPad, which was a man-machine graphical communication system [1]. It consisted of a direct manipulation interface, where visible objects on the screen are directly manipulated with a pointing device. Even the common mouse was developed at Stanford Research Laboratory (now SRI) in 1965 to be a cheap replacement for light-pens, which had been used at least since 1954 [2].

1.2.1 Text Editors

Although it seems unlikely, text editors were in fact a breakthrough in HCI. In 1962 at the Stanford Research Lab, D.C. Engelbart proposed, and later implemented a word processor with automatic word wrap, search and replace, user-definable macros, scrolling text, and commands to move, copy, and delete characters, words, or blocks of text. In 1965, Stanford's TVEdit was one of the first CRT-based display editors that was widely used [3].

1.2.2 HyperText

In 1954, the idea for hypertext, where documents are linked to related documents was credited to Vannevar Bush's famous MEMEX idea [4]. In 1967, the Hypertext Editing System from Brown University had screen editing and formatting of arbitrary-sized strings with a light pen [5]. In 1969, the Hypertext Editing System, jointly designed by Andy van Dam, Ted Nelson, and two students at Brown University was distributed extensively [6]. In 1976, The University of Vermont's PROMIS was the first Hypertext system released to the user community. It was used to link patient and patient care information at the University of Vermont's medical center.

In 1977, the project titled *ZOG: A Man-Machine Communication Philosophy* from Carnegie Mellon University was another early hypertext system [7]. In 1983 at the University of Maryland, Ben Shneiderman's *Hyperties* was the first system where highlighted items in the text could be clicked on to go to other pages [8]. In 1990, Tim Berners-Lee used the hypertext idea to create the World Wide Web at the government-funded European Particle Physics Laboratory (CERN). Mosaic, the first popular hypertext browser for the World-Wide Web was developed at the Univ. of Illinois' National Center for Supercomputer Applications (NCSA).

1.2.3 CAD and Video Games

Work in computer graphics has continued to develop algorithms and hardware that allow the display and manipulation of ever more realistic-looking objects such as CAD/CAM machine parts or medical images of body parts [9].

The first graphical video game including the first computer joysticks was probably SpaceWar by Slug Russel of MIT in 1962 [10]. The early computer Adventure game was developed by Don Woods into a more sophisticated Adventure game at Stanford in 1966 [11]. In 1970, Conway's game of LIFE was implemented on computers at MIT and Stanford [10].

1.3 Upcoming Areas of Human Computer Interaction

The means by which humans interact with machines continues to evolve rapidly. The reason for this swift development is because of the ever changing demands of the public as well as the emergence of better technology. For example, hardware costs have decreased and memory and speed of computer systems has increased greatly. This motivates the search for innovative methods of input such as voice, gesture, and pen. The fact that 'size does matter' indicates that smaller devices are preferred as they are less cumbersome and more portable.

Some of the upcoming areas of HCI include Gesture Recognition, 3-D, and Virtual and Augmented Reality. Gesture recognition has been used in commercial CAD systems since the 1970s, and came to universal notice with the Apple Newton in 1992 [11]. The "Lincoln Wand" by Larry Roberts was an ultrasonic 3D location sensing system, developed at Lincoln Labs in 1966. An early use was for molecular modeling [12]. The original work on Virtual Reality was performed by Ivan Sutherland from 1965-1968.

Knowing that human gestures come in many forms such as hand gestures, body gestures and facial expressions, Hand Gesture Recognition in particular is an area in which effort and interest has been invested with the aim of developing techniques that reduce the complexity of interaction between humans and computers. In this paper human computer interactions through hand gestures will be considered.

1.4 The Nature of Gesture

Gestures are expressive, meaningful body motions – physical movements of the fingers, hands, arms, head, face, or body with the intent to convey information or interact with the environment. Hand gestures in particular are those movements made by a person's hands while he or she is communicating with others. Gestures in essence complement speech and may in some cases be more effective or may even replace speech. Gestures are used to communicate meaningful information (semiotic); to manipulate the environment (ergotic); or to discover the environment through tactile experience (epistemic) [13].

Although the concept of gesture is loosely defined and depends on the context of the interaction, there are four basic common gesture types and can be defined as [14]:

- Iconic: representational gestures depicting some feature of the object, action or event
- Metaphoric: gestures that represent a common metaphor, rather than the object or event
- Beat: small, formless gestures, often associated with word emphasis
- Deictic: pointing gestures that refer to people, objects, or events in space or time [15].

The vast majority of automatic recognition systems are for deictic gestures (pointing), emblematic gestures (isolated signs) and sign languages (with a limited vocabulary and syntax).

As mentioned earlier, it is desirable to have a human computer interface that is more natural and that can take advantage of human traits. With the ever decreasing costs in visual equipment, it is expected that in the near future nearly every system will possess the capability to 'see' its user and to be able to interact in a two-way manner as opposed to the one way communication that is prevalent today. Hand gesture recognition certainly is a field that is very rapidly opening the doors to constructive interfaces between man and machine.

1.4.1 Representation of Gesture

Gestures can be static, where the user assumes a certain pose or configuration, or dynamic, defined by movement. Some gestures have both static and dynamic elements where the pose is important in one or more of the gesture phases; this is particularly relevant in sign languages. When gestures are produced continuously, each gesture is affected by the gesture that preceded it, and possibly by the gesture that follows it.

This *co-articulation* may be taken into account as a system is trained if the representation supports it. There are several aspects of a gesture which may be relevant and therefore may need to be represented explicitly.

Hummels et al [16] describe four aspects of a gesture which may be important to its meaning:

- Spatial information: where it occurs, locations a gesture refers to
- Pathic information: the path which a gesture takes
- Symbolic information: the sign that a gesture makes
- Affective information: the emotional quality of a gesture

In order to infer these aspects of gesture, we first have to sense human position, configuration, and movement. This can be done directly with sensing devices such as magnetic field trackers, instrumented gloves, and data-suits (which are attached to the user) or indirectly using cameras and computer vision techniques. Each sensing technology differs along several dimensions, including accuracy, resolution, latency, range of motion, user comfort, and cost.

1.4.2 Hand Gesture Input

Using hand gestures as an input media is not a new idea. In 1979, the "Put that There" experiment already used primitive gestural input [17]. More recently, the availability of new devices to track hand gestures and the advent of virtual reality systems have popularized the concept of gestural input.

With respect to hand gesture input, three main directions have been investigated [18]:

- Virtual Reality Systems: user interacts mainly by means of direct manipulation of the objects of the application.
- Multi-Modal Interfaces: provides natural and powerful interaction by using the natural human-to-human communication means such as speech combined with gesture and gaze.
- Recognition of Gestural Languages: for example deaf sign language.

The expected advantages of hand gesture input can be summarized as follows:

- Natural interaction: Gestural input makes use of the natural human communicative skills to interact with a computer resulting in natural and easy to learn interfaces.
- Terse and Powerful interaction: Devices such as the Dataglove capture 16 (non independent) dimensions. The theoretical throughput of such devices is very high compared to traditional devices such as the keyboard or mouse.

- Direct interaction: With an ideal hand gesture input device, the hand would become *the* device. This makes it possible to emulate other devices (like the mouse or keyboard). The user becomes free to wander around without any device to carry and interact with the surrounding machinery by adequate gestures.

1.4.3 Gesture Recognition

There is no standard way to perform gesture recognition – a variety of representations and classification schemes are used. However, most gesture recognition systems share some common structure. Recognizing gestures is a pattern recognition task which typically involves transforming the input into an appropriate representation and then classifying it from a database of predefined gesture representations. As gestures are highly variable from one person to another and from one example to another with a single person, it is essential to capture the essence of the gesture and use this as a representation. Static gesture, or pose recognition can be accomplished by a straightforward implementation using template matching, geometric feature classification, neural networks, or other standard pattern recognition techniques to classify pose. Dynamic gesture recognition, however, requires consideration of temporal events. This is typically accomplished through the use of techniques such as time-compressing templates, dynamic time warping, Hidden Markov Models (HMMs), and Bayesian networks.

Most of the recent works related to hand gesture interface techniques have been categorized as: glove-based methods used in tracking systems and vision-based methods

1.4.3.1 Data gloves

People naturally use their hands for a wide variety of manipulation and communication tasks. Besides being quite convenient, hands are extremely dexterous and expressive, with approximately 29 degrees of freedom (including the wrist). In his comprehensive thesis on whole hand input, Sturman [19] showed that the hand can be used as a sophisticated input and control device in a wide variety of application domains, providing real-time control of complex tasks with many degrees of freedom. There are a number of commercially available tracking systems which can be used during gesture recognition, primarily for tracking eye gaze, hand configuration, and overall body position. For several years, commercial devices have been available which measure, to various degrees of precision, accuracy, and completeness, the position and configuration of the hand. One of these devices is the data glove- an instrumental glove or exoskeleton device mounted on the hand and fingers. Some advantages of data gloves include:

- Direct measurement of hand and finger parameters (joint angles, 3D spatial information, wrist rotation)
- Provides data at a high sampling frequency
- Easy to use
- No line-of-sight occlusion problems
- Relatively low cost versions available
- Data is translation-independent (within the range of motion)

Disadvantages of data gloves include:

- Calibration can be difficult
- Tethered gloves reduce range of motion and comfort
- Inexpensive systems can be very noisy
- Accurate systems are expensive
- The user is forced to wear a somewhat cumbersome device

Body suits are another method in which certain trackers or sensors are attached to the users clothing. The 3D position of these markers can be measured thereby allowing the perception of complex movement patterns.

Sensing technology has a long way to go to overcome the associated disadvantages. On the other hand however, passive sensing using computer vision techniques is beginning to make headway as a user-friendly interface technology.

1.4.3.2 Vision-based Gesture Recognition

The most significant disadvantage of the tracker-based systems is that they are cumbersome and expensive. This detracts from the immersive nature of a virtual environment by requiring the user to don an unnatural device that cannot easily be ignored and which often requires significant effort to put on and calibrate. Even optical systems with markers applied to the body suffer from these shortcomings. What many have wished for is a technology that provides real-time data useful for analyzing and recognizing human motion that is passive and non-obtrusive. Computer vision techniques have the potential to meet these requirements.

Vision-based interfaces use one or more cameras to capture images and understand those images to produce visual features that can be used to interpret human activity and recognize gestures. Typically the camera locations are fixed in the environment, although they may also be mounted on moving platforms or on other people. For the past decade, there has been a significant amount of research in the computer vision community on detecting and recognizing faces, analyzing facial expression, extracting lip and facial motion to aid speech recognition, interpreting human activity, and recognizing particular gestures. Unlike sensors worn on the body, vision approaches to body tracking have to contend with occlusions. From the point of view of a given camera, there are always parts of the user's body that are occluded and therefore not visible. Multiple cameras can be used, but this adds correspondence and integration problems.

Vision can use properties like texture and color in its analysis of gesture, while tracking devices do not. Unlike special devices which measure human position and motion, vision uses a multipurpose sensor; the same device used to recognize gestures can be used to recognize other objects in the environment and also to transmit video for teleconferencing, surveillance, and other purposes. Currently, most computer vision systems for recognition follow the pattern shown in Figure 1.3.

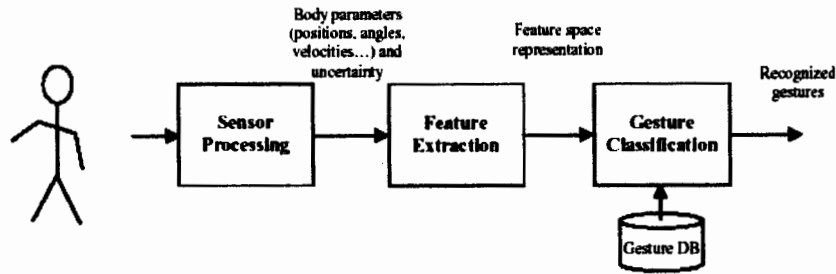


Figure 1.3 Vision based recognition process

Analog cameras feed their signal into a digitizer board, or framegrabber. There may be a preprocessing step, where images are normalized, enhanced, or transformed in some manner, and then a feature extraction step. The features – which may be any of a variety of two- or three-dimensional features, statistical properties, or estimated body parameters – are analyzed and classified as a particular gesture if appropriate.

Vision-based systems for gesture recognition vary along a number of dimensions, most notably:

- Number of cameras – How many cameras are used?
- Speed and latency – Is the system real-time?
- Structured environment – Are there restrictions on the background, movement, etc.?
- User requirements – Must the user (not) wear gloves, long sleeves, rings etc.?
- Primary features – What low-level features are computed (edges, regions, histograms)?
- Two or three-dimensional representation – Does the system construct a 2D or 3D model of the body?

The answers to the questions of course mostly depend on the application that is to be developed. In case the system is for a single user, one camera is probably sufficient. In case however the system is used for teleconferencing purposes, then multiple cameras may be necessary. Speed of course is an important issue especially in real time systems. In such applications the user expects a command to be executed as soon as it is issued. Certain constraints regarding background and/or foreground may be necessary to help with subsequent processing. During the development certain decisions must be made as to which features must be extracted and which features can be ignored. A successful vision system can therefore be developed whilst keeping the above points in mind.

CHAPTER 2

Literature Survey

2. Literature Survey

A great deal of research and development has been done in the area of hand gesture recognition. The literature survey for this field in general can be very comprehensive. However, in this context the recognition is taking place in a workspace or zone. This chapter therefore discusses that work which has been done in a context similar to the current one.

2.1 Vision Based Gesture Recognition

Ryan Garver stated the development of a gesture based interface system [20]. The goal of the project was to build a system that uses natural gestures for computer control within the *working zone*. In order to limit the complexity of this project there have been several assumptions made about the environment in which this system is to be used. Most of these restrictions are designed to help with hand segmentation. The first restraint is that the background remains fairly constant and well textured; this is necessary to facilitate the use of background subtraction. Another stipulation made is that lighting will remain fairly constant; skin color methods tend to be sensitive to lighting changes.

2.1.1 Hand Segmentation

This module performs static hand segmentation from individual frames. The hand segmentation system is made up of three image processing algorithms: dynamic background subtraction, statistical skin color segmentation, and depth thresholding. By combining these methods a relatively reliable segmentation system is constructed. The first step in this hand segmentation is the background subtraction. The result is a trimmed down image containing only foreground information. On the resulting image the skin color filter is applied. From this an image of skin color pixels that lie in the foreground of the image is retrieved. Morphological filters clean up the image, and then a depth threshold segments the hands from the face, forearms, and other unwanted \skin colored pixels.

2.1.2 Gesture System

The two primary methods of gesture use can be described as pointing gesture, and command gesture. The pointing gesture module of the system creates a rough association from movements made by the user to movements made by the mouse. Along with this is a method for conveying mouse clicks, this together with the mouse motion of the pointing system provides a fairly generic user interface with practical applications. The second portion of the project is the command gesture module. The command gesture module is designed to translate discrete hand motion into a keyboard or some other command that executes a certain piece of functionality. An important first step in the gesture analysis is the classification of gestures between pointing and command. For the sake of speed and simplicity this was implemented by looking at the number of hands and at the basic shape of the hands in question. To engage a pointing gesture the user must insert a single hand into the working zone.

If the hand shape is fist-like the system begins interpreting the gesture as pointing. If two hands are inserted into the working zone the gesture is interpreted as a command gesture and processed accordingly.

2.1.2.1 Pointing Analysis

The pointing gesture system uses the change of position of the center of the hand as a tie to the change in position of the mouse cursor. Using this information a sense of acceleration in the mouse pointer is created. The faster the user moves their hand the more the mouse will move. This allows the user to take advantage of the whole area of the screen with out needing to do the gesture equivalent of picking up the mouse. If the user needs to make a small movement of a few pixels, the user can just make slow movements.

2.1.2.2 Command Gesture System

The first step in preparing the gesture for executing a command is to divide the hand motion into three portions: start, gesture, finish i.e. finding the beginning of the actual gesture and separating it from the rest of the hand motion. The gesture is marked by a change in position of more than a prescribed distance, and then only after a certain short period of time from when the hands enter the working area and are captured. The end is delimited by the removal of the hands from the working area. This produces a gesture sequence that is then processed by a series of HMMs. Each HMM represents a specific gesture, and the performed gesture is classified as the representative model with the highest probability of correspondence.

2.2 Real Time Hand Gesture System Based on Evolutionary Search

Juan Wachs et al considered a vision based system that can interpret a user's gestures in real time to manipulate objects within a medical data visualization environment [21].

2.2.1 System Overview

A webcam placed above the screen captures sequence of images. The hand is segmented using color, B/W threshold and various morphological image processing operations. The location of the hand in each image is represented by the 2D coordinates of its centroid; and mapped into one of eight possible navigation directions of the screen to position the cursor of a virtual mouse. The motion of the hand is interpreted by a tracking module. At certain points in the interaction it becomes necessary to classify the pose of the hand.

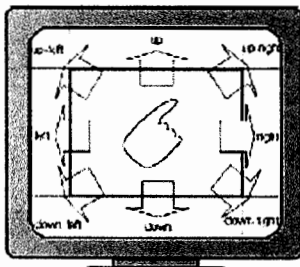


Figure 2.1 Directional Zones

The image is then cropped tightly around the blob of the hand and a more accurate segmentation is performed. The postures are recognized by extracting symbolic features, from the sequence of images. The classification is used to bring up X-rays images, select a patient record from the database or move objects and windows in the screen.

2.2.2 Segmentation

In order to track and recognize gestures, the CAMSHIFT algorithm is used. For CAMSHIFT, a probability distribution image of the hand color is created using a 2D hue-saturation color histogram. This histogram is used as a look-up-table to convert the acquired camera images into a corresponding skin color through a process known as back propagation. Thresholding to black and white followed by morphological operations is used to obtain a single component for further processing to classify the gestures. The initial 2D histogram is generated in real-time by the user in the 'calibration' stage of the system. The calibration process is initiated by the detection of hand motion within the template region. In order to avoid false motion cues originated by non hand motion a background maintenance operation is maintained. A first image of the background is stored right after the application is launched, and then background differencing is used to isolate the moving object (hand) from the background. Since background pixels have small variations due to changes in illumination over an extended period of time, the background image is dynamically maintained. Background variations are identified by threshold the absolute difference between two consecutive frames.

2.2.3 Pose Recognition

Gestures are classified using a simple finite state machine. When the doctor wishes to move the cursor over the screen, he moves his hand out of the 'neutral area' to any of the 8 directions. While the hand is in one of the 8 quadrants, the cursor moves in the requested direction. The interaction is designed in this way because the doctor will often have his hands in the 'neutral area' without intending to control the cursor. The user can precisely control pointer alignment. When a doctor decides to perform a click, double-click, or drag with the virtual mouse, he/she places the hand in the 'neutral area' momentarily. This method differentiates between navigation and precise commands.

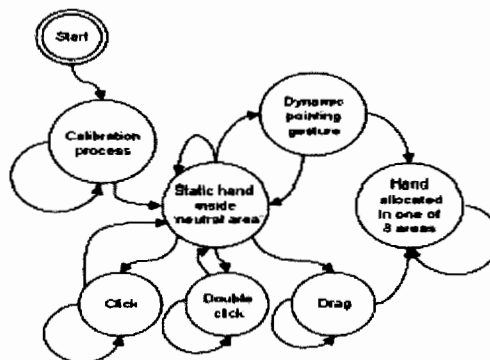


Figure 2.2 Finite State Machine for Recognition

2.3 Computer Vision Based Gesture Recognition for an Augmented Reality Interface

Moritz Störring et al presented a computer vision based gesture interface that is part of an Augmented Reality Interface [22]. It can recognize a 3D pointing gesture, a click gesture and 5 static gestures. The gesture recognition system presented in this paper is part of an AR project, the goal of which is to develop a multi-user AR system for round-table meetings, e.g., for architects. The system should contain a sufficient number of different gestures in order to make a useful interface, and have a low computational complexity. This should be achieved without introducing any enhancements like markers and Infrared lighting.

2.3.1 Counting Fingers

Hand and fingers can be approximated by a circle and a number of rectangles. By applying a polar transformation around the center of the hand and counting the number of fingers present in each radius, a gesture can be recognized. When only 1 finger is recognized this is a pointing gesture and the tip is the actual position where the user is pointing. A click gesture is defined as movement of the thumb in 3 static states: in the first and third state the thumb is next to the index finger. In the 2nd state the thumb has an angle with the index finger. Movement between these states can be recognized by changes in the width of the bounding box i.e. it grows and then shrinks.

2.4 Accurate Recognition of Large Number of Hand Gestures

Atid Shamaie et al [23] introduced a hierarchical gesture recognition algorithm to recognize a large number of gestures. Given an image sequence the problem is to find a gesture in a large database of predefined gestures which is the most similar to that. A hierarchical algorithm is proposed that uses two levels of filtering and a third level for decision making. The first two filtering levels extract the most probable gestures from the database and the final level's decision is based on graph matching for a final selection between a few gestures passed through the previous filters.

2.4.1 Hand Movement Tracking

It is assumed that a hand gesture starts from a neutral position. This condition can be the hand hanging by the side or the hand in lap. This is considered as the hand out of region of interest (ROI). The region of interest is defined as the area in which the hand gesture must be done to be recognized by the system. Before the user starts to gesture, he/she must move the hand into the ROI. Every gesture starts with a particular shape of hand. The position of hand is defined by the centre of the rectangle containing the hand.

2.5 CHARADE: Remote Control of Objects Using Free Hand Gestures

Thomas Baudel et al presented an application that uses hand gesture input to control a computer while giving a presentation [24]. The display presented to the audience is an active surface that reacts to the speakers gestures, yet the speaker can still gesture to the audience whilst operating other systems. The system has a well defined means to detect the *intention* of the gesture in order to avoid every gesture being interpreted by the system- whether it was intended or not. The model is based on the notions of active zone and gestural commands.

2.5.1 Structure of the Model

The model is based on the notions of active zone and gestural commands. The *active zone* is a 2D area, typically the projection of a computer screen on a wall. Gestures are interpreted only when the user designates this area, i.e. when the projection of the hand enters the active zone. The recognition of a command involves three steps: detection of the intention to address a command to the system, segmentation of gestures (recognition of start and end positions), and classification (recognition of a gesture in the command set). As soon as a command is recognized, it is issued. Gestures that are not recognized are simply ignored. The user wears gloves.

2.5.2 Detection of Intention

Gestures are interpreted only when the projection of the hand is in the active zone. This allows the user to move and perform gestures in the real world. It also makes it possible to use several active zones to address different systems. Commands are ended by leaving the end zone or using an end position.

2.5.3 Segmentation of Gesture

Start and end positions are defined by the wrist orientation and finger positions. These dimensions are quantized in order to make positions both easier to recognize by the system and more predictable by the user. Seven orientations of the wrist, four bendings for each finger, and two for the thumb are being used. This gives theoretically 3,584 positions, among which at least 300 can be obtained with some effort and from 30 to 80 are actually usable (depending on the user's skill and training).

2.5.4 Classification

Different gestures are classified according to their start position and dynamic phase. The dynamic phase uses the path of the projection of the hand, the rotation of the wrist, the movements of the fingers, and the variation of distance between the hands to the active zone (allowing for push-like gestures).

2.6 Interaction via Motion Observation

Foyle et al described the development of a novel HCI [25]. A standard webcam watches the user and responds to their movements so that the user can operate the system, play games, and move a mouse pointer without external devices. The user then moves their hand around in the visual area of the camera, and a continuous stream of images is captured. For tracking, background subtraction is employed to find which objects are 'new' to the scene, and assumes that the biggest object was the object that was to be tracked and its location can be derived. The system incorporates a weighting system, such that connected areas hold a higher value than non-connected areas, and as a result the largest object in the image will be located correctly. Instead of using a mouse pointer, the interface uses a concept of "zones." As shown in Figure 2.3, the camera's field of view is split into a grid of zones and the system observes which zone the user's hand is located in. If the user's hand is located in the top right zone, this is shown on screen by the illumination of an object in the top right area of the interface. For a mouse click, the user simply selects an object by moving into the corresponding zone, and then pauses for half a second. To provide as static background as possible, the camera was oriented such that it pointed down onto a surface, such as a desk.

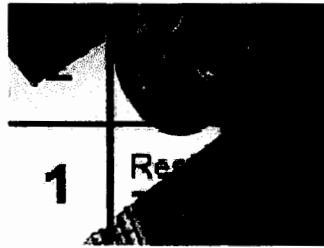


Figure 2.3 Camera view divided into zones

CHAPTER 3

System Design

3. System Design

Once the requirements have been gathered and understood, it is possible to write up a description of what functions the system is to perform with respect to the application and how it is to carry out those functions. Figure 3.1 shows the block diagram of the system. In this chapter, a detailed description of the system and its modules is given from the software point of view, along with the hardware requirements, configurations, and settings.

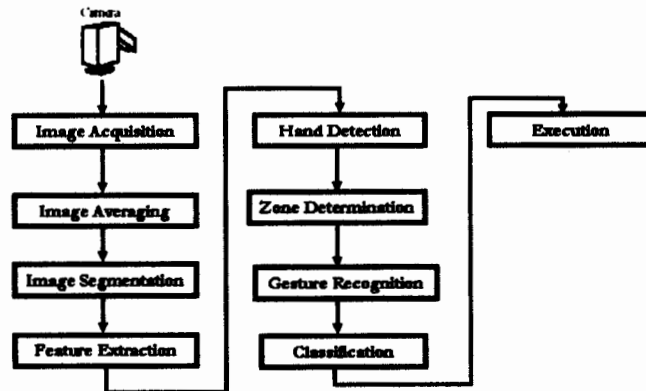


Figure 3.1 Block diagram of system

3.1 System Setup

In order to avoid the problems of immersion, the user places his or her hand in a workspace. Within this workspace all gesture recognition will take place i.e. the system only attempts to recognize those gestures performed in a certain zone. This will eliminate the problems associated with immersion. The user can also freely gesticulate and communicate with other persons and systems without the system trying to interpret every move. The user can also set up multiple devices, each with their own workspace so that multiple devices can be controlled. A black mat, similar to a large mouse pad and roughly 2-by-2 feet is designated as the workspace. It is partitioned into three blocks. The bottom two blocks are used to control the slide navigation and the top third block is used to jump to different slides. A simple web camera is used for image acquisition. The camera is fixed on a tripod such that it faces downwards. The only object within view of the camera is assumedly the mat and subsequently the user's hand. Frames are acquired at a fixed rate of 6 frames per second.

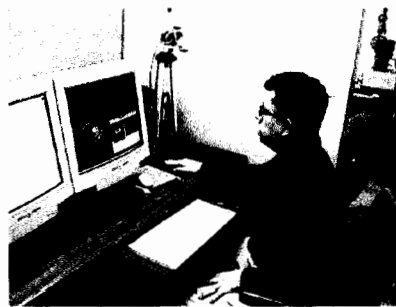


Figure 3.2 Possible system setup

3.2 Gesture Vocabulary

One of the challenges of gesture recognition is to define a practical gesture vocabulary. When defining the hand gestures that will be used as input to an interface system, certain points must be kept in mind such as the ease of the gesture. The gesture should be natural and simple to remember. Gestures from sign language are not recommended as they are highly structured and can be difficult for an amateur to perform and remember. Cultural dependence should also be kept in mind. Certain gestures exist that are culture specific i.e. they occur in one culture but not in another. For example, the traditional Indian greeting is to hold the two hands together, palm to palm. So when selecting a gesture it is preferable if it is not culture dependent so that it does not become illogical in another culture. The gestures should not be too complex to perform so that even those who are aged or who have motor or physical disabilities will not face any discomfort. The vocabulary should also be limited- enough gestures should exist to meet the aims of the application to be developed. In this way, both the user and the system will benefit in having to learn fewer gestures. The pose of the gesture should be logical i.e. it should represent the function that is to be performed and lie within the context of its use. In case of a pointing gesture for example; it makes more sense to define it as an outstretched index finger. The gestures employed here are shown in Figure 3.3.



Figure 3.3 The gesture vocabulary for our system consists of the above five gestures

Each gesture is identified by the number of outstretched fingers. From the user's perspective, it is likely that the gestures will be considered easy to use, perform, and remember. It is not necessary for the user to stretch their fingers to the maximum extent, although there should be some space separating the fingers. Examples of gestures used by different authors are shown in Figures 3.4a and 3.4b.

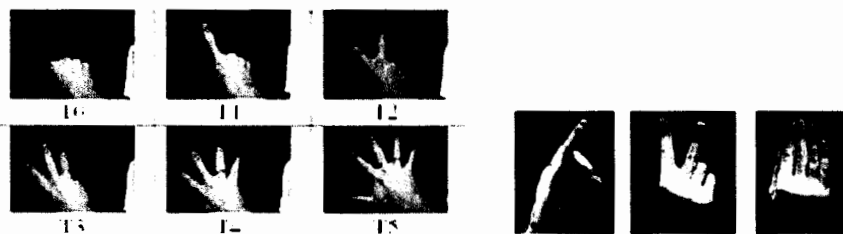


Figure 3.4 a) Gesture lexicon samples

b)

The variety of the gestures is enough to create an interface with several useful and interesting applications. From the system's point of view, the gestures are such that they are likely to be recognized. As the hand is parallel to the camera plane, the problem of recognition is reduced from three dimensions to two thus reducing the system's burden.

The application to control Microsoft PowerPoint ® presentations will use the gestures in Figure 3.3 as valid input.

With respect to the zone in which the gestures are to be performed, in Figure 3.5 the block labeled 1 is the first zone and is used to navigate back and forth in the slides. Holding one finger up will cause the presentation to move to the previous slide whereas holding two fingers up will cause it to move to the next slide. The second zone is 2. One finger will cause the presentation to jump to the first slide whereas two fingers will cause it to jump to the last slide. Holding all fingers up will cause the presentation to exit. The third zone, 3, is the top area and here the user can jump to a specific slide. With these gestures, the user can go directly to the first 4 slides.

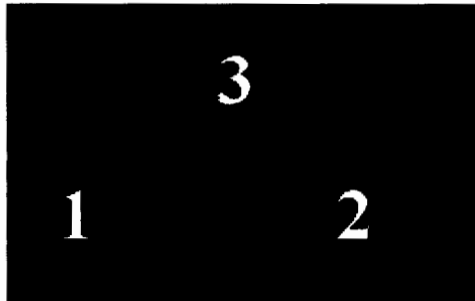


Figure 3.5 Division of workplace into zones

3.3 Image Acquisition

As the system operates in real time, a live video stream must be captured using an attached standard web camera. Frames are acquired at a low rate as opposed to the standard rate of 30 Hz. This is because the user's hands enter the zone for a short period of time before exiting again and to compensate hand motion effects. The system is employed to automatically control a presentation, so instead of interacting with the vision system for prolonged periods of time, the user will be more occupied with communicating with the environment and other people. The accompanying illumination should not be too low so that the hand is visible within the zone and it should not be too high to avoid problems of reflectance. It was not necessary to have a high resolution for the images as we are not extracting very low level features of the hand. For example, for fingerprint recognition very high resolution images would be required so that the intricacies of the fingerprint can be seen. Also, computational expense is avoided by using low resolution images.

3.4 Frame Averaging

It is not known when the user will insert his or her hand in the workspace. When the hand is inserted, it remains there for a short while only. In order to get a good snapshot of the object that entered the zone, frame averaging is performed. In this way, an average of the acquired frames is calculated so that a general or basic picture of the object can be taken and further processed. Problems of illumination and motion blur can be avoided through averaging. Averaging is done not by nested calls to individual arithmetic functions but by

performing a linear combination of the images. When calls are nested to arithmetic functions, values are truncated and rounded to the next nearest value. Thus, accuracy is lost in the final result. By performing a linear combination, all arithmetic operations are performed at once before truncating and rounding the final result. Although the live video stream is in RGB format i.e. it is a colored video stream, all three planes are not necessary for subsequent processing. Only the Red plane is extracted as skin tones are known to contain large amounts of red.

3.5 Hand Segmentation

The averaged image in fact captures the fundamental nature of the gesture. The black background color was selected to facilitate hand segmentation. It is expected that the only object that will appear in the workspace is the hand. In case other objects happen to appear on the zone, the largest object will be taken as the hand. As the human skin contains more red only the red plane is used. The segmentation process therefore, is fairly simple. A fixed threshold is used to extract the hand from the background.

3.6 Image Preprocessing

Once the hand object has been segmented from the background, it must be preprocessed. Several morphological operations are applied on the object to clean it and remove spurious pixels. The cleaning is necessary so that in later steps the object can be thinned properly.

3.6.1 Image Cleaning

This is a morphological operation that removes *isolated* pixels i.e. individual 1's surrounded by 0's, such as the center pixel in the pattern shown in Figure 3.6.



Figure 3.6 Image Cleaning

3.6.2 Smoothing Using Mode

By checking the mode of the pixels in a neighborhood, the extra uncorrelated pixels may be removed. If five or more pixels in its 3-by-3 neighborhood are 1, then that pixel is set to 1 otherwise it sets it to 0.

3.6.3 Image Erosion

Erosion is a well known morphological operation that erodes or shrinks an image from the outer boundaries. The amount of erosion depends on the structuring element. In this case, the structuring element is such that it erodes at a depth of one pixel.

An example is shown in Figure 3.7. All foreground pixels that have at least 1 neighbor that belongs to the background will be eliminated. The red pixels indicate pixels that have been removed.

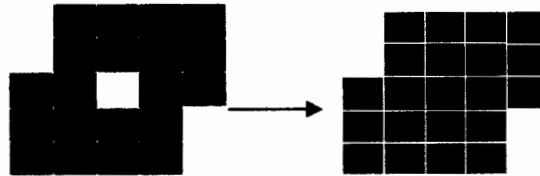


Figure 3.7 Image Erosion

3.7 Connected Component Labeling

Once the unnecessary pixels have been removed, the next step is to distinguish one object from the other in the image. This is done by labeling the connected components of the image. Two pixels are said to be *4-connected* if they show horizontal or vertical adjacency. If this adjacency includes the diagonal neighbors as well, then the pixels are said to be *8-connected*. In our system, 4-connectivity has been used.

3.8 Feature Extraction

Once all the objects have been distinctly labeled, they need to be analyzed to determine whether or not they meet the requirements to represent a hand.

3.8.1 Area of Object

Firstly, the size of the object is calculated in terms of its area. The area of an object is represented by the number of pixels composing that object. It is expected that only hands will be in the work area, so the largest object will be that of the user's hand and all other objects are discarded as they may be noise or redundant pixels belonging to some smaller object.

3.8.2 Minimum Enclosing Rectangle

The Minimum Enclosing Rectangle (MER) is the smallest rectangle that completely encloses the region. By finding the upper left coordinates and the lower right coordinates of the region a corresponding MER can be drawn around the object. The coordinates obtained will thereafter be used to determine which zone the user's hand is located in. The appropriate command can then be executed.



Figure 3.8 Minimum Enclosing Rectangle of hand object

3.9 Skeletonization

It is important to find a representation of the gesture that depicts the essence of the gesture. Thinning an object reduces it from a solid or compact object to lines. It removes pixels so that an object without holes shrinks to a minimally connected stroke. The pixels which are removed are the redundant pixels; once they are removed the image is simpler and shows the underlying structure of the object.

The problem while performing thinning lies in determining which pixels are redundant. The pixels to be removed are marked in a first pass and removed in a second pass over the image. This is repeated until there are no more redundant pixels, at which point the remaining pixels are those belonging to the skeleton of the object. Thinning must not be confused with *erosion*. There are some very important differences between the two. Erosion can be used to completely delete an object whereas thinning must never do this. The skeleton must remain intact and must have the following properties:

- It should consist of thin regions, one pixel wide.
- The pixels comprising the skeleton should lie near the center of cross section of the region.
- Skeletal pixels must be connected to each other to form the same number of regions that existed in the original figure.

Unlike the human face, the human hand presents a greater degree of uniformity. Differences in hands such as color, size, presence or absence of hair, rings, moles, scars etc., have a negligible effect on the performance of our system. By thinning the cleaned hand image, the underlying structure or skeleton of the human hand can be obtained. It was found that when the hands were held in the specified poses, there was a great deal of consistency in the thinned images of hands belonging to different people.



Figure 3.9 Top row- hands belonging to 1 male and 2 females.
Bottom row- Corresponding thinned images

3.10 Zone Determination

When the features have been extracted and the hand image has been thinned, the zone where the user's hand is located in must be determined. This is done with the help of the MER coordinates obtained during the feature extraction process. The zone is divided into 3 partitions- a top and two bottom halves. The two bottom halves correspond to zones 1 and 2 respectively and the top corresponds to zone 3. When the MER coordinates are compared, they are checked to see which zone they fall in. If the upper left and lower right coordinates fall within one of the zones then the appropriate command is executed. The boundaries of the zones are kept thick to avoid overlapping of one zone over the other.

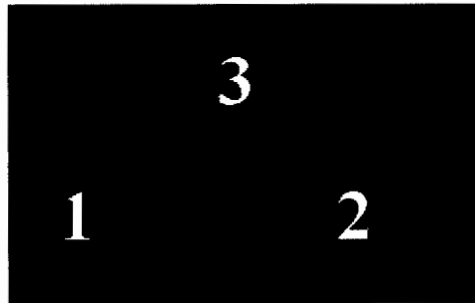


Figure 3.10 Workspace partitioned into zones

3.11 Gesture Recognition

A variety of classification schemes are used for gesture recognition. However, most computer vision based gesture recognition systems share some common structure. Recognizing gestures is a pattern recognition task which typically involves transforming the input into the appropriate representation and then classifying it from a database of predefined gestures. Static gesture or pose, recognition can be accomplished by a straightforward implementation using template matching, geometric feature classification, neural networks, or other standard pattern recognition techniques to classify pose. Dynamic gesture recognition, however, requires consideration of temporal events. This is typically accomplished through the use of techniques such as time-compressing templates, dynamic time warping, Hidden Markov Models (HMMs), and Bayesian networks. As gestures are highly variable from one person to another and from one example to another with a single person, it is essential to capture the essence of the gesture and use this to represent the gesture. The approach used also depends on the nature of the gesture.

Jerome Martin et al [27] used Principal Component Analysis (PCA) for hand posture classification. In the paper by Moritz Störring et al [22], the hand was approximated by a circle and a number of rectangles where the number is equal to the number of outstretched fingers. A polar transform was done around the center of the hand and the number of rectangles (fingers) in each radius was counted.

In the research by M.A. Foyle et al [25], the system was not restricted just to hands but to general object movement. Objects that were 'new' to the scene were tracked and the largest of those objects was considered the hand. Ryan Garver's [20] system performed gesture recognition based on the number of hands located in the work area. The presence of a single hand indicated a pointing gesture whereas the presence of two hands indicated a command gesture. In the paper by Juan Wachs et al [21] Haar features are calculated to estimate and classify the hand pose.

3.11.1 Determination of the Number of Fingers

The thinned hand images clearly show a structural correspondence among the hands of different people. The hands of several males and females of different age and size were used to support the fact that hands are indeed rather uniform when held in a particular pose. Factors such as age, size, and marks on the hand in the form of scars, moles, and warts are insignificant and can be ignored. Two functions were used to count the number of fingers- one function found the endpoints of the line image and the next discarded all those lines that did not fulfill the requirement of a particular length.

3.11.1.1 Finding Finger End Points

Each gesture defined is associated with the number of fingers extended by the user. The thinned images contain connected lines. To count the number of fingers, the endpoints of the lines were used instead of the entire line image. The number of endpoints gives the number of fingers extended. As fingers are extended in an upward direction, the bottom of the image is not needed and any lines or endpoints present at the bottom of the image can be discarded.

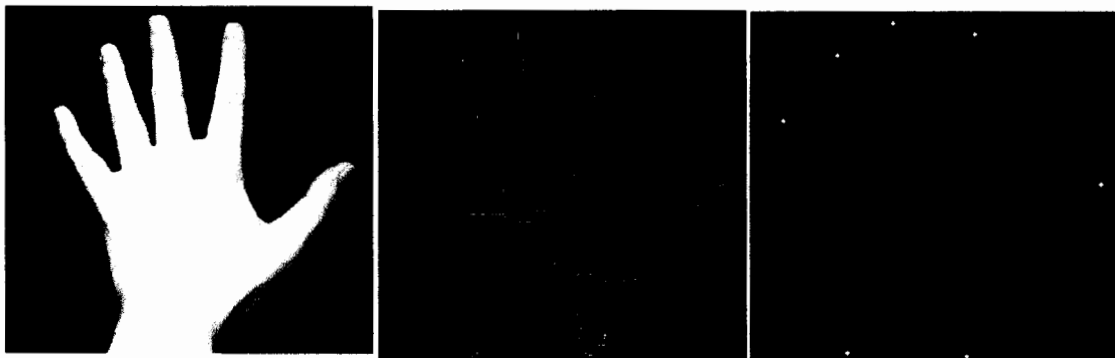


Figure 3.11 a)Original

b) Thinned image

c) Endpoints of thinned image

3.11.1.2 Calculation of Finger Length

When a person's fingers are extended, the knuckles may appear prominent. In the thinned image, the knuckles or wrist may appear as small lines. Fingers however should be longer lines. Therefore, all lines that are below a certain length are discarded and not counted as a finger.



Figure 3.12 a) Original image



b) Lines depicting fingers. Short lines discarded

3.12 Interfacing with PowerPoint

To be able to control the presentation, the application must somehow be linked to Microsoft PowerPoint. This was done using ActiveX. Microsoft PowerPoint was designated as an

ActiveX server so that certain commands could be utilized by the MATLAB application. These functions were executed based on the gesture input.

3.13 Application Control

When the user runs the application, they have the option of changing the resolution of the acquired images. After doing so, they can get accustomed with the zone and its partitions. By previewing a live video stream, they can adjust the camera view if necessary and check the position of their hand in each zone. Once they have an idea of where their hands appear in the preview window with respect to the workspace, they can begin to train themselves.

3.13.1 Training

During the training, the user is shown the workspace. He or she can insert and remove his or her hand gestures from the work area and a visual message will inform them what gesture had been recognized. This will familiarize the user to the zones and the gesture poses.

3.13.2 Running the Presentation

Once the user feels he or she has become familiar with the zones, their boundaries, and the gestures, he or she can run a presentation. The user can select any Microsoft PowerPoint presentation. After selection, the presentation runs in the slideshow mode and it can be controlled by the use of the hand gestures as input. There is no need for the mouse or keyboard at this point although they are fully functional at the time the presentation is running.

CHAPTER 4

Implementation

4. Implementation

The previous chapters dealt with understanding the theoretical nature of the problem. The next step is the implementation of the suggested solution. This chapter introduces the tool used (MATLAB 7.0) and describes the functions used during the implementation phase.

4.1 MATLAB 7.0

The research work was implemented in MATLAB 7.0 which is a product of MathWorks Corp. MATLAB® is a high-performance language for technical computing. It integrates computation, visualization and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include mathematical analysis and computation, data acquisition, data analysis, application development; including graphical user interface. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows the solving of many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non-interactive language such as C or Fortran. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis. MATLAB features a family of add-on application-specific solutions called toolboxes. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Toolboxes are available for many areas, including Image Processing.

4.2 Image Acquisition

The Image Acquisition Toolbox is a collection of functions that extend the capability of the MATLAB® environment. The toolbox supports a wide range of image acquisition operations, including acquiring images through many types of image acquisition devices, viewing a live video stream; starting acquisitions; configuring callback functions that execute when certain events occur; and bringing the image data into the MATLAB workspace.

As the application runs in real time, it is necessary to configure certain input device parameters. The input device in this case was a simple, standard web camera. The device specific parameters were the video resolution (this is set by the user) and the frame rate. The frames that are logged can be stored in memory only or in memory and disk, in case the user decides to save his or her gestures as an AVI file. Logging the frames in memory only is generally faster.

Each time a certain number of frames have been acquired certain functions are called and executed. Given below is the sequence of functions that are called along with their descriptions.

4.3 Frame Averaging

Once the data or frames have been captured, every two frames are averaged together. Only the red plane is used for getting the average. By using a linear combination of images, a greater degree of precision is maintained instead of nesting mathematical functions inside each other. The image is also rotated 180 degrees to adjust it according to the camera position.

FRAME AVERAGING (Frames)

Start Loop

- 1- Get first frame.
- 2- Get second frame.
- 3- Rotate frames by 180 degrees.
- 4- Get the red planes for consecutive frames
- 5- Get average by performing linear combination.

End Loop

4.4 Image Preprocessing

Here the averaged image is binarized and then cleaned using a variety of morphological filters applied in succession. The filters clean the image by removing isolated pixels and spurious pixels. Very small holes are filled in by changing a pixel's value according to the majority (mode) in its eight neighborhood.

IMAGE PREPROCESSING (Averaged Frame)

- 1- Convert grayscale image to binarized image:
 - a. If pixel's value is below 0.5, set to 0.
 - b. If the pixel's value is above 0.5, set to 1.
- 2- Clean image:
 - a. Remove each pixel whose value is 1 and is surrounded by 0's.
- 3- Smooth using mode:
 - a. If five or more pixels in a pixel's 8-neighborhood are 1, set the pixel to 1 else set it to 0.
- 4- Erode image:
 - a. If a foreground pixel has at least one neighbor belonging to the background, delete the pixel.

4.5 Region Labeling

Before an object can be described or recognized, its position in the image must be known, and all pixels belonging to the object must be identified. This function first segments the image into regions and then labels them. Since the pixels in a connected region are all ultimately connected to each other, it is necessary to find only one pixel belonging to the region. This pixel is known as the seed pixel. All other pixels connected to the seed pixel are then found. The area and bounding box are calculated. The object inside the bounding box is extracted and further processed.

REGION LABELING (Preprocessed Frame)

```
Start Loop through image
    1- Get seed pixel and set its value to 1
    2- Set all 4-neighbors of seed pixel to 1
End Loop
```

4.6 Area Calculation

Region labeling may result in a number of regions being produced. It is of interest to use only that region which contains the hand object. The area property of the hand will be used to determine if an object is a hand. Although it is assumed that the only object that appears in the camera's field of view is the hand, other small objects may appear due to noise. The object with the largest area is taken to be the hand.

CALCULATE AREA (Labeled Regions)

```
Start Loop through Number of Regions
    1- Get area of each region
        a. Count the number of pixels belonging to that region
    2- If the area satisfies a threshold, it is taken as hand
        Object else it is noise.
End Loop
```

4.7 Minimum Enclosing Rectangle

The system waits until a hand object is found. After it is known that a region contains a hand, the minimum enclosing rectangle (MER) of the hand object must be found. The MER is the smallest rectangle that will completely enclose the object. The coordinates of the MER will be used to determine which zone the user's hand is in.

MINIMUM ENCLOSING RECTANGLE (Region with Largest Area)

```
1- Find x-coordinates of the uppermost left corner of object
2- Find coordinates of the bottommost right corner of object
3- H = height of object by counting rows
4- W = width of object by counting columns
5- Draw a rectangle of height H and width W that starts from the
    uppermost coordinates to the bottommost coordinates.
```

4.8 Thinning

It was shown that the basic structure of the human hand is the same regardless of age or sex. The extra pixels of the segmented object comprise the thickness of the lines that form the image. It is desirable to remove these extra pixels. This is done by thinning. The problem lies in determining which pixels are redundant.

THIN IMAGE (Hand Object)

- 1- Delete pixel only if it has more than 1 and fewer than 7 neighbors.
- 2- Delete a pixel only if its counting index is 1.
- 3- Delete a pixel only if at least one of its neighbors in the 1,3, 5 direction is a background pixel.
- 4- Delete a pixel only if one of its neighbors in the 3,5, 7 direction is a background pixel.
- 5- Delete a pixel only if at least one of its neighbors in the 7,1, 3 direction is a background pixel.
- 6- Delete a pixel only if one of its neighbors in the 1,5, 7 direction is a background pixel.

The first rule ensures that the end points of the skeleton are not eroded and that pixels are stripped from the boundary of the region, not from the inside.

The second rule ensures that formerly connected regions do not become separate. A pixel with a counting index of 1 is connected to one other region only. The higher it's counting index, the more regions it is connected to.

The remaining rules are to remove the bias of the selection of pixels that are removed. By doing passes over the image in different directions, all parts of the object are considered equally.

4.9 Zone Determination

The zone of the segmented object is found with the help of the bounding box coordinates. If the coordinates lie within the ranges of the work area partitions, then it belongs to that particular zone.

ZONE DETERMINATION (MER of Hand Object, Dimensions of Image)

- 1- Get image dimensions in rows and columns.
- 2- $col2 = \text{floor}(\text{column}/2)$.
- 3- $row2 = \text{floor}(\text{row}/2)$.
- 4- $ypos = y\text{-coordinate of MER}$.
- 5- $xpos = x\text{-coordinate of MER}$.
- 6- $w = \text{width of MER}$.
- 7- $h = \text{height of MER}$.
- 8- Determine zone:


```

      IF ((ypos>=0 & ypos<=col) & (xpos>=0 & xpos<=row2))
        zone=3

      ELSEIF ((ypos>=0 & ypos<=col2) & (xpos>=row2 & xpos<=row))
        zone=1 (Bottom Left)
      
```

```

ELSEIF (ypos>=col2 & ypos<=col) & (xpos>=row2 & xpos<=row)
    zone=2 (Bottom Right)
ELSE
    msgbox(' Your hand is not in the correct zone','Zones');

END

```

4.10 Gesture Recognition

As was described earlier, the gestures are recognized by counting the number of fingertips, and a finger is represented as a line of particular length. Given below is the pseudo code for finding the fingertips.

4.10.1 Determination of Finger End Points

The end points of the thinned binary image are found and marked. An endpoint is a foreground pixel that has exactly one foreground neighbor. The endpoints are subsequently counted to determine how many fingers have been counted. The same is repeated for all other gestures with the only difference being the number of fingers found.

DETERMINE ENDPOINTS (Thinned Object)

- 1- Establish a lookup table that is persistent.
- 2- Find the endpoints using the lookup table.

4.10.2 Calculation of Finger Length

The length of the finger should be long, so all other lines in an image that may represent knuckles or some other small object should be eliminated. Skeletonization reduces an object to a set of connected lines. The lined image must first be decomposed into a set of disconnected lines before the length of each line can be calculated. This is done by deleting each pixel that has three or more neighbors.

DETERMINE FINGER LENGTH (Thinned Image)

Start Loop through Image

- 1- For each foreground pixel, count the number of foreground neighbors it has in each direction
- 2- If the number of neighbors is equal to or exceeds 3, set that pixel to 0 (background).

End Loop

In order to be accepted as a finger, the length of each disconnected line is calculated and compared against a threshold. If the length is equal to or exceeds the threshold, it is counted as a finger otherwise it is not.

CHAPTER 5

Results and Conclusion

5. Results and Conclusion

After analyzing the problem definition, gathering requirements, proposing a solution, and implementing the system at hand, the next step is to derive results that describe the performance of the system. This chapter describes the results of the application developed. Screenshots are provided to show the processing of the system.

5.1 Image Preprocessing

Every image needs to undergo certain preprocessing steps to make it suitable for the subsequent steps that include segmentation, feature extraction, and finally recognition. There was no need to enhance the brightness or contrast of the image nor was there any need to bring out details by application of smoothing and sharpening filters. There was however the need to apply some morphological filters that would clean the image and remove unwanted pixels that may be in the form of noise. Following are the results after applying the morphological operations.

5.1.1 Image Cleaning and Segmentation Results

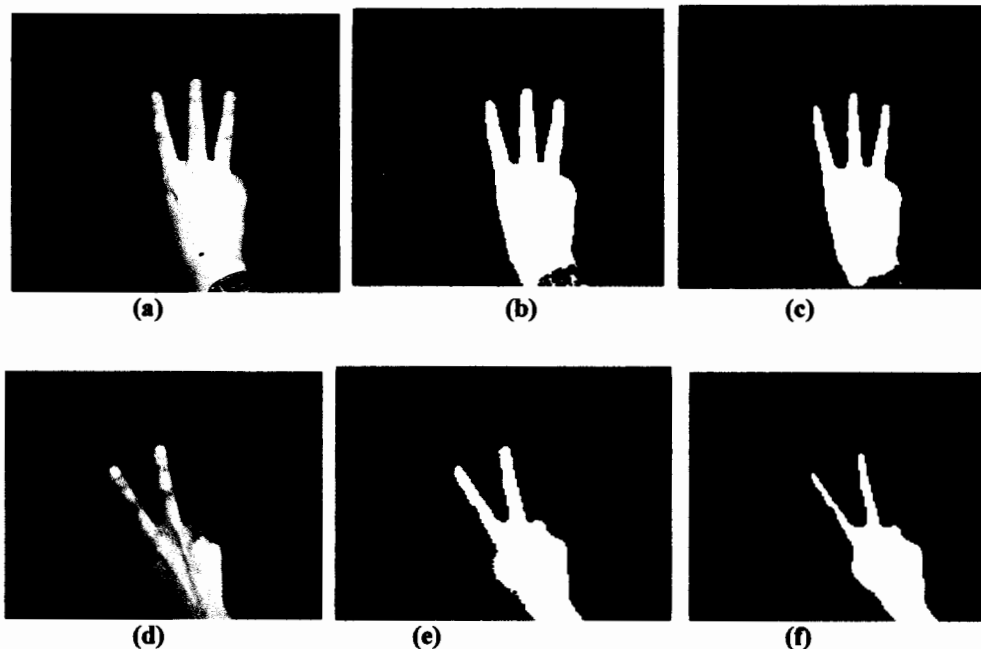


Figure 5.1 Cleaning and segmentation results: a) Original b) Binarized c) Cleaned d) Original e) Binarized f) Cleaned

As can be seen from Figure 5.1, the morphological filters have done a good job in removing redundant pixels. The date in Figure 5.1a has been completely removed and the watch dial of the wristwatch has been almost entirely eliminated. In the second sample of Figure 5.1d, the outer contours of the fingers and hand are cleaner and smoother.

5.2 Thinning

Thinning or skeletonization is the process by which an image is reduced to lines one pixel in width. By reducing a hand image to a bare skeleton, recognition of fingers can be realized.

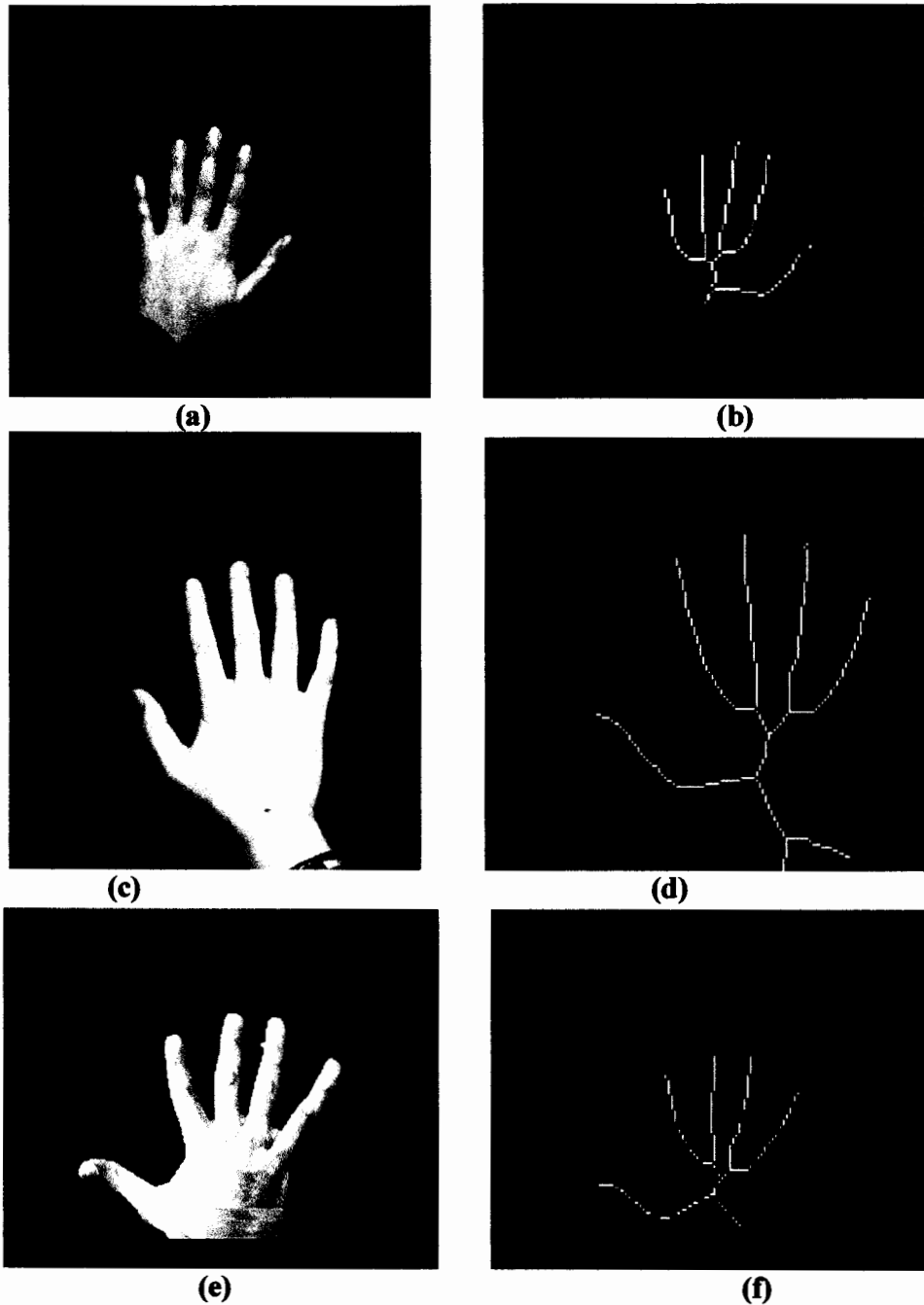


Figure 5.2 Thinned Images: Figures a, c, e are the original. Figures b, d, f are the thinned images

The hands in Figure 5.2 belong to three different people- one male and two females. Although humans physically differ a great deal from the outside, their underlying skeletal structure is fundamentally the same. Thinning reduces a potential hand to its essential skeletal structure. This structure essentially consists of an arm extending/dividing into a number of lines representing fingers. By capturing that structure and its features, the fingers can be counted with high accuracy.

5.3 Calculating Finger Length

Counting the endpoints of the lines in a thinned image is not sufficient to declare recognition of gesture. In this case, the length of the fingers was used as well to verify the number of extended fingers. Certain lines may appear due to noise or knuckles.

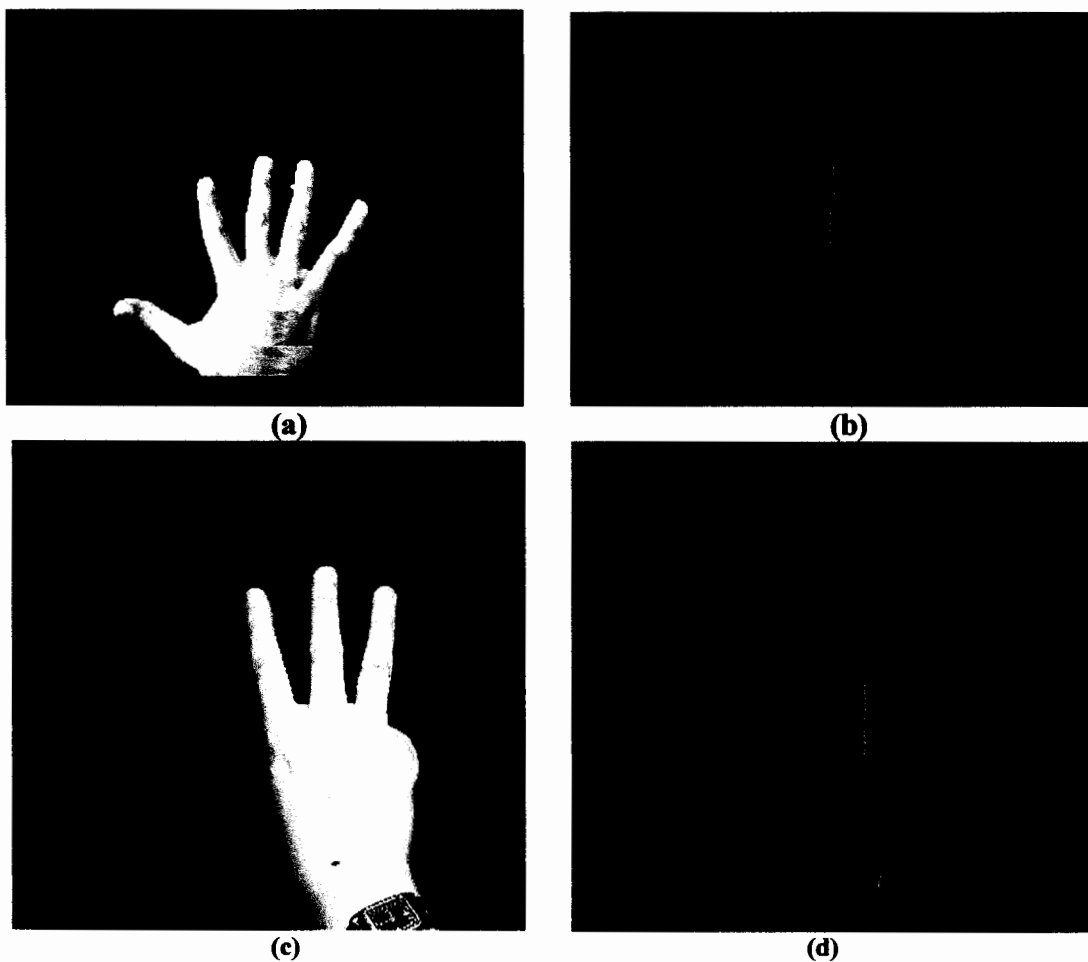


Figure 5.3 Counting fingers based on their length: a) Original b) Lengths computed c) Original d) Lengths computed

The fingers in Finger 5.3 appear as long extensions in the upwards direction. Before the length of the fingers can be calculated, the lines of the thinned image must be separated. This can be done by removing those skeletal pixels connected to three or more foreground pixels. Small lines that sprout from the wrist are not included in the counting of fingers. The different colors in Figure 5.3b and Figure 5.3d indicate different objects.

5.4 Thresholding Lengths

After the image has been thinned and the lines disconnected from one another, it must be decided whether the length of the line justifies a finger. A threshold is used to determine whether the length of a line is sufficient to depict a finger. If the length exceeds the threshold it is counted as a finger otherwise it is not.

5.5 Gesture Recognition Results

The GUI is shown in Figure 5.4. The user can first train him or herself and then operate the PowerPoint presentation.

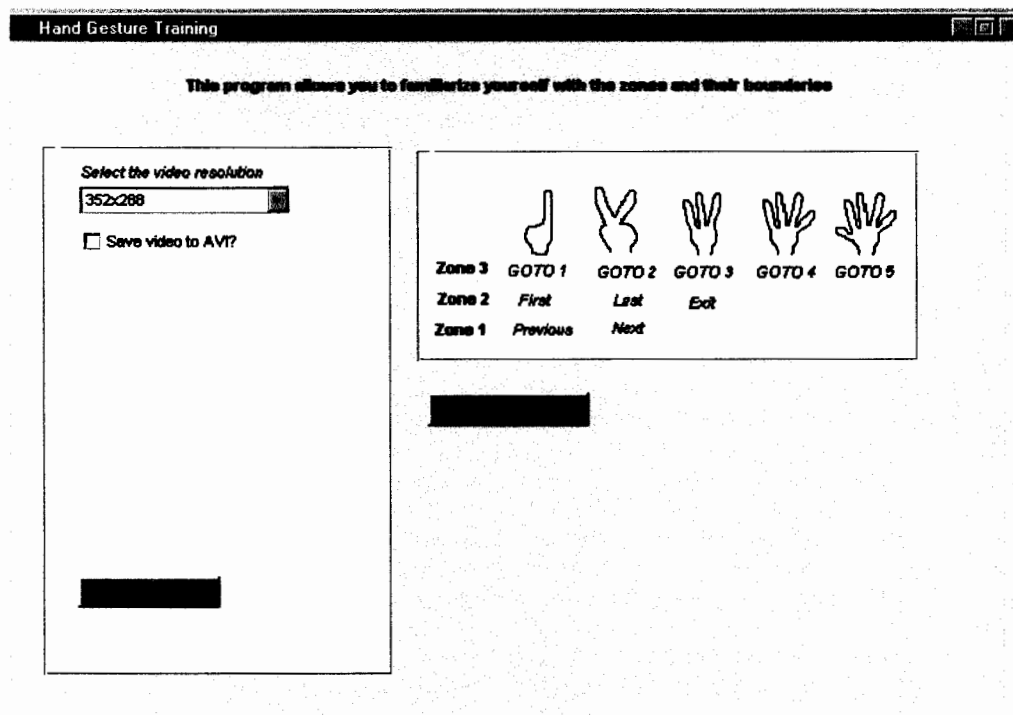


Figure 5.4 Application interface

Figure 5.5 shows the *preview* window in which the user can view his or her hand while the real time video is being captured. This allows the user to be able to coordinate his or her hand movement and positions within the zone.

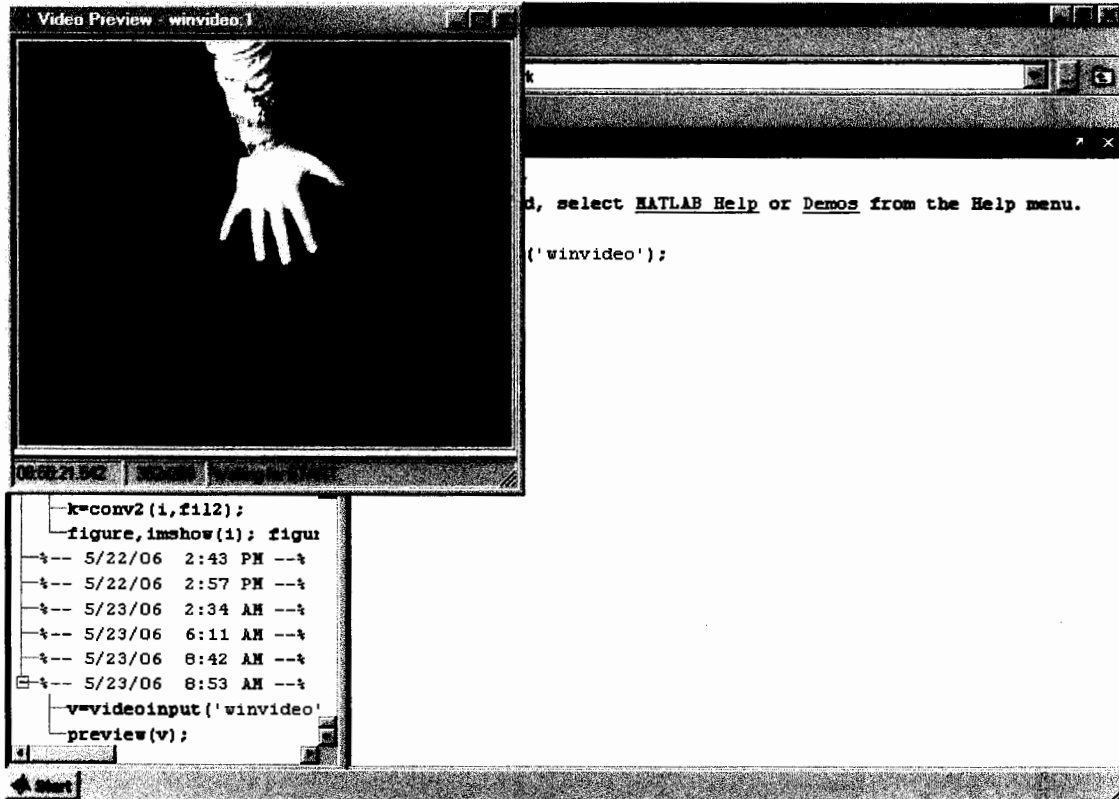


Figure 5.5 Live video previewing window

5.5.1 Recognition of Previous Slide

When the user enters their hand in the camera view in a particular zone, the system attempts to recognize the gesture. Shown in Figure 5.6 is the result for the Previous Slide hand gesture which consists of extending a single finger in zone 1.

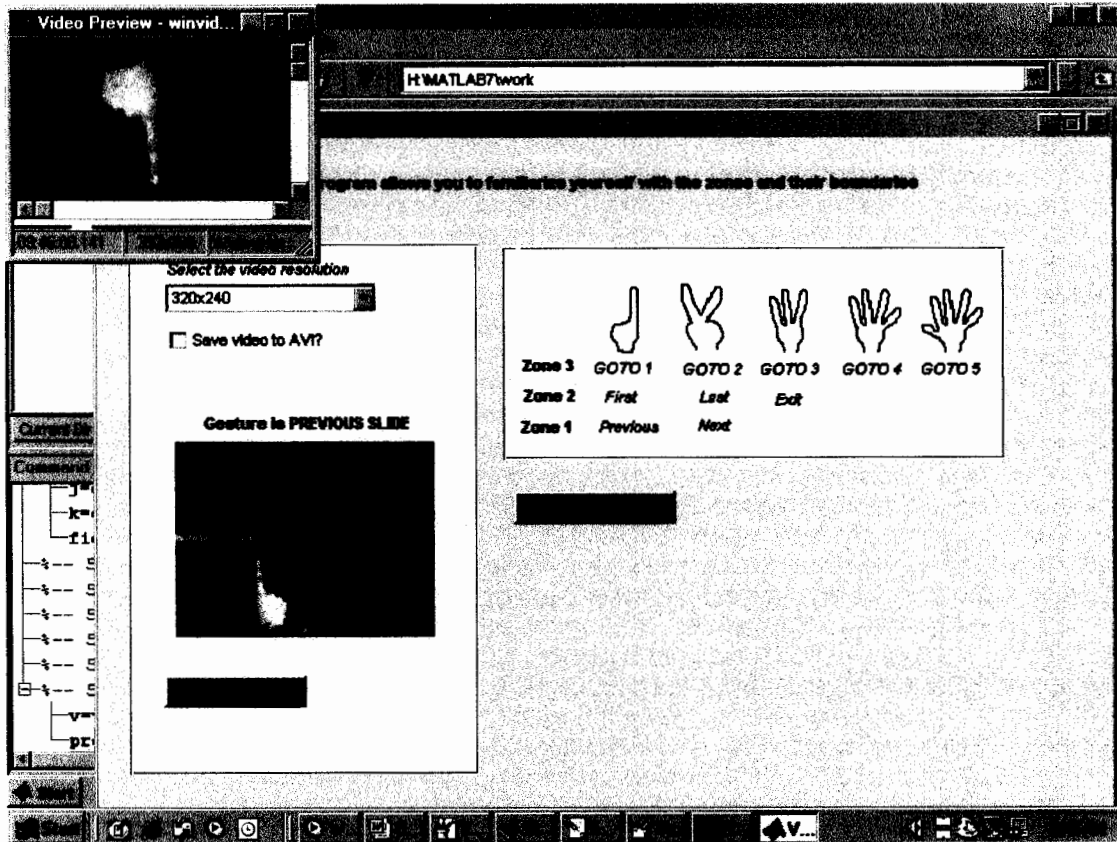


Figure 5.6 Recognition of Previous Slide gesture recognition

As can be seen in Figure 5.6, the user is very clearly informed of what gesture has been performed. The user can also see the division of the workspace into three distinct zones.

5.5.2 Recognition of Next Slide

The Next Slide gesture takes place in the same zone (zone 1) and consists of two extended fingers.

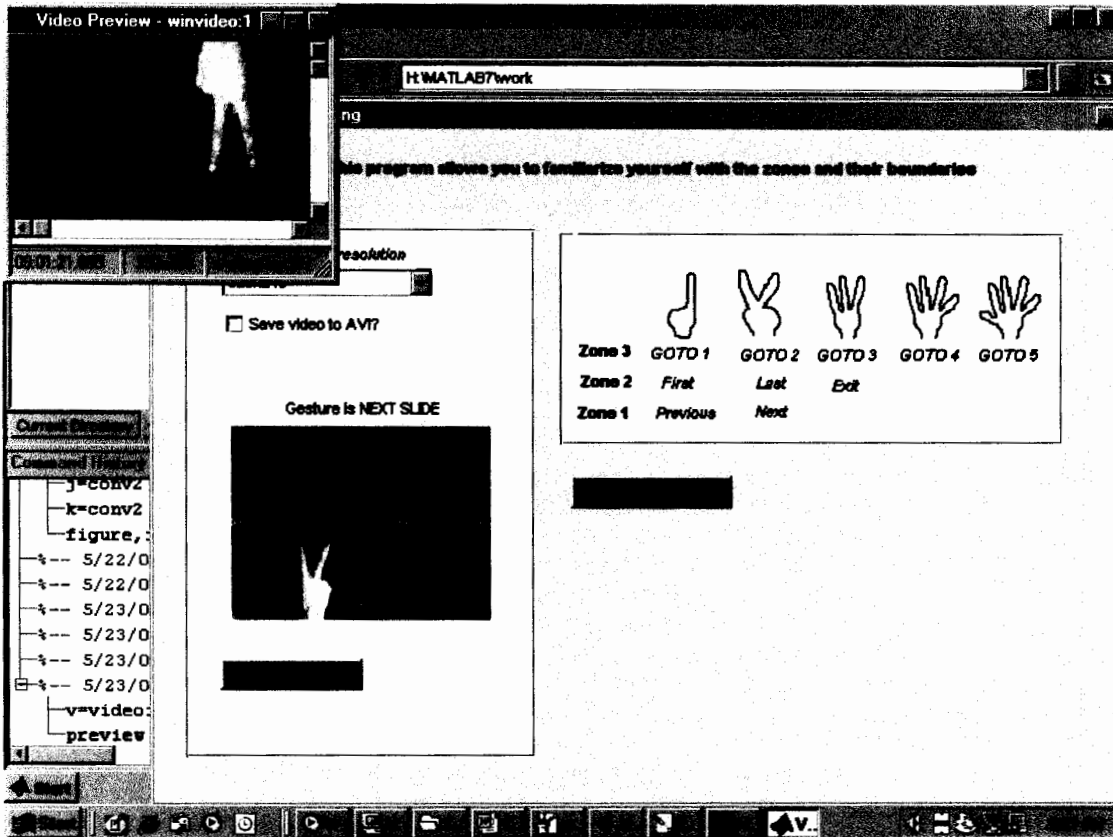


Figure 5.7 Recognition of Next Slide gesture recognition

It is important to note that it is not necessary to fully stretch the fingers apart, as it is not desirable to put the user in an uncomfortable position. It is necessary however to have some space between the fingers.

5.5.3 First Slide

The First Slide gesture is the same as the Previous Slide gesture- a single finger extended in zone 2, as shown in Figure 5.8.

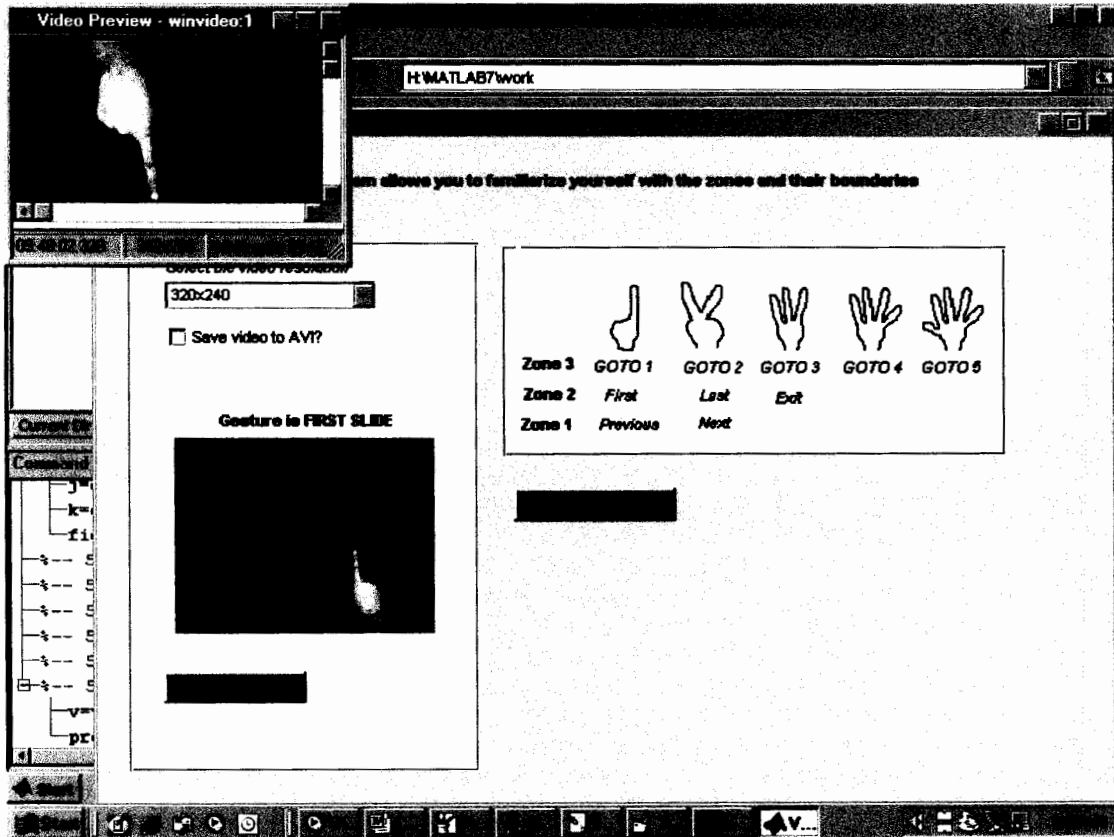


Figure 5.8 Recognition of First Slide gesture recognition

Notice that quite a bit of the user's arm is showing but this does not affect the recognition.

5.5.4 Last Slide

The gesture for Last Slide is the same as the gesture for the Next Slide i.e. extension of two fingers. The zones, however, are different therefore the gesture classified and command executed is different. As can be seen in Figure 5.9, the boundaries of the zone change color when the user moves their hand from one zone to another. This is to facilitate the user.

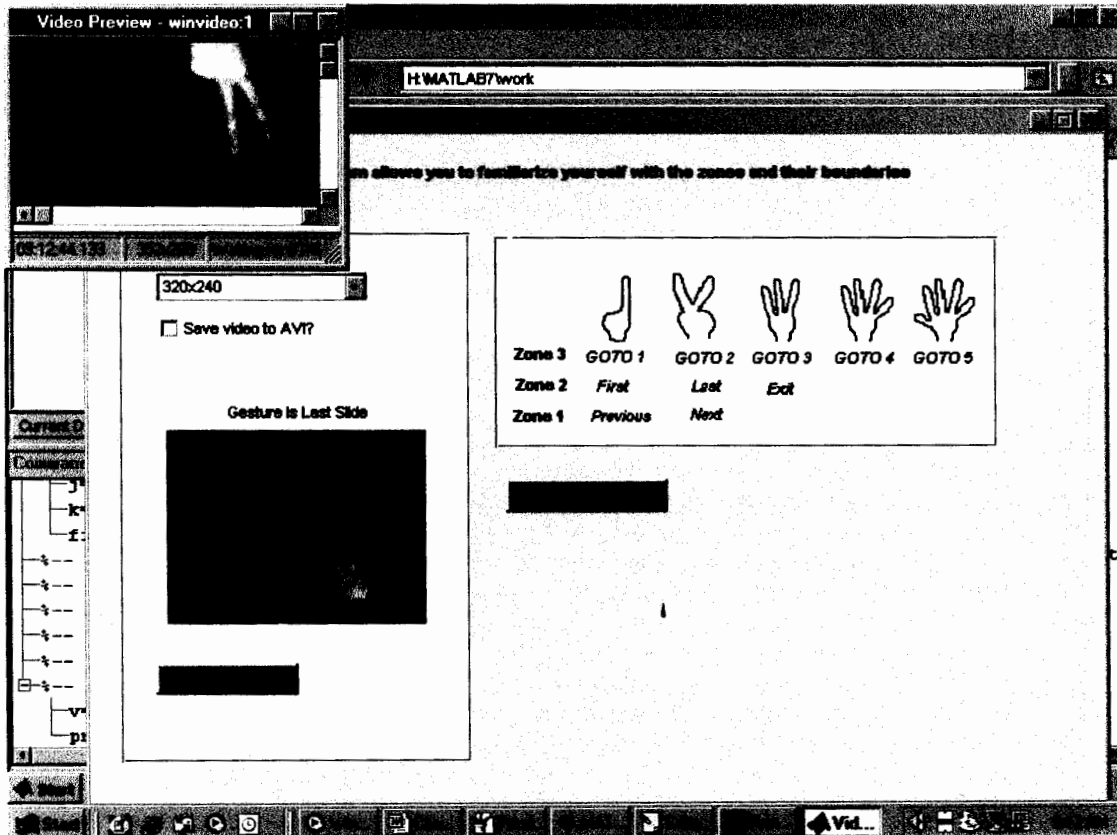


Figure 5.9 Recognition of Last Slide gesture recognition

Note that although the hand appears blurred in Figure 5.9 (this type of blurring is known as motion blur and it is incurred when the user moves his or her hand into or out of the zone) the gesture is still classified correctly. This implies that the image averaging and cleaning results are sufficient to guarantee correct classification.

5.5.5 GOTO Slide 1

By extending just the index finger (or any one finger) in zone three, the user can jump to the first slide, as shown in Figure 5.10.

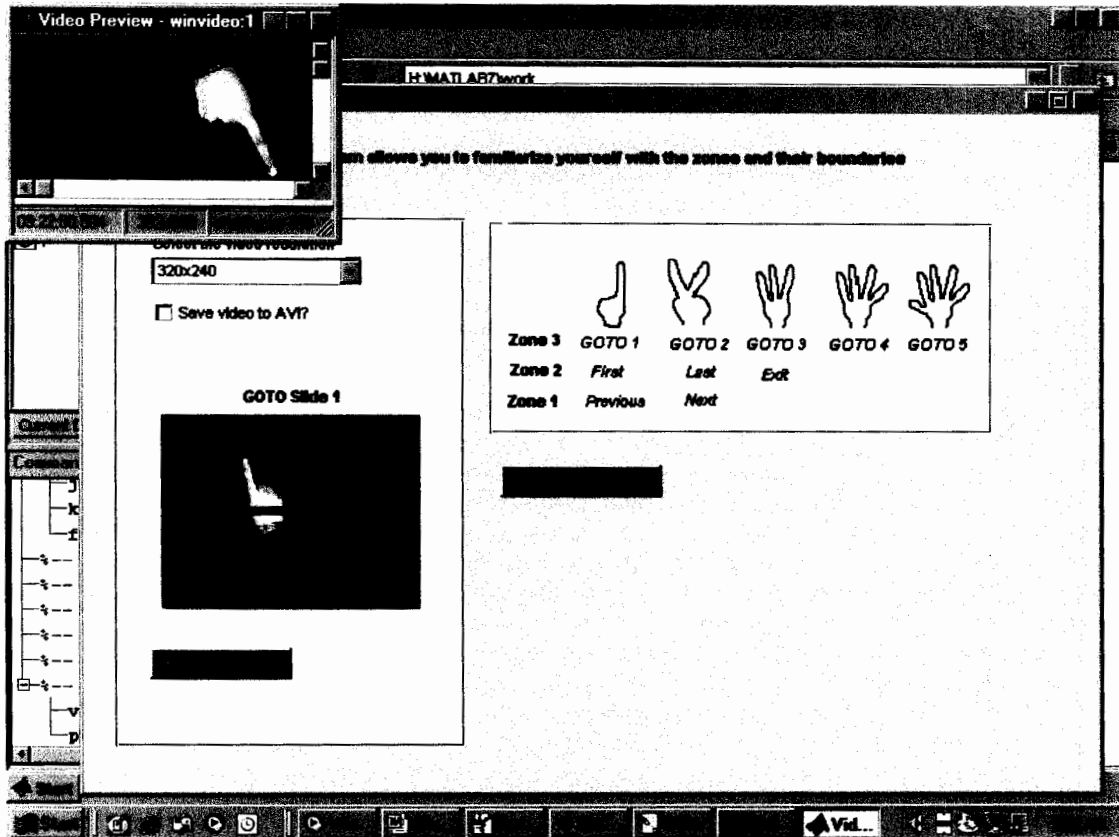


Figure 5.10 GOTO Slide 1 gesture recognition

It is clearly visible from Figure 5.10 that the user's hand is not contained entirely within the zone. However, the system still recognizes the gesture due to the thick boundaries that prevent overlapping of gestures. Again the zone boundaries change color.

5.5.6 GOTO Slide 2

This gesture is made by extending any two fingers- for example, the index and the middle finger.

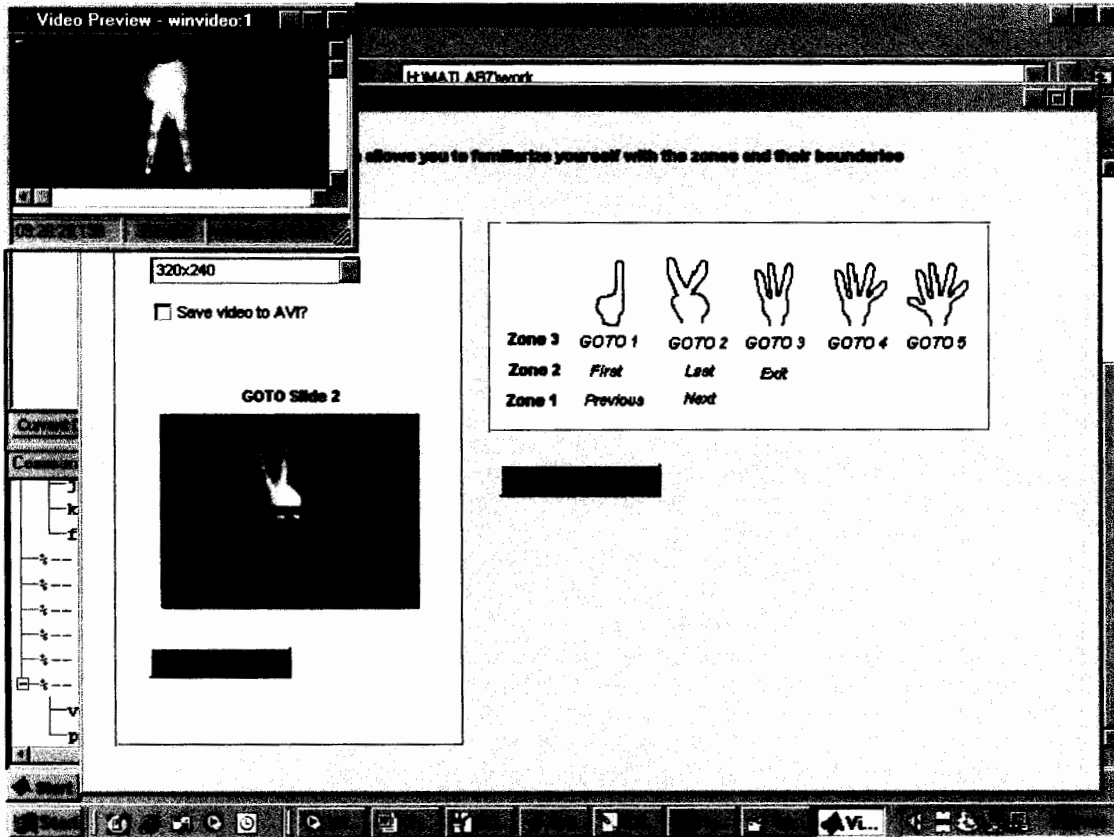


Figure 5.11 GOTO Slide 2 gesture recognition

Note that although there is motion blur, classification is still correct.

5.5.7 GOTO Slide 3

Results for this gesture are shown in Figure 5.12. By extending three fingers the user can jump to the third slide.

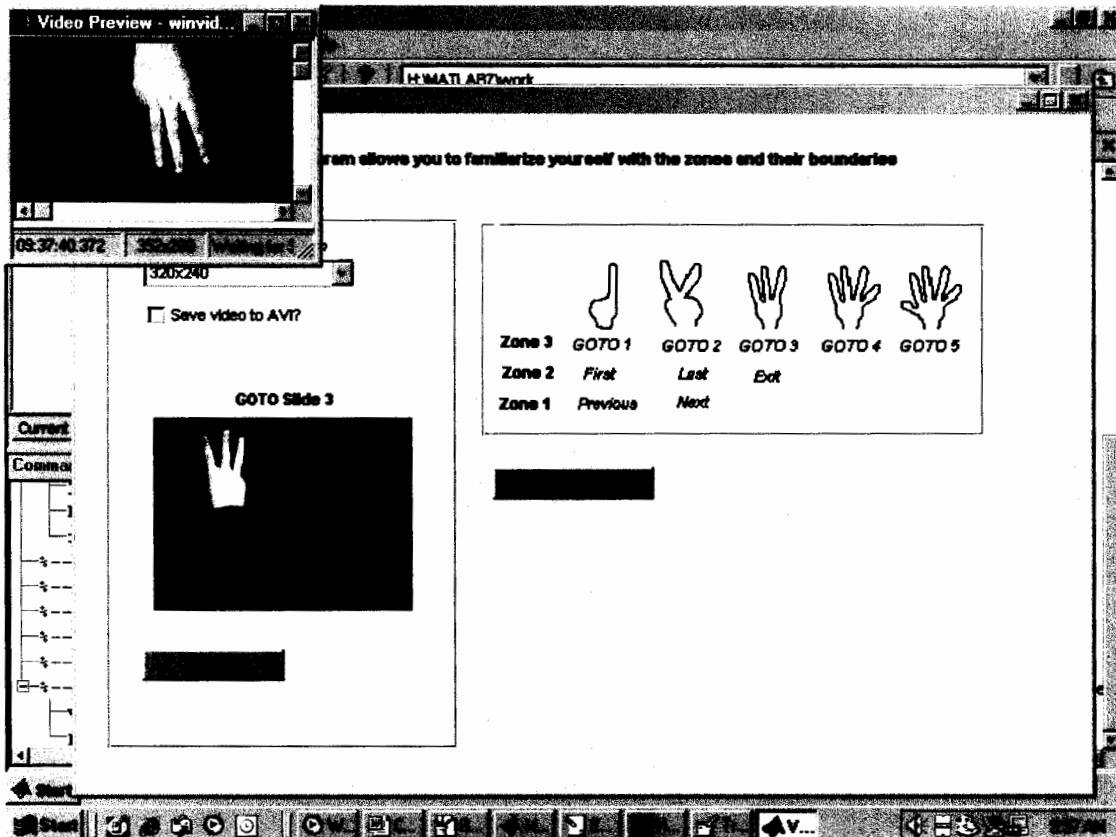


Figure 5.12 GOTO Slide 3 gesture recognition

5.5.8 GOTO Slide 4

Within the third zone, the user can place his or her hand anywhere i.e. to the left, right or center. The area in which recognition is taking place is larger.

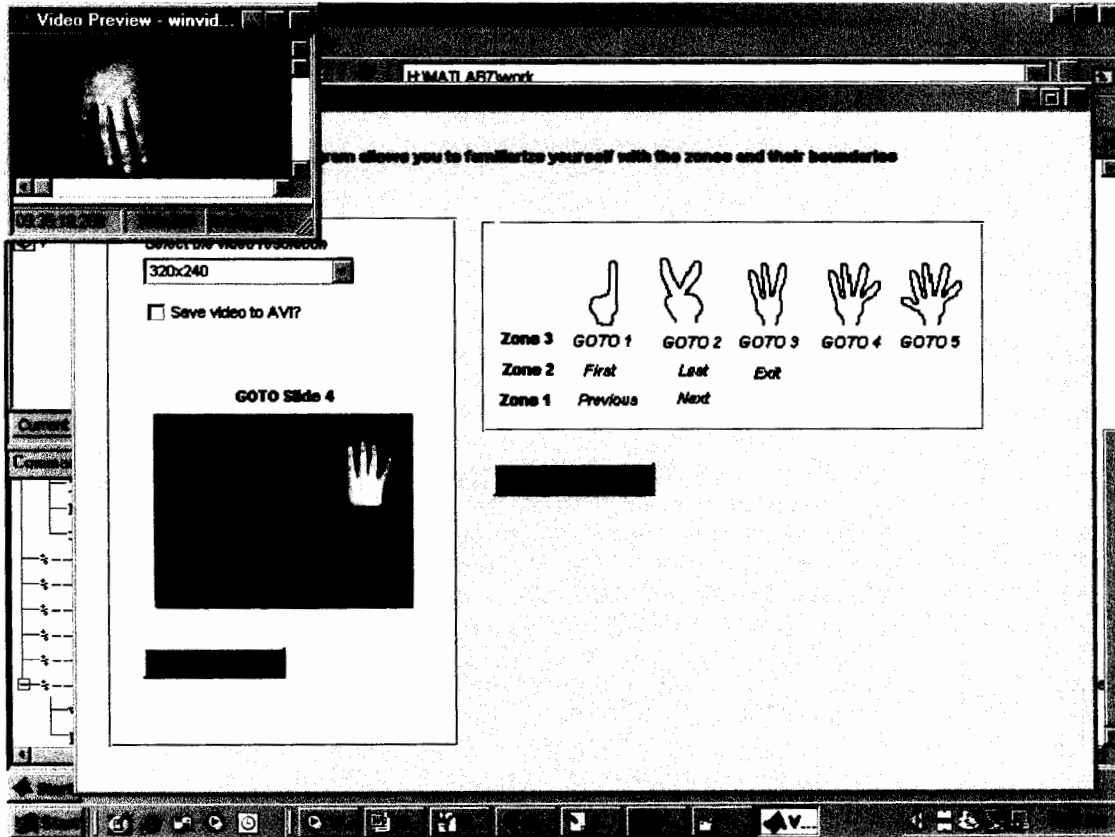


Figure 5.13 GOTO Slide 4 gesture recognition

Note that the user's fingers are relaxed as shown in Figure 5.13. They are spaced apart, but they are not deliberately stretched.

5.5.9 Exit Slideshow

By opening the hand, the user can exit the presentation. The PowerPoint presentation will end, as shown in Figure 5.14.

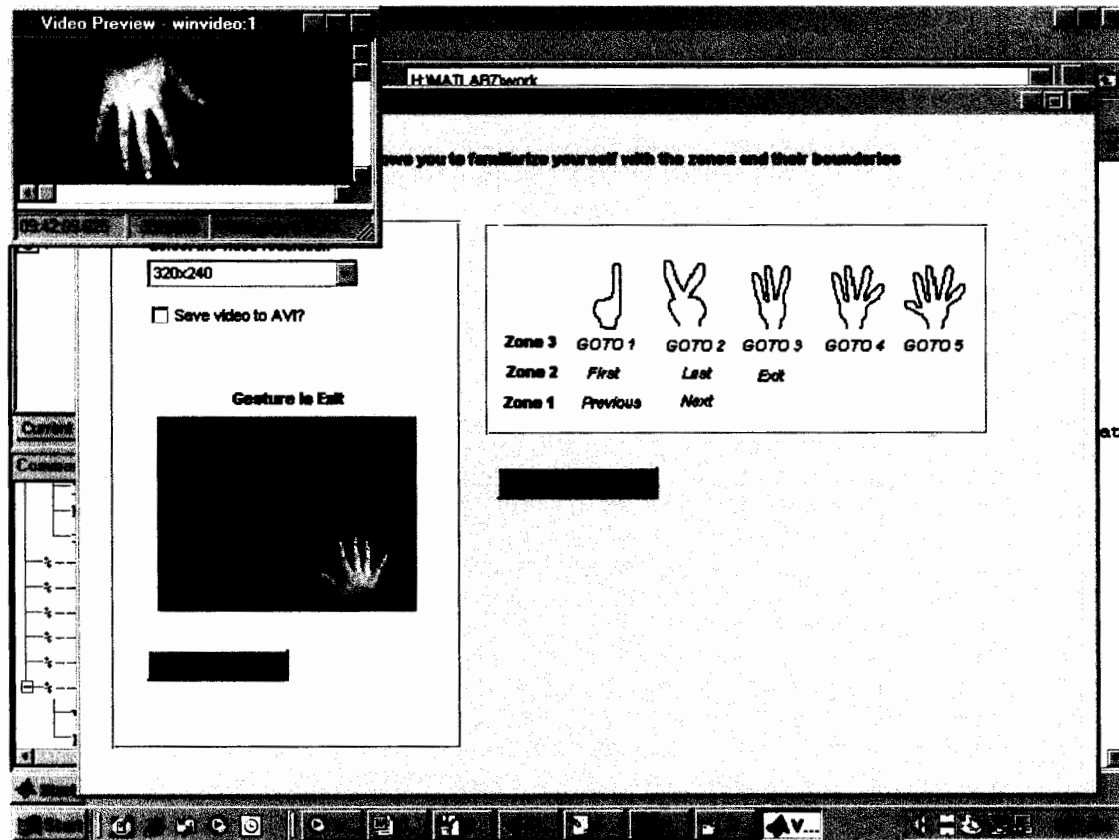


Figure 5.14 Exit Slideshow gesture recognition

5.5.10 Begin Presentation

Once the user feels that he or she has become familiar with the zones, their boundaries, and the gestures, he or she is ready to begin presenting. The user can select any Microsoft PowerPoint® presentation and control it using hand gestures as input. The mouse and keyboard, although fully functional, are not needed to navigate the slides as shown in Figure 5.15.

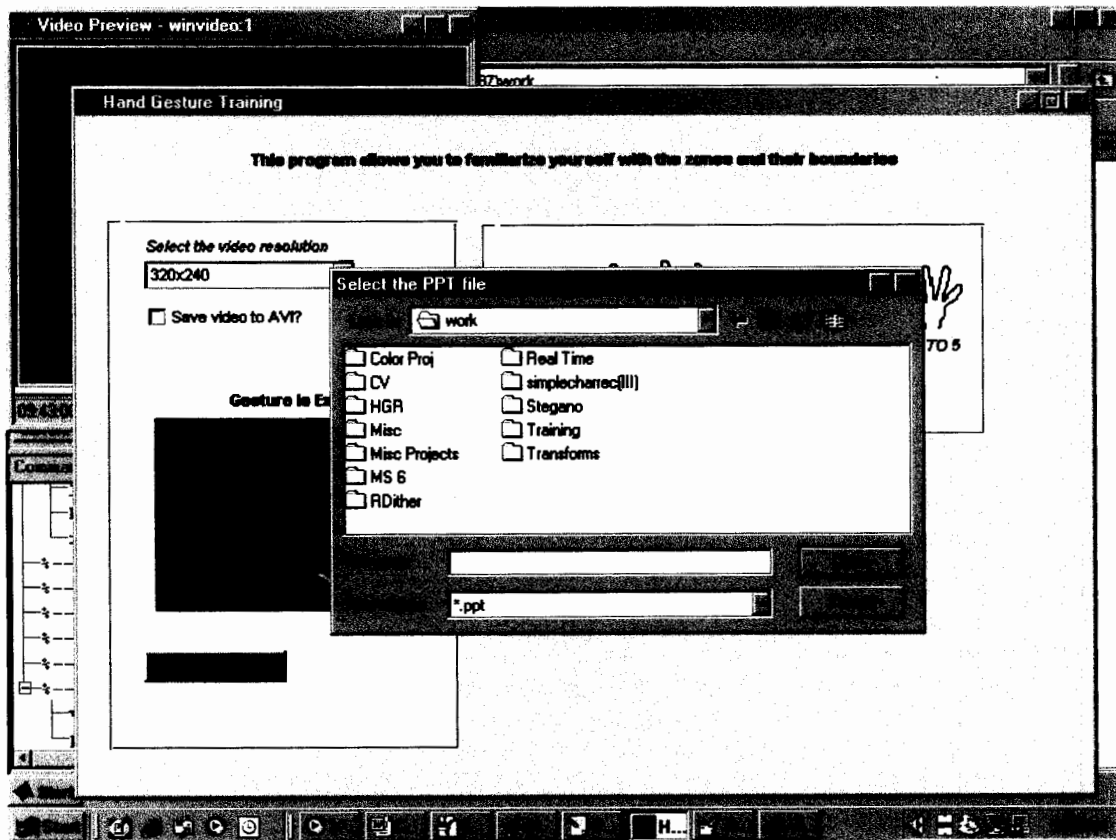


Figure 5.15 Begin Presentation-select *.ppt file

5.6 Analysis

A recognition system has been developed that allows a user to control a Microsoft PowerPoint® presentation via hand gestures. A workplace divided into three zones is designated as the place of recognition. The features of the application developed can be appreciated by comparing it with the features of other, similar applications.

Research Name	Method	# Zones	Recognition Rate	Miscellaneous
Vision Based Gesture Recognition	HMM	1	N/A	1 or 2 hands
Real Time Hand Gesture System Based on Evolutionary Search	FSM	8	95-100	-
Computer Vision Based Gesture Recognition for an Augmented Reality Interface	Counting Fingers	N/A	N/A	-
Accurate Recognition of Large Number of Hand Gestures	3-layer algorithm	1	89.6	-
CHARADE: Remote Control of Objects Using Free Hand Gestures	Driver algorithm	1	N/A	16 gestures, uses DataGlove
Interaction via Motion Observation	Image Subtraction	5	N/A	-

Table 5.1 Analysis of features of similar research work

The research work presented has several different features-

- It is a vision based system that does not employ the use of a DataGlove.
- It consists of three zones.
- It uses five gestures to execute nine different commands.
- The same gesture can be used in different zones to imply different meanings.
- It uses a method based on the skeleton of the human hand.
- The system can be used by left and right handed users.

Given in Table 5.2 are some objective results for the recognition rate of the developed system

Gesture	Male Success Rate %	Female Success Rate %
Previous	90~100	90~100
Next	90~100	90~100
Exit	80~100	80~100
GOTO 3	80~100	80~100
GOTO 4	80~100	80~100

Table 5.2 Objective results for recognition rate

The recognition results for gestures containing five fingers are lower because sometimes the thumb does not get counted as a finger. When three fingers are extended problems some time arise due to knuckles.

5.7 Conclusion

The diversity of computer users has increased greatly over the past several decades to include people of all age and caliber. This variety includes the very old and the very young. Also included are those with physical disabilities. The field of hand gesture recognition continues to gain popularity with time as it is becoming painfully clear that the conventional methods of data input restrict the user in many ways and are not a natural or convenient way for data input. Thus, in this research a human computer interface system has been developed that allows a user to give a presentation to an audience in real time using Microsoft PowerPoint®. The user can simultaneously control the machine and communicate with other machines or persons. Constraints include the preference for long sleeved, dark clothing and the orientation of the camera. Some of the distinctive features of this research are:

5.7.1 Elimination of Immersion Syndrome

As all gestures are to be recognized in the working zone, the user does not face the problems of being fully immersed by the system. This is because the vision system only attempts to recognize those gestures that take place within the workspace. As no other gesture is captured by the system, they are ignored. If several devices are set up, each with their own work area, or perhaps some sharing work areas, then the user can easily communicate with the devices without the confusion of which system the gesture is for.

5.7.2 Natural Gestures to Allow Ease of Use

Conventional interfaces including mice and keyboards pose problems for those individuals with physical disabilities. Motor-impaired users are often incapable of using their limbs efficiently and they may suffer from tremors. Persons of age find it difficult to hold the mouse for long periods of time or to curve their fingers on the keyboard. A computer vision based system that uses hand gestures as input is ideal especially when the gestures are such that they resemble natural, human like gestures. In this way, not only is it easy to execute the gesture but also it is easy to remember what the gesture is and what it is user for. The gestures were carefully chosen so that the commands they execute would appear to match the shape of the gesture.

5.7.3 Open to Left or Right Handed Users

The gesture lexicon and subsequent recognition is such that it does not matter whether the user is left or right handed. As recognition is based on the number of fingers extended, it does not matter what hand is used and the user can also alternate between both hands. Also, there is only the need for one hand at a time.

5.7.4 Same Gestures in Different Zones

In this research, the purpose of the zone was to facilitate the user, so that they can avoid the Immersion Syndrome and the system so that it does not attempt to recognize all gestures despite the intention. In other zone based systems, the authors have used different gestures for each zone. In this work however, as the zones are distinct, the gestures can be repeated i.e. the same gesture can be used to execute different commands. According to which zone the gesture is taking place, the corresponding action will be taken. In this way, the gesture lexicon is reduced and the user as well as the system does not have to remember so many gestures

5.7.5 Real Time Processing

As the development was done in MATLAB, the system is nearly real time. The speed of the application can be improved by implementing the system using Microsoft Visual C++. The system acquires images in real time, so this application can be an ideal commercial product for speakers who frequently give presentations in different conferences, seminars, universities etc.

5.8 Future Enhancements

Some suggestions for future enhancement include the description of new gestures to extend the functionality of the system. For example, mouse movement control could be incorporated into the system so that when the user moves their hand the mouse follows the path of their hand. This would mean the development of a tracking system. The system could be modified to work without the need for an explicitly defined zone or if the zone boundaries were relative to the hand position. Also, the gesture system could be extended into a sign language system for the deaf.

APPENDIX

A.1 Hardware and Software Installation Notes

In order to run the designed application, MATLAB 7.0 must be installed on the system. The processor speed should be at least 266 MHz. To install MATLAB 7.0, insert the CD in the CD-ROM or DVD-ROM. Setup will automatically run and begin to install the software on your system. The toolboxes which must be installed are the Image Processing Toolbox and the Image Acquisition Toolbox. Installation of the documentation can be skipped.

For image acquisition, an acquisition device must be attached to the computer and running. The device can be a professional grade image acquisition device, such as a frame grabber, or a generic Windows image acquisition device, such as a Webcam. The camera used during design was Creative PC-CAM 350. To use the Image Acquisition Toolbox, you must install and configure your image acquisition device. This includes

- a) Installing the frame grabber board in your computer.
- b) Installing any software drivers required by the device. These are supplied by the device vendor.
- c) Connecting a camera to a connector on the frame grabber board.
- d) Verifying that the camera is working properly by running the application software that came with the camera and viewing a live video stream.

Note: after installation of the acquisition, it must be checked whether MATLAB supports such a device. Using the `imacqwinf()` function, you get several pieces of information that the toolbox needs to uniquely identify the image acquisition device you want to access. One of these pieces of information is known as an Adaptor. An adaptor is the software that the toolbox uses to communicate with an image acquisition device via its device driver. The toolbox includes adaptors for certain vendors of image acquisition equipment and for particular classes of image acquisition devices.

A.2 Frame Averaging

```
% Getting an average image from the frames captured
for i=1:NumFrames

    % First frame and second frames
    img1=data(:,:,i);
    img2=data(:,:,i+1);

    % Rotate to adjust to camera positions
    img1=imrotate(img1,180);
    img2=imrotate(img2,180);
    img3=imlincomb(0.5,img1,0.5,img2);
    %Get the red planes for consecutive frames
    imgRed1=img1(:,:,1);
    imgRed2=img2(:,:,1);
    imgRed1=im2double(imgRed1);
    imgRed2=im2double(imgRed2);
    % Get average
    ravg=imlincomb(0.5,imgRed1,0.5,imgRed2);

end
```

A.3 Image Preprocessing

```
% Preprocessing the image
bw=im2bw(ravg,0.5);
bw0=bwmorph(bw,'clean');
bw1=bwmorph(bw0,'spur');
bw2=bwmorph(bw1,'majority');
bw3=bwmorph(bw2,'erode');
```

A.4 Region Labeling

```
% Getting the region properties-will get the filled image &
BoundingBox
[L N]=bwlabel(bw3);
stats=regionprops(L,'Area',
'FilledImage','BoundingBox','ConvexImage','Centroid');

areaa=[stats(i).Area];
if(areaa>=1500)

    [im_fill Bbox ch1]=RegionProp(bw3,i);
    ypos=Bbox(1);
    xpos=Bbox(2);
    h=Bbox(3);
    w=Bbox(4);

    B=bwmorph(im_fill,'clean');
```

```
B1=bwmorph(B,'spur');
B3=bwmorph(B1,'majority');
```

A.5 Thinning

```
% Applying thinning
Bthin=bwmorph(B3,'thin',Inf);
```

A.6 Zone Determination

```
col2=floor(col/2);
row2=floor(row/2);

% Top left Quadrant
if ((ypos>=0 & ypos<=col) & (xpos>=0 & xpos<=row2))
    zone=3

% Bottlom Left Quadrant
elseif ((ypos>=0 & ypos<=col2) & (xpos>=row2 & xpos<=row))
    zone=1

% Bottom Right Quadrant
elseif ((ypos>=col2 & ypos<=col) & (xpos>=row2 & xpos<=row))
    zone=2;
else
    msgbox(' Your hand is not in the correct zone','Zones');
end
```

A.7 Gesture Recognition

A.7.1 Determination of Finger End Points

```
% Finding the number of fingers extended by counting endpoints
[stats2 Nend bbox distance]=MarkEnds(RGB1,g);

if (Nend==1)% Have to check for previous gestures
    msg='P'
    set(R,'String',' ');
    set(R,'String','Gesture is PREVIOUS SLIDE');
    % Draw lines so the user can see where their hand is
    set(gcf,'CurrentAxes',Display);
    imshow(img3);
    %This will draw lines, coordinates are given as [y y],[x x]
    line([col2 col2],[row2 row], 'Color','y','LineWidth',5);
    line([1 col],[row2 row2], 'Color','y','LineWidth',5);
end
```

A.7.2 Calculation of Finger Length

```
[m n]=size(bthin);
bthin=im2uint8(bthin);
% bthin(m-35:m,1:n)=0;
for i=1:m
    for j=1:n
        count=0;
        if(bthin(i,j)==255)
            if(bthin(i-1,j)==255 )
                count=count+1;
            end

            if(bthin(i,j-1)==255)
                count=count+1;
            end

            if(bthin(i,j+1)==255)
                count=count+1;
            end

            if(bthin(i+1,j)==255)
                count=count+1;
            end

            if(bthin(i-1,j-1)==255)
                count=count+1;
            end

            if(bthin(i-1,j+1)==255)
                count=count+1;
            end

            if(bthin(i+1,j-1)==255 )
                count=count+1;
            end

            if(bthin(i+1,j+1)==255 )
                count=count+1;
            end

            if(count>=3)
                bthin(i,j)=0;
            end

        end
    end end
```

References

- [1] Sutherland, I.E. "SketchPad: A Man-Machine Graphical Communication System," in *AFIPS Spring Joint Computer Conference*. 1963. 23. pp. 329-346.
- [2] English, W.K., Engelbart, D.C., and Berman, M.L., "Display Selection Techniques for Text Manipulation." *IEEE Transactions on Human Factors in Electronics*, 1967.
- [3] Tolliver, B., *TVEdit*. Stanford Time Sharing Memo Report, Number, March, 1965.
- [4] Van Dam, A. and Rice, D.E., "On-line Text Editing: A Survey." *Computing Surveys*, 1971. 3(3): pp. 93-114
- [5] Bush, V., "As We May Think." *The Atlantic Monthly*, 1945. 176(July): pp. 101-108. Reprinted and discussed in *interactions*, 3(2), Mar 1996, pp. 35-67.
- [6] Van Dam, A., et al. "A Hypertext Editing System for the 360," in *Proceedings Conference in Computer Graphics*. 1969. University of Illinois.
- [7] Robertson, G., Newell, A., and Ramakrishna, K., *ZOG: A Man-Machine Communication Philosophy*. Carnegie Mellon University Technical Report, Report, Number, August, 1977.
- [8] Koved, L. and Shneiderman, B., "Embedded menus: Selecting items in context." *Communications of the ACM*, 1986. 4(29): pp. 312-318.
- [9] ACM SIGCHI Curricula for Human Computer Interaction by Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong, and Verplank Copyright © 1992, 1996 ACM SIGCHI
- [10] Brad A. Myers. "A Brief History of Human Computer Interaction Technology." *ACM interactions*. Vol. 5, no. 2, March, 1998. pp. 44-54.
- [11] Levy, S., *Hackers: Heroes of the Computer Revolution*. 1984, Garden City, NY: Anchor Press/Doubleday pg. 49, 132
- [12] Levinthal, C., "Molecular Model-Building by Computer." *Scientific American*, 1966. pp. 42-52.
- [13] Cadoz, "Les réalités virtuelles," Dominos, Flammarion, 1994.
- [14] D. McNeill, *Hand and Mind: What Gestures Reveal about Thought*. Chicago: University of Chicago Press, 1992.
- [15] McNeill, D. (1985). So you think gestures are nonverbal? *Psychological Review*. Vol 92(3) 350-371, Jul
- [16] C. Hummels and P. Stappers, "Meaningful gestures for human computer interaction: beyond hand gestures," Proc. Third International Conference on Automatic Face and Gesture Recognition, Nara, Japan, Apr. 1998.
- [17] Thomas Baudel and Michel Beaudouin-Lafon, "Charade: An Interaction Model Designed For Hand Gesture Input," *Communications of the ACM* 1993
- [18] Thomas Baudel, Michel Beaudouin-Lafon, Annelies Braffort, Daniel Teil, "An Interaction Model Designed For Hand Gesture Input"
- [19] J. Sturman, "Whole-hand Input," Ph.D. Thesis, MIT Media Laboratory, Cambridge, MA, February 1992.
- [20] Ryan Garver, "Vision Based Gesture Recogniton," 2003.
- [21] Juan Wachs, Helman Stern, Yael Edan, Michael Gillam, Craig Feied, Mark Smith & Jon Handler, "Real Time Hand Gesture System Based on Evolutionary Search," Genetic & Evolutionary Computation Conference, USA June 2005

[22] Moritz Störring, Thomas B. Moeslund, Yong Liu, and Erik Granum, "Computer Vision-Based Gesture Recognition For An Augmented Reality Interface," In 4th IASTED International Conference on Visualization, Imaging, and Image Processing, pages 766-771, Marbella, Spain, Sep 2004.

[23] Atid Shamaie Alistair Sutherland, "Accurate Recognition of Large Number of Hand Gestures," 2nd Iranian Conf. on Machine Vision and Image Processing, 2002

[24] Thomas Baudel and Michael Beaudouin-Lafon, "CHARADE: Remote Control of Objects Using Free Hand Gestures," Communications of the ACM, 1993, pp 28-35.

[25] M A Foyle and R J McCrindle, "Interaction via Motion Observation," Proc. 5th Intl Conf. Disability, Virtual Reality and Assoc. Tech., Oxford, UK, 2004