Cognitive Science and Technology

Prashan Premaratne

Human Computer Interaction Using Hand Gestures

Springer

Cognitive Science and Technology

Series Editor
DAVID M.W. POWERS, Information, Science & Technology (363),
Adelaide, South Australia, Australia

For further volumes:
http://www.springer.com/series/11554

Prashan Premaratne

Human Computer Interaction Using Hand Gestures

Springer

Prashan Premaratne
School of Elec., Comp. and Telecom. Eng.
The University of Wollongong
North Wollongong
New South Wales
Australia

ISSN 2195-3988 ISSN 2195-3996 (electronic)
ISBN 978-981-4585-68-2 ISBN 978-981-4585-69-9 (eBook)
DOI 10.1007/978-981-4585-69-9
Springer Singapore Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014932266

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

World would be an empty place for me without the laughter of my children whom I dedicate this book to: my daughter Lihini Savannah Mandakini and son Shaakya Gavin Kalpesh.

“Strength does not come from winning. Your struggles develop your strengths. When you go through hardships and decide not to surrender, that is strength”.

Arnold Schwarzenegger

Preface

People write books for different reasons. One is an opportunity to express the thoughts that would be valuable for contemporaries and the future generations. My quest for writing this has been to summarize my sustained research in this area for the past 10 years. A decade ago when I joined the School of Electrical, Computer and Telecommunications Engineering at the University of Wollongong as an academic, one of my goals was to develop a practical systems that would bring computer vision closer to one's day to day activities. I was inspired by my own research at the University of Adelaide as a Research Fellow working for CSSIP developing a system to automatically classify ships for Australian Naval Forces. Despite lacking support to garner any patents, I published the first developmental work in 2005. Then a refined version of this work which was published in IET Computer Vision received national and international media accolades. Soon after, the big players in the consumer electronics giants such as Samsung patented the technology for mobile phones and televisions. Few years later, this probably prompted Microsoft to develop Kinect camera which use hand gesture recognition for gaming. Today, hand gesture control is considered as viable option for gaming and non-critical input to computers and smart devices. Samsung smart television can be controlled using hand gestures. The consumer electronics control system that I developed made use of my best abilities in classification and system development. When I first developed a Static Hand Gesture Recognition system which garnered worldwide publicity, it was the first attempt by a researcher in the world to use Hu moments approach for hand gesture recognition. Over the years, I have incrementally developed the theory and technology to implement such a system in a Smartphone or many other devices with minimal power for hand gesture recognition.

There has been depth of research into Sign Language since 1990s. This research is also benefitted with any developments in Hand Gesture Recognition. I believe that, miniscule devices optimized for receiving and being able to recognize human gestures will play a vital role in future of computer human interactions. The challenge now is to develop the theory further with supplementing information not just vision in order to accurately detect and recognize hand gestures.

Every effort has been made to reflect most of the successful algorithms and approaches in hand gesture recognition in this book so that it would be versatile for any graduate student who is embarking on research into man-machine interaction. The readers are encouraged to use Matlab for developing fast routines to implement any type of hand gesture related processing. If Matlab would result in unexpectedly longer processing time, they are directed to use OpenCV (Open Computer Vision) platform that is fast becoming very popular with academics and developers which already has numerous computer vision related functional libraries. The youtube also presents a vast collection of evidence to the usefulness of OpenCV with thousands of routines already written in C++ to process images.

August 2013 Dr. Prashan Premaratne

Acknowledgement

In 2011, one of my very good friends, Prof. Sisil Kumarawadu of the University of Moratuwa, Sri Lanka invited me to co-author a book titled ‘Mathematical Modeling of UAV, Surface and Ground Vehicles’ for Springer Publications. Due to difficulty in obtaining sabbatical leave at that time, the project never went ahead. However, this opened up another opportunity when I was invited by Springer Publishers to author this book. This fulfils my lifelong dream to author a book that would benefit research students in their pursuit of man-machine interaction using computer vision. A very big thank goes to my dear friend Prof. Sisil Kumarawadu.

I should also acknowledge my wife Vimasha’s patience over the last 6 months when I was working at the University for almost 10 h a day for most of the week including weekends. My former Masters by Research student, Mr. ZhengMao Zou who undertook some preliminary work in human computer interaction, specifically in Support Vector Machines classification was helpful for writing the chapter on Classification. My current PhD student, Mr. Shuai Yang has made many significant contributions in bringing this final manuscript together.

Finally, this book would not have been completed on time if my sabbatical leave had not been approved. Many hurdles prevailed yet, Prof. John Patterson, my Senior Deputy Vice Chancellor at the University of Wollongong rightfully intervened with his usual charm. Finally, the obstacles were free and my dream was realized. Yet, my acknowledgement will not be complete if not for the prompt communications and great advice from Dr. Loyola D’Silva (Loy) and his editorial team at Springer Publications. Thank you all!

16 December, 2013 Dr. Prashan Premaratne

Contents

About the Author

Prashan Premaratne In 1994 Dr. Premaratne was awarded a full scholarship to pursue undergraduate studies under the John Crawford Scholarship Scheme at the University of Melbourne. He obtained his Bachelor of Engineering (Electrical and Electronics with Hons) from the University of Melbourne in 1997 and joined Fujitsu (Singapore) Pty. Ltd. as a Network Software Engineer at Network Software Development Division. In 1999 he was awarded a National University of Singapore Postgraduate Scholarship, National Science and Technology Board of Singapore award and a Motorola Research grant to pursue a Doctor of Philosophy degree in Electrical and Computer Engineering. He was awarded his PhD in 2002 for his work on “Blind Deconvolution for Image Restoration”.

Dr. Prashan Premaratne is a Senior Member of IEEE and has been a Guest Editor of Intelligent Computing Theories, LNCS7995, Lecture Notes in Computer Science, Springer, Bio-Inspired Computing and Applications 7th International Conference on Intelligent Computing, for Revised Papers Series: Lecture Notes in Computer Science, Vol. 6840; Guest Editor of International Journal of Information Technology Vol. 11 No. 12—Special issue on Security and Financial Series Analysis); Guest Editor—International Journal of Wavelets, Multiresolution and Information Processing (IJWMIP); and Guest Editor on a special issue in Neurocomputing Journal (Elsevier, 2006). He is also an ARC OzReader in Mathematics, Information and Communication. He was the Tutorial Chair of ICIC2008 conference in Shanghai, China and was the Program Committee Chair of ICIC2006 conference held in Kungming, China. He has also published over 75 IEE/IEEE refereed publications in signal and image processing and has been invited to chair sessions in major IEEE conferences such as, IASTED-EAS 2012, ICIAFS 2010, ICIAFS 2008, ISIMP2004, in Sri Lanka, Hong Kong and in China. He is also a reviewer for more than 8 IEEE accredited journals such as Electronics Letter, IET Image Processing, IET Computer Vision, International Journal of Vision and Image Understanding.

Chapter 1
Introduction

The first human interaction with a computer goes back to the days when punch cards were used to program computers. Then the output of the computers was also punched holes on cards. This was a very slow way of communication and the interaction was totally different to what we have today.

In 1970's with the invention of the computer mouse and the invention of the remote controller for the television, more robust and visual input output from a computer or an electronic system was available. This could be stated as the start of true 'human computer interaction'. The computer mouse and the keyboard together can be termed as the genuine contenders for the notion of human computer interaction. Since all the electronics devices whether it is a DVD player, Television or a digital camera, contain multiple microprocessors and microcontrollers. Therefore, they are in fact computers and the interaction we make with as in input and output can be loosely termed human computer interaction.

An acceptable controller that issues commands to a computer should be able to deliver the following:

- Reliability in issuing the intended command
- Reproducibility of any command
- Ease of use for the human in the long run
- Shorter learning curve for the first time user
- Durability as an instrument so that the investment is secured for a long time
- Very low maintenance

Figure 1.1 shows a diagram that depicts the human computer interaction using hand gestures. In this system, any human gesture is captured by a focussed camera which in turn is processed and identified by a system and is interpreted or actioned by an electronic device.

There are compelling arguments as to the reasons why large percentages of modern users still are content with the computer mouse and keyboard. The following dot points highlight some of the reasons:

- Older generation resists change
- Has been used of over 40 years with incremental change over the years
- No learning curve for the user

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_1,

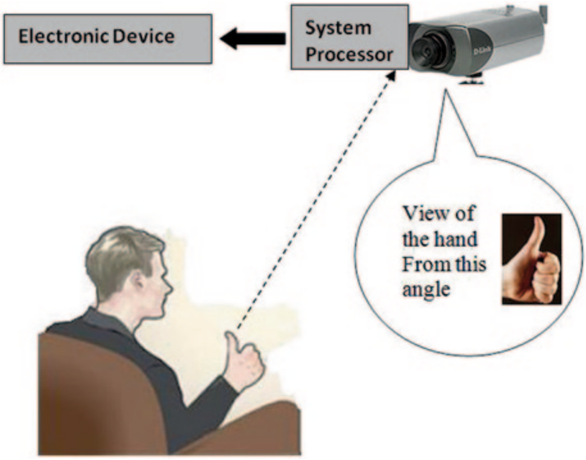


Fig. 1.1 Human computer interaction using hand gestures- how a human issues a command to a machine

- Very high reliability in communication with a computer or many other devices
- There is no radically different solution with better results
- Computer mouse and keyboards are 'good enough' for slow computer human communication

However, the computers and many other consumer electronics devices that we interact with today are fast changing with many applications developed not just for traditional sense of interaction. This in turn needs fast response from users and the interactions have been revolutionized recently. Some of these developments are:

- New generation wants to experiment with different modes of controllers
- Gaming is becoming very popular and traditional controllers are not fast enough for fast moving games
- Traditional controllers are either tethered or difficult to manoeuvre and restricts the user
- New applications need fast and more subtle inputs that traditional controllers can't offer.

The revolutions that changes the way users interact with computers are growing everyday with fast modes of communication. However, there are inherent limitations of the human body that would ultimately decide the best mode of interaction. For now, hand gestures are becoming a mode that is fast, natural and easy for the users to master yet providing a plethora of commands to control any device with subtle changes. The most important attributes of a new computer human communication interface are:

- High accuracy
- Ease of use without holding any equipments/instruments in hand
- Shorter user learning cycle
- Lower cost
- Offer capabilities that are not available with any traditional interface

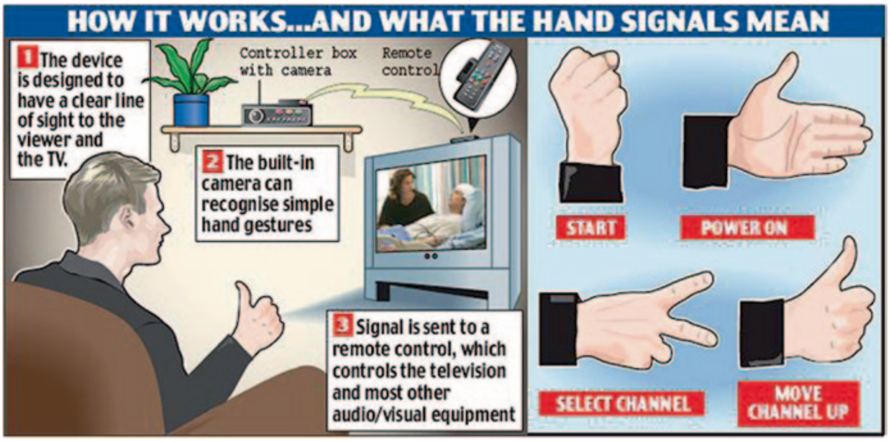

HOW IT WORKS...AND WHAT THE HAND SIGNALS MEAN
1 The device is designed to have a clear line of sight to the viewer and the TV.
Controller box with camera
Remote control
2 The built-in camera can recognise simple hand gestures
3 Signal is sent to a remote control, which controls the television and most other audio/visual equipment
START
POWER ON
SELECT CHANNEL
MOVE CHANNEL UP

Fig. 1.2 The description of the hand gesture recognition system developed by Dr. Prashan Premaratne as it appeared in Daily Mail UK (14 April 2007)

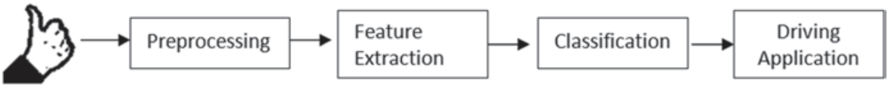


Fig. 1.3 Different stages of a hand gesture based control system

Hand gesture recognition using computers has been attempted for decades since early 1980's. Yet, a truly practical system based on computer vision that could operate in the living room with minimal computer power was developed only in 2005 at the University of Wollongong. This system as shown in this artistic impression of Fig. 1.2 was able to control up to 16 devices at anytime with 10 gestures with extremely high accuracy.

1.1 Chapter Description

This book is laid out in a logical way for a postgraduate student to appreciate the steps needed to realize a human computer interaction. Chapter 2 describes the historical developments that lead to the concept of human computer interaction starting with glove based system and how the developments in computer vision lead to hand markers to coloured gloves that resulted in quite accurate and reliable glove based systems today. Computer vision based hand gesture control systems conform to a multi-stage processing architecture as depicted in Fig. 1.3. The chapters from 3 onwards are laid out in that order. Chapter 3 describes the variety of preprocessing techniques employed by almost all computer vision based systems. It is unthinkable that a computer vision algorithm can be developed without preprocessing as the inherent raw image feed is typically characterized by noise, irregularities, out of focus, and non-uniformly lit images. Since the systems developed to date lack the human ingenuity, morphological filtering has become imperative for image understanding.

Chapter 4 will discuss various feature extraction techniques that have been attempted and succeeded in hand gesture classification. Every effort has been made to include all successful literature discussing variety of feature extraction techniques with good classification outcomes. Techniques which are unreliable have been dropped so that novice student researcher is not sent on a 'wild goose chase'.

Similar to the philosophy of Chaps. 4 and 5 discusses the classification algorithms which have been carefully chosen to be hand gesture specific. All evidence of successful classification outcomes of classification techniques have been discussed in depth with evidence from published research. Fundamental strategies have been discussed to impart an in-depth knowledge to the reader with the hope that new algorithms would be developed in future with the insight gained.

Chapter 6 discusses one of the single most prominent applications of hand gesture recognition: the sign language. It discusses the Auslan, the Australian sign language and ASL, the American Sign Language that is akin to Latin in European languages for formal sign languages in the world. Computer vision today draws much rigour for its algorithms due to vastness of sign languages in the world. The sheer number of gestures performed by hand and the human brains capacity to understand each and everyone of them will engage the field of computer vision for decades to come. The application of sign languages will determine the success of any preprocessing, feature extraction or classification approach and will always be ultimate application determines the success of human computer interaction.

The book will come to an end with 'Future Trends in Hand Gesture Recognition' in Chap. 7. This will also introduce many other hand gesture based consumer level appliances and applications that is already heralding this mode of interaction.

Chapter 2
Historical Development of Hand Gesture Recognition

The history of hand gesture recognition for computer control started with the invention of glove-based control interfaces. Researchers realized that gestures inspired by sign language can be used to offer simple commands for a computer interface. This gradually evolved with the development of much accurate accelerometers, infrared cameras and even fibreoptic bend-sensors (optical goniometers). Some of those developments in glove based systems eventually offered the ability to realize computer vision based recognition without any sensors attached to the glove. These are the coloured gloves or gloves that offer unique colours for finger tracking ability that would be discussed here on computer vision based gesture recognition. Over past 25 years, this evolution has resulted in many successful products that offer total wireless connection with least resistance to the wearer and will be discussed in Chap. 7. This chapter discusses the chronological order of some fundamental approaches that significantly contributed to the expansion of the knowledge of hand gesture recognition.

2.1 History of Data Glove

This book is never going to be complete without the historical development of hand gesture recognition based on computer vision without giving the due recognition for the evolution of hand gesture system based on data glove. Data glove in essence is a wired interface with certain tactile or other sensory units that were attached to the fingers or joints of the glove, worn by the user. The tactile switches, optical goniometer or resistance sensors which measure the bending of different joints offered crude measurements as to determine a hand was open or closed and some finger joints were straight or bent. These results were mapped to unique gestures and were interpreted by a computer. The advantage of such a simple device was that there was no requirement for any kind of pre-processing. With very limited processing power on computer back in 1990s, these systems showed great promise despite the limited manoeuvrability due to tethers that connected the glove to the computer. Figure 2.1

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_2,

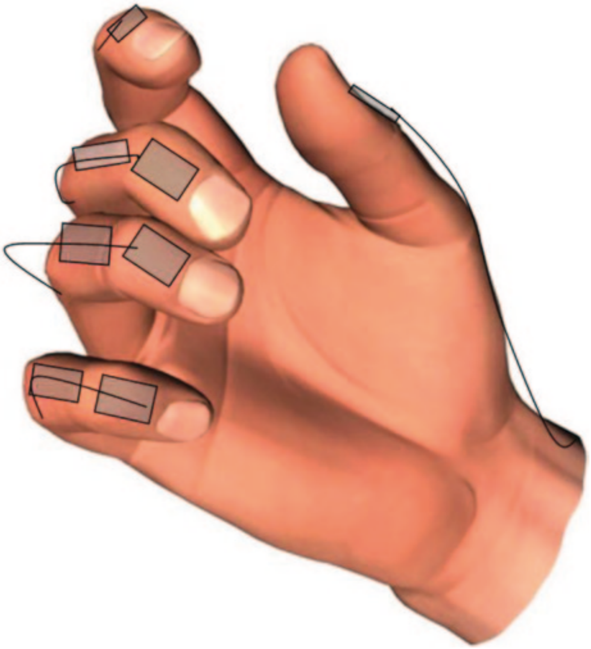


Fig. 2.1 Artistic impression of a sensor glove that places sensors on finger joints

shows an artistic impression of a data-glove or a sensor glove that strategically places variety of sensors to monitor the flexing of fingers to form different gestures.

Today, there exists gloves that are wireless and easy to wear unlike the ones we had 20 years ago. The following sections of this chapter will discuss the history of some of these devices and their performance scores in interpreting hand gestures.

By looking at the evolution of data gloves, there were two distinct categories emerged over the years.

1. Active data glove—consisted of few or variety of sensors on the glove to measure flexing of joints or acceleration and had a communication path to the host device using wired or wireless technology. These gloves are known to restrain the user of artistic ability.
2. Passive data glove—consisted only of markers or colours for finger detection by an external device such as a camera. The glove did not have any sensors onboard.

The first glove prototypes to emerge included the Sayre Glove, the Massachusetts Institute of Technology (MIT)-LED glove and the Digital Entry Data Glove [1]. The Sayre Glove which was developed in 1977 used flexible tubes with a light source at one end and a photocell at the other, which were mounted along each finger of the glove. Bending fingers resulted in decreasing the amount of light passed between the LED and the photodiode. The system thus detected the amount of finger bending using the voltage measured by a photodiode [2].

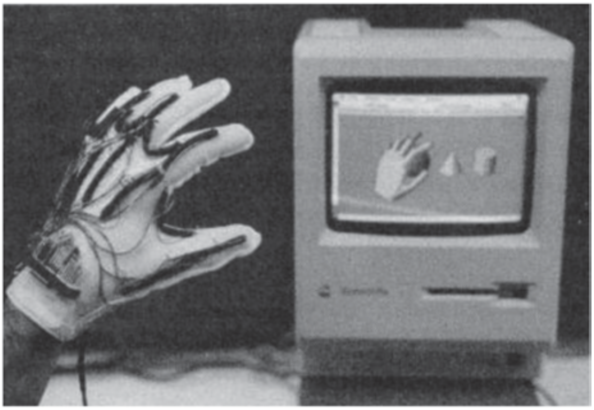


Fig. 2.2 The Z^{TM}Glove developed by Zimmerman [2]

The first glove to use multiple sensors was offered by the 'Digital Entry Data Glove' which was developed by Gary Grimes in 1983. It used different sensors mounted on a cloth. It consisted of touch or proximity sensors for determining whether the user's thumb was touching another part of the hand or fingers and four "knuckle-bend sensors" for measuring flexion of the joints in the thumb, index, and little finger. It also had two tilt sensors for measuring the tilt of the hand in the horizontal plane and two inertial sensors for measuring the twisting of the forearm and the flexing of the wrist [1]. This glove was intended for creating "alphanumeric characters" from hand positions. Hand gestures were recognized using hard-wired circuitry, which mapped 80 unique combinations of sensor readings to a subset of the 96 printable ASCII characters.

These gloves had limited accuracy and were tethered to computers using cumbersome wiring. They were meant for very specific applications and as proof of concept. They never received any attention beyond experimental tools and were never commercialized.

During 1980s, the sensor technology developed rapidly due partly to cold war fears and the natural expansion of industry in many European countries. These sensor technology paved way for rapid developments in computer technology and peripherals. Many leading research teams around the world started new computer peripherals with a market orientation using then recently developed new technical knowledge. The first commercially available Data Glove appeared in 1987 [1]. This was an improved version of the first DataGlove developed by Zimmerman in 1982 which is shown in Fig. 2.2 [3]. The technology was similar to the one used in Sayre Glove in 1977. However, the 1987 version carried fibre optics instead of light tubes and was equipped with 5–15 sensors increasing its ability to distinguish different gestures. The multiple sensors available on the DataGlove made it popular among researchers of different fields and number of similar devices was developed. Data Glove inspired development of Power Glove [4–6], which was commercialized by Mattel Intellivision as a control device for the Nintendo video game console in 1989. The Power Glove used resistive ink to measure the flexion of the finger joints

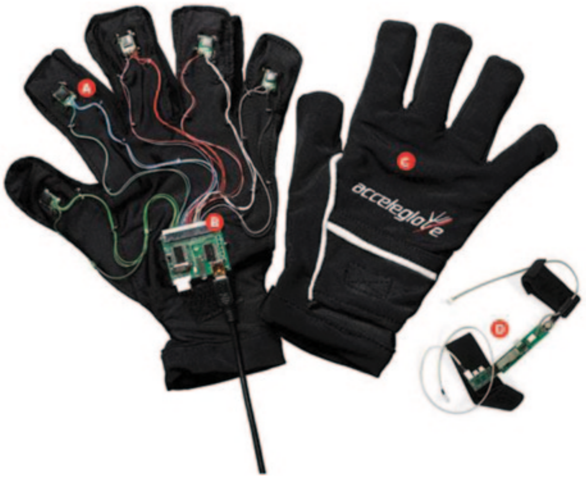


Fig. 2.3 MIT Acceleglove with its multiple sensors. [8]

[1]. There were other development such as Super Glove [4] developed Nissho Electronics in 1995 consisted of 10–16 sensors and used resistive ink printed on boards sewn on the glove cloth. An updated version of the Power Glove, the P5 Glove, was commercialized by Essential Reality, LLC, in 2002 [7].

2.2 What's Out There Today?

The following section details the state of the data glove today. A number of these are now commercially available for different types of human computer interaction (HCI). These data gloves are mainly aimed at researchers to develop sophisticated systems to make the HCI a reality.

2.2.1 MIT Data Glove

From its developments in early 1980s, MIT Data Glove has evolved dramatically offering different capabilities with different models. Currently developed under MIT spinoff company AnthroTronix, acceleGlove as shown in Fig. 2.3, is a user programmable glove that records hand and finger movements in 3D. The other models available from them include 5DT's Data Glove for virtual reality that cost between $ 1000–$ 5000. The company initially developed Data Gloves for US Defence for controlling robots. Their acceleGlove is also used in video games, sports training, or physical rehabilitation.

As shown in Fig. 2.3, an accelerometer rests just below each fingertip and on the back of the hand. The accelerometers can detect the three dimensional orientation of the fingers and palm with respect to the gravity when a gesture or any movement

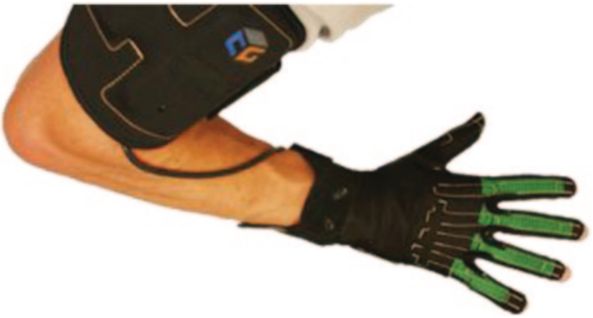


Fig. 2.4 CyberGlove III for motion capture

is made. The accuracy of these measurements is within a few degrees which allow programs to distinguish slight changes in hand position. The glove has openings for finger tips which would allow the user to type or write while wearing the glove.

2.2.2 CyberGlove III

The CyberGlove III (MoCap Glove) developed by CyberGlove Systems is a device that aims to record gestures accurately for motion capturing for movie making and graphic animation industry as shown in Fig. 2.4. The streamlined industrial designs that they developed allows for rigorous physical mobility in hand motion capturing for motion movies and graphic animation industry today [9]. The device also consists of Wi-Fi for data communication with a transmission range of 30 m. The unit contains 22 sensors and can operate for 2–3 h with the rechargeable battery onboard. The SD memory card offers motion recording option for motion capture animation purposes but the device is not aimed at computer or any other peripheral control.

2.2.3 CyberGlove II

CyberGlove has been developed to deliver many data inputs due to different flexing of joints motion from other areas of the hand. The 18-sensor data glove features two bend sensors on each finger, four abduction sensors, and sensors measuring thumb crossover, palm arch, wrist flexion, and wrist abduction. Different version of this glove that contains 22-sensors has three flexion sensors per finger, four abduction sensors, a palm-arch sensor, and sensors to measure wrist flexion and abduction. Each sensor is extremely thin and flexible making the sensors almost undetectable in the lightweight elastic glove. As shown in Fig. 2.5, one version of the glove offers open finger tips that would allow a user to type, write and grasp objects easily. The CyberGlove motion capture system has been used in many applications including digital prototype evaluation, virtual reality biomechanics, and animation.

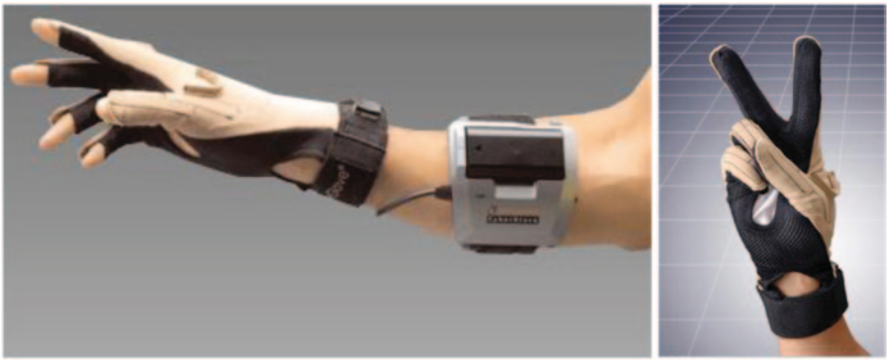


Fig. 2.5 CyberGlove II [10]

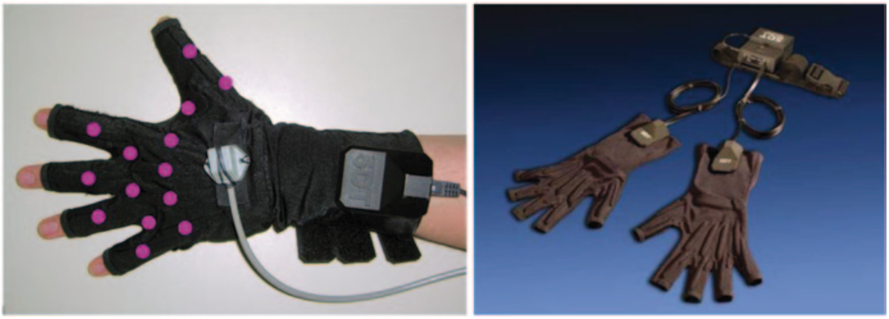


Fig. 2.6 5DT Motion Capture Glove and Sensor Glove Ultra. (Courtesy of [10]). (*left*) Current version, (*right*) early version

2.2.4 Fifth Dimension Sensor Glove Ultra

The 5DT Data Glove Ultra is another glove based gesture recognition device with very high precision flexor resolution. With its arrays of sensors, it provides 10-bit flexor resolution which is aimed at highly natural motion capture for movie industry [11]. The Sensor Glove Ultra is known to produce high data quality with low cross-correlation between different sensor metrics for realtime animations using Bluetooth data transfer. Figure 2.6 shows early and current version of Sensor Glove by Fifth Dimension (5D).

2.2.5 X-IST Data Glove

X-IST Data Glove from Inition [12], provides a motion capture solution with finger tip touch sensors that can be used for music related application. Since the unit is

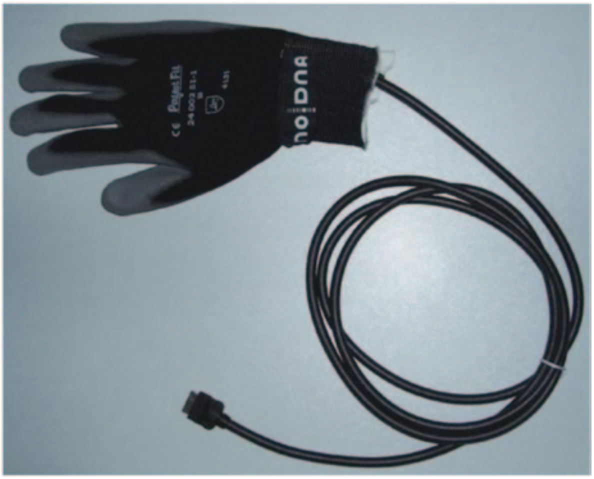


Fig. 2.7 X-IST Data Glove. [12]

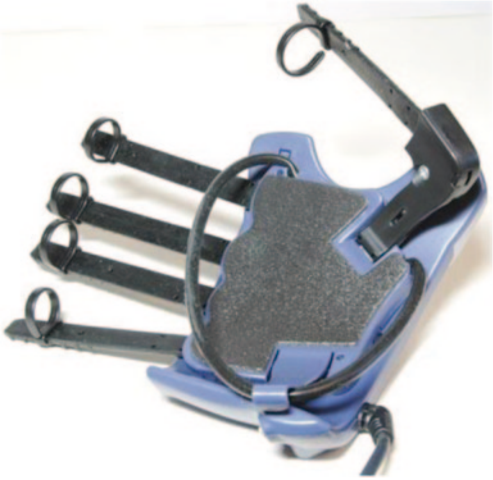


Fig. 2.8 P5 Glove—a cheaper alternative. [13]

wired to an usb interface, the user is not completely at rest. Each finger joint flex is measured along with the tilt of the hand. A cable connects user to the computer peripheral as shown in Fig. 2.7.

2.2.6 P5 Glove

P5 Glove has been developed by MindFlux as a way to provide cheaper alternative to many expensive wired gloves available in the market that can be used for gaming [13]. The P5, as shown in Fig. 2.8, incorporates a bend sensor and remote tracking technologies, which provides users intuitive interaction with 3D and virtual

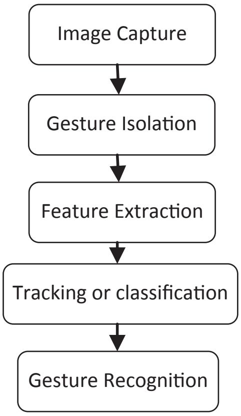


Fig. 2.9 Typical computer vision based gesture recognition approach

environments, such as games, websites and educational software. This is one of the very few technologies currently reaching the user as a means for controlling machines using peripherals other than, mouse, joystick or keyboard.

Today these gloves are no longer limited for use in computer human interactions as was illustrated above. Some of them are indeed for interacting with a computer for gaming and other natural like communication. Yet others are for 3D movie animation and some are for healthcare applications such as monitoring of vital signs to physiotherapy on injured or healing hands and fingers.

2.3 Vision Based Hand Gesture Recognition

Using cameras to recognize hand gestures started very early along with the development of the first wearable data gloves. There were many hurdles at that time in interpreting camera based gestures. Coupled with very low computing power available only on main frame computers, cameras offered very poor resolution along with color inconsistency. The theoretical developments that lead to identifying skin segmentation were in its infancy and were not widely recognized for its good performance that we see today. Despite these hurdles, the first computer vision gesture recognition system was reported in 1980s. Figure 2.9 shows a flow diagram of a typical gesture recognition strategy.

2.3.1 Hand Gesture Recognition Using Colored Gloves

The MIT-LED glove was developed at the MIT Media Laboratory in the early 1980s as part of a camera-based LED system to track body and limb position for real-time computer graphics animation [14]. A camera sitting in front of the user could capture number of LEDs as they were studded on the glove. This resulted in

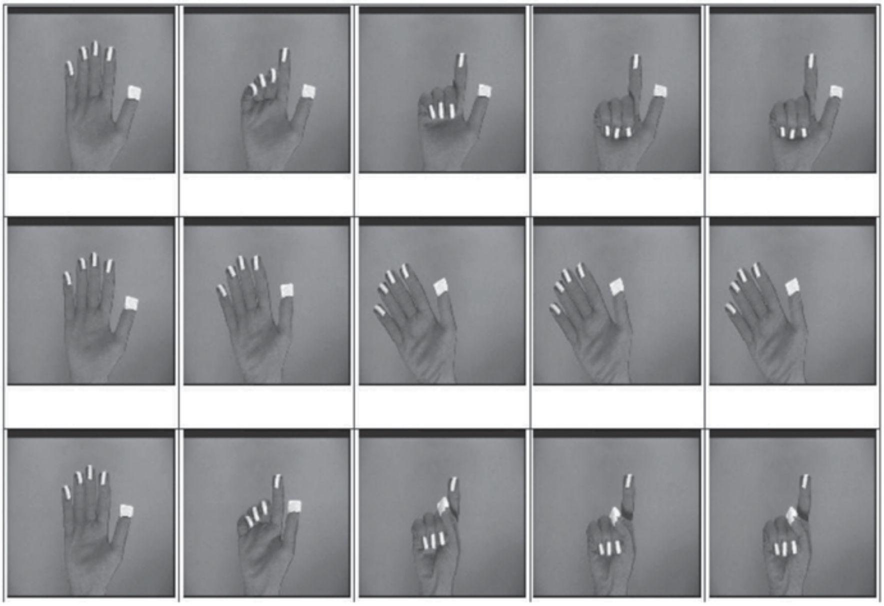


Fig. 2.10 Gesture tracking using coloured finger tips of the glove. (Courtesy of Davies et al. [15])

different illumination patterns for different gestures that could be interpreted by a computer. However, the performance was poor due to occlusions and the variations of any gesture performed by different users.

One of the first instances of gesture recognition using a glove with finger tip markers was reported by Davies et al. [15]. They used colored markers on fingertips as shown in Fig. 2.10, and a gray scale camera to track the finger tip movement and their relative organization to determine seven hand gestures. They managed to realize their system on image sampling speed of only 4 Hz on a SPARC-1 computer without any specialized hardware. Given the state of image capturing and computer processing power available in 1994, the system demonstrated the capability of computer vision approach as a viable contender against wired glove techniques for gesture recognition.

In 1996 Iwai et al. [16] proposed a colored glove technique in which 10 finger regions were identified. They used multiple colors to designate different parts of the finger and sections of the palm in order to avoid the occlusion problem many computer vision approaches suffered. In the occlusion problem, certain parts of the hand or fingers are covered by occlusion and the camera is unable to interpret the gesture accurately. When different color regions denoted different sections of the hand (fingers, palm), the system could rely on the color and the boundary to make informed decisions. They used a decision tree method to automatically recognize limited number of gestures.

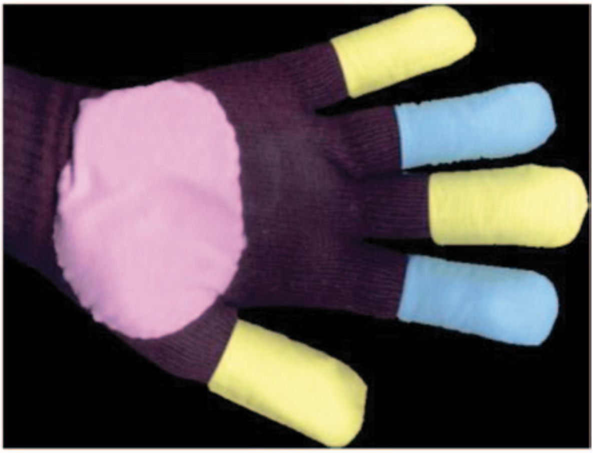


Fig. 2.11 A simple colored glove developed by Lamberti et al. [17]

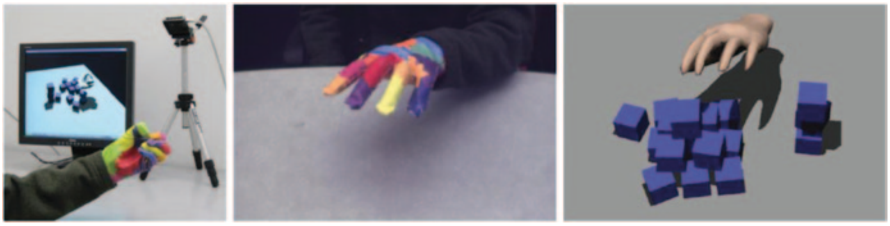


Fig. 2.12 A colored glove system for virtual reality applications. (Courtesy of [18])

In recent years, more and more research was concentrated on vision based hand gesture recognition. Compared to non-vision based recognition (data glove or electro-magnetic waves etc.), vision based recognition are more natural and comfortable [17], as it does not constrain the flexibility of hand movements. Based on the data glove and electromagnetic waves, a coloured glove has been developed Lamberti et al. [17] which is easy to wear without constraining the user. As shown if Fig. 2.11, the colored glove contains separate color to track the palm and fingers are marked with alternating colors. The aim of this approach has been to develop a very low cost approach against the dataglove with much more flexibility and very low computation requirements so that disabled users can make use of the technology in a classroom environment.

Articulated hand-tracking systems have been widely used in virtual reality but are rarely deployed in consumer applications due to their price and complexity. MIT researchers, R. Wang and J. Popovic recently developed a simple and inexpensive for 3D articulated user-input using the hands [18]. Their approach, as shown in Fig. 2.12, uses a single camera to track a hand, wearing an ordinary cloth glove that is imprinted with a custom pattern. The pattern is designed to simplify the pose estimation problem, allowing them to employ a nearest-neighbor approach to track hands in an interactive manner.

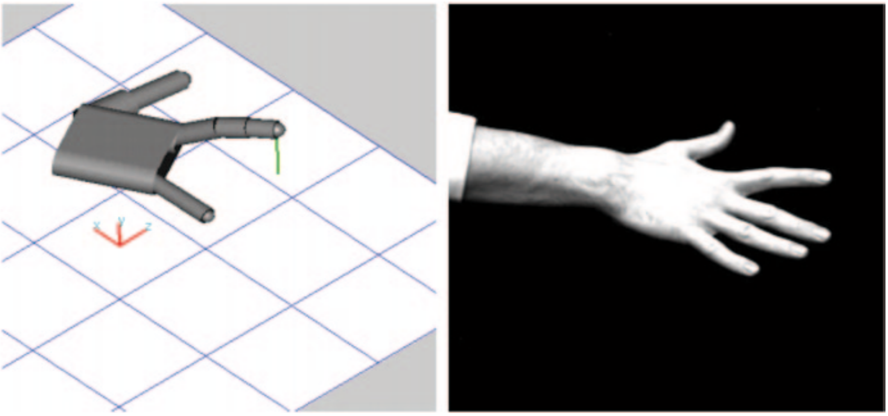


Fig. 2.13 The hand model (*left*) is derived using simple features extracted from two cameras for the first time in the world as reported by Rehg and Kanade. (Courtesy of [19])

2.3.2 Beyond Markers—Vision Based Systems

The first instance of a hand recognition system that totally relied on computer vision without markers was reported by Rhg and Kanade in 1993 [19]. Their system stood out from similar work of hand gesture recognition research then as the user was finally at ease without the requirement to wear any glove or colored markers for the first time. The system was called DigitEyes [19]. They demonstrated hand tracking at speeds of up to 10 Hz using line and point features extracted from gray scale images of ordinary unmarked hands. Most previous real-time visual 3D tracking work had addressed objects with 6 or 7 spatial degrees of freedom (DOF) by then [20, 21]. Rhg and Kanade presented tracking results for branched kinematic chains with as many as 27 DOF for human hand model which resulted in a highly articulated model. The success of this computer vision approach was mainly due to their ability to extract simple and useful features from human hand, which was in its infancy then. In order to avoid occlusion across complicated backgrounds, they used two cameras effectively as shown in Fig. 2.13.

1993 has been a year that contributed significantly to the research on gesture recognition based on computer vision. Each contribution made use of 10 Hz object tracking using a very common computer vision platform. Darrel and Pentland developed a method for learning, tracking, and recognizing human gestures using a view-based approach to model articulated objects [22]. In their approach, objects were represented using sets of view models and space-time patterns such as gestures were matched to stored gesture patterns using dynamic time warping.

In other words, because the parameter values underlying the object pose are associated with the set of correlation scores, they characterized a gesture by the pattern of view model scores directly. A gesture was then simply recognized by finding the closely matched space-time pattern of the model scores. This approach did not require transforming sets of view model scores to parameters and then matching in parameter space, but simply matching in the model-score space directly. Real-time performance was achieved by using special-purpose correlation hardware and view prediction to prune as much of the search space as possible. Furthermore, both view models and view predictions were learned from examples.

Because the parameter values underlying the object pose are associated with the set of correlation scores, we can characterize a gesture by the pattern of view model scores directly. Recognizing a gesture requires determining which stored gesture most closely matches the observed space-time pattern of model scores. In other words, rather than transforming sets of view model scores to parameters and then matching in parameter space, simple matching in the model-score space directly is sufficient.

The research carried out around the early 1990s always suffered from low camera resolution to inadequate computing power. The features that would be derived from a single camera not always resulted in successful outcomes for many research projects. Utsumi et al. countered this trend using a stereo rig in order to determine the Centre Of Gravity (COG) of the palm and to calculate the 3D positions of finger tips [23]. However, they failed to implement the system in realtime, yet they were successful in estimating the COG and fingertips location which were later used by other researchers as stepping stone to research on hand gesture recognition.

Freeman et al. in 1995 developed a new set of stable features using orientation histograms [24]. These offered a very stable and unique set of features which were very robust for hand gesture recognition. Figure 2.14 shows that gestures look different to the naked eye under different color schemes, yet orientation histograms are capable of providing features that are invariant to different illumination. Their research heralded a new era where stable features were the dominant factor in designing robust gesture recognition. The research conducted in the following years was keen to find unique but stable features so that the recognition performance would increase.

Dynamic gestures were difficult to analyse when limited computing power was available. However, steady development of computer processing power along with better camera technology incrementally moved research towards dynamic gesture recognition. One of the obstacles in dynamic gesture recognition was to device an effective way to identify information from moving video which contained moving gestures. In 1995 Cui, Swets and Wen demonstrated that this could be attempted by automatic visual partitioning [25]. They theorized that movement of the hand gesture (dynamic gesture) could be decomposed into two components: global and local motions. The global motion captured gross motion in terms of position and the local motion characterized deformation, orientation and gesture changes. This led to a development of three stage framework for spatiotemporal event recognition.

As illustrated in Fig. 2.15, the first stage acquires the sequence which involves motion detection to extract a moving image sequence and motion-based visual attention which identifies windows that include moving object. They mapped this temporal window to a standard temporal length to form ‘motion clip’. The speed information was available from the mapping performed in this stage. The second

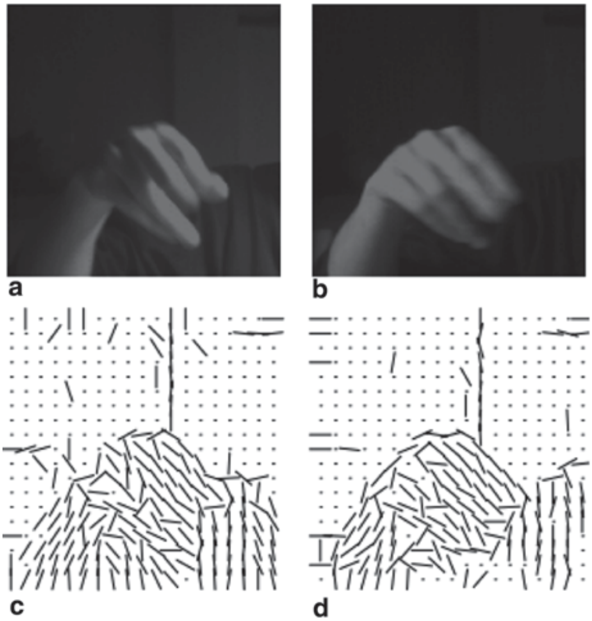


Fig. 2.14 Showing the robustness of local orientation to lighting changes. Pixel intensities are sensitive to lighting changes. **a** and **b** are same gestures under two different lighting conditons. **c** and **d** are the orientation maps of these (**a**) and (**b**) which are much more stable. (Courtesy of [24])

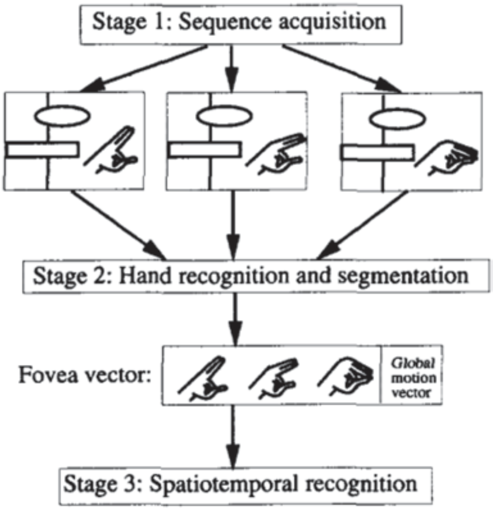

Stage 1: Sequence acquisition
Stage 2: Hand recognition and segmentation
Fovea vector:
Global motion vector
Stage 3: Spatiotemporal recognition

Fig. 2.15 Three-stage framework for spatiotemporal event recognition. [25]

stage was object recognition followed by segmentation. During segmentation, the object of interest was framed using a rectangular window from each image in the sequence and then is mapped to a fixed size fovea image. This step normalized the image in which the object of interest appears at a standard position with a standard size. They further simplified the processing by assuming a simple background which eliminated the need to deal with skin tone regions in their work. They used Most Expressive Features (MEF) and Most Discriminating Features (MDF) approach to find methods for good classification results. The reported best performance was 96 %.

Some of the many drawbacks using a single camera include occlusion and the difficulty in removing the cluttered background. Many researchers around 1998 started using multiple cameras to mitigate the effect of occlusion which resulted in many misclassifications. Such misclassifications often led to unfavourable reviews by the research community as poor results offered no solution for the advancement of the research. The leading researchers were very eager to bring the advantages of the Human Computer Interaction to the day to day life especially for control problems and gaming. Even though, the multiple cameras offered better results, having multiple cameras which were physically apart, created cluttered solution.

Shimade, Shirai, Kuno and Miura proposed the best solution that is achievable by a single camera in 1998 [26]. Their approach offered the best solution remove the occlusion problem using a single camera. They proposed a method to precisely estimate the pose (joint angles) of a moving human hand and check whether that was a valid gesture using 3D model. The monocular image sequence that they used did not contain any depth information. In their approach, given an initial rough shaped 3-D model, possible pose candidates are generated in a search space efficiently reduced using silhouette features and motion prediction. Then, they selected the candidates with high posterior probabilities to obtain the rough poses. The feature correspondence was established even under quick motion and self-occlusion.

Next, In order to refine both the 3-D shape model and the approximate pose under the depth ambiguity in monocular images, they proposed an ambiguity limitation method by loosely constraining the knowledge of the object represented as inequalities. The method calculated the probability distribution satisfying both the observation and the constraints. When multiple solutions were possible, they were preserved until a unique solution is determined. They produced experimental results to show that the depth ambiguity is incrementally reduced if the informative observations were obtained. Captured gestures and how they can be represented using a 'rough' pose is depicted in Fig. 2.16.

Multiple cameras solve the problem of occlusion when a gesture can be seen from many angles which are not possible to be detected by a single camera. However, as obvious, multiple cameras need calibration and other constraints to determine that multiple view belong to the same gesture at a given time. When two cameras are used, stereo vision strategies easily establishes the correspondence problem.

In 1998, Segen and Kumar of Bell Laboratories developed a vision based 3D hand interface for man machine interaction [27]. Their setup consisted of stereo setup to extract the *x*, *y*, *z* location of pointing fingers so that its location could be used to interact with a computer interface displayed in front of the user. The system used 60 frames per second and was very responsive. The system responded to 4 types of gestures: *Point*, *Reach*, *Click* and *Ground*. The system allowed them to use different combinations of those gestures to manipulate Virtual Reality-type games.

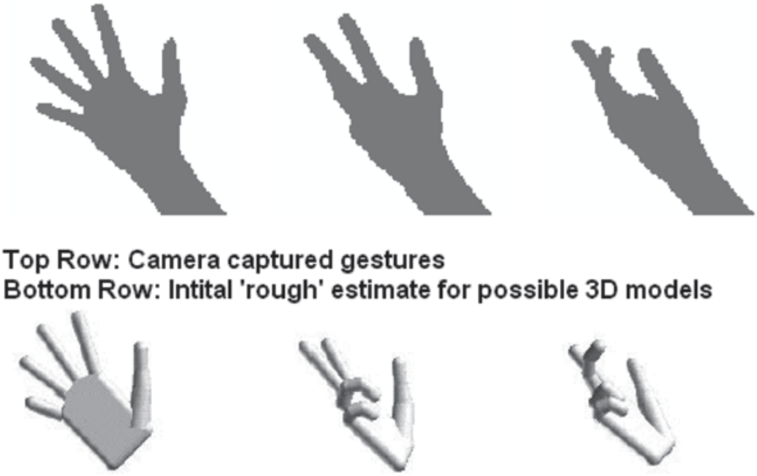

Top Row: Camera captured gestures
Bottom Row: Intital 'rough' estimate for possible 3D models

Fig. 2.16 Captured gestures and initial estimates of the possible 3D models. (Courtesy of [26])

For instance, *Click* gesture to fire a gun in 'doom' and 'Reach' gesture to open doors in virtual reality. Their fundamental gestures are shown in Fig. 2.17 and gesture associations with particular functions are shown in Fig. 2.18.

Segen and Kumar demonstrated that multiple applications can be run with few basic gestures with an appropriate computer interface. This lead to the development of many gestural interfaces especially, for control over the year.

Similar to the work carried out by Segen and Kumar, Utsumi and Ohya proposed a hand gesture tracking system using multiple cameras in 1999 [28]. The approach that they pursued tracked 3D position of the posture of the hand with multiple viewpoint images. They also used the multiple views to reduce self-occlusion and hand-hand occlusion as they were very prominent problems faced by sing camera gesture tracking. Each hand position was tracked with a *Kalman* filter and the motion vectors were updated with image features in selected images that did not include hand-hand occlusion. 3D hand postures were estimated with a small number of reliable image features. These features were extracted based on distance transformation, and they were robust against changes in hand shape and self-occlusion which resulted in a "best view" image which was selected for each hand for shape recognition. Fourier descriptors were used as features to describe postures and the system was proposed as a user interface device in virtual environments replacing the reliance of the glove type devices which had many disadvantages as outlined at the start of the chapter. The camera setup of this multi-camera system is shown in Fig. 2.19 where the left image shows a side view and the right image shows the front view.

The hardware technologies vastly improved since the year 2000. This dramatically increased the resolution of the camera and the interface. USB 2.0 replaced the slow-speed USB 1.0 and that alone resulted in the development of very high quality

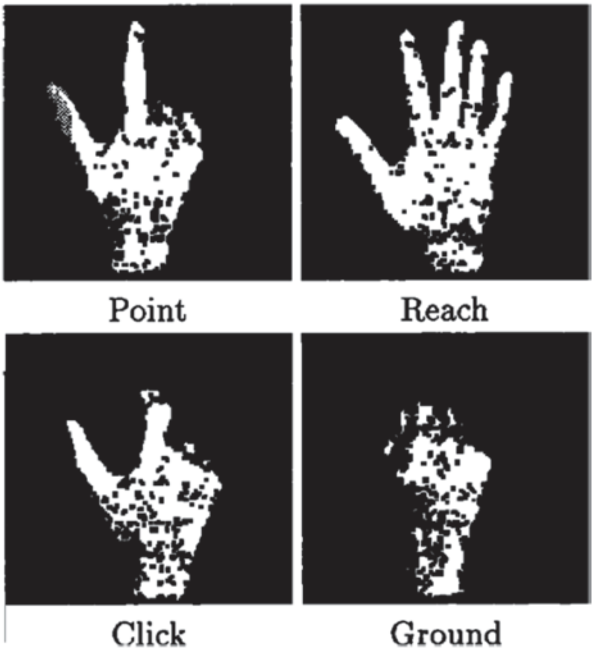


Fig. 2.17 Gestures used in virtual reality by Segen and Kumar [27]

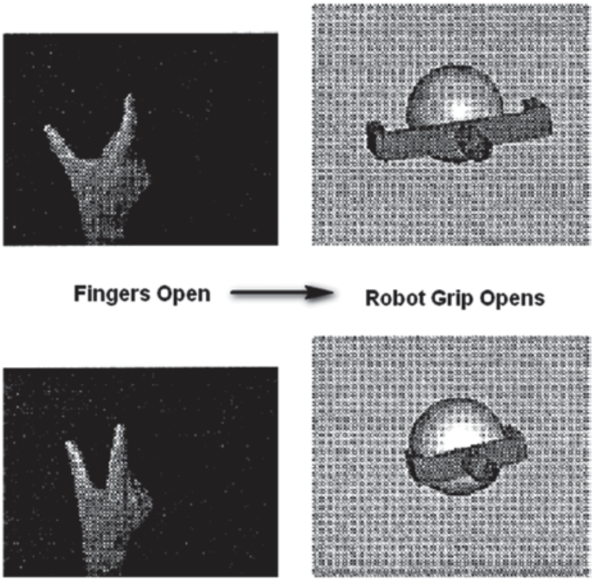


Fig. 2.18 Controlling a robot grip with fingers. [27]

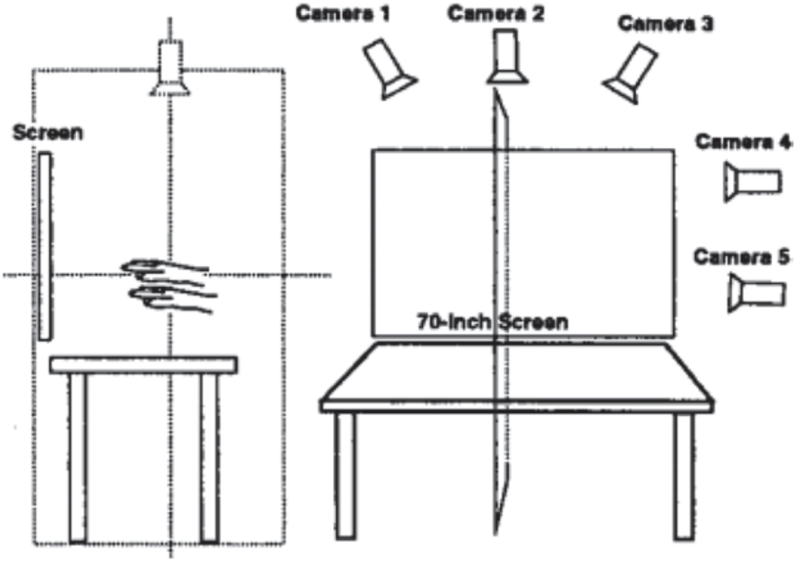


Fig. 2.19 Tracking hands with multiple viewpoints. [28]

cameras accessible to the researchers at very low price. Prior to that, the researchers always relied on good quality cameras with frame grabbers with costs running to several 1,000 $. When the cameras were available with better resolution and low noise, researchers devised new ways to rely on feature extraction from the high quality images available instead of sophisticated multi camera system.

In 2003, Chen et al. [29] introduced a hand gesture recognition system to recognize dynamic hand gestures against a static background. The system consisted a realtime hand tracking and extraction, feature extraction, hidden Markov model (HMM) training, and gesture recognition. They initially applied a real-time hand tracking and extraction algorithm to trace the moving hand and extract the hand region. Then the hand region was used to extract Fourier descriptors (FD) to characterize spatial features and the motion analysis to characterize the temporal features. They combined both spatial and temporal features of the input image sequence into the feature vector. This feature vector was used in a Hidden Markov Model to recognize input gesture. The gesture to be recognized was separately scored against different HMMs. The model with the highest score indicated the corresponding gesture. Experimental results indicated that the recognition rate above 90 % was possible with 20 different gestures. Figure 2.20 depicts the flow diagram of their hand tracking system.

Eigen Values offer a remarkable concise representation of image data. The features (Eigen Values) are completely uncorrelated thus providing the simplicity to represent image data. Only few higher values of the Eigen Coefficients would be adequate to represent an image. In 2005, Gastaldi et al. [30], developed a dynamic gesture recognition system that relied on stereo vision with Eigen Values as the

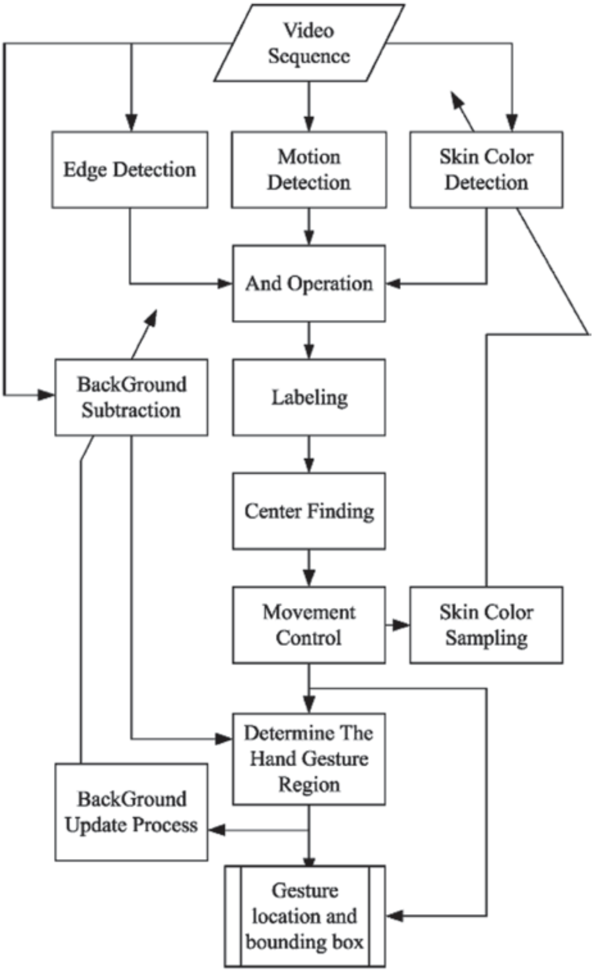


Fig. 2.20 The flow diagram of the hand tracking system based on Fourier descriptors and Hidden Markov Models. (Courtesy of Chen et al. [29])

features. They used a stereo system to detect the movements of the fingertips as shown in Fig. 2.21. This observation resulted in monitoring only five temporal sequences. Since there was very high correlation found among these temporal sequences, they managed to represent these 5 sequences by a representative single temporal sequence projected in a subspace obtained by the principal component analysis (PCA) of the feature space. Using the PCA technique, they obtained five Eigen Vectors and five Eigen values for each gesture. They concluded that by using only 3 Eigen values, more than 99 % of the information carried out by the original sequence could be retained. They projected those curves on a 3D space corresponding to the principal subspace associated to the gesture of highest dimensionality. Figure 2.22 shows the five curves of the five gestures which are easily distinguishable. This is due to the fact that by projecting the five fingertip sequences that characterize the gesture, in a common 3D subspace, temporal variations are omitted

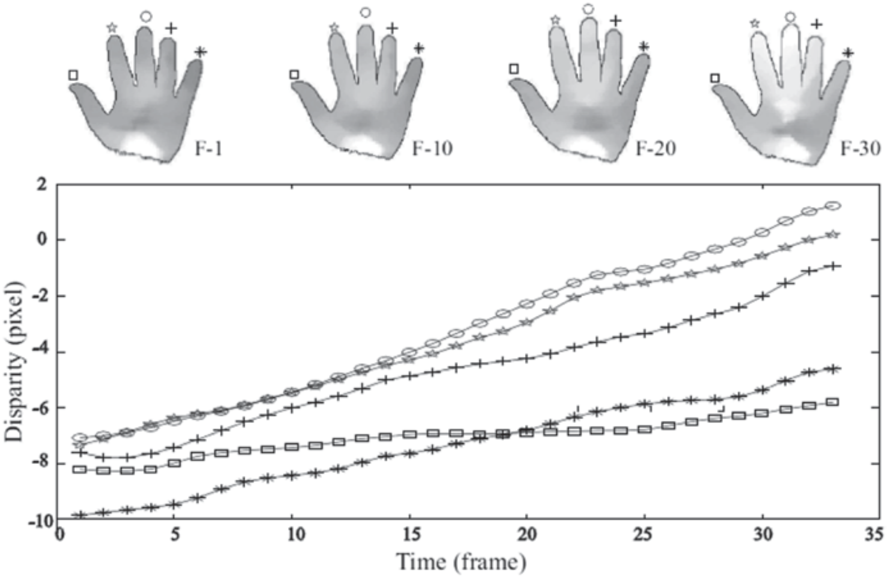


Fig. 2.21 Dynamic disparity information associated to the raising of the finger tips while keeping the wrist fixed on the table and the corresponding time evolutions of disparity information at the finger tips which characterizes the hand movement. (Courtesy of [30])

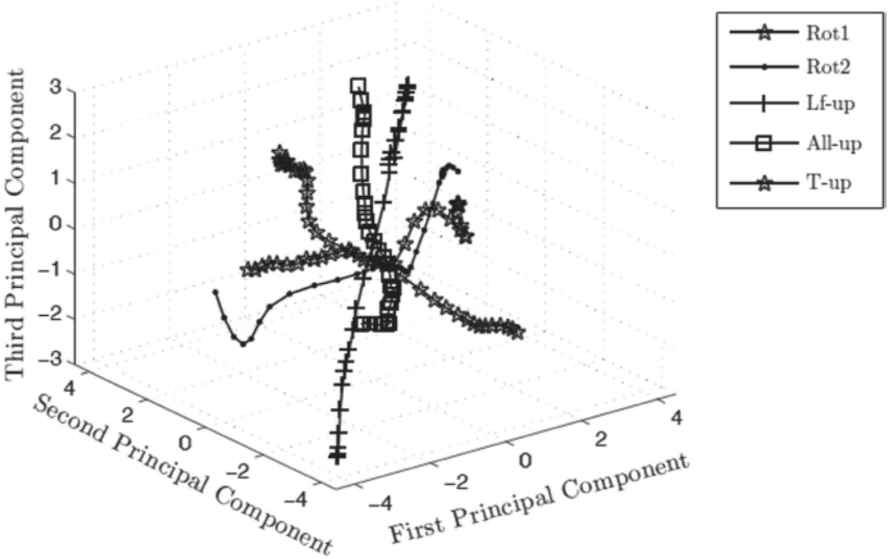
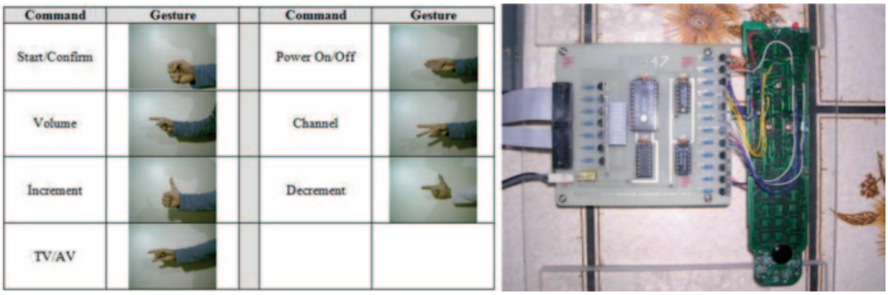


Fig. 2.22 Projection on the common 3D principal subspace of the five temporal sequences that characterize five gestures. (Courtesy of [30])



Command	Gesture	Command	Gesture
Start/Confirm		Power On/Off	
Volume		Channel	
Increment		Decrement	
TV/AV			

Fig. 2.23 Hardware interface (*left*) Mapping of commands to gestures (*right*)

which facilitates establishing relationships about depth variations on the finger-tips during hand movement.

In 2005, Premaratne et al. [29] developed hand posture recognition scheme which provided 100 % accuracy for selected 10 gestures. These gestures were used to control consumer electronics control devices such as televisions and DVD players. The approach used moment invariant features to uniquely represent hand gestures and an elaborate skin segmentation and morphological filtering mechanism to remove background and noisy regions. Figure 2.23 denotes the command mapping to hand postures and developed hardware interface which used a computer parallel port to communicate the recognized gestures to control various switches to perform different functions. The approach was invariant to gesture size (scale) and rotation and lighting variations and the skin segmentation was very robust against many extreme skin colours.

The classification stage of many hand gesture recognition system use a single classification stage. Such classification stages do provide strong classification when the selected features are unique and stand apart from other gestures. However, as it turns out, human gestures especially when using sign language (a detailed discussion is on Chap. 6), gestures vary incrementally in shape from one gesture to the other. This type of scenario can be better served with a classifier with multiples stages where approximate initial classification is further classified using more details at a later stage.

Bing and Ejima proposed a multi-stage classifier for hand gesture recognition in 2006 [30]. Their hand detector was based upon a tree structure of boosted cascaded of weak classifiers. The head of the tree forms the general hand detector and its sole purpose was to find all possible hand hypotheses in the image. Successful hypotheses were then passed onto the branches of the tree where specific cascades designed only to detect hands of a specific shape are used to determine the exact pose of the hand in the image. To build shape specific detectors the data set must be broken up or clustered into similar shapes that are for a specific shape. These clusters were expected to contain sufficient variations for shapes to allow the classifier to generalize the process. They used an approach from their previous research to perform an intelligent selection of training images for training data [31]. The framework of *tree of hand detector* is shown in Fig. 2.24.

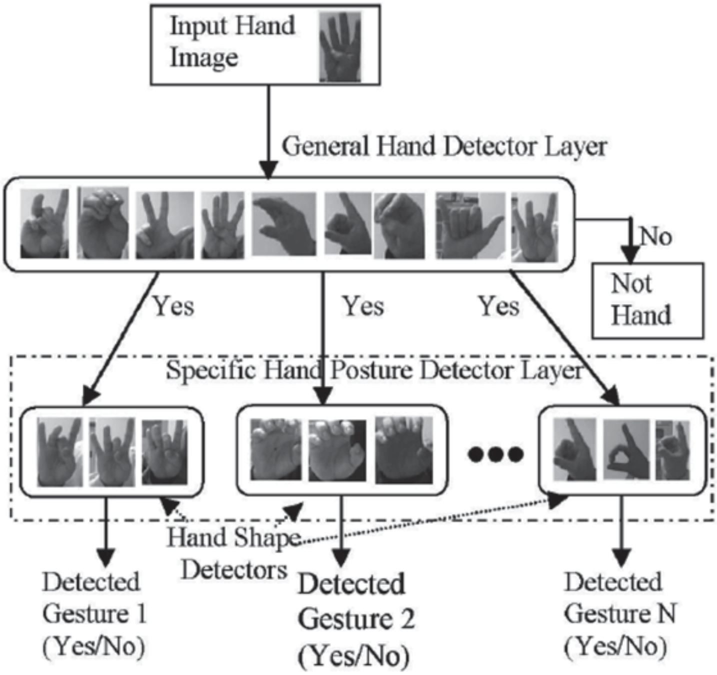

Input Hand Image
General Hand Detector Layer
No
Not Hand
Yes
Yes
Yes
Specific Hand Posture Detector Layer
Hand Shape Detectors
Detected Gesture 1 (Yes/No)
Detected Gesture 2 (Yes/No)
Detected Gesture N (Yes/No)

Fig. 2.24 The framework of a *tree of hand detectors*. (Courtesy of [30])

In order to detect hands in an image, an exhaustive detection across all possible position and scales were performed using a heuristic coarse-to-fine strategy to speed up this process. This approach was needed as the majority of the positions and scales would not contain hands. The structure of the detector cascades resulted in many parameterizations that would be rejected in the first few layers of the top strong classifier, which required only a very small amount of computation. The results obtained for a set of 36 different gestures of American Sign Language showed a 99.5 % correct recognition rate. These results demonstrated that multilevel classifier could boost the classification accuracy in hand gesture recognition.

Out of many features that researchers have exploited, skeletonization does not offer the robustness that some researchers claim [32, 33]. The reason being that it is impossible to automatically make skeletons as shown in Fig. 2.25 as Berci and Szolgay claim due to inherent noise that always result in unusual branching in skeletonization [34].

As mentioned previously, arguably the best gesture recognition can be offered by a camera system with depth information. This is quite evident from the work published by Appenrodt et al. [35]. Since stereo cameras could also suffer from occlusions, they used a multiple stereo camera setup with interesting results as shown in Fig. 2.26.

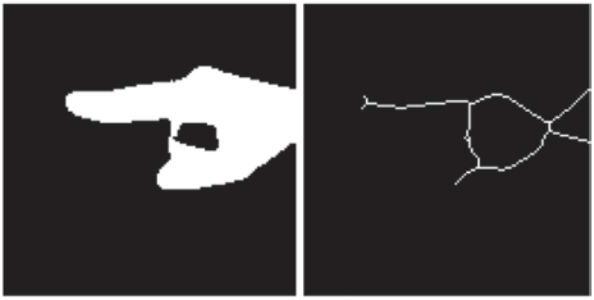


Fig. 2.25 Hand gesture and it skeleton. (Courtesy of [32])

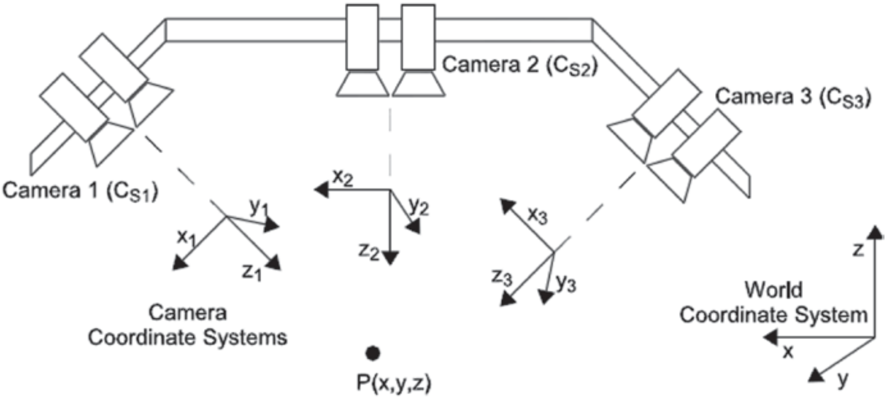


Fig. 2.26 Multi stereo camera system. (Courtesy of [35])

Their work offers a rare insight into the self-occlusion problem encountered by many researchers. In Fig. 2.27, a front view stereo camera still misses a finger due to self occlusion. Appenrodt et al. generated a hand model to describe the gestures accurately in order to avoid the self-occlusion problem. They used color information as well as depth information for detecting the hand of the user. Based on these 3D information of the scene, a representative point cloud of the hand was computed which is shown in blue points. They used an *Iterative Closest Point* (ICP) algorithm to fit the hand model (red) into the point cloud (Fig. 2.28). Thereby, the change of angle from each joint was calculated for the best correlation between model and illustration. Hence, by analyzing the model in each time step (t) allows the tracking of fingertips' local motion in video sequences. In addition, the global motion (translation) of the hand was calculated by tracking the palm.

Since the single camera system suffered from self-occlusion, they developed a multi stereo camera system to combine these information from different viewpoints. They obtained very encouraging results that did not suffer from self-occlusion as shown in Fig. 2.28.

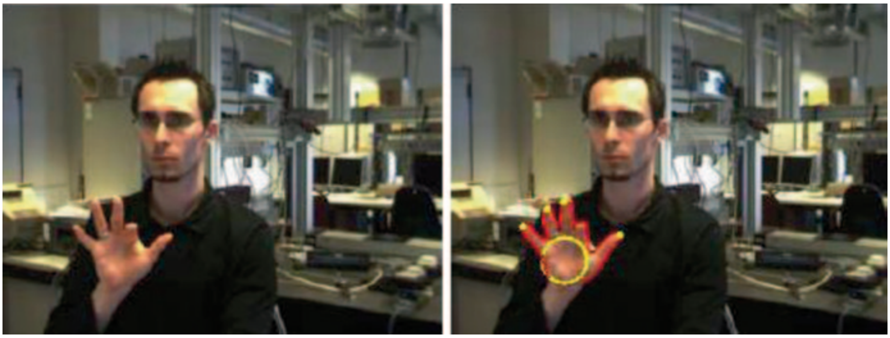


Fig. 2.27 Single camera system misses a finger in fingertip detection (on *right*). (Courtesy of [35])

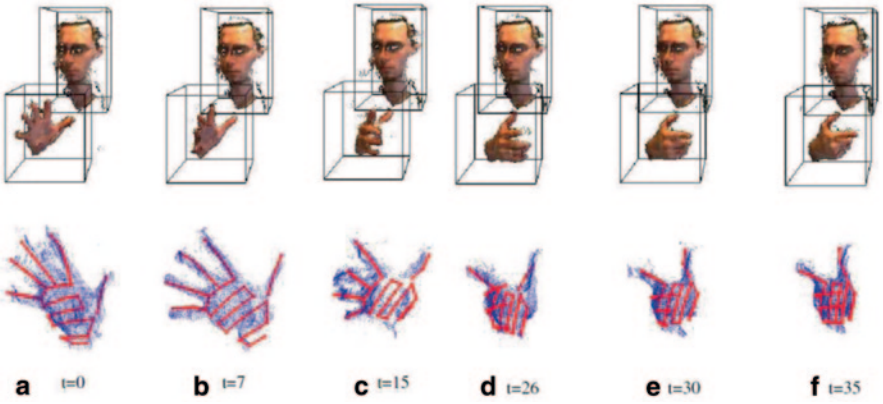


Fig. 2.28 Fingertip detection without self-occlusion using a multi camera system. (Courtesy of [35])

Hand gesture recognition systems have come a long way from its humble beginnings form wired gloves with few mechanical switches to very sophisticated ergonomically designed gloves with human comfort in mind. The basic resistance switches today have paved way for accelerometers which will offer more than 27 DOF with 3D coordinates of every finger. Their use has deviated from initial sign language to movie making to sophisticated control gear.

The development of the computer vision based gesture recognition will have to go a long way in realizing what has been achieved by glove based systems. No single no prominent strategy in camera setup to feature extraction to classification has been established as the research indicates different trends in myriad of ways. Yet, a powerful application such as sign language stands to challenges the brightest minds to develop the best of approaches in the above areas for a cohesive solution.

References

1. Dipietro, L., Sabatini, A.M., Dario, P.: Survey of glove-based systems and their applications. IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. **38**(4):461–482 (2008)
2. Zimmerman, T.G.: Optical flex sensor. US Patent 4,542,291, (1982)
3. Zimmerman, T., Lanier, J., Blanchard, C., Bryson, S., Harvill, Y.: A Hand Gesture Interface Device. In: Proceedings of the Human Factors in Computing System and Graphics Interface, pp. 189–192 (1987)
4. LaViola, J.J.: A survey of hand posture and gesture recognition techniques and technology. Providence, RI, Technical Report CS-99-11, Brown University (1999)
5. Eglowstein, H.: Reach out and touch your data. Byte **15**(7):283–290 (1990)
6. Gardner, D.L.: The Power Glove. Des. News **45**, 63–68 (1989)
7. http://www.essentialreality.com. Accessed Sept. 2, 2013
8. http://www.technologyreview.com/article/414021/open-source-data-glove/. Accessed Sept. 2, 2013
9. http://info.organicmotion.com/motion-capture-blog/bid/306947/CyberGlove-III-Gives-MoCap-a-Hand. Accessed Sept. 2, 2013
10. http://www.cyberglovesystems.com/products/cyberglove-ii/specifications. Accessed Sept. 2, 2013
11. http://www.5dt.com/?page_id=34. Accessed Sept. 2, 2013
12. http://inition.co.uk/3D-Technologies/x-ist-data-glove. Accessed Sept. 2, 2013
13. http://www.mindflux.com.au/products/essentialreality/p5glove.html. Accessed Sept. 2, 2013
14. Sturman, D.J., Zeltzer, D.: A survey of glove-based input. IEEE Comput. Graph. Appl. **14**(1):30–39 (1994)
15. Davies, J., Shah, M.: Recognizing Hand Gestures. ECCV-94, (1994)
16. Iwai, Y., Watanabe, K., Yagi, Y., Yachida, M.: Gesture recognition using Colored Gloves. Proceedings of ICPR ’96, pp. 662–666 (1996)
17. Lamberti, L., Camastra, F.: Real-Time Hand Gesture Recognition using a Color Glove. In: Maino G. and Foresti G.L. (eds.) ICIAP 2011, Part I, Lectures Notes on Computer Science Series (LNCS) **6978**, pp. 365–373. (2011)
18. Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. ACM Trans. Graphics Proc. ACM SIGGRAPH. **28**(3):63 (2009)
19. Rehg, J.M., Kanade, T.: DigitEyes: Vision-based Human Hand Tracking. Proceedings of European Conference on Computer Vision (1994)
20. Gennery, D.: Visual tracking of known three-dimensional objects. Int. J. Comput. Vision **7**(3):243–270 (1992)
21. Kang, S.B., Ikeuchi, K.: Grasp recognition using the contact web. In: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, Raleigh, NC, (1992)
22. Darrell, T., Pentland, A.: Space-Time Gestures. In: Looking at People Workshop, Chambery, France, pp. 335–340 (1993)
23. Utsumi, A., Miyasato, T., Kishino, F.: Multi-Camera Hand Pose Recognition System Using Skeleton Image. IEEE International Workshop on Robot and Human Communications, pp. 219–224 (1995)
24. Freeman, W.T., Michal, R.: Orientation Histograms for Hand Gesture. International Workshop of Automatic Face and Gesture Recognition (1995)
25. Cui, Y., Swets, D., Weng, J.: Learning-Based Hand Sign Recognition. Proceedings of the International Workshop on Automatic Face and Gesture Recognition, pp. 201–206 (1995)
26. Shimada, N., Shirai, Y., Kuno, Y., Miura, J.: Hand gesture estimation and model refinement using monocular camera-ambiguity limitation by inequality constraints. Proceedings of Third IEEE International Conference on Automatic Face and Gesture Recognition. pp. 268–273 (1998)

27. Segen, J., Kumar, S.: Gesture VR: Vision-based 3D hand interace for spatial interaction. Proceeding MULTIMEDIA '98, Proceedings of the Sixth ACM International Conference on Multimedia, pp. 455–464 (1998)
28. Utsumi, A., Jun, O.: Multiple-hand-gesture tracking using multiple cameras. IEEE Comput. Soc. Con. Comput. Vision Pattern Recognit. **1**, 473–478 (1999)
29. Chen, F.S., Fu, C.M., Huang, C.L.: Hand gesture recognition using a real-time tracking method and hidden Markov models. Image Vision Comput. **21**, 745–758 (2003)
30. Gastaldi, G., Pareschi, A., Sabatini, S., Solari, F., Bisio, G.M.: A Man Machine Communication System based on the Visual Analysis of Dynamic Gestures. IEEE International Conference on Image Processing, pp. 397–400 (2005)
31. Premaratne, P., Safaei, F., Nguyen, Q.: Moment Invariant Based Control System Using Hand Gestures. In: Intelligent Computing in Signal Processing and Pattern recognition. Book Series Lecture Notes in Control and Information Sciences, Huang, D.-S., Li K., Irwin G.W. (eds.) ICIC 2006, LNCIS vol. 345, pp. 322–333. Springer Berlin Heidelberg (2006)
32. Nguyen, D.B., Ejima, T.: A New Approach Dedicated to Hand Gesture Recognition. Proceedings of the 5th International Conference Cognitive Informatics, pp. 62–67 (2006)
33. Binh, N.D., Shuchi, B., Ejima, T.: Real-Time Hand Tracking and Gesture Recognition System. Proceedings of International Conference on Graphics, Vision and Image Processing (cVIP-o), pp. 362–368 (2005)
34. Berci, N., Szolgay, P.: Vision based Human Machine Interface via Hand Gestures. 18th European Conference on Circuit Theory and Design, ECCTD, 496–499 (2007)
35. Appenrodt, J., Handrich, S., Al-Hamadi, A., Michaelis, B.: Multi stereo camera data fusion for fingertip detection in gesture recognition systems. International conference of soft computing and pattern recognition (SoCPaR), pp. 35–40 (2010)

Chapter 3
Pre-processing

Computer vision is aimed at simulating the human visual system in order to extract useful information for machines to make decisions. A visual camera is usually used for this purpose which detects brightness, colour, texture and dimensions of an object in focus. When a camera captures scenery, it contains both 'wanted' as well as 'unwanted' information. If the camera is focussed on a person's hand looking for a possible gesture, then the 'unwanted' objects in the scenery would be the background which may contain the person's body, clothing, other people, pets, walls, windows, curtains or any other equipment. Since the system is developed to respond to gestures, the system would try to extract only the 'wanted' information. However, as the system would not have the level of intelligence as a human, it relies on 'clues' to extract only the 'wanted' objects.

Recognizing the 'wanted' information poses many challenges in computer vision. In the case of hand gesture, how a machine would identify a hand with various gestures that it could produce with different looking skin tones from around the world is difficult problem. This problem is even more compounded when hand gestures are captured in varying lighting conditions as the same hand would look different under different lighting conditions. Yet, the amount of knowledge that has been gathered in the past few decades will offer potential solutions to sift 'wanted' information from 'unwanted' clutter. This chapter will discuss many concepts of skin segmentation, morphological filtering, noise removal, and depth measurements of objects in order to identify the 'wanted' information reliably in the context of hand gesture recognition.

The next section will detail the approach that a machine would take to look for human hand called 'skin segmentation'. Once an object resembling human skin is detected, the system would expect to extract further information from this skin-tone region. However, due to poor lighting and other imperfections in the camera sensor, the extracted skin looking region may turn out to be 'noisy' resembling rough edges and missing parts in a skin region. These imperfections would be removed using a process called morphological filtering as would be discussed later. Finally, recent developments in the camera technology that derives depth information together with visual information provides opportunities to remove unwanted areas in an image using depth information would be discussed at the end of this chapter.

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_3,

3.1 Skin Segmentation

Skin segmentation is the process of looking for skin-like regions or skin tone in a visual image. The purpose of skin segmentation lies in the applications of computer vision such as people detection and tracking, face detection and tracking and gesture recognition and tracking. Once detected, this information will lead to applications in door access control, crowd counting, robotic control and human computer interaction, removing pornographic content using internet filters and many other video applications. There have been other instances of applications in automatic video annotations where newscasters were detected using skin color present in face and hand regions [1] and in image retrieval from image archives. There are many similar applications where background is controlled or unlikely to contain skin color tones, skin color detection is used to detect human faces and hands in face recognition in controlled environments [2].

Human skin is relatively easy to detect in controlled environments. However, detection in uncontrolled settings such as in consumer digital photographs is generally difficult. The appearance of skin in an image depends on illumination, geometry and color when the image was captured [3]. The humans are known to be adept at recognizing color of objects in different illumination conditions known as color constancy. This is however, is not trivial for a machine to achieve with our present level of understanding of imaging. Algorithms need to be robust enough to deal with variations in lighting or illumination, color resolution, and imaging noise. There are also other issues where skin-tone colors are found in wood, leather, certain clothing, hair, sand, paints, etc. These materials cause the classifiers to record false positives when looking for skin-tones.

3.1.1 The Problem of Skin Detection

Skin detection problem is recognized as a classification problem in many computer vision problems. In many common approaches, skin tones belonging to many ethnicities around the world under different lighting conditions are used to build databases to develop algorithms to classify them effectively. As would be discussed in the following sections, it has been found that the standard RGB color space is not the optimum color space for skin detection. Researchers have used experimental data to conclude that different color spaces have varying capabilities at extracting features or learning parameters to have better performance when extracting information to classify skin tones. As shown in Fig. 3.1, it is logical to select a color space where skin tones are represented more compactly. In this graph, Asian, African and Caucasian skin colors in *R* and *G* color spaces occupy different regions (sub images (a), (b) and (c)). However, these apparent different skin colors are confined to smaller area where they cluster together in UV, C_bC_r regions. This fact highlights why UV (from YUV color space) or C_bC_r (from YC_bC_r color space) is better than RGB color space in detecting skin tones.

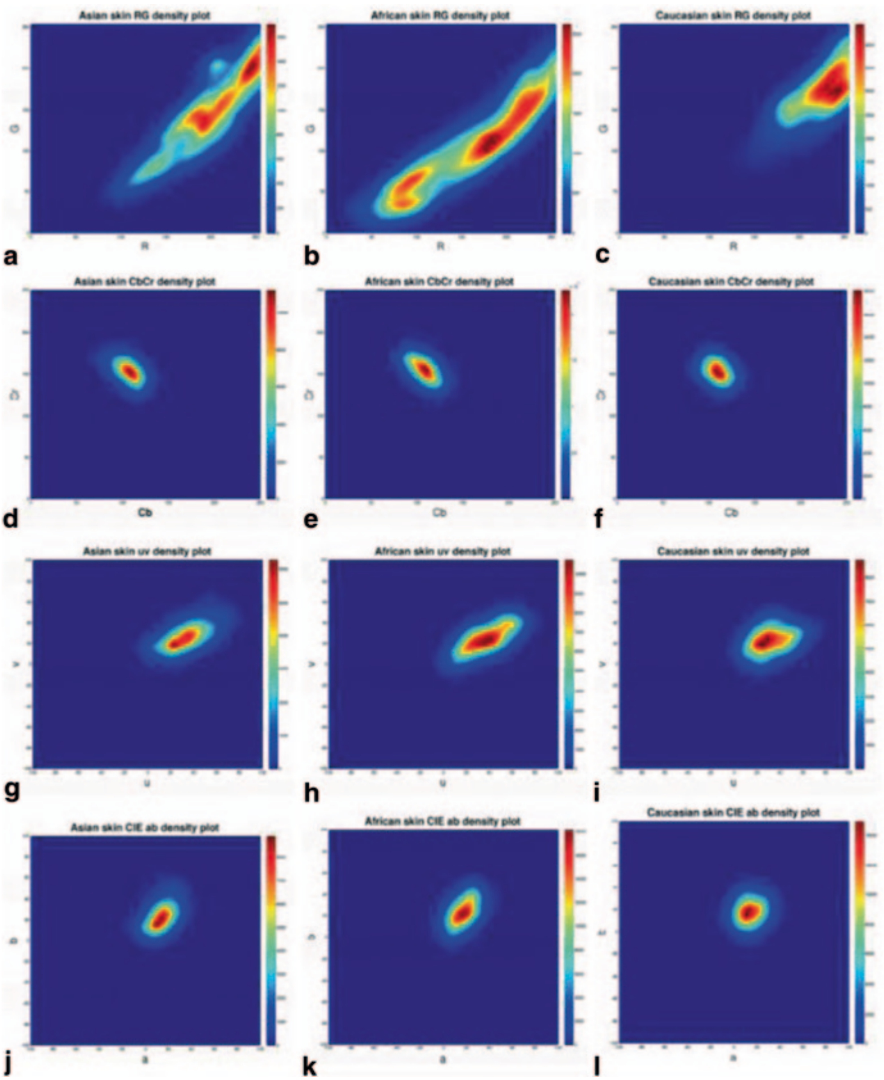


Fig. 3.1 Skin color tones do differ dramatically with ethnicity from different parts of the world when compared in RGB color space but is more stable in $C_B C_R$, CIE Lab and UV spaces. (Courtesy of [3])

Forsyth and Fleck [4] have reasoned why human skin color has limited range of hues despite the appearance of different skin tones from different parts of the world. The skin tone of any ethnicity is simply formed by combination of blood which is red and melanin which is brown. Therefore, despite the appearance, human skin color clusters in a small area in any color space. Researchers have experimented with different color spaces in order to find a color space which is invariant to illumination conditions [3].

There are two prominent approaches to skin segmentation practiced by researchers in this discipline; pixel based and region based. Pixel-based method classifies each pixel as skin or non-skin individually, independently from its neighbors. Methods utilizing color falls into this category. The region based method tries to take the spatial arrangement of skin pixels into account during the detection stage to enhance the performance. Region-based methods rely on additional knowledge such as texture of the color being investigated.

Skin color segmentation can be defined as the process of discrimination between skin and non-skin pixels. However, ambient light, shadows and the non-uniformity of imaging sensors in different cameras produce different tones that would result in different skin tones for the same person at different times. This makes it important that skin color determination is carried out in an appropriate color space where illumination or lighting conditions does not affect the decision making. Furthermore, due to variety of different skin colors from different parts of the world, it would be intriguing to see whether skin segmentation could be effectively carried at all using machine vision. The following section on color spaces will answer these questions.

3.1.2 Appropriate Color Space for Skin Segmentation

RGB is the most prominent additive color space consisting of Red (R), Green (G) and Blue (B) channels. These channels are highly correlated and contain luminance or brightness information along with the chrominance value. However, due to the presence of luminance information in each channel, any color observed does not linearly correspond to human perception. In other words, due to presence of luminance, two slightly different colors (R, G, B combined) with different luminance may appear to be the same. As was shown in Fig. 3.1, RGB color space skin color for different ethnicities would spread so widely that its use in skin segmentation in the presence of other objects would be questionable.

There are other classes of color spaces in existence because of Television transmission. The YUV contains Y luminance and U and V chrominance components. Unlike RGB, Y contains the entire luminance component making U and V independent or invariant to illumination. YIQ is a similar color space which is used in NTSC Television format. YC_bC_r carries similar information to that of YUV and is used in JPEG based image compression standard. Figure 3.1 shows the benefit of using these color spaces opposed to RGB as they provide compact clusters invariant to ethnic background that would facilitate simpler classification approach [5–9].

Perceptual color spaces which have been developed the way how artists describe color, and its properties have also been used for skin segmentation research. Color spaces such as HSI, HSV and HSL are commonly used as they are much closer to human perception than the television broadcast related color spaces. Hue (H) has been described as the color and Saturation (S) which describes how ‘pure’ the color and brightness (I, V or L). HSV can be mapped from RGB using nonlinear mapping. Similar to YUV approach, H and S values are used for skin segmentation where intensity or the brightness value is disregarded to remove the sensitivity of the illumination on skin segmentation results [5, 9].

Such complexities can only be overcome if an approach can be devised where skin segmentation is invariant to most of these variables yet resulting in an acceptable discriminatory power of skin vs non-skin regions in an image. The answer lies in some color spaces other than the most common RGB. Red, Green and Blue (RGB) color space is the most common color space used to represent images. RGB is an additive color space with Red, Green and Blue components carrying highly correlated information. John and Rehg [10] and Brand and Mason [11] have demonstrated that skin segmentation is possible in RGB space. However, there is overwhelming evidence that suggests RGB color space is not effective for skin segmentation for variety of skin color from different parts of the world. Researchers had proposed using normalized RGB to obtain chromaticity information to classify skin pixels effectively. However, normalized RGB is plagued by uneven illumination [12–15]. The skin segmentation thresholds for RGB are given by Kovac et al. [16]:

For uniform daylight illumination:

$$R > 95, G > 40, B > 20$$
$$Max\{R, G, B\} - min\{R, G, B\} < 15$$
$$|R - G| > 15, R > G, R > B$$

Flashlight or daylight lateral illumination:

$$R > 220,\ G > 210,\ B > 170$$
$$|R - G| \le 15,\ B < R,\ \ B < G.$$

3.1.2.1 Normalized RGB

There have been efforts to remove discrepancies observed when different color combinations with varying intensity appearing similar in RGB space. One such suggestions is normalized color space given by following expressions:

$$r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B}, b = \frac{B}{R+G+B} \tag{3.1}$$

Here $r = 1 - g - b$ due to normalization. Hence, determining any two normalized colors will completely define the color space. Gomez and Morales used a constructive induction approach to determine the skin map [17, 18]. Using the normalized RGB values they determined that the following thresholds resulted in best skin segmentation performance:

$$\frac{r}{g} > 1.185, \frac{r \cdot b}{(r+g+b)^2} > 0.107 \text{ and } \frac{r \cdot g}{(r+g+b)^2} > 0.112$$

3.1.2.2 YCbCr, YUV and YIQ

Due to the linear nature of transformation between RGB and YC_bC_r, YC_bC_r is often used in people surveillance and skin color segmentation [19–22]. The YC_bC_r values are less computationally intensive to achieve compared to the HSV values and are computed as follows:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112 \\ 112 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.2}$$

The Y, U, V and YIQ values are similarly calculated from RGB using a linear conversions:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}, \tag{3.3}$$

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.595716 & -0.274453 & -0.321263 \\ 0.211456 & -0.522591 & 0.311135 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{3.4}$$

As shown if Fig. 3.2, a hand gesture looks different in different color spaces. Yet, YC_bC_r offers the ability to separate skin tone from non-skin regions as shown in Fig. 3.3.

YC_bC_r thresholds for skin segmentation are:

$$77 \le C_b \le 127 \text{ and } 133 \le C_r \le 173.$$

3.1.2.3 HSV, HIS, HSL—Hue, Saturation and Intensity (Value, Lightness)

Researchers have devised HSV (Hue Saturation and Value) and YC_bC_r color space to separate luminance and chrominance information. This separation of brightness information from chrominance leads to reduction in uneven illumination [23]. The HSV values are derived using the following expressions using RGB components:

$$H = \arccos \frac{\frac{1}{2}((R-G)+(R-B))}{\sqrt{(R-G)^2+(R-B)(G-B)}} \tag{3.5}$$

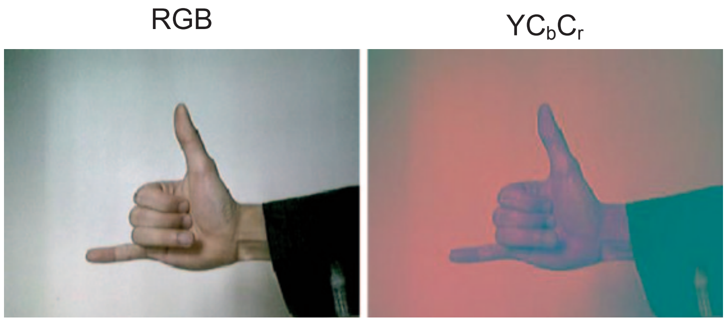


Fig. 3.2 Hand gesture in RGB and YC_bC_r color spaces

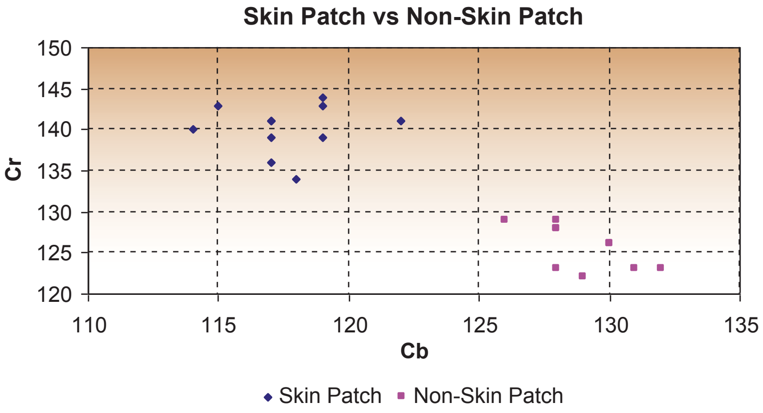


Fig. 3.3 Correlation between C_r and C_b for Skin Patch and Non-Skin patch pixels

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

$$V = \frac{1}{3}(R + G + B). \tag{3.7}$$

Tsekeridou and Pitas [18, 24], have obtained thresholds for skin segmentation using the following thresholds:

$$V \geq 40;$$
$$0.2 < S < 0.6;$$
$$0^{\circ} < H < 25^{\circ} \text{ or } 335^{\circ} < H < 360^{\circ}.$$

Starting from a training data set composed of skin color samples, Garcia and Tziritas computed the color histogram in HSV color space, and estimated the shape of the skin color cluster [18, 25]. They found a set of planes by successive adjustments depending on segmentation results, developing the thresholds shown below which define six bounding planes found in the HSV color space case, where $H \in \left[-180^{\circ}\ 180^{\circ}\right]$:

$$\begin{aligned}
&V \geq 40;\\
&H \leq (-0.4V + 75)\\
&10 \leq S \leq (-H - 0.1V + 110)\\
&\mathit{if}\ H \geq 0 \quad S \leq (0.08(100 - V)H + 0.5V)\\
&\mathit{if}\ H < 0 \quad S \leq (0.5H + 35).
\end{aligned}$$

Hue-saturation based color spaces stems from the humans desire to numerically specify the notions of tint, saturation and tone. Hue represents the dominant color (as in dominant wavelength) whereas saturation defines the 'colorfulness' of an area with respect to its brightness [26]. The amount of light or luminance, historically measured in lux, has lead to the notions of 'intensity', 'lightness' or 'value'. The user is directed to the following references for deeper notions of color spaces in skin segmentation [27–31].

There are direct relationships among the brightness and the chrominance values which attempt to conceal the chrominance information. In 1999, Fleck et al. developed an alternative way of hue and saturation computation using log opponent values to reduce the dependence of chrominance on the illumination levels [32].

The polar coordinate system of Hue-Saturation spaces, as shown in Eq. 3.5, results in a cyclic form. This is inconvenient color space for parametric skin color models that need tight cluster of skin colors for best performance. A different representation of Hue-saturation using Cartesian coordinates can be used [19, 33]:

$$X = S\cos H, \qquad Y = S\sin H$$

HSL and HSV are the two most common cylindrical-coordinate representations of points in an RGB color model. The two representations rearrange the geometry of RGB in an attempt to be more intuitive and perceptually relevant than the cartesian (cube) representation. Developed in the 1970s for computer graphics applications, HSL and HSV are used today in color pickers, in image editing software, and less commonly in image analysis and computer vision [34]. The relationship between RGB and HSL, and HSV are as follows:

$$\begin{aligned}
M &= \max(R, G, B)\\
m &= \min(R, G, B)\\
C &= M - m
\end{aligned} \tag{3.8}$$

$$H' = \begin{cases} undefined, & if\, C=0 \\ \dfrac{G-B}{C} \bmod 6, & if\, M=R \\ \dfrac{B-R}{C} + 2, & if\, M=G \\ \dfrac{R-G}{C} + 4, & if\, M=B \end{cases} \tag{3.9}$$

$$H = H' \times 60^o \tag{3.10}$$

$$\begin{aligned} I &= \frac{1}{3}(R+G+B) \\ V &= M \\ L &= \frac{1}{2}(M+m) \end{aligned} \tag{3.11}$$

$$\begin{aligned} S_{HSV} &= \begin{cases} 0, & if\ C=0 \\ \dfrac{C}{V}, & otherwise \end{cases} \\ or& \\ S_{HSV} &= \begin{cases} 0, & if\ C=0 \\ 1-\dfrac{m}{I}, & otherwise \end{cases} \end{aligned} \tag{3.12}$$

$$S_{HSL} = \begin{cases} 0, & if\ C=0 \\ \dfrac{C}{1 - |2L - 1|}, & otherwise \end{cases} \tag{3.13}$$

3.1.2.4 TSL—Tint, Saturation and Lightness

A normalized chrominance-luminance TSL space is a transformation of the normalized RGB into more intuitive values, close to hue and saturation in their meaning [19].

$$\begin{aligned} S &= \left[9/5(r'^2 + g'^2)\right]^{1/2} \\ T &= \begin{cases} \arctan(r'/g')/2\pi + 1/4, & g' > 0 \\ \arctan(r'/g')/2\pi + 3/4, & g' > 0 \\ 0, & g'=0 \end{cases} \\ L &= 0.299R + 0.587G + 0.114B \end{aligned} \tag{3.14}$$

where $r'=r-1/3$, $g'=g-1/3$ and r, g are defined as in Eq. 3.1 [19]. Terrillon et al. [35] have compared nine different color spaces for skin modelling with a unimodal Gaussian joint probability density functions (only chrominance components of the color spaces were used). They argue that normalized TSL space is superior to other color spaces for this task.

3.1.2.5 CIELAB Color Space

CIELAB color space has been devised to be perceptually uniform color space. According to Poynton et al., perceptual uniformity refers to "*Digital image representation is perceptually uniform if a small perturbation of a component value—such as the digital code value used to represent red, green, blue, or luminance—produces a change in light output at a display that is approximately equally perceptible across the range of that value*" [36]. Hence uniform color spaces were defined in such way that all the colors are arranged by the perceptual difference of the colors. However, the perceptual uniformity in these color spaces is obtained at the expense of heavy computational transformations. As shown in Eqs. 3.15, 3.16 and 3.17, the computation of the luminance (L) and the chroma (a, b) is obtained through a non-linear mapping of the XYZ coordinates [37]. CIE (Commission International d'Eclairage) specifies three: CIE*XYZ, CIE*Lab, and CIE*Luv. In CIE*Lab or CIELab, the three components represent luma or luminance (or illumination) component and ab represent the chroma or color information [38]. The relationship between RGB, and XYZ and a, b components are:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4125 & 0.3576 & 0.1804 \\ 0.2127 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9502 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.15}$$

$$L = \begin{cases} 116\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 & if \frac{Y}{Y_n} > 0.008856 \\ 903.3 & otherwise \end{cases} \tag{3.16}$$

$$a = 500\left[\frac{X^{\frac{1}{2}}}{X_n} - \frac{Y^{\frac{1}{3}}}{Y_n}\right], \qquad b = 200\left[\frac{Y^{\frac{1}{2}}}{Y_n} - \frac{Z^{\frac{1}{3}}}{Z_n}\right]. \tag{3.17}$$

The threshold values for skin segmentation under CIE LAB are: [39]

$a_{\text{max}}=14$, $a_{\text{min}}=2$, $b_{\text{max}}=18$, $b_{\text{min}}=0.7$. Figure 3.4 depicts the results of skin segmentation under different color spaces.

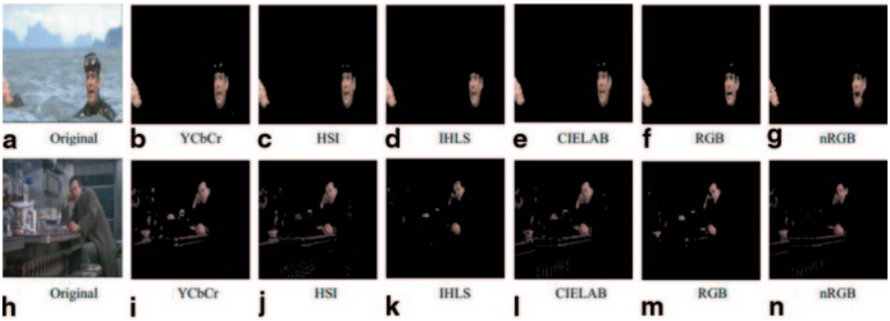


Fig. 3.4 Example results of skin detection using static skin filters in different color spaces. *Black* shows non-skin. (Courtesy of [39])

The goal of skin segmentation is the rapid decision making of skin vs non-skin regions. This can be accomplished by a set of rules which would define valid regions for skin in different color spaces. In the previous sections, for each color space, skin color thresholds were presented that were developed using extensive research over the years.

Pixel Based Skin Classification Using Non-parametric Skin Modelling The features used in skin classification are the values from color spaces. The problem then reduces to identifying a test pixel falls into the compact boundary or outside. Brand and Mason [40] constructed a simple one-dimensional skin classifier which would asses if the ratio between R and G channels falls in between particular upper and a lower bound. There are other approaches where the skin color region in a two dimensional color space (U, V or C_b, C_r, etc.) is modelled using an elliptical boundary model [41]. The model parameters are estimated with the help of a large skin patch database.

There are other classification strategies using Bayesian probabilistic approaches using the knowledge of statistics. The classification relies on finding the $P(skin|color)$ which is the probability of any *color* pixel being skin. This information is almost impossible to be determined given that any color space having extremely high number of colors. However, by rewriting this expression using the following way simplifies the problem:

$$P(skin\,|\,color) = \frac{P(color\,|\,skin)P(skin)}{P(color\,|\,skin)P(skin) + P(color\,|\,nonskin)P(nonskin)}.$$

Since finding information of $P(\text{color}|skin)$ can be achieved using information gathered by recording human skin color from every part of the globe. Also the denominator signifies the total probability of observing color which does not affect the classification as it is a constant. Therefore the problem reduces to finding $P(skin|color)$ which can be estimated using histograms [13, 20, 28, 42–44], mixture of Gaussian models [30, 45] to approximate probability density functions.

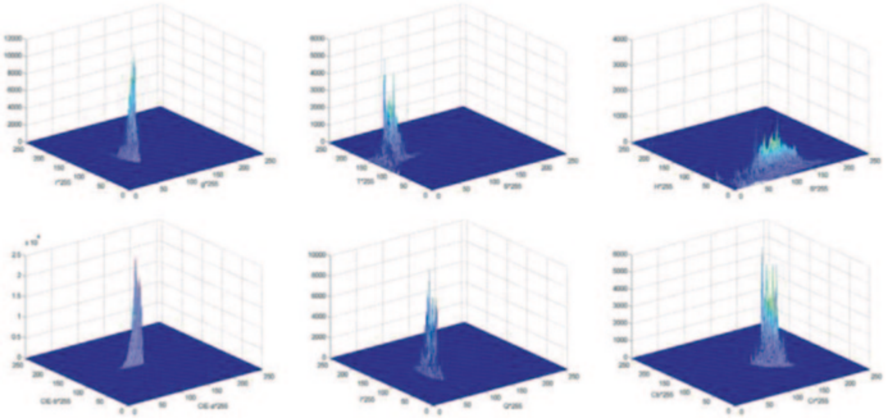


Fig. 3.5 Cumulative histograms of the training skin color pixels in different chrominance spaces: normalized r-g, T-S, H-S, CIE-ab, I-Q, Cb-Cr [47]

3.1.2.6 Region Based Skin Segmentation

Scientists overwhelmingly agree that for effective skin segmentation, it is natural to treat skin or non-skin as regions instead of individual pixels [45]. This would reduce the amount of noise that is present when isolated skin-tone 'patches' are erroneously classified as skin. Some of the early work on region based skin segmentation was reported by Yang and Ahuja on Gaussian mixture model for skin classification opposed to the predominantly simple thresholding or a single Gaussian distribution to characterize the properties of skin color [45]. They used multiscale segmentations to find elliptical regions for face detection. Hence, their model is biased toward elliptical objects. Kruppa et al. proposed a simple generative skin patch model combining shape and color information [46]. Their model was parametric and represented the spatial arrangement of skin pixels as compact elliptical regions. Those parameters were estimated by maximizing the mutual information between the model-generated skin pixel distribution and the distribution of skin color as observed in the image.

As shown in Fig. 3.5, histograms can be developed for different color spaces using variety of skin tones representing variety of human races from the world for an effective classifier [47]. Such knowledge can then be used effectively for skin segmentation as shown in Fig. 3.6 [47].

Poudel et al. proposed a segmentation technique based on the notion of superpixel [48–50], to group similar color pixels together. Each superpixel was classified as skin or non-skin by aggregating pixel-based evidence obtained using a histogram based Bayesian classifier similar to [11].

The result was further improved with Conditional Random Field (CRF), which operate over superpixels instead of pixels. Even though the segmentation cost is an overhead over the pixel-based approach, it greatly reduces the processing cost further down the line, such as smoothing with CRF. Furthermore, aggregation of pixels into regions helps to reduce local redundancy and the probability of merging

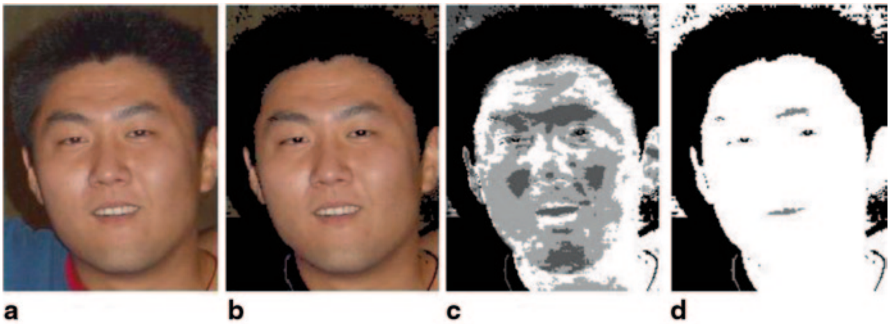


Fig. 3.6 An example of image segmentation. (**a**) The original image, (**b**) the result after pre-processing, (**c**) the result of the original FCM, (**d**) the result of the improved FCM. (Courtesy of [47])

unrelated pixels [51]. Since superpixels preserve the boundary of the objects, it helps to achieve very accurate object segmentations [52]. Their method not only outperformed the current state-of-the-art pixel-based skin color detection methods but also extracted larger skin regions while still keeping the false-positive rate lower, providing semantically more meaningful results. This could in turn benefit higher-level vision tasks, such as face or hand detection.

3.2 Morphological Filtering

Computer vision relies on identifying shapes and structures in image acquisition. As was discussed in the section of skin segmentation, once a shape is isolated as a binary image with numerous imperfections, morphological filtering is commonly used to remove imperfections in shapes to understand the image content. In particular, the binary regions produced by simple thresholding are distorted by noise and texture. Morphological image processing pursues the goals of removing these imperfections by accounting for the form and structure of the image.

Morphological filtering is a broad set of non-linear image processing operations that can be used to process images based on shapes. These operations apply a structuring element of different shapes to an input image. The output image usually retains its original size. The structural element denotes the size of the window that would operate on a neighbourhood of a pixel to create the output. The size and shape of the neighbourhood can be chosen to construct a morphological operation that is sensitive to specific shape(s) in the input image.

Before the detailed theory of morphological filtering is discussed, it would be useful to see an example of a computer vision application in the context of computer human interaction to ascertain the usefulness of this process. Figure 3.7 shows that under good lighting conditions, the skin segmented hand gesture contain few noise patches. When the lighting deteriorates, the resulting thresholded image contains more noise patches as shown in Fig. 3.7 (right bottom). In order for computer vision

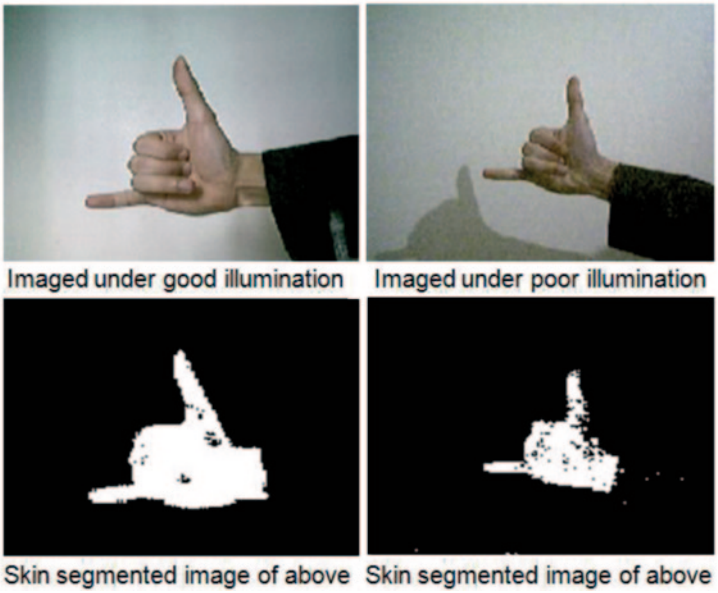
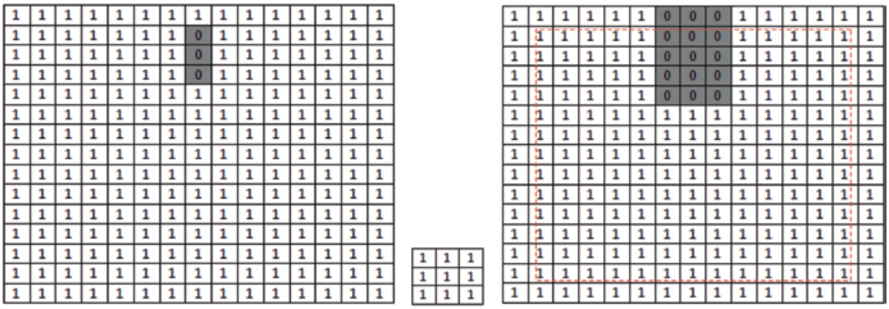

Imaged under good illumination
Imaged under poor illumination
Skin segmented image of above
Skin segmented image of above

Fig. 3.7 The above images show that under poor illumination, skin segmentation results in multiple undesired artefacts. Even the well-lit images produce undesirable regions as shown in images of *left*

system to be effective, the skin segmented extracted gesture should be solid white for further processing. Also, the noise spots shown in Fig. 3.7 (left bottom and right bottom) should be removed. The only operation that facilitates this requirement is morphological filtering as would be discussed next.

3.2.1 Basic Operations; Erosion and Dilation

Dilation and erosion are considered to be the most basic morphological operations. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The size of the structuring element (SE) determines the number of pixels added or removed from the objects in an image. In dilation and erosion, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image [53, 54]. There are set rules that define the process either as dilation or erosion. The morphological filtering process is mostly binary in nature however; these operations can also be used on gray scale images. These operations can be applied on gray scale images when their light transfer functions are unknown and therefore their absolute pixel values are of no or minor interest. In binary operation, the outcome is either 1 (which is the highest intensity value) or 0 (which is lowest intensity possible). In dilation, the value of the output pixel is the maximum value of all the pixels in



1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1	1	1
1	1	1
1	1	1

1	1	1	1	1	1	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fig. 3.8 Binary image of size 15 × 15 is operated on with a structuring element which performs 'erosion' and the result is shown on the *right*. Only a 13 × 13 sized area contain the valid signal after erosion marked with *red broken line*

the input pixel's neighborhood. In a binary image, if any of the pixels is set to the value 1, the output pixel is set to 1. The erosion rule states that the value of the output pixel is the minimum value of all the pixels in the input pixel's neighborhood. In a binary image, if any of the pixels is set to 0, the output pixel is also set to 0. Figure 3.8 shows the operation of a structuring element of size 3 × 3 on a binary image of size 15 × 15. The outcome of this is shown in the right hand size matrix of Fig. 3.8. The 'red' broken line marks the boundary of the valid signal after the operation as outside of this region is considered invalid due to the size of the structuring element. Figure 3.9 shows the outcome using a 3 × 3 structural element. As can be seen, this leaves skin tone regions intact. Hence the size of the structural element is very important. The size of the structural element depends on the size of disconnected or noisy artefacts that remains after skin segmentation.

Figure 3.10 shows that the outcome of any morphological filtering is sensitive to the size of the structuring element as an inappropriate size would simply result in a more complicated image that a computer vision system is unable to utilize. Fairly large structural elements erode the information contained in the useful object such as skin segmented hand gesture. Only close observation of the objects to be preserved and removed would justify the size of the structural element.

3.2.1.1 Mathematical Definition of Morphological Filtering—Erosion and Dilation

Mathematically, erosion is defined for an Image *I* by a structural element *S* as follows:

$$I \ominus S = \left\{ I \middle| S_I \subseteq I \right\}$$

Where S_I refers to *S* translated with *I*.

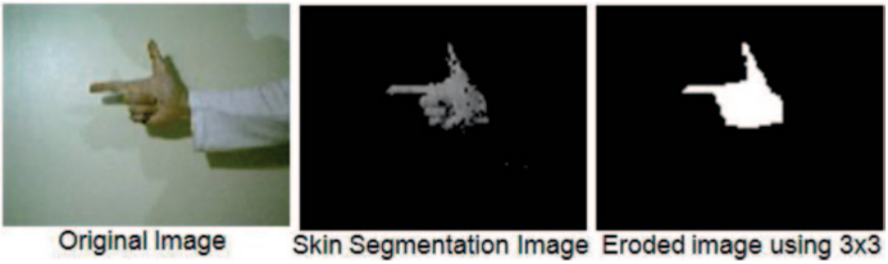


Fig. 3.9 Result of erosion using a 3 × 3 structural element

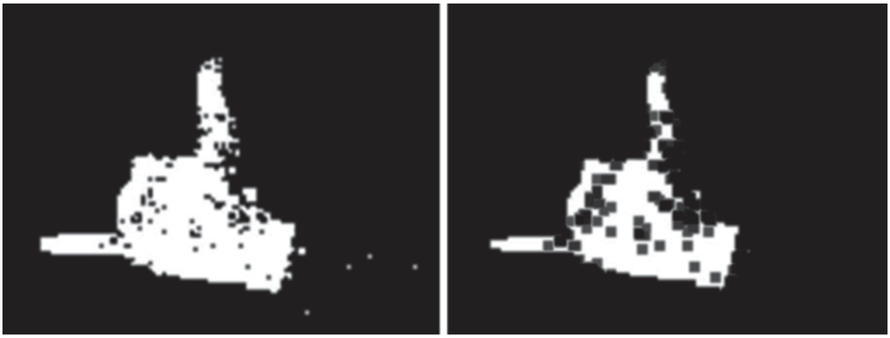


Fig. 3.10 Erosion of a noisy hand gesture using a structural element of size 7 × 7. Here the result shows large *square holes* in the resulting image signalling that the size of the structuring element is not appropriate for this operation

A complementary operation to that of erosion is dilation. It is defined simply as the erosion of the complement of a set. If I^c denotes the complement of I, then the dilation of a set I by a set S is denoted by $I \oplus S = (I^c \ominus S)^c$. This helps us to easily understand dilation in terms of erosion. Figure 3.11 shows the effect of dilation on a structure using a 3 × 3 structural element.

Figure 3.12 shows the outcome of 'filling' when dilating an eroded image. However, this process outline that dilation with larger structural elements will not necessarily fill image gaps. Morphological operations such as erosion and dilation can be performed on gray scale images as shown in Fig. 3.13 and 3.14. In Fig. 3.13, the result of erosion using a structural element of size 6 × 6 square results in disfiguring the letters and darkening the image. On the contrary, dilation result in similar disfigurement of lettering yet, lightening the image as shown in Fig. 3.14.

Processing Pixels at Image Borders (Padding Behavior) In morphological filtering, origin of the structuring element is centred over the pixel of interest in the input image. For pixels at the edge of an image, parts of the neighborhood is defined by the amount that structuring element can extend beyond the border of the image.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1	1	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1	1	1
1	1	1
1	1	1

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	0	0	0	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fig. 3.11 Results of dilation using a 3 × 3 element. See that the *vertical line* has completely disappeared as its width was less than the width of the structural element

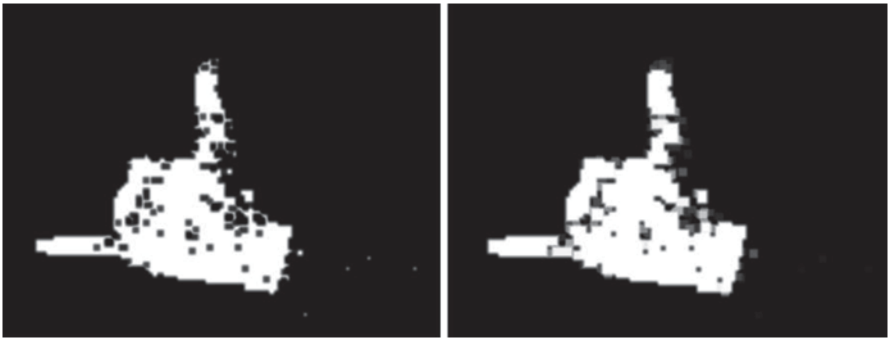


Fig. 3.12 Dilation of an eroded image with a structural element of size 5 × 5 (*left*) and 7 × 7 (*right*)

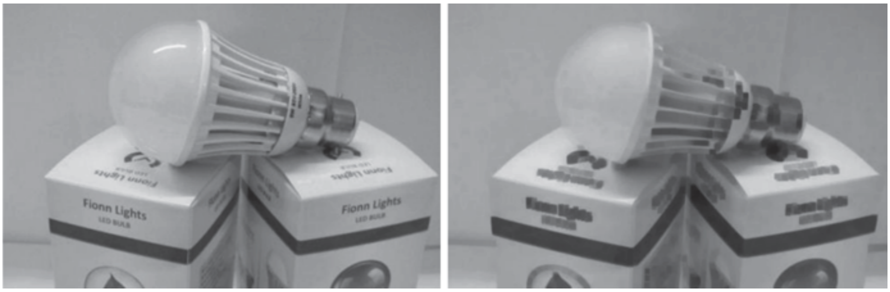


Fig. 3.13 Erosion of a gray scale image by a 6 × 6 structural element. Original (*left*), eroded (*right*). The image content is much darker after erosion

To process border pixels, the morphological functions assign a value to these undefined pixels as if the functions had padded the image with additional rows and columns. The value of these padding pixels varies for dilation and erosion operations. Pixels beyond the image border are assigned the minimum value afforded by

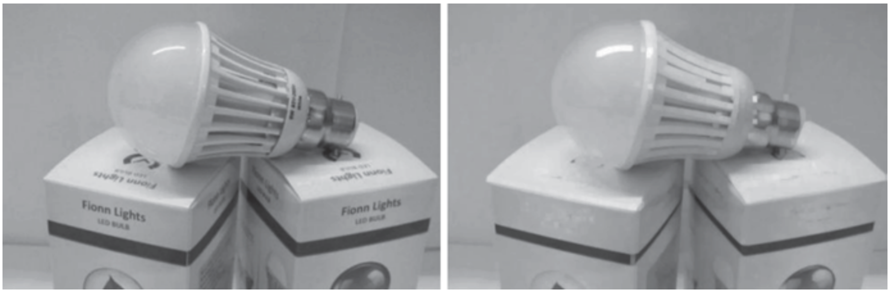


Fig. 3.14 Dilation of grayscale image by a 6 × 6 structural element. The image is lighter than before after dilation. Original (*left*) and the dilated image (*right*)

the data type. For binary images, these pixels are assumed to be set to 1. For gray scale images, the maximum value for uint8 images is 255. For dilation of binary images, these pixels (padding pixels) are assumed to be set to 0 whereas for gray scale images, the minimum value for uint8 images is 0.

3.2.2 Opening and Closing

Erosion and dilation are used in many other morphological filtering to achieve different outcomes for computer vision applications. Hand gesture recognition in its binary representation usually result in many holes and noisy unconnected artefacts. These areas need to be filled up producing solid gestures while removing the artefacts without affecting the gesture.

Opening can be described using more fundamental operations. Opening is so called because it can open up a gap between objects connected by a thin bridge of pixels. In this case, the dilation and erosion should be performed with a structuring element that has been rotated by 180°. Typically, the structural elements are symmetrical, so that the rotated and initial versions of it do not differ. Any regions that have survived the erosion are restored to their original size by the dilation. All pixels which can be covered by the SE with the SE being entirely within the foreground region will be preserved. All foreground pixels which cannot be reached by the structuring element without lapping over the edge of the foreground object will be eroded away. Opening is idempotent which refers to the fact that repeated application has no further effects.

Closing is the operation of filling holes in the regions while keeping the initial region sizes. In other words, closing (opening) of a binary image can be performed by taking the complement of that image, opening (closing) with the structuring element, and taking the complement of the result. The formal mathematical definitions of opening and closing are defined next.

Opening Opening is performed by erosion followed by dilation resulting in eliminating protrusions and smoothing contours. Both of these operations are attempted

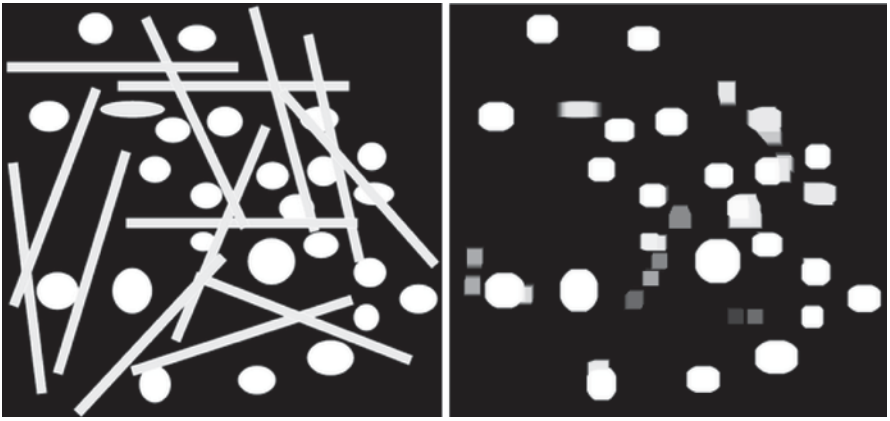


Fig. 3.15 Opening with a 10 × 10 square structuring element

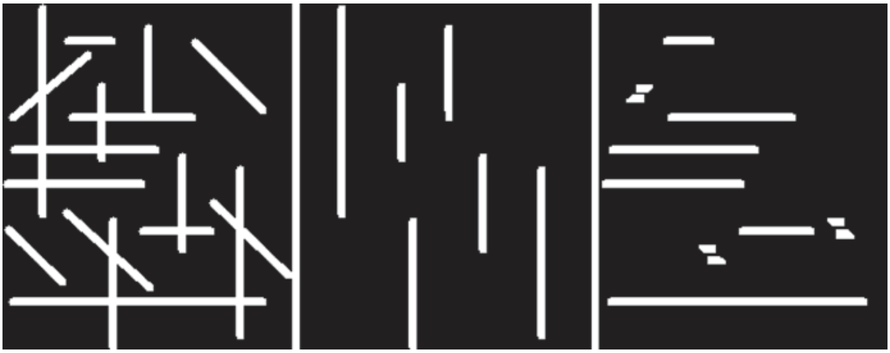
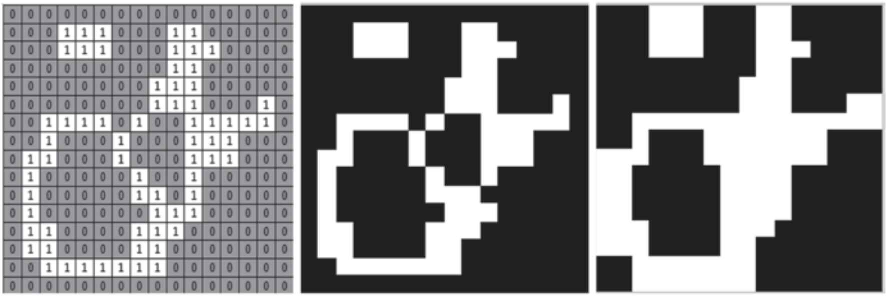


Fig. 3.16 Opening the image on the *left* with a 3 × 9 structuring element (result shown in the *middle*), opening with 9 × 3 structuring element (*right*)

using the same structural element. The mathematical symbol of opening is '∘' and the definition opening using erosion and dilation is given by:

$$I \circ S = (I \ominus S) \oplus S$$

Opening is known as a filtering mechanism to remove clutter to enhance image intelligibility especially for computer vision. As shown in Fig. 3.15, using a specific type of structuring element with specific size, the long thin objects are removed from the image. This would be advantageous for removing clutter for medical diagnosis or counting certain type of objects removing unnecessary ones. The effect of the choice of the structuring element size is illustrated in Fig. 3.16. A SE of size 3 × 9 will result in leaving vertical bars intact and the 9 × 3 will remove the vertical bars leaving only the horizontal ones.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	1	1	0	0	0	0	0
0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0
0	0	1	1	1	1	0	1	0	0	1	1	1	1	1	0
0	0	1	0	0	0	1	0	0	0	1	1	1	0	0	0
0	1	1	0	0	0	1	0	0	0	1	1	1	0	0	0
0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0
0	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0
0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 3.17 Closing a 16 × 16 image with a 3 × 3 square structuring element. The figure on the *left* shows an image as matrix with '1' associated with *white* and '0' associated with *black*. The operations are performed on the host image in the *middle* with the results shown on the *right*

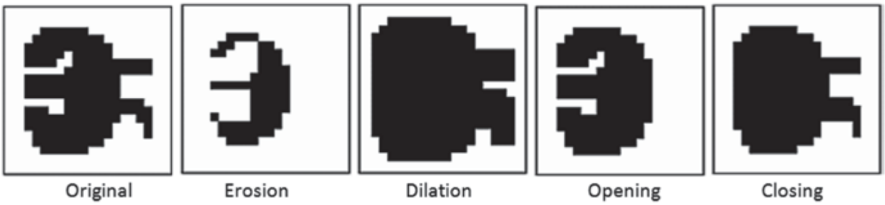


Fig. 3.18 Comparison of different processes of fundamental morphological filtering with an illustration of their use on a binary image, courtesy of [55]

Closing Closing is performed using dilation followed by erosion resulting in smoothing contours and fusing narrow breaks and long thin gulfs. This eliminates small holes and fills gaps in contours. As in opening, same structural element is used for both dilation and erosion. It would be interesting to understand the closing process as a structural element operates on the host image. For the initial dilation, the SE slides around outside each foreground region. All background pixels which can be covered by the SE with the SE being entirely within the background region will be preserved. All background pixels which cannot be reached by the structuring element without lapping over the edge of the foreground object will be turned into foreground. This scenario is illustrated in Fig. 3.17 when operated on by a 3 × 3 square structuring element. Opening is also known to idempotent as Opening. The symbol of closing is '$\bullet$' and is defined using dilation and erosion as follows:

$$I \bullet S = (I \oplus S) \ominus S.$$

The morphological operations described so far can be compared with each other based on their effect on the host image as shown in Fig. 3.18. Hand gesture recognition research relies heavily on these fundamental operations when using computer vision to register gestures. This chapter will further discuss other morphological operations such as hit and miss transform, thickening, thinning followed by skeletonization as they are commonly used in hand gesture recognition research.

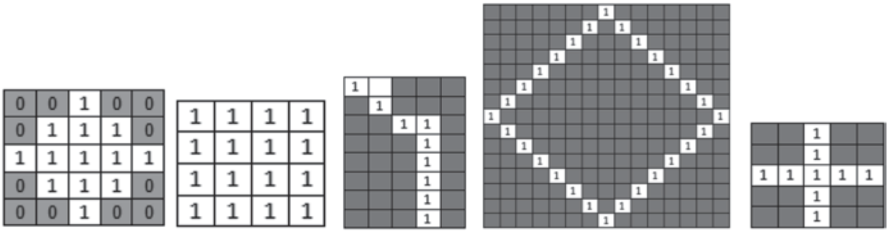


Fig. 3.19 Variety of structuring elements; disc, Square, irregular and asymmetric, very large structuring element and a cross. The *darkened squares* contain zero

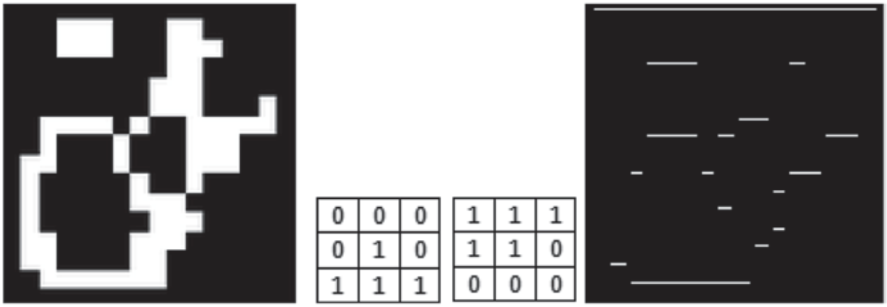
3.2.3 Structuring Element (SE)

A structuring element is a matrix consisting of only 0's and 1's that can have any arbitrary shape and size. The pixels with values of 1 define the neighborhood. One dimensional or two dimensional structuring elements are typically much smaller than the image being processed. The center pixel of the structuring element is known as the origin which identifies the pixel being processed. The pixels in the structuring element containing 1's define the neighborhood of the structuring element. 3D structuring elements use 0's and 1's to define the extent of the structuring element in the x- and y- axes with z signifying height values to define the third dimension. The operation of morphological filtering on binary images can be better understood by considering compound operations like opening and closing as filters. Their resemblance to filters of shape, opening with a disc shaped structuring element which smooths corners from the inside and closing with a disc results in smoothing corners from the outside. They also can filter out any image details that are smaller in size than the structuring element (e.g. opening is filtering the binary image at a scale defined by the size of the structuring element). Only those portions of the image that fit the structuring element are passed by the filter; smaller structures are blocked and excluded from the output image. The size of the structuring element is most important to eliminate noisy details but not to damage objects of interest.

The structuring elements do not have much restriction apart from the fact that they should not increase the energy of the resulting process. Any shape and size can be selected for structuring element. However, it would be advantageous to select a shape that would easily achieve the purported purpose in the morphological process. Some of the different shapes used are shown in Fig. 3.19.

3.2.4 Hit-and-miss Transform

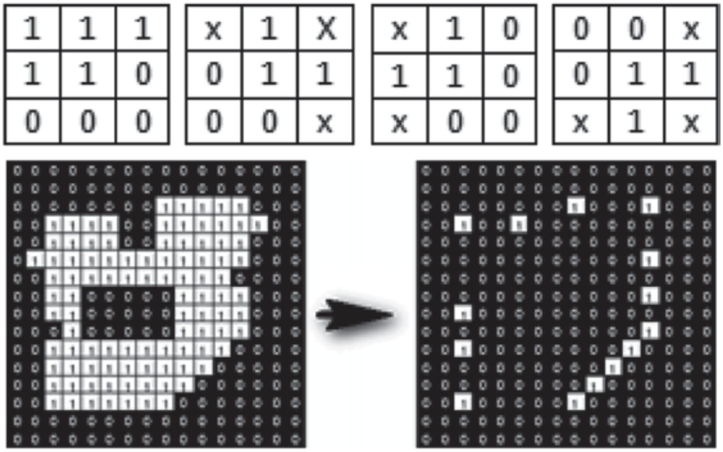
Hit-and-miss Transform is used to look for particular patterns of foreground and background pixels for very simple object recognition. It is well-known that all other morphological operations can be derived from it [57–59]. The transform operates by assessing whether the foreground and background pixels in the structuring



0	0	0
0	1	0
1	1	1

1	1	1
1	1	0
0	0	0

Fig. 3.20 Binary image developed in Fig. 3.17 operated on by two transforms to achieve the result shown on *right*



1	1	1
1	1	0
0	0	0

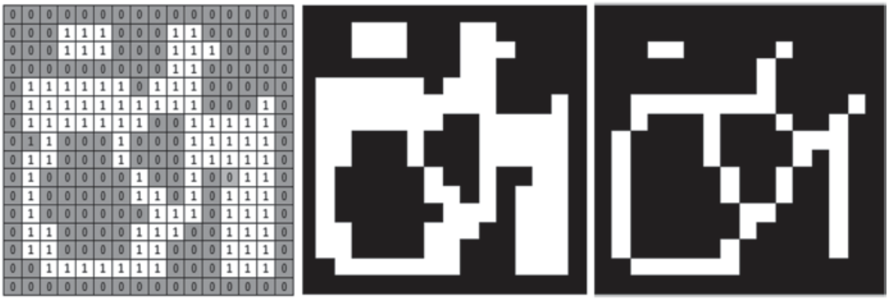
x	1	X
0	1	1
0	0	x

x	1	0
1	1	0
x	0	0

0	0	x
0	1	1
x	1	x

Fig. 3.21 Corner detection using hit-and-miss transform. The four transforms are shown on the *top* row with 'x' marking 'don't care' states. *Bottom* images show the original image transforming to corner detection where only the corner pixels remain

element exactly matches the foreground and background pixels in the image. If they match, then the pixel underneath the origin of the structuring element is set to the foreground color. The transform consists of 0's and 1's with usually a 1 at the origin. The transform matrix could also contain 'don't care' values which refers to either '0' or '1' which are not going to affect the outcome of the result significantly. An image can be operated on by more than one structural element one after the other. Figure 3.20 shows a binary image operated on by two structuring elements shown in the middle and the result on the right. Figure 3.21 shows how four miss-and-hit transforms can be used for corner detection on a binary image. In this, each transform operate on the input image and the results are 'OR'ed using logical processing to arrive at the final corner detection.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	1	1	0	0	0	0	0
0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	1	1	1	1	1	1	0	1	1	1	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	0	0	0	1	0
0	1	1	1	1	1	1	1	0	0	1	1	1	1	1	0
0	1	1	0	0	0	1	0	0	0	1	1	1	1	1	0
0	1	1	0	0	0	1	0	0	0	1	1	1	1	1	0
0	1	0	0	0	0	0	1	0	0	1	0	0	1	1	0
0	1	0	0	0	0	0	1	1	0	1	0	1	1	1	0
0	1	0	0	0	0	0	0	1	1	1	0	1	1	1	0
0	1	1	0	0	0	0	1	1	1	0	0	1	1	1	0
0	1	1	0	0	0	0	1	1	0	0	0	1	1	1	0
0	0	1	1	1	1	1	1	1	0	0	0	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 3.22 Thinning of a binary image. The image on the *left* shows the pixel arrangement where some regions are 4 pixels wide. The sections with 1 pixel width remain unchanged

3.2.5 Thinning

Morphological thinning is used to remove selected foreground pixels from binary images after edge detection where lines are often thicker than one pixel in width. Thinning will result in lines only one pixel wide. Hit-and-miss Transform can be used to perform thinning operation. In this approach, the effectiveness of thinning is determined by the structuring element [60, 61]. The mathematical definition of the thinning is given by the following relationship when using hit-and-miss transform:

$$Thin(I \text{ by } S) = I - HitandMiss(I, S)$$

Where logical subtraction is defined by A − B = A ∩ *NOT* B. The thinning of a binary image is shown in Fig. 3.22.

3.2.6 Thickening

Thickening is a morphological operation that is used to grow selected regions of foreground pixels in binary images similar to dilation or closing. It has several applications, including determining the approximate convex hull of a shape, and determining the skeleton by zone of influence [57–61]. Thickening is normally only applied to binary images, and it produces another binary image as output [58]. The definition of the Thickening can be given by the following relationship using Hit-and-Miss Transform:

$$Thicken(I \text{ by } S) = I \cup HitandMiss(I, S).$$

Thus the thickened image consists of the original image and any additional foreground pixels switched on by the hit-and-miss transform. Figure 3.23 shows the application of Thickening on a binary image.

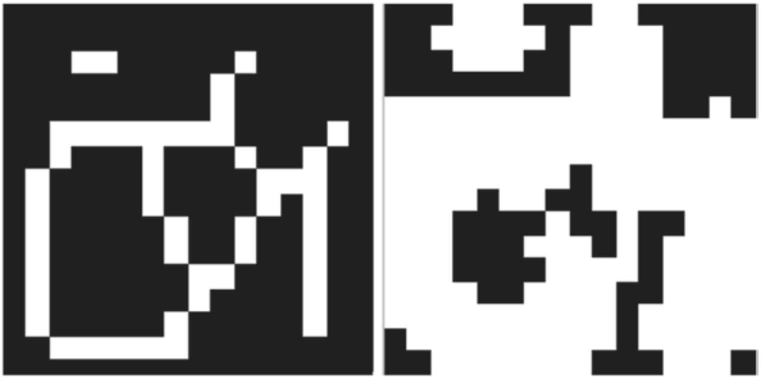


Fig. 3.23 Thickening of one pixel thick object (on the *left*). The result is shown on the *right*

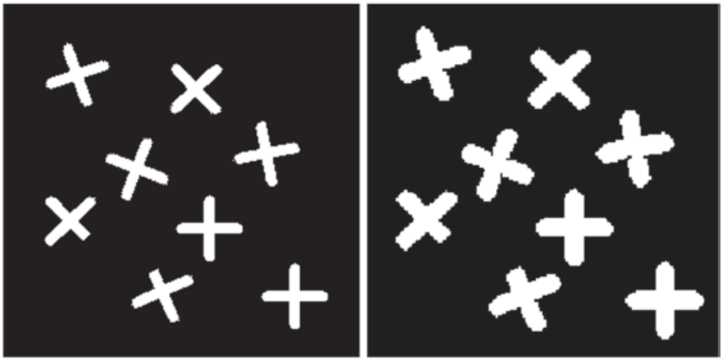


Fig. 3.24 Some objects retains shape if they are located adequately apart during the transformation

As was depicted in Fig. 3.23, the shape of the original object is somewhat obscured after thickening. However, this may not always be the case if the shapes are adequately located far apart and the SE is of specific size as shown in Fig. 3.24.

3.2.7 Skeletonization

Skeletonization is the process for reducing foreground regions in a binary image to a skeletal remnant that largely preserves the extent and connectivity of the original region. This in essence throws away most of the original foreground pixels. The skeleton is useful because it provides a simple and compact representation of a shape that preserves many of the topological and size characteristics of the original shape. This results in providing an approximate length of a shape by considering just the end points of the skeleton and finding the maximally separated pair of end

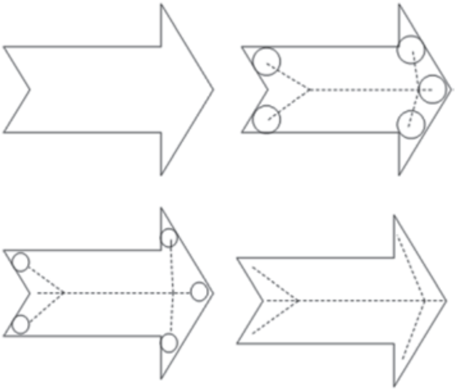


Fig. 3.25 Skeletonization is the process of continuously eroding of a structure (object) with ever decreasing structural element until it can be carried no further

points on the skeleton. Similarly, this will also lead to distinguishing many qualitatively different shapes from one another on the basis of how many 'triple points' there are (i.e. points where at least three branches of the skeleton meet).

Using the previous definition of erosion, skeletonization can be defined as the process where an object is eroded multiple times with ever decreasing size of structural element as follows:

$$Skeleton(I,S) = \bigcup_{k=0}^{K} I_k \ominus S_k$$

This process is illustrated in Fig. 3.25. Skeletonization is often used in text scanning to prune the thick edges so that optical character recognition and hand written recognition can be implemented in machines.

As with thinning, slight irregularities in a boundary will lead to spurious spurs in the final image which may interfere with recognition processes based on the topological properties of the skeleton. Figure 3.26 clearly illustrates this. Despurring or pruning can be carried out to remove spurs of less than a certain length but this is not always effective since small perturbations in the boundary of an image can lead to large spurs in the skeleton.

Skeletonization can result in a remarkable gesture identity if any gesture captured by camera can be turned into an accurate model. However, as was seen Fig. 3.26, skeletonization can result in much more complicated unintelligible realizations which offer no value for hand gesture recognition. However, recently, there have been few reported cases of research which were based on skeletonization of hand gestures for gesture recognition.

Ionescu and Coquin reported a hand gesture recognition method based on the 2D skeleton representation of the hand [61]. They represented each gesture with a hand skeleton and this skeleton was compared with a skeleton in a database for a match. They used Baddeley's distance [62], as a measure of dissimilarities between model parameters. Even though the results were promising yet, they suffered from

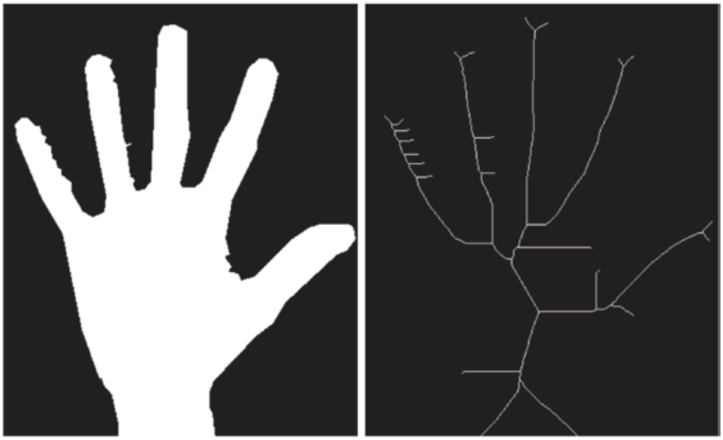


Fig. 3.26 Skeletonization in hand gesture recognition can sometimes lead to unforseen scenarios where even a slight imperfection on a binarized gesture can result in completely unintelligible results

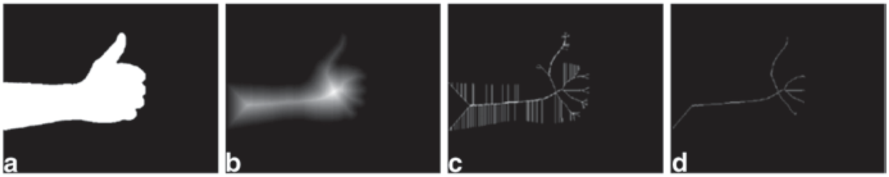


Fig. 3.27 Skeleton extraction: (**a**) hand region (binary image), (**b**) chamfer distance image (*white* corresponds to the greatest distance), (**c**) the skeleton obtained after connecting the centers of maximal discs, and (**d**) the skeleton obtained after spurious hole filling, pruning, and beautifying the previous skeleton. (Courtesy of [61])

occlusion and was limited to very few hand gestures. The directions of the camera were unconventional as it captured images from side which was unnatural for computer human interface as shown in Fig. 3.27.

Reddy et al. proposed an approach for calculating local orientation histograms of skeleton of the hand by using distance transformation techniques [63]. They relied on the local histograms as features due to their invariance to translation, rotation and scaling. Skeleton was computed for each and every hand posture in the entire hand motion and superimposed on a single image called as Dynamic Signature of the particular gesture type. Then the gesture was recognized by matching the image signature (features of local orientation) against the entries in the gesture alphabet. They used Image Euclidean distance measure as the metric to determine image similarities.

There are compelling reasons for using skeleton of the hand for gesture recognition. Skeletons provide compact representation of an object and preserve the topology of the object. Skeleton is robust against translation rotation and scaling

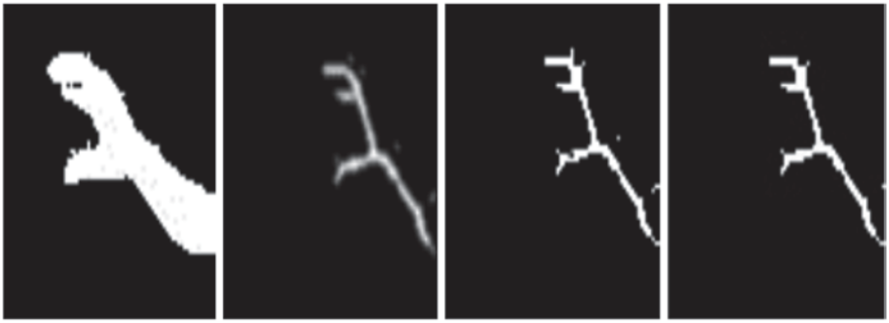


Fig. 3.28 Hand gesture skeletons for gesture recognition. (Courtesy of [63])

[64]. Skeleton is also extracted by using several methods such as chamfer distance transform [65], and morphological thinning [66] (Fig. 3.28).

Wu et al. in 2012 presented research using the skeleton of hand using depth information for hand gesture recognition [64]. They presented a method of recognizing hand gestures in the form of point clouds recorded by Kinect sensor. Initially, through Laplacian-based contraction and further processing, they extracted skeleton points from point clouds of hands. Then a novel partition-based descriptor and correspondence algorithm was applied to classify these skeletons and therefore to recognize gestures. In the process of recognition, the issue of scale variance and rotation variance were solved. They used 3D models downloaded from Princeton 3D Model Search Engine to be standard gestures, then record gestures using Kinect sensor. The recognition accuracy for 12 gestures was about 85 % on average. They finally verified their claims using performance analysis where the results proved both its accuracy and robustness. They demonstrated that skeleton-based method of recognition has great potential for further exploration. Figure 3.29 shows the stages of gesture skeletonization and their 12 gestures in Fig. 3.30.

3.3 Gesture Extraction Using Color and Depth Information

One of the major challenges in gesture recognition is to reliably capture the gesture alone from the clutter of the background. This is a non-trivial task as it has been shown over the years [68–74]. As was discussed in the previous section of skin segmentation, skin detection tries to separate the gesture from the background. However, this problem is compounded when the background contains skin-tone regions. Since the cameras are essentially 2D devices unlike the human eye, there is no information a camera can supply to separate hand gesture from another person in the background. However, if a stereo vision or another setup that detects

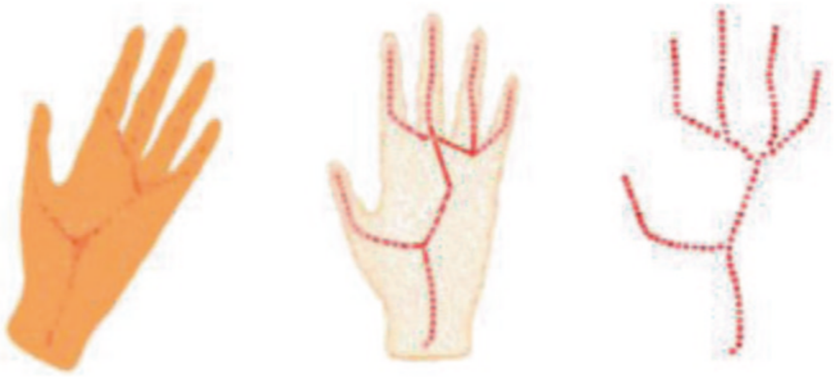


Fig. 3.29 Key points achieved from Laplacian-based contraction and index-based compaction. (Courtesy of [64])

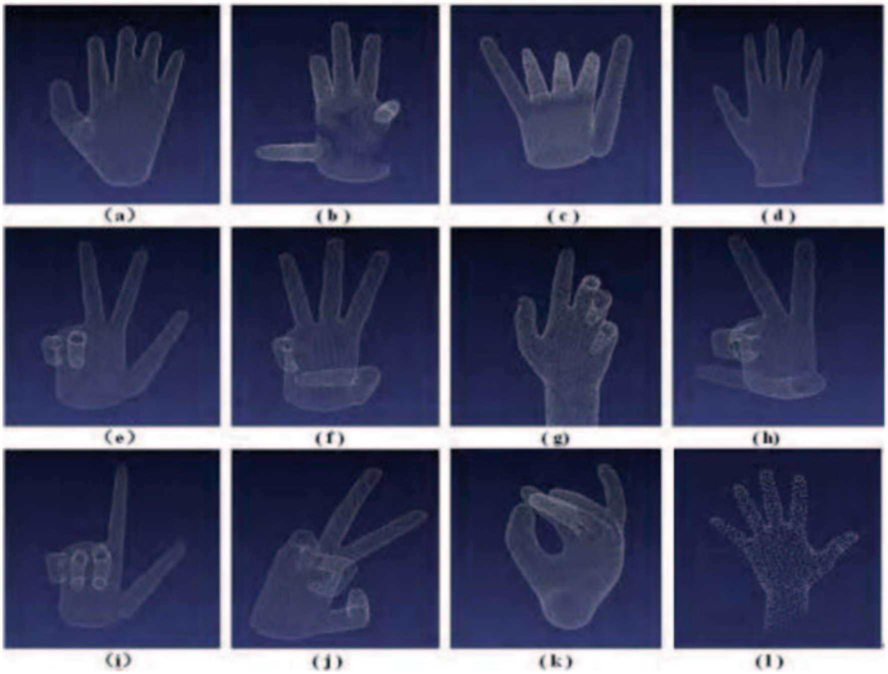


Fig. 3.30 Twelve cloud gestures used by Wu et al. (Courtesy of [67])

depth can be used, the complexity of the problem unravels as depth to the objects becomes available.

Yet, this section will discuss why this depth information alone is not reliable for background-foreground separation based on research carried out over the past 15 years. Very recently, there has been a glimmer of hope due to new breed of in-

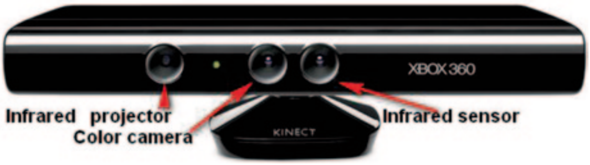


Fig. 3.31 Kinect Camera developed for Microsoft Xbox

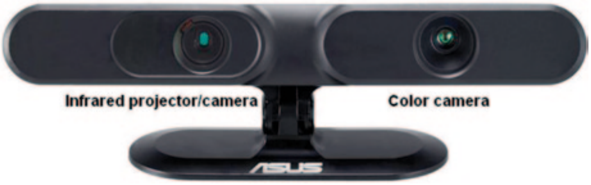


Fig. 3.32 Asus Xtion camera which has identical imaging capabilities to that of Kinect but with a personal computer compatible USB interface

expensive consumer grade cameras which are increasingly being used in an effort to retrieve depth information. Some of these devices are Kinect from Microsoft an Asus Xtion (both are manufactured by the same Taiwanese company with similar capabilities). Instead of stereo vision setup, these camera relies on infradred structured lighting projection and image capture through both infrared and color camera. The technology behind these cameras differs from the traditional depth camera; stereo vision. The novel technology is based on structured lighting which a well-understood phenomenon that is used in stereoscopy [75, 76]. The distortion pattern of the projected infrared structured light pattern is observed by the infrared camera to detect the depth to the objects and this information is fused with color image information so that every pixel has a depth parameter. Previously, non-stereoscopy systems relied on Time of Flight (TOF) cameras which have been confined to high end research due to their prohibitive cost. Currently, the next generation of Kinect camera is going to be released at the end of 2013 and is equipped with TOF technology opposed to Infrared light projection and the switch has been due to some of the limitations especially in resolution of infrared sensors compared to the CMOS imaging sensor.

Figure 3.31 depicts the Kinect camera with its onboard infrared projector, infrared receiver and color CMOS camera. The CMOS and the Infrared sensor both have a resolution of 640 × 480 at 30 fps. However, its depth perception is confined to 320 × 240. This results in many visual pixels not having proper depth information leading to edge anomalies in depth-color view. The Asus Xtion also has the same resolution in its sensors which is shown in Fig. 3.32. However, their physical appearance differ due to the Kinect having panning capability where as Xtion is simply has a front facing configuration.

Recently, there has been increased interest in applications of computer vision to traffic monitoring on highways to security surveillance in restricted areas. One of the preliminary tasks in such applications is to extract the foreground or objects from the background. Many early works relied on *background subtraction* which would simply look for the image difference before and after objects have

Fig. 3.33 Swiss Ranger 4000 by Mesa Imaging

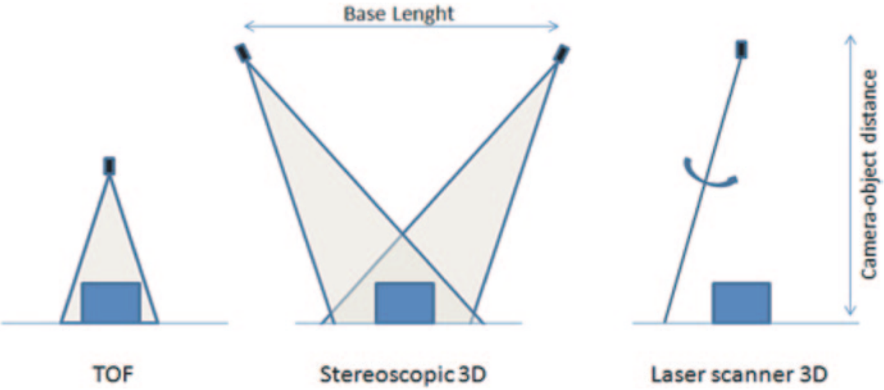


Fig. 3.34 TOF camera uses only one camera and needs a lesser distance from the camera to the object as shown on the *left*. Stereoscopy and laser scanners need more camera-object distance to be effective

been observed. As it turns out, same image sensor would produce slightly different picture with incrementally small color variations and noise when imaging an object few seconds apart. This problem is also compounded when natural lighting changes in day and temperature differences due to wind at night. Hence, simple subtraction of two images will not result in the foreground being revealed. It would contain undesirable sections of the background that would lead to false positives if decisions are made immediately without further elaborate processing. Such discrepancies in imaging sensors and technologies have called for more advanced hardware that would tackle some of the issues mentioned above.

Swiss high tech company Mesa Imaging had developed a TOF camera which dominated the market for many years for commercial imaging equipment that provided basic hardware as shown in Fig. 3.33. As shown in Fig. 3.34, TOF camera technology stands out from stereoscopy and laser scanning technology. Laser scanning technology has never been used for human surveillance as it is objectionable as a safe mode of information gathering due to high intensity lasers being used that

Fig. 3.35 Kinect II with its TOF technology

would cause eye damage. Stereoscopy devices need extensive special arrangement which is not suitable for the above applications. TOF technology stands out from these technologies however, they offer limited resolution opposed to the massive visual resolution offered by modern camera sensors. Therefore, the alignment of depth information with their visual counterpart usually results in more error prone low resolution scenario.

In 2009, ZCam, a company which developed TOF technology to develop a camera gesture interface to use human gestures to engage with gaming activities was taken over by Microsoft. It is rumoured that ZCam technology has enabled the Microsoft to develop a more advanced Kinect to use TOF camera technology at an extremely low cost compared to what has been commercially available from vendors such as Mesa Imaging. Kinect II released in the second half of 2013 is shown in Fig. 3.35. This is a very positive move for gaming enthusiasts as well as researchers in computer vision as Microsoft has a tendency to develop technology for mass market at reasonable costs. Its depth perception will increase from current 320 × 240 to 512 × 424 pixels which would be very valuable for emerging applications.

3.3.1 Image Registration

Image registration is the process of aligning two-dimensional images to a different three-dimensional space. In the case of a 3D camera, the registration process aligns the depth and colour streams together so that operations on either stream can then be related to the other stream. When the distance between the two perspectives of each camera (IR and Colour) is known, an approximation between pixels in each frame is determined. That is, if an object is closer to the camera (as known by the Depth component), the offset of that pixel to a pixel in the colour image array is larger than an object further away. There are numerous techniques for completing this operation, as listed below.

3.3.1.1 Edge/Key-Point Detection

A major option for image registration is the selection of key points, edges, surfaces or objects, then transferring those into another reference point. There are a number

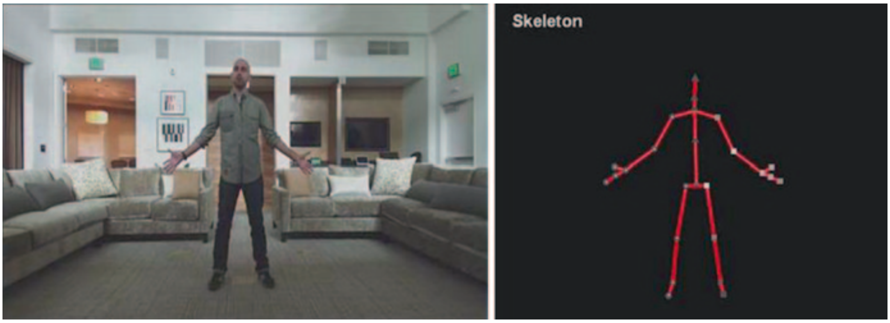


Fig. 3.36 Kinect markerless motion capture produced by BerkelTools

of methods for segmenting images into various objects which differ in complexity and accuracy [77]. Some methods prefer to isolate the different objects by locating edges or distinct rapid changes in the image [78]. Other methods search for continuous regions of consistent surfaces and segment within that section [79]. For greater accuracy, both methods can be combined for a hybrid-style algorithm. This method of key-point identification is found in ORB [80] and BRISK [81]. Another consideration of key-point locating involves searching for corners of objects in a scene, as these represent the orientation and boundary of an object, as covered by Rosten's work on image mapping [82]. All three options are available as part of the OpenCV computer vision library [83].

While often considered in the process of stitching together separate two-dimensional images to infer depth information of the scene, this method can also be used with a depth stream. The intention has been to improve the accuracy of the registration between an object in the colour stream and the comparative depth stream [84]. The key-point referencing method used was comparable to the results produced by the PrimeSense method coordinated by the camera. Some of the capabilities of Kinect II combined with BerkelTools offer new mode of gaming environment as shown in Fig. 3.36.

3.3.2 *Stereo Triangulation for Depth Estimation—Passive Stereo Vision*

Stereo vision is the concept of determining the lengths and sizes of objects in 3D space as done by humans and many other animals. It is a process which has been inspired by natural world where most of the fauna having two eyes. Mimicking such a system, engineers use two cameras which are few centimetres apart to create two slightly different views of the same scene (or object(s)). As shown in Fig. 3.37, in the ideal case of epipolar geometry which describes the mechanism of stereo vision, the *dot* in the diagram produces A_1 and A_2 in two camera views. However,

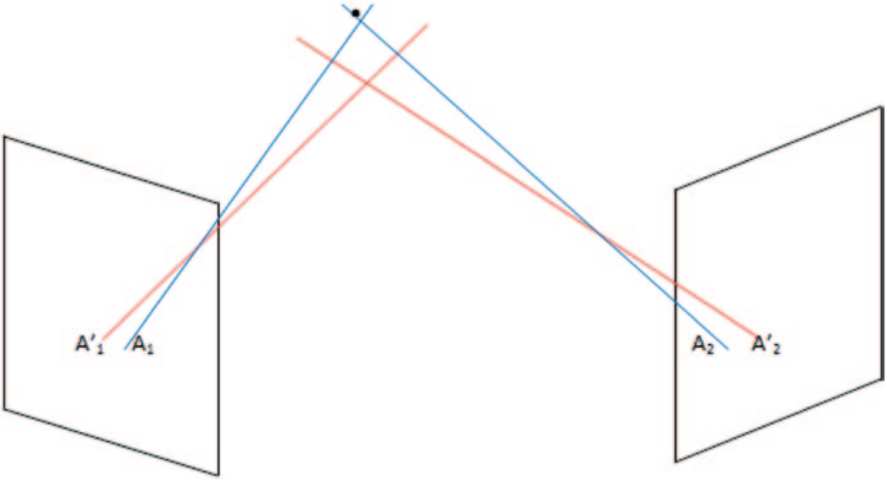


Fig. 3.37 Stereo vision disparity as seen by two cameras

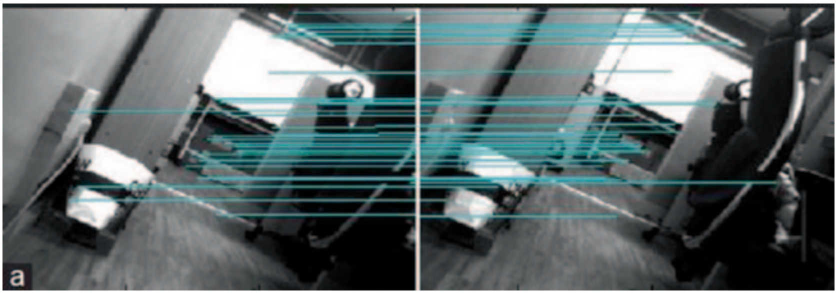


Fig. 3.38 Stereo disparity and correspondence. The *green* line connects the features identified in the *left* camera with the matching feature on the *right*

due to many imperfections in this system such as camera focussing issues, instead of the exact intersection of the two blue lines where the *dot* is, only measurements of A’$_1$, A’$_2$ are mostly available and used for triangulation. If the focal lengths and the camera and the distance between cameras are known (these are well-known for any stereo setup) the distance of the camera viewing plane to the *dot* (object) can be estimated using basic algebra. However, since ordinary images have complex scenery opposed to well-defined points, many other factors come into play reducing the applicability of simple stereovision in many imaging application. Figure 3.38 shows an image pair used in stereovision based robotic navigation attempt. These two images, even though are almost identical have been captured by a stereo camera pair with a slight change in point of view. Unlike Fig. 3.37 (triangulation diagram), the scenery has many points of interest. This issue now leads to identifying correspondence between each point, seen on the left camera with that of the right camera,

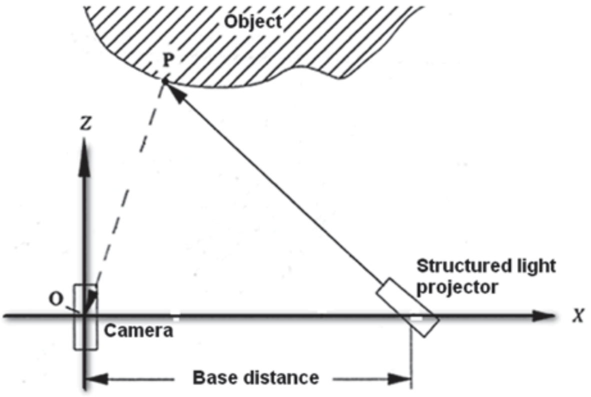


Fig. 3.39 Depth measurement using structured lighting with a single camera and a projector

known as the correspondence problem. The most prominent approach to find the correspondence relies on correlation between identical rows and constraint known as disparity that regulates mismatches [85–88].

Stereo vision often fails when used in featureless or textureless surroundings such as snow or highly repetitive patterns or uniform surroundings. Furthermore, if the forward looking cameras would not find any nearby imaging surfaces, it would also fail. Due to these limitations, stereovision along is not used for navigation especially in outdoor surroundings. However, stereovision does offer viable solutions for computer human interaction which usually takes place in indoors. Yet, the amount of processing power needed to solve the correspondence problem has dissuaded the commercial applications as seen by the investments of Microsoft on alternative technology that would be discussed next.

3.3.3 Active Stereo Vision-Coded Structured Light

Active stereo vision refers to the set up where instead of two cameras are setup to acquire images, a light pattern projector replaces one camera as shown in Fig. 3.39. In structured light imaging, a predefined light pattern is projected onto an object and simultaneously observed by a camera. The appearance of the light pattern in a certain region of the camera image varies with the camera-object distance. This effect is utilized to generate a distance image of the acquired scene. The predefined light patterns can be generated using many approaches as would be discussed in the next section. Some setups may use two cameras or multiple cameras to reduce the likelihood of occlusion by the object being imaged. Since this light pattern is visible to the human eye, such stereo systems are objectionable when used in public places such as airports for 3D face recognition or other types of surveillance [76].

In coded structured light, a light pattern is coded so that correspondences between image points and points of the projected pattern can be easily found. Therefore, coded structured light is considered to one of the most reliable ways for re-

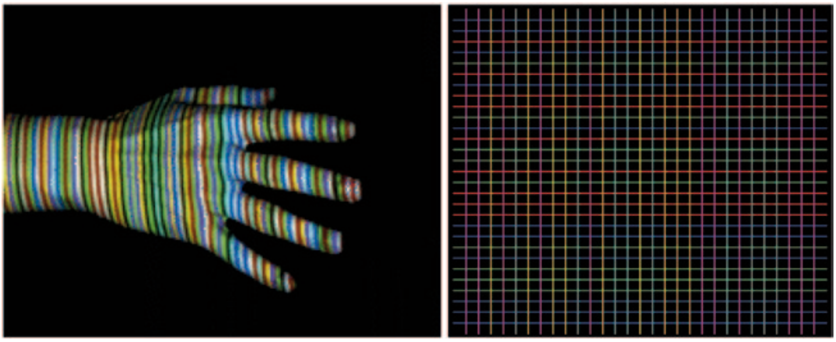


Fig. 3.40 125 slits encoded with a De Bruijn sequence of 8 colors and window size of 3 slits (*left*), courtesy of [99]. De Bruijn series spatial codification pattern (shown on *right*) [100]

covering the surface of objects [89]. Once the code pattern on the object is captured and decoded, the decoded points can be triangulated for 3D coordinates to recover the structure.

There are three prominent approaches for light pattern coding in practice. Time-multiplexing which is commonly used, is based on temporal coding. In this approach, a set of patterns are successively projected on the surface being imaged and captured at the same time. The codeword for any pixel is the result of multiplexing of the sequence of projected patterns on to that pixel. The codeword generation can be realized using, binary codes, *n-ary* codes, gray code combined with phase shifting and using hybrid techniques which are combination of time-multiplexing and neighbourhood strategies [90–93]. Time-multiplexing results in dense 3D points and high accuracy compared to other approaches. It is also suitable for objects with color as binary or *n-ary* codes are resilient against color objects. However, due to its reliance on multiple projections, the approach is limited to static objects.

The second techniques for light pattern coding are the approach based on spatial codification. The techniques used in this category generate a unique global pattern. The codeword for a single pixel can be determined by observing the pattern for its neighbors [94]. However, occlusions or non-neutral colors can lead to errors as not all neighborhood pixel patterns can be reliably retrieved. Some of these spatial neighbourhood strategies include De Bruiin sequences, *M-arrays* and strategies based on non-formal codification [95–98]. The technique is applicable to dynamic objects. Figure 3.40 shows a De Bruijn series coding pattern and how it is used for 3D depth measurements.

The third technique in light pattern generation is *direct codification*. In this approach, each pixel gets its own color (color intensity) to represent the pattern [101–106]. However, the observed color from any pixel does not solely depend on the projected color. It also depends on the color of the surface. Since different color objects reacts to colors differently, this strategy is only suitable for objects with neutral color object or objects with pale colors [94]. Some implementations

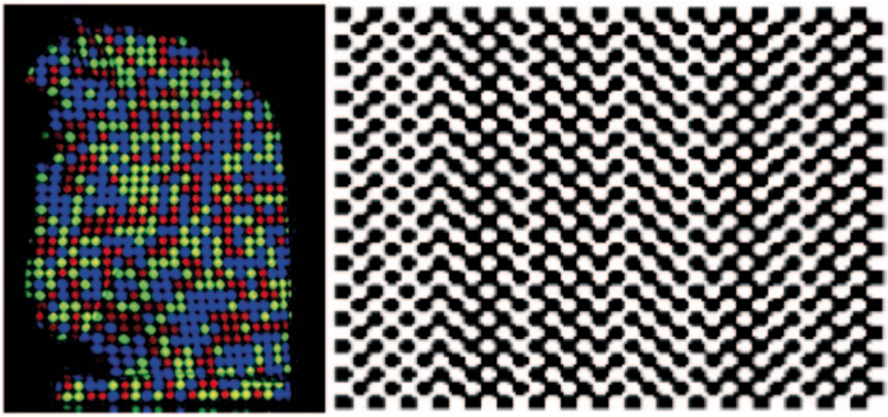


Fig. 3.41 M-arry represented with an array of *coloured dots* (*left*), courtesy of Morano et al. [107]. M-array proposed by Vuylsteke et al. represented with shape primitives [100]

of this direct codification relies on capturing many reference frames with change of colors. Therefore this is not generally used for dynamic scenes.

Even though objectionable to the human user, structured light based stereo vision can be very effective for 3D scanning in indoors. Many researchers have used different types of grid patterns and coded color schemes to determine the depth of the objects using the observed distortion due to object depth and shape of the predefined pattern. Such color schemes and code patterns as shown in Fig. 3.41 can help in reducing the correspondence problem if more than two cameras are used. However, since the light diffracts much more than a laser beam, the resolution from visible light based stereo vision is limited.

3.3.4 Infrared Structured Light for Active Stereo Vision

Recently, researchers have developed invisible infrared lighting patterns or structured light to obtain depth information without any public backlash. This has resulted in the successful Microsoft Kinect using infrared structured light patterns to develop low cost, yet, effective gaming apparatus. The light pattern used in Kinect is known as a speckle pattern which resembles speckle noise. However, it is a well structured pattern, projected using an infrared laser through a plastic material which houses the pattern. The distortion of this pattern is compared with the original speckle pattern to determine the depth of the objects. One of the problems of this approach which is also common to the visible structured light is the shadows created by occlusion of objects. In addition to the shadows, the depth-images contain noise due to limited resolution of the IR camera. Since the vision camera is of much higher resolution, a single infrared point of the captured pattern may not be

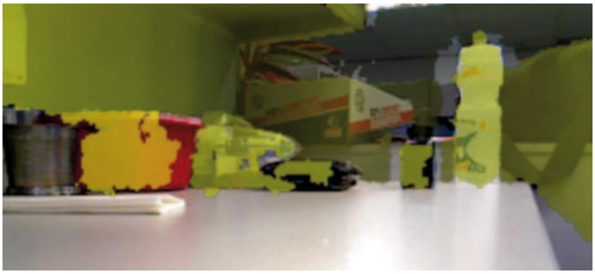


Fig. 3.42 Color Camera view with depth information fused using Kinect

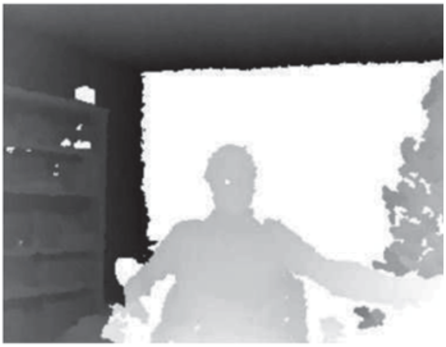


Fig. 3.43 Depth information is easy to ascertain in a gray-scale image fused with depth information using kinect

assigned to a single pixel. Therefore the position must be interpolated by the portion of luminosity of the two adjacent pixels. This interpolation is sensitive to external infrared radiation such as direct sun light. For many devices such as Kinect, increasing distance from the camera leads to poor resolution and errors due to misalignment of infrared pixels with CMOS pixels. There are other errors due to offset of the CMOS and the Infrared receiver. There are methods for calibrating the camera to modify the image, such that the depth map from the infrared camera and CMOS input are aligned [108–110]. The scene reconstruction then follows a process used in 3D game design known as texturing. The model generated from the depth map data is "painted" with a texture, in this case, the video input from the CMOS camera as shown in Figs. 3.42 and 3.43. This can be completed in real-time through the use of modern graphics processors.

Kinect uses an infrared speckle as shown in Fig. 3.44. The known pattern is compared with its offset when the pattern falls on a surface and distorted. This disparity measure results in triangulation to reveal the depth map. In Kincet, the color visual image sensor has much higher pixel density than its infrared sensor. This results in false depth map that is not really an issue in Kinects applications of gaming. However, since it is difficult to improve the resolution of the depth map using infrared speckle patterns, Microsoft has resorted to TOF camera technology for their upcoming Kinect II as was discussed before.

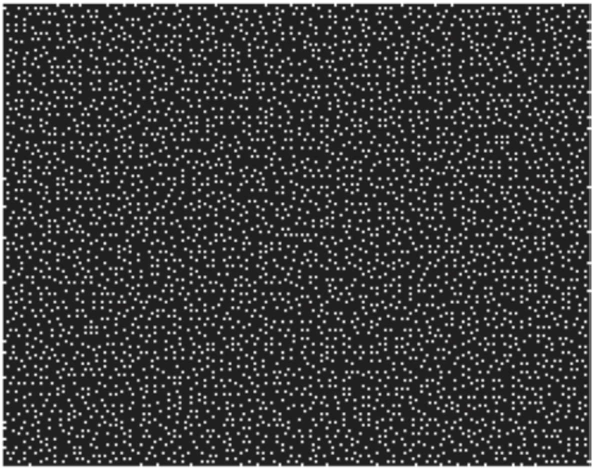


Fig. 3.44 Speckle pattern used in Kinect. (Courtesy of [111])

3.3.5 Time of Flight (TOF) Camera for Depth Information

The advantage of TOF compared to triangulation methods in passive or active stereo vision is that the whole system is very compact where the illumination (pattern projector) is placed just next to the camera lens, whereas the other systems need a certain minimum base line. In contrast to laser scanning systems, no moving parts are present in the system.

The working principle of the TOF camera can be understood as using extremely short or narrow light pulses to illuminate a target (any object in its flying path) and record the return of the pulse on every pixel on the camera sensor with its time of arrival. This is done extremely precisely so that even 1 cm depth differences of any part of the object being scanned can be differentiated. The pulse width of the illumination determines the maximum range the camera can handle. In case of pulse width of 50 ns, the scanning range is restricted to 7.5 m. These short times show that the illumination unit is a critical part of the system. These short pulses can currently be generated with special LEDs or lasers.

When infrared structured light is used in the presence of background light, the CMOS camera sensor receives an additional part of the signal. This results in disturbing the distance measurement. In order to eliminate the background contribution of the signal, the whole measurement can be performed a second time with the infrared illumination switched off. If the objects moves and are further away than the distance range, the measurements result in error. Here, a second measurement with the control signals delayed by an additional pulse width helps to suppress such objects. Other systems work with a sinusoidally modulated light source instead of the pulse source.

As mentioned previously in this chapter, the ZCam's time-of-flight camera system features a near-infrared (NIR) pulse illumination component as well as an image sensor with a fast gating mechanism. Based on the known speed of light, ZCam coordinates the timing of NIR pulse wave emissions from the illuminator with the gating of the image sensor so that the signal reflected from within a desired depth range is captured exclusively. The amount of pulse signal collected for each pixel corresponds to where within the depth range the pulse was reflected from, and can thus be used to calculate the distance to a corresponding point on the captured subject [112, 113].

Due to the fast timing required for light-based time-of-flight, the ZCam uses custom hardware for illumination and gating. The illuminator is a series of NIR laser diodes around the lens barrel, switched by special high-speed driver circuits that produce pulses with a rise time and fall time of less than 1 ns [113]. The time-of-flight camera is optically matched with a corresponding video camera, allowing the RGB video and range imaging to integrate together.

This chapter methodically developed the required knowledge for preprocessing that is vital in understanding object shapes. When undesirable noise and artefacts are present, morphological filtering based processing can restore objects so that they can be understood by computer vision. With the use of depth information, cluttered backgrounds can be removed to reveal the foreground which typically contains the hand gestures for human computer intearaction.

References

1. Abdel-Mottaleb, M., Elgammal, A.: Face detection in complex environments from color lmages. Proceedings of the International Conference on Image Processing (ICIP), 622–626 (1999)
2. Alshebani, Q., Premaratne, P., Vial, P.: An Embedded Door Access Based on Face Recognition System: A Survey. To appear in (ICSPCS), 2013, Australia, (2013)
3. Ahmed, E., Crystal, M., Dunxu H.: Skin Detection-a short Tutorial. Encyclopedia of Biometrics by Springer-Verlag Berlin, Heidelberg, 1218–1224 (2009)
4. Forsyth, D.A., Fleck, M.M.: Identifying nude pictures. Proceeding of Third IEEE Workshop on Applications of Computer Vision, 103–108 (1996)
5. Albiol, A., Torres, L., Delp, E.: Optimum color spaces for skin detection. In: Proceedings of the International Conference on Image Processing (ICIP), 122–124 (2001)
6. Shin, M.C., Chang, K.I., Tsap, L.V.: Does colorspace transformation make any difference on skin detection? WACV '02: Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision, 275 (2002)
7. Zheng, Q.F., Zhang, M.J., Wang, W.Q.: A hybrid approach to detect adult web images. PCM 2 3332, 609–616 (2004)
8. Lee, Y., Yoo, S.I.: An elliptical boundary model for skin color detection. In: Proceedings of the International Conference on Imaging Science, Systems, and Technology, (2002)
9. Senior, A., Hsu, R.L., Mottaleb, M.A., Jain, A.K.: Face detection in color images. IEEE Trans. PAMI 24(5), 696–706 (2002)
10. Menser, B., Wien, M.: Segmentation and tracking of facial regions in color image sequences. Proceeding of SPIE Visual Communications and Image Processing, 731–740 (2000)
11. Jones, M.J., Rehg, J.M.: Statistical color models with application to skin detection. In: Proceeding of CVPR'99 1, 274–280 (1999)
12. Beetz, M., Radig, B., Wimmer, M.: A person and context specific approach for skin color classification. 18th International Conference on Pattern Recognition (ICPR 2006), (2006)
13. Soriano, M., et al.: Skin detection in video under changing illumination conditions. 15th International Conference on Pattern Recognition, (2000)

14. Kawato, S., Ohya, J.: Automatic skin-color distribution extraction for face detection and tracking. 5th International Conference on Signal Processing Proceedings (WCCC-ICSP 2000), (2000)
15. Park, J., et al.: Detection of human faces using skin color and eyes, IEEE International Conference on Multimedia and Expo (ICME 2000), (2000)
16. Kovac, J., Peer, P., Solina, F.: 2D versus 3D color space face detection. 4th EURASIP Conference on Video/Image Processing and Multimedia Communications, 449–454 (2003)
17. Gomez, G., Morales, E.F.: Automatic feature construction and a simple rule induction algorithm for skin detection. Proceedings of ICML workshop on Machine Learning in Computer Vision, 31–38 (2002)
18. Gasparini, F., Schettini, R.: Skin Segmentation using Multiple Thresholding. Proceedings of SPIE 6061, 128–135 (2006)
19. Vezhnevets, V., Sazonov, V., Andreeva, A.: A Survey on Pixel-Based Skin Color Detection Techniques, In Proceedings of GRAPHICON-2003, (2003)
20. Zarit, B.D., Super, B.J., Quek, F.K.H.: Comparison of five color models in skin pixel classification. ICCV'99 Int'l Workshop on recognition, analysis and tracking of faces and gestures in Real-Time systems, 58–63 (1999)
21. Hsu, R.-L., Abdel-Motalleb, M., Jain, A. K.: Face detection in color images. IEEE Trans. PAMI **24**(5), 696–706 (2002)
22. Ahlberg, J.: A system for face localization and facial feature extraction. Technical Report no. LiTH-ISY-R-2172, Linkoping University, (1999)
23. Sebastian, P., Yap, V.V., Comley, R.: The effect of colour space on tracking robustness. 3rd IEEE Conference on Industrial Electronics and Applications (ICIEA 2008), 2512–2516 (2008)
24. Tsekeridou, S., Pitas, I.: Facial feature extraction in frontal views using biometric analogies. Proceedings of IX European Signal Processing Conference 1, 315–318 (1998)
25. Garcia, C., Tziritas, G.: Face detection using quantized skin color regions merging and wavelet packet analysis. IEEE Transaction on Multimedia. **1**, 264–277 (1999)
26. Poynton, C.A..: Frequently Asked Questions About Colour. In ftp://www.inforamp.net/pub/users/poynton/doc/colour/ColorFAQ.ps.gz (1995)
27. Skarbek, W., Koschan, A.: Colour image segmentation—a survey. Technical Report, Institute for Technical Informatics, Technical University of Berlin, (1994)
28. Sigal, L., Sclaroff, S., Athitsos, V.: Estimation and prediction of evolving color distributions for skin segmentation under varying illumination. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2, 152–159 (2000)
29. Mckenna, S., Gong, S., Raja, Y.: Modelling facial colour and identity with gaussian mixtures. Pattern Recognit 31, **12**, 1883–1892 (1998)
30. Jordao, L., Perrone, M., Costeira, J., Santos-Victor, J.: Active face and feature tracking. In Proceedings of the 10th International Conference on Image Analysis and Processing, 572–577 (1999)
31. Fleck, M., Forsyth, D.A., Bregler, C.: Finding nacked people. In Proceedings of the ECCV 2, 592–602 (2002)
32. Brown, D., Craw, I., Lewthwaite, J.: A som based approach to skin detection with application in real time systems. In Proceedings of the British Machine Vision Conference, (2001)
33. http://en.wikipedia.org/wiki/HSL_and_HSV
34. Terrillon, J.-C., Shirazi, M.N., Fukamachi, H., Akamatsu, S.: Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In Proceedings of the International Conference on Face and Gesture Recognition, 54–61 (2000)
35. Poynton, C., Funt, B.: Perceptual uniformity in digital image Representation and display. Color Research and Applications, (2013)
36. Kaur, A., Kranthi, B.V.: Comparison between YCbCr color space and CIELab color space for skin color segmentation. Int. J. Appl. Info. Syst. **3**(4), 30–33 (2012)
37. Singh, S.K., Chauhan, D.S., Mayank, V., Singh, R.: A robust skin color based face detection algorithm. Tamkang J. Sci. Engg. **6**(4), 227–234 (2003)

38. Khan, R., Khan, Z., Aamir, M., Sattar, S.Q.: Static filtered skin detection. IJCSI International Journal of Computer Science Issues. **9**(2), 257–261 (2012)
39. Poudel, R.P.K., Nait-Charif, H., Zhang, J.J., Liu, D.: Region-based skin color detection. VISAPP 1, 301–306 (2012)
40. Hikal, N.H., Kountchev, R.: Skin color segmentation using adaptive PCA and modified elliptic boundary model. ICACSIS. **2011**, 407–412 (2011)
41. Chen, Q., Wu, H., Yachida, M.: Face detection by fuzzy pattern matching. In Proceedings of the Fifth International Conference on Computer Vision, 591–597 (1995)
42. Schumeyer, R., Barner, K.: A color-based classifier for region identification in video. Vis. Commun. Image Process. SPIE. **3309**, 189–200 (1998)
43. Birchfield, S.: Elliptical head tracking using intensity gradients and color histograms. In Proceedings of CVPR '98, 232–237 (1998)
44. Yang, M.H., Ahuja, N.: Detecting human faces in color images. In International Conference on Image Processing 1, 127–130 (1998)
45. Kruppa, H., Bauer, M., Schiele, B.: Skin patch detection in real-world images. In: Van Gool, L. (ed.), Pattern Recognition, Lecture Notes in Computer Science **2449**, 109–116 (2002)
46. Chang, F., Ma, Z., Tian, W.: A region-based skin color detection algorithm advances in knowledge discovery and data mining. Lecture Notes in Computer Science 4426, 417–424 (2007)
47. Ren, X., Malik, J.: Learning a classification model for segmentation. In IEEE International Conference on Computer Vision 1, 10–17 (2003)
48. Moore, A.P., Prince, S., Warrell, J., Mohammed, U., Jones, G.: Superpixel lattices. In IEEE Conference on Computer Vision and Pattern Recognition, 1–8 (2008)
49. Soatto, S.: Actionable information in vision. In Proceedings of the International Conference on Computer Vision 25, 17–48 (2009)
50. Fulkerson, B., Vedaldi, A., Soatto, S.: Class segmentation and object localization with superpixel neighborhoods. In Proceedings of International Conference on Computer Vision 5, 670–677 (2009)
51. Brand, J., Mason, J.: A comparative assessment of three approaches to pixellevel human skin-detection. In Proceedings of the International Conference on Pattern Recognition 1, 1056–1059 (2000)
52. Soille, P.: Morphological Image Analysis Principles and Applications, 2nd ed., XVI, 391 (2003)
53. www.cs.princeton.edu/~pshilane/class/mosaic/
54. Smith, S.W.: The Scientist and Engineer's Guide to Digital Signal Processing, Chap. 25.
55. www.mmorph.com/html/morph/mmopen.html/
56. Gonzalez, R., Woods, R.: Digital Image Processing, Addison-Wesley Publishing Company, 518–548 (1992)
57. Davies, E.: Machine Vision: Theory, Algorithms and Practicalities, Academic Press, 149–161 (1990)
58. Haralick, R., Shapiro, L.: Computer and Robot Vision 1, Addison-Wesley Publishing Company, Chap. 5, 168–173 (1992)
59. Jain, A.: Fundamentals of Digital Image Processing, Prentice-Hall, Chap. 9. (1989)
60. Vernon, D.: Machine Vision, Prentice-Hall, Chap. 4 (1991)
61. Ionescu, B., Coquin, D.: Dynamic hand gesture recognition using the skeleton of the hand. EURASIP J. Appl. Signal Process. **13**, 2101–2109 (2005)
62. Coquin, D., Bolon, P.: Applications of Baddeley's distance to dissimilarity measurement between gray scale images. Pattern Recognit. Lett. **22**(14), 1483–1502 (2001)
63. Reddy, K.S., Latha, P.S., Babu, M.R.: Hand Gesture Recognition Using Skeleton of Hand and Distance Based Metric, D.C. Wyld et al. (eds.) ACITY 2011, CCIS, 198, 346–354 (2011)
64. Borgefors, G.: Distance transformations in digital images. Comp. Vis. Graphics Image Process. **34**(3), 344–371 (1986)
65. Chehadeh, Y., Coquin, D., Bolon, H.: A skeletonization algorithm using chamfer distance transformation adapted to rectangular grids. In: Proceedings of 13th IEEE International Conference on Pattern Recognition (ICPR 1996) 2, 131–135 (1996)

66. Hasthorpe, J., Mount, N.: The generation of river channel skeletons from binary images using raster thinning algorithms. School of Geography, University of Nottingham
67. Wu, S., Jiang, F., Zhao, D.: Hand Gesture Recognition based on Skeleton of Point Clouds. 2012 IEEE fifth International Conference on Advanced Computational Intelligence (ICACI), 566–569 (2012)
68. Premaratne, P., Ajaz, S., Premaratne, M.: Hand Gesture Tracking and Recognition System Using Lucas-Kanade Algorithm for Control of Consumer Electronics. Neurocomputing Journal, (2012)
69. Premaratne, P., Nguyen, Q.: Consumer electronics control system based on hand gesture moment invariants. IET Comp. Vis. **1**(1), 35–41 (2007)
70. Zou, Z., Premaratne, P., Premaratne, M., Monaragala, R., Bandara, N.: Dynamic hand gesture recognition system using moment invariants. 5th International Conference on Information and Automation for Sustainability, 108–113 (2010)
71. Herath, D.C., Kroos, C., Stevens, C.J., Cavedon, L., Premaratne, P.: Thinking head: Towards human centred robotics. 11th International Conference on Control, Automation, Robotics and Vision (ICARCV), 2042–2047 (2010)
72. Premaratne, P., Ajaz, S., Premaratne, M.: Hand Gesture Tracking and Recognition System for Control of Consumer Electronics. Springer Lecture Notes in Artificial Intelligence (LNAI) 6839, 588–593 (2011)
73. Premaratne, P., Nguyen, Q., Premaratne, M.: Human computer interaction using hand gestures. Adv. Intell. Comput. Theor. Appl. Commun. Comput. Info. Sci. **93**, 381–386 (2010)
74. Premaratne, P., Safaei, F., Nguyen, Q.: Moment invariant based control system using hand gestures Intelligent Computing in Signal Processing and Pattern recognition, Book Series Lecture Notes in Control and Information Sciences vol. 345, 322–333 (2006)
75. Premaratne, P., Safaei, F.: Feature based Stereo Correspondence using Moment Invariant. Proceedings of the IEEE International Conference on Information and Automation for Sustainability, 104–108 (2008)
76. McGuire, D., Premaratne, P.: A System for the 3D Reconstruction of the Human Face using the Structured Light Approach. The 5th Workshop on the Internet Telecommunications and Signal Processing, 1–7 (2006)
77. Ding, Y., Ping, X., Hu, M., Wang, D.: Range image segmentation using randomized Hough transform. In Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on 2, 807–811 (2003)
78. Jiang, X., Bunke, H.: Edge Detection in Range Images Based on Scan Line Approximation. Comp. Vis. Image Underst. **73**(2), 183–199 (1999)
79. Besl, P.J., Jain, R.C.: Segmentation through Variable-Order Surface Fitting. IEEE Trans. Pattern Anal. Mach. Intell. **10**(2), 167–192 (1988)
80. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. International Conference on Computer Vision, (2011)
81. Leutenegger, S., Chli, M., Siegwart, R.: Brisk: Binary robust invariant scalable keypoints. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, Luc J. Van Gool (eds.) ICCV, 2548–2555 (2011)
82. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In European Conference on Computer Vision 1, doi: 10.1007/11744023 34. http://edwardrosten.com/work/rosten_2006_machine.pdf., 430–443 (2006)
83. OpenCV, W.G.: Opencv 2.4.5.0 documentation. (2013)
84. Herrera, D.C., Kannala, J., Heikkil, J.: Joint depth and color camera calibration with distortion correction. IEEE Trans. Pattern Anal. Mach. Intell. **34**(10), 2058–2064 (2012)
85. Howard, I., Rogers, B.: Seeing in depth. (2002)
86. Coutant, B.E., Westheimer, J.: Population distribution of stereoscopic ability. Ophthal. Physiol. Optics. **13**(1), 3–7 (1993)
87. Liesbeth, I.N., Mazyn, Lenoir, M., Montagne, G., Geert, J., Savelsbergh, P.: The contribution of stereo vision to one-handed catching. Exp. Brain Res. **157**(3), 383–390 (2004)
88. Salas, J., Tomasi, C.: People detection using color and depth images. Pattern Recognition, Lecture Notes in Computer Science 6718, 27–135 (2011)

89. Payeur, P., Desjardins, D.: Structured light stereoscopic imaging with dynamic pseudo-random patterns. Image Analysis and Recognition. Lect. Notes Comput. Sci. 5627, 687–696 (2009)
90. Desjardins, D., Payeur, P.: Dense stereo range sensing with marching pseudo-random patterns. Fourth Canadian Conference on Computer and Robot Vision (CRV '07), 216–226 (2007)
91. Grin, P.M., Narasimhan, L.S., Yee, S.R.: Generation of uniquely encoded light patterns for range data acquisition. Pattern Recog. 25(6), 609–616 (1992)
92. Morita, H., Yajima, K., Sakata, S.: Reconstruction of surfaces of 3D objects by M-array pattern projection method. Second International Conference on Computer Vision, 468–473 (1998)
93. Salvi, J., Pagès, J., Batlle, J.: Pattern codification strategies in structured light systems. Pattern Recognit. 37(4), 827–849 (2004)
94. van Aardenne-Ehrenfest, T., de Bruijn, N.G.: Circuits and trees in oriented linear graphs. Simon Stevin. 28, 203–217 (1951)
95. Han, Y.K., Yang, K.: New M-ary power residue sequence families with low correlation. Proceedings of IEEE International Symposium on Information Theory (ISIT2007), 2616–2620 (2007)
96. Han, Y.K., Yang, K.: New M-ary sequence families with low correlation and large size. IEEE Trans. Inf. Theory 55(4), 1815–1823 (2009)
97. Kim, Y.-S., Chung, J.-S., No, J.-S.: and Chung, H.: New families of M-ary sequences with low correlation constructed from Sidel'nikov sequences. IEEE Trans. Inf. Theory 54(8), 3768–3774 (2008)
98. Zhang, L., Cudess, B., Seitz, M.: Rapid Shape Acquisition Using Color Structured Lightand Multi-pass Dynamic Programming. 1st IEEE International Symposium on 3D Data Processing, Visualization, and Transmission, 1–13 (2002)
99. Vuylsteke, P., Oosterlinck, A.: Range image acquisition with a single binary-encoded light pattern. Pattern Analy. Mach. Intell. 12(2), 148–163 (1990)
100. Carrihill, B., Hummel, R.: Experiments with the intensity ratio depth sensor. Comp. Vis. Graphics Image Process. **32**, 337–358 (1985)
101. Hung, D.: 3d scene modelling by sinusoid encoded illumination. Image Visi. Comp. **11**, 251–256 (1993)
102. Tajima, J., Iwakawa, M.: 3-D data acquisition by rainbow range finder. International Conference on Pattern Recognition, 309–313 (1990)
103. Geng, Z.J.: Rainbow 3-dimensional camera new concept of high-speed 3-dimensional vision systems. Opt. Eng. **35**(2), 376–383 (1996)
104. Wust, C., Capson, D.W.: Surface profile measurement using color fringe projection Mach. Vis. Appl. **4**, 193–203 (1991)
105. Sato, T.: Multispectral pattern projection range finder. Proceedings of the Conference on Three-Dimensional Image Capture and Applications II 3640, SPIE, 28–37 (1999)
106. Morano, R.A., Ozturk, C., Conn, C., Dubin, S., Zietz, S., Nissanov, J.: Structured light using pseudorandom codes. Pattern Anal. Mach. Intell. **20**(3), 322–327 (1998)
107. Sali, E., Avraham, A.: Three-Dimensional Mapping and Imaging. http=://www.faqs.org/patents/app/20100265316#ixzz299280m00 (2010). Accessed Oct 2010
108. Shpunt, A., Mor, Z.: Non-Uniform Spatial Resource Allocation for Depth Mapping. http=://www.faqs.org/patents/app/20110211044#ixzz299LnJhHM (2011). Accessed Sept 2011
109. Zalevsky, Z., Shpunt, A., Maizels, A., Garcia, J.: Method and System for Object Reconstruction. http://www.sumobrain.com/patents/WO2007043036.html (2007). Accessed April 2007
110. http://azttm.wordpress.com/2011/04/03/kinect-pattern-uncovered/
111. Katz, S.: Boxing with ZCam. Engineering TV. (2009)
112. Iddan, G.J., Yahav, G.: 3D imaging in the studio. Proceedings of SPIE 4298, (2003)
113. Iddan, G.J., Yahav, G.: 3D imaging in the studio.Three-Dimensional Image Capture and Applications IV, Brian D.C., Joseph H.N., Roy P.P. (eds.), Proceedings of SPIE 4298, 48–55 (2001)

Chapter 4
Feature Extraction

An image is worth 1,000 words. Yet, a machine to describe a picture or a color image is not trivial. Of course, some measurements can easily be estimated such as different colors, their intensities, size and dimensions of certain objects if the object can be specified. Yet, the most difficult aspect is to make the decisions as to what constitute an object. In a scene consisting of hand gesture or gestures and a cluttered background, difficulty lies in interpreting these items. Perhaps, the hand gesture recognition offers some help compared to other problems as skin detection can be used to define a hand as was discussed under Pre-processing in Chap. 3. Yet, even when a hand is detected and isolated, what configuration the hand shows is again a difficult question to address.

Feature extraction attempts to extract certain measureable inputs that can be used to classify a section of a signal. If the isolated section of an image contains what the humans interpret as a hand sign with a 'thumbs up' gesture, then it is important to extract information that would make this gesture unique compared to other possible gestures. The success of any classification relies on the ability to develop unique and robust features. As would be detailed in Chap. 6 on Sign Languages, even the same user would not be able to precisely perform the same gesture again. That is to say any gesture has certain variability and the certain degree of uniqueness among other gestures. Humans have evolved in a more subtle way to remember and understand this variability and uniqueness. To develop machine capabilities to interpret this information from an image is not trivial. Therefore a robust feature or set of features should uniquely describe the gesture in order to achieve reliable recognition. In other words, different gestures should result in different good discriminable features. Furthermore, shift and rotation invariant features lead to a better recognition of hand gestures even if the hand gesture is captured from a different angle.

This chapter contains few sections on different approaches to extract features that would make successful classification avoiding false positives. It would contain orientation histogram based feature extraction, the highly successful moment Invariant feature extraction; Principal Component Analysis based feature extraction, other feature extraction methodologies based on color and few other feature extraction strategies that results in successful gesture classification.

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_4,

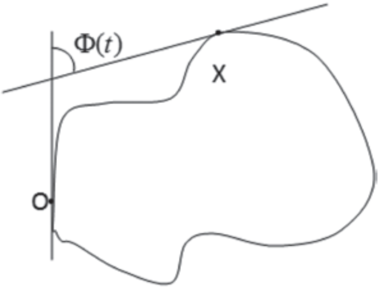


Fig. 4.1 Description of a point *X* with respect to origin *O* using Fourier descriptor

Before the discussion starts on successful features for better classification, it would be versatile to describe the attributes of a good feature. In the context of hand gesture recognition, good features are:

1. Compact set of data representing a unique gesture
2. Sufficient separation of feature clusters. Variety of distance measures such as Euclidean, Mahalanobis, etc. can be used to measure the distance between one gesture cluster and the other gesture clusters. The inter-cluster difference should be sufficient so that statistical variation of same gesture by different users at different times should not confuse the gesture classification.
3. The features should cluster well for different users with different hand sizes and different skin colors and gesture orientations (the features should be invariant)
4. The features obtainable in realtime

4.1 Fourier Descriptors (FD)

Fourier descriptors have been the first features used to describe shapes in image processing and computer vision [1–7]. They have been used for fingerprint recognition as way back as in 1972 due to its simplicity in describing contours which are invariant to scale, shift and rotation [2]. Due to these attributes, they are equally suitable for describing hand gestures.

Figure 4.1 outlines a closed contour that can be described effectively by Fourier Descriptor. To describe point X on the curve as shown above using the arc length *s* from the origin O, a relationship has to be established using the angle that is formed when two tangents from O and X meets as shown above. Then this point is uniquely described by the angular variation $\boldsymbol{\Phi}(t)$ such that:

$$\boldsymbol{\Phi}(t) = \Phi(t) - t \; where \; t = 2\pi s / L.$$

In order to introduce the property of scale invariance, the length of the arc is normalized such that entire contour spans an angle of 2π. This function is real, continuous, and periodical with a period 2π and hence can be described by a Fourier series:

$$\boldsymbol{\Phi}(t) = \sum_{k=0}^{\infty} a_k \exp(-jkt).$$

The set of modules of the coefficients a_k is called Fourier descriptors which can be used to describe various shapes such as leaves, finger prints and human hand postures.

Researchers have used extensions to basic Fourier descriptors to analyse shapes with increasing complexity. Lin and Hwang [8] showed that an alternative representation of the Fourier series is possible using elliptic Fourier features. In their approach, a shape was interpreted as a specific composition of feature ellipses having fixed axis lengths and fixed relative positions and orientations. It was shown that a shape can be represented by a set of ellipses which were rotation and translation invariant. Each ellipse also contained invariant major and minor axis lengths and each pair of ellipses had a specific position and orientation. Lin and Jungthirapanich [9] further developed the 2D elliptic Fourier descriptor to a 3D descriptor. Harding and Ellis developed the concept further with to show that using the FD on a set of trajectory data, it would be possible to recognize a range of pointing gestures that is invariant to natural variations due to the single individual or a 'normal' population. The 2D spatial data of a sequence of hand centroids was obtained using a single camera, but had the potential to be extended to 3D spatial data.

4.1.1 Elliptic Fourier Descriptors

As shown in Fig. 4.1, a point on a contour can be described by a coordinate pair which can be represented by a complex number $z(k)=x(k)+jy(k)$, so that the discrete Fourier transform of $z(k)$ is [10]:

$$a(u) = \frac{1}{N}\sum_{k=0}^{N-1} z(k) e^{\frac{-j2\pi uk}{N}} \tag{1}$$

For $u=0, 1,.. N-1$.

The obtained $a(u)$ coefficients describe the contour. In order to attain translation invariance of this feature, the DC component of the Fourier series given by $a(0)$ removed from the sequence and the rest of the components are scaled by $a(1)$ so that the feature incorporates scale invariance [10]. The origin of the sequence is encoded into the phase of a(u). The consequence of origin selection is illustrated in Fig. 4.2 as it would change the orientation of the contour. An ellipse can be modeled as a positive and negative sequence of differing amplitudes. If the phase shift affecting both sequences is θ the orientation angle, then the sequence is:

$$A_{pos} e^{+j(\varphi+\theta)} \; and \; A_{neg} e^{-j(\varphi+\theta)}$$

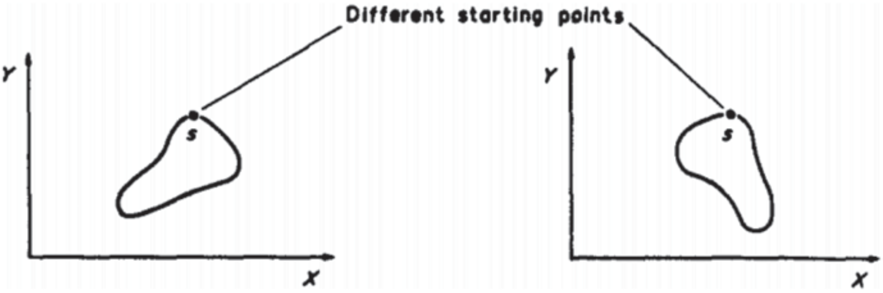


Fig. 4.2 Different starting points due to different orientations. (Courtesy of [8])

The orientation angle, θ can be found by taking the average of the positive and negative sequence phase. The direction and shape of the ellipse depends upon the magnitude of A_{pos} or A_{neg}. The relative size of A_{pos} and A_{neg} affects direction of revolution of the ellipses.

The representation of closed contours based on elliptical basis functions is described in detail by Lin and Hwang [8]. They mathematically demonstrated that a closed contour can be described by its Fourier descriptor feature matrices. A shape can be viewed geometrically as the locus generated by properly moving the feature ellipses.

Harding and Ellis [10], developed hand tracking method based on the work of Lin and Hwang. As shown in Eq. 1, the complex frequency domain data generated by the Fourier descriptor technique is generated by a discrete Fourier Transform algorithm. The number of harmonics generated was equal to the number of samples, *N*. The sample lengths were all normalized to the same length (64) by a multirate process. They used sixty four samples to encode a typical gesture that was completed within two seconds, at a sample rate of 30 frames per second, and additionally aided the speed of FFT implementation. Figure 4.3 shows ‘elliptic corkscrews’ and the overall trajectory for a gesture- *To Left Should and Return*.

Conseil et al. [11] developed a Fourier descriptor based method to represent hand gestures in an attempt to compare the performance accuracy of Fourier descriptor to Hu Moment based (this is discussed in Sect. 4.3.1) approaches. They used Triesch hand posture database and defined their own gesture vocabulary, with 11 gestures, and performed the acquisition of a large number of images, with 18 persons, and approximately 1,000 images per gesture per person [12].

They claimed that the tests were performed on a more realistic database, with various hand configurations realized by non-expert users. The learning was done with manually selected images of an expert user, with nearly 500 images per gesture. In the tests, they used 6 Fourier descriptors and initially validated the learning stage by running classification on the learning images, and obtained recognition rates of 98.11 % for Hu moments and 99.96 % for Fourier descriptors. Then images of the other users were classified using this learning data, with approximately 1,000 images per gesture for each user. They obtained a total of 86.22 % for Fourier

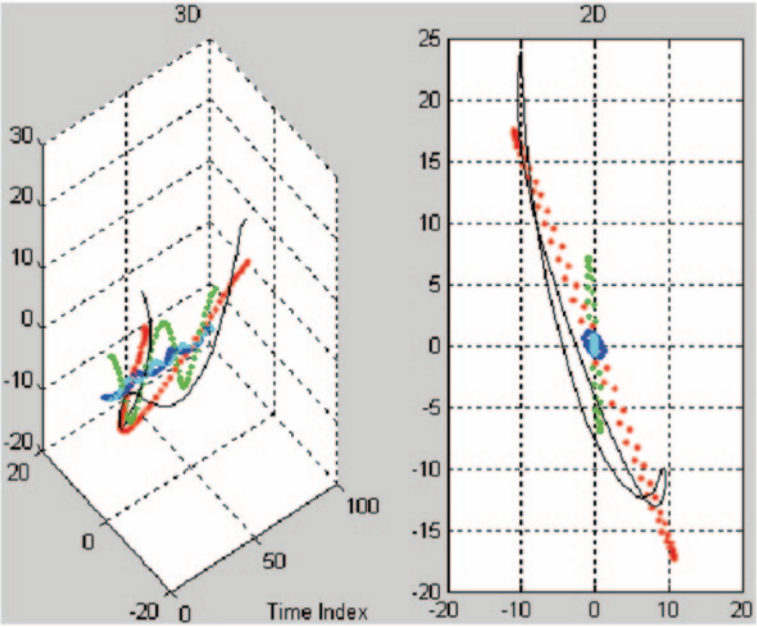


Fig. 4.3 *Left*: 3D, *Right*: 2D, view of the first 4 'elliptic corkscrews [8]','.' and overall trajectory ('-') of gesture 'To left shoulder and return'. (Courtesy of [10])

descriptors versus 71.08 % for Hu moments. They also observed that FD outperformed Hu moments in terms of discrimination between visually close gestures. Figure 4.4 shows that the low frequency coefficients contain information on the general form of the shape and the high frequency coefficients contain information on the finer details of the shape.

One of the earliest works of hand gesture recognition using gesture feature extraction was attempted by Utsumi et al. in 1995 [18]. They proposed very simple feature extraction method that relied on centre of gravity of the hand and the finger locations based on Fourier descriptors. However, they used multiple cameras and tracked 3D position, posture, and shapes of human hands from multiple viewpoint images. This reduced self-occlusion and hand-hand occlusion by employing multiple-viewpoint and viewpoint selection mechanism. Each hand position was tracked with a Kalman filter and the motion vectors were updated with image features in selected images that did not include hand-hand occlusion. In their approach, 3D hand postures were estimated with a small number of reliable image features using COG and fingertip positions. These features were extracted based on distance transformation, and were found to be robust against changes in hand shape and self-occlusion. Finally, a "best view" image was selected for each hand for shape recognition. The shape recognition process was based on Fourier descriptors. The outline of their approach is depicted in Fig. 4.5.

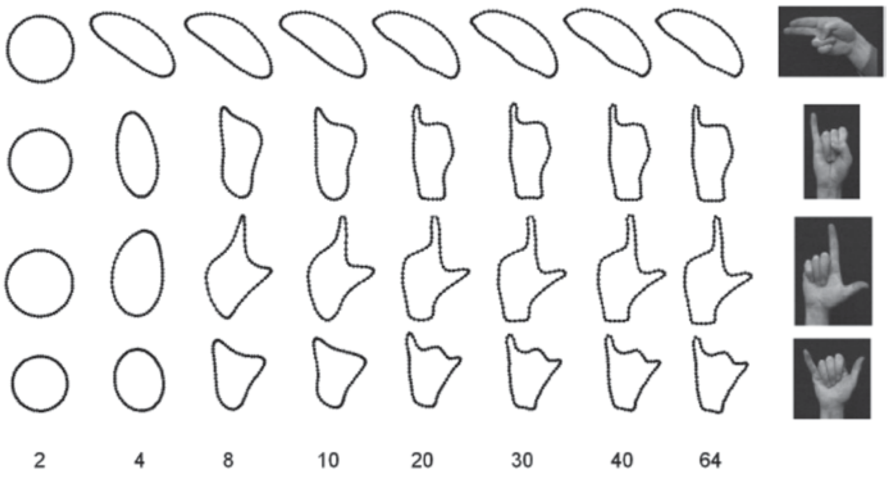


Fig. 4.4 Example of reconstruction with FD, as a function of the cut-off frequency, with an initial contour sampled at 64 points. (Courtesy of [12])

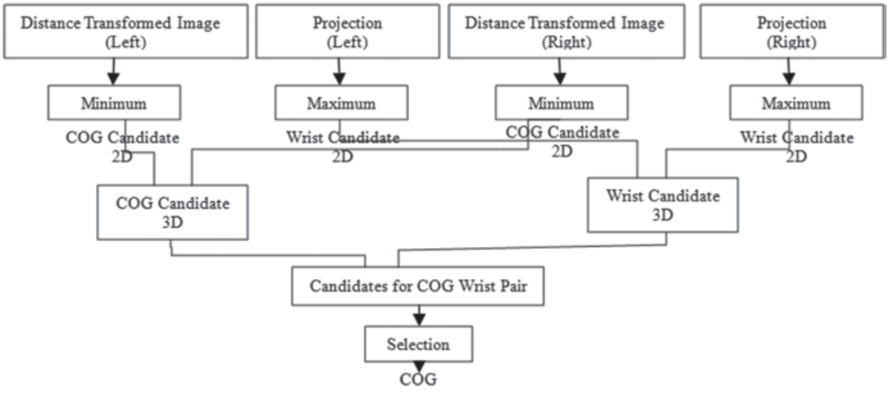


Fig. 4.5 COG Detection. (Courtesy of [13])

4.1.2 Modified Fourier Descriptors

Licsár and Szirányi applied a boundary-based Fourier descriptors for feature extraction based on widely used for shape description method used for content-based image retrieval systems [14, 15]. The extracted features were classified using neural networks classification algorithms [16, 17] resulting in about 91 % recognition rate for 6 gestures. In their method, the gesture contours were classified by the nearest neighbor rule and the distance metric based on the Modified Fourier Descriptors (MFD) [15]. This metric is invariant to the rotation, transition, reflection and scaling of shapes. The strategy requires that the examined shape should be defined by

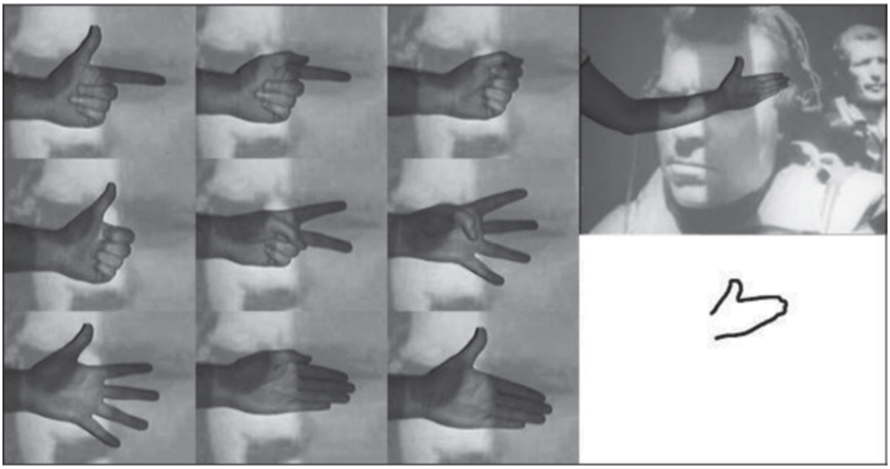


Fig. 4.6 Gesture vocabulary and segmentation result. (Courtesy of [14])

a feature vector, which is periodic, to expand it into Fourier series. The approach generated a feature sequence between the two wrist points, as shown in Fig. 4.6, along the shape boundary leading to more unambiguous features. This is due to the fact that the shape contours of the palm when showing only the index or the thumb finger is very similar to each other, while the contour between wrist points are distinctively different. The defined boundary sequence was constructed as a complex sequence of the *x* and *y* coordinates of the boundary points. These boundary points were then used to calculate the discrete Fourier transform (DFT) of this complex sequence. They further used the magnitude values of the DFT coefficients to retain invariance to rotation and extended the MFD method to obtain symmetric distance computations. They reported that when the trainer and the user were the same, recognition rates were above 97 % while different users resulted in an accuracy around 86 %.

4.2 Contour Description using 1D Sequence

Fourier descriptors always had a strong appeal as an excellent descriptor of the shape boundary or contour with invariance for translation, scale, rotation and reflection or mirror image offered by MFD techniques. One of the drawbacks in the Fourier descriptor is that the non-smooth contours result in very poor description of the shape resulting in classification error. Even though many researchers willingly state this in their research, this is indeed the reason why many others deviated from the very promising Fourier descriptors. Malima et al. in 2006 reported a new development inspired by Fourier descriptors to recognize hand gestures [18]. Their approach had limited focus and was not intended to develop a highly accurate

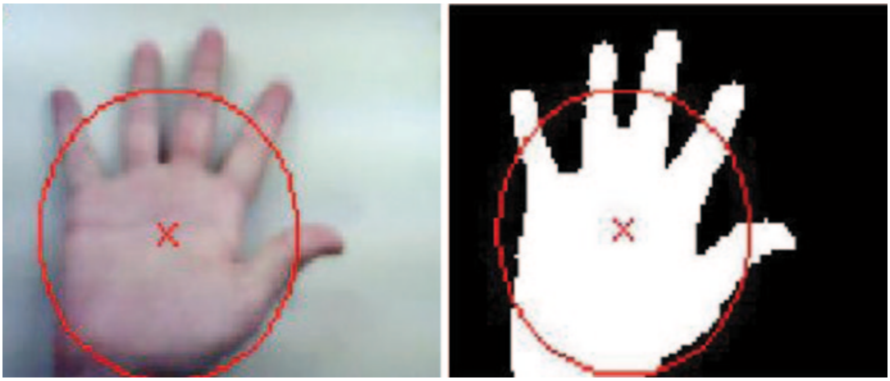


Fig. 4.7 Original image (*left*) with circle overlapped and the skin segmented binary image (*right*) with the circle with COG as the center. (Courtesy of [18])

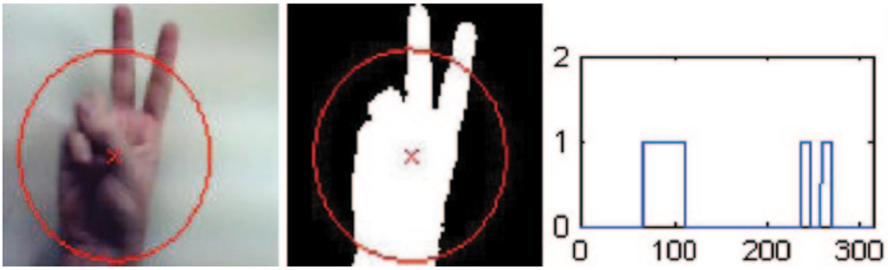


Fig. 4.8 Circle overlapping the hand (*left*), binary image (*middle*) and the zero-to-one transitions [18]

system as their gesture recognition was used to control a robot arm. Nevertheless, the approach had many positive developments.

As shown in Fig. 4.7, the initial step in extracting features was to select the region of importance. This is achieved by drawing a circle whose radius is 0.7 of the fartherest distance from the Centre of Gravity (COG). Such a circle is likely to intersect all the fingers active in a particular gesture as demonstrated in Fig. 4.7. Once the skin segmentation is performed and the image is binarized, the 1D signal or the feature vector that describes the gesture is obtained by tracking the circle constructed in the previous step. As conceivable, the uninterrupted 'white' portions of this signal correspond to the fingers or the wrist. The total transitions of zeros to one can be counted to indicate the signal. By subtracting one from this number removes the transition due to the wrist. Estimating the number of fingers leads to the recognition of the gesture. This process is shown in Fig. 4.8.

This algorithm simply counts the number of active fingers without any regard to which particular fingers are active. Different combination of active fingers may result in the same configuration. A user may potentially use any finger combination for 'on' or 'off' state to activate robotic commands which limits its use as a solid

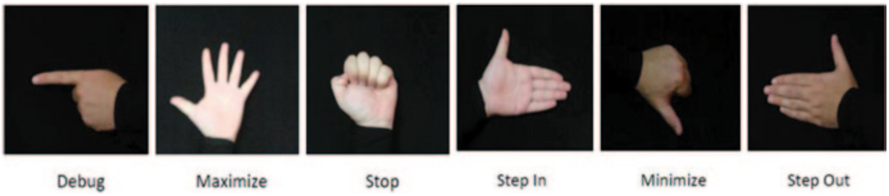


Fig. 4.9 Set of gestures used by Hasan and Misra [19]

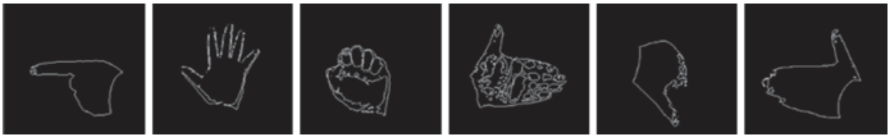


Fig. 4.10 Edges of the gestures. (Courtesy of [19])

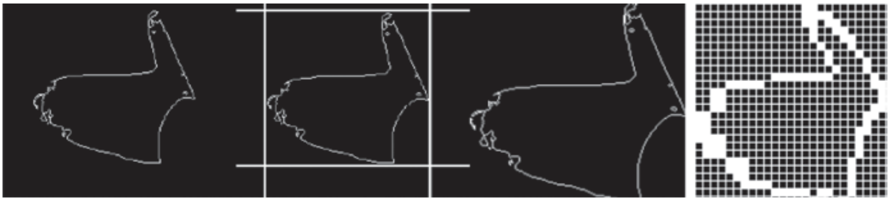


Fig. 4.11 Normalization operation and features calculation via dividing the gesture edge map with remapping. (Courtesy of [19])

hand gesture recognition approach. This algorithm is scale invariant as any size of hand or image of a hand will result in the same 1D signal. It is also rotation invariant, since the orientation of the hand does not hinder the algorithm from recognizing the gesture. In addition, the position of hand is also not an issue leading to translation invariance.

Fourier descriptor-based methods predominantly use edge contours as the source of features. Hasan and Misra proposed an approach where the edge map of gestures were remapped to 25×25 blocks with each block comprising the output of the edge map due to 5×5 pixels. The edge detection is achieved by convolving the binary image with a Laplacian Mask. Figure 4.9 shows the set of hand gestures they were using with skin segmented edge maps shown in Fig. 4.10. These edge maps were then normalized as shown in Fig. 4.11 and mapped to a 25×25 block feature map representation as shown on Fig. 4.11 (right most). This represents a hand gesture feature vector of size 625 (25×25) and the pixel value of the 25×25 block is determined by the following calculation:

$$B = \sum pixel\, value\, in\, each\, 5 \times 5\, block.$$

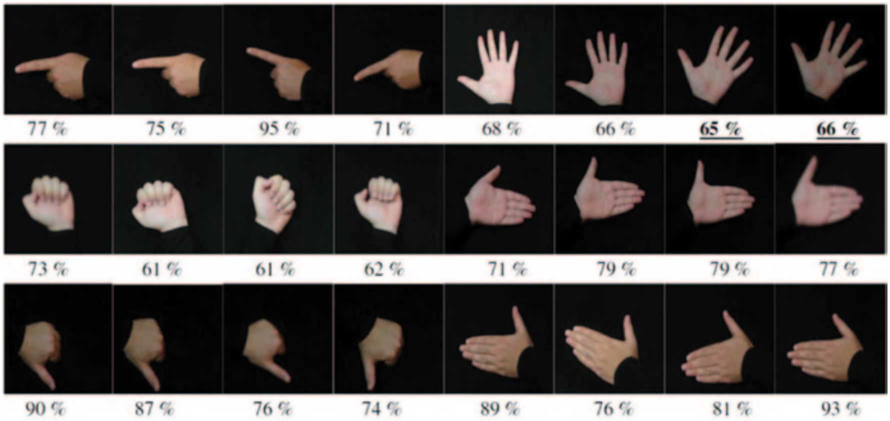


Fig. 4.12 Recognition accuracy for each gesture. (Courtesy of [19])

Their primary objective was to establish a system which could identify specific human gestures and utilize these gestures to control machines in a natural way. They used HSV (Hue, Saturation and Value) color model for segmentation and identified a feature vector of 25 × 25 after remapping of edges as discussed earlier. Their experiment showed that more that 65 % of these features were zero values which leads to minimum storage requirements and the recognition rate achieved surpassed 91 % using 36 training gestures and 24 different testing gestures. Their classification results are shown in Fig. 4.12 different gestures.

Li [20] attempted feature extraction techniques that were similar to Hasan and Misra, to classify hand gestures for robot control. It was called Fuzzy C-Means clustering technique however, the feature extraction stage remain very basic. In this approach, the segmented hand shape was converted into a feature vector. Their system used the approach designed by Wachs and Kartoun [21]. In this approach, a feature vector of the image with 13 parameters was created where the first feature is the aspect ratio of the hand's bounding box. The other 12 features were the values representing a coarse discretization of the image, where each grid cell is the mean gray level in the 3 by 4 block partition of the input image. The mean values of each cell represented the average brightness of those pixels in the image. Figure 4.13 illustrates typical user gestures, their binary representation after skin segmentation and the block mean gray scale values and the resultant feature vector on the third bottom row.

Initial research carried out prior to year 2000 focussed on less image processing tasks compared to what is attempted today. The major reasons behind this were the amount of computing power available on ordinary desktop computers to the resolution and the accuracy of cameras and the maturity of the developmental tools and programming languages available at that time. Obtaining the hand outline for human computer interaction was first proposed by Segan and Kumar [22]. In their effort, the outline of the hand is extracted using an edge tracking algorithm. The

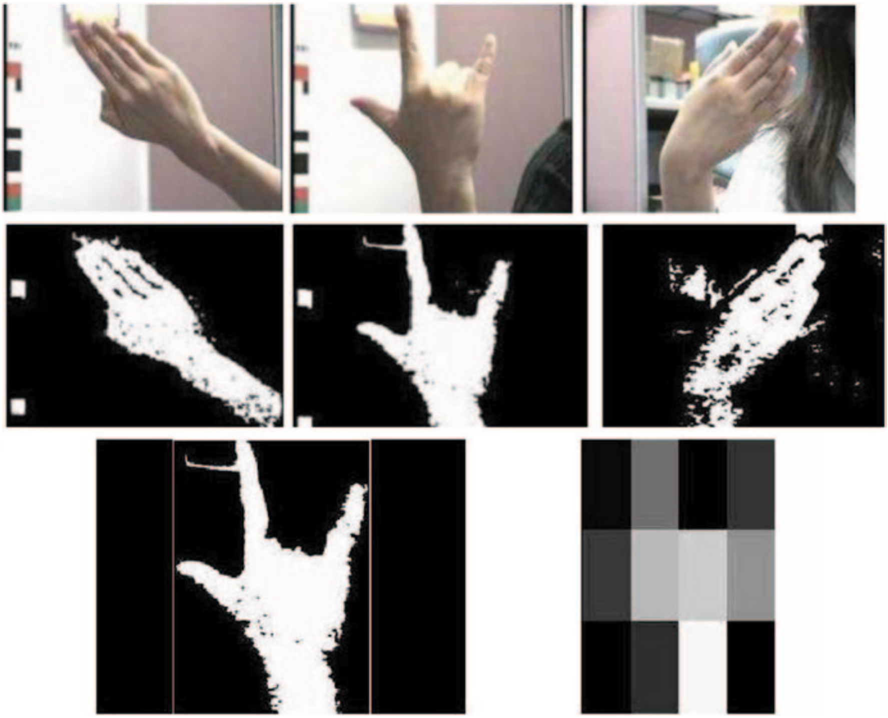


Fig. 4.13 Hand gestures in HSV space (*top row*), their binary representation after skin segmentation (*middle row*) and a gesture and its gray scale block feature vector (*bottom row*). (Courtesy of [20])

system was capable of recognizing both hand postures and gestures which was a remarkable feat at that time. In this approach, the local features were represented by the local extrema of the outline; peaks and valleys. The peaks are found at the finger tips, whereas the valleys are rather found in the regions where two integers join the palm of the hand. This is shown in Fig. 4.14.

Segan and Kumar restricted their system to identify one of four possible gesture classes: *Point*, *Reach*, *Click*, and *Ground*, shown in Fig. 4.15. *Point* and *Reach* are static gestures, while *Click* is a dynamic gesture that involves quick bending of the index finger. The Ground class includes all gestures other than the remaining three, as well as an empty image.

An image that belongs to the Point class, was further analyzed to compute the position and orientation of the pointing finger in the image plane, that is a three degrees of freedom (3DOF) pose (x; y;). The classification method consisted of two stages: an initial classification based on analysis of local features, and final classification involving a finite state machine.

Extracting the hand outline of the connected regions was extracted by comparing the input image with a previously acquired background image. After extracting the regions, the boundary of each region was represented as a list of pixel positions in a

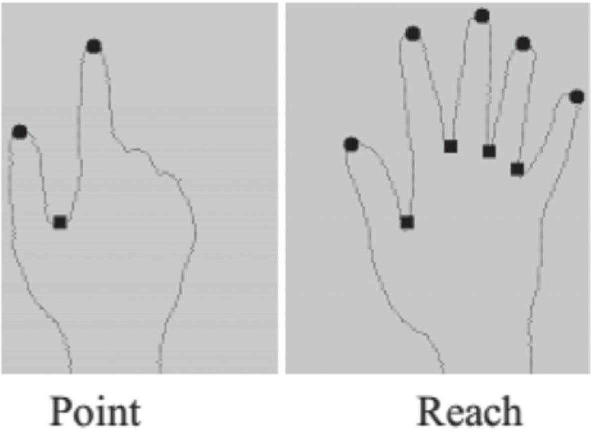


Fig. 4.14 Peaks (*circles*) and valleys (*squares*) used in initial gesture classification. (Courtesy of [22])

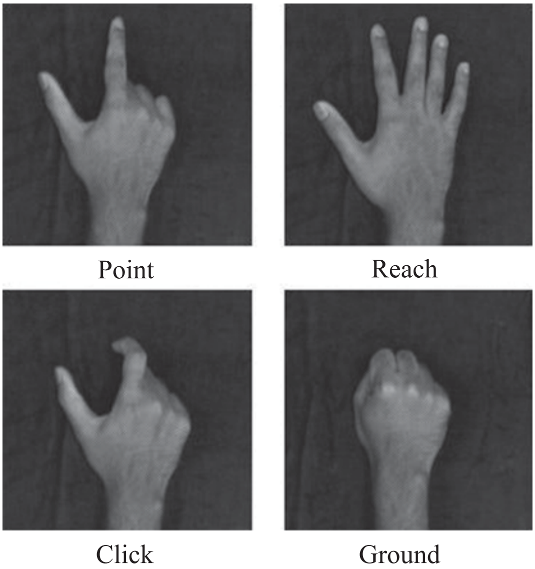


Fig. 4.15 Four possible gesture classes outlined by Segan and Kuma. (Courtesy of [22])

clockwise order. A heuristic screening of the regions based on perimeter length led to the identification of a hand.

The boundary of the region selected as a possible "hand" is further analyzed to extract local features. At each point the k-curvature measure at each point. The k-curvature is the angle $C(i)$ between two vectors $[P(i-k); P(i)]$ and $[P(i); P(i+k)]$, where k is a constant. The points along the boundary where the curvature reached a local extremum, that is the "local features", were then identified. Some of these local features were labelled "peaks" or "valleys". Peaks were defined as having a positive curvature above P_{thr} and the 'Valleys' were defined as having a negative curvature less than V_{thr} [22].

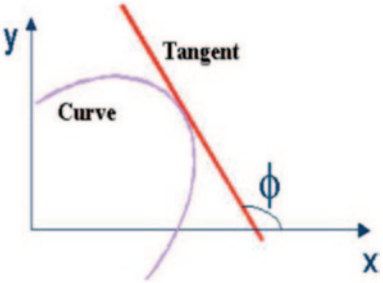


Fig. 4.16 The curvature of a planar curve. (Courtesy of [23])

One advantage of such features is the quick exclusion of inappropriate gestures using the number of peaks and valleys as indicators. One of the disadvantages was that this simplistic approach limited the available gestures to a minimum of four.

4.2.1 Contour Description using Curvature Scale Space Features

In a race to develop ideal features that would separate hand gestures apart and brings each gesture by different users closer, Chang et al. presented a novel feature extraction approach based on Curvature Scale Space (CSS) for translation, scale, and rotation invariant recognition of hand postures [23]. Initially, the CSS images were used to represent the shapes of boundary contours of hand postures followed by extraction of multiple sets of CSS features to overcome the problem of deep concavities in contours of hand postures [23]. These CSS images can then be classified using techniques such as nearest neighbour classification to establish matchings between multiple sets of input CSS features and the stored CSS features for hand postures. Chang et al. produced results to show the proposed approach was able to extract multiple sets of CSS features from input images with good recognition accuracy.

Mokhtarian and Mackworth [24, 25] first proposed the object contour-based shape descriptor based on the CSS image of the contour [23]. The CSS descriptor provides translation, scale and rotation invariant features of curves.

The curvature κ of a planar curve is defined as the derivative of the tangent angle φ with respect to the arc length s, as shown in Fig. 4.16 [23]. The curvature κ is written as follows [23–25]:

$$\kappa = \frac{d\varphi}{ds}$$

and Letting $T = \{x(u),\ y(u) \big| u \in [0,1]\}$ where T is the planar curve and u is the normalize arc length parameter.

Curvature κ can be expressed in terms of u and σ, standard deviation as

$$\kappa(u,\sigma) = \frac{X_u(u,\sigma)Y_{uu}(u,\sigma) - X_{uu}(u,\sigma)Y_u(u,\sigma)}{(X_u(u,\sigma)^2 + Y_u(u,\sigma)^2)^{3/2}}$$

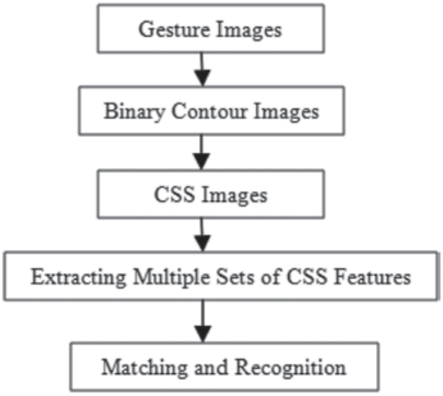


Fig. 4.17 Curvature scale space feature extraction and gesture matching. (Courtesy of [23])

Where

$$\begin{aligned} X_u(u,\sigma) &= x(u) * g_u(u,\sigma), \\ X_{uu}(u,\sigma) &= x(u) * g_{uu}(u,\sigma), \\ Y_u(u,\sigma) &= y(u) * g_u(u,\sigma), \end{aligned}$$

And

$$Y_{uu}(u,\sigma) = y(u) * g_{uu}(u,\sigma),\ \textit{where} * \textit{denotes convolution},$$

$$\mathrm{g}_u(u,\sigma) = \frac{\partial}{\partial u} g(u,\sigma)$$

and

$$\mathrm{g}_{uu}(u,\sigma) = \frac{\partial^2}{\partial u^2} g(u,\sigma).$$

The function defined implicitly by $\kappa(u,\sigma) = 0$ is the CSS image of T [23–25].

In Chang et al.'s approach as outlined in Fig. 4.17, when an image is captured with a potential hand gesture, its contours are extracted using edge detection. It is important to have a continuous contour for the next steps to be successful. Then the contour is successively low-pass filtered with a kernel. For 201, 534, 640, 724 and 731 iterations, the curvature of the curves determine the CSS image. This process is illustrated in Fig. 4.18. With each passing of low-pass filter, the contour smoothens as expected reducing the curvature in many regions.

A good set of features would be expected to be stable when a unique hand gesture is made. Unfortunately, CSS is somewhat unstable with subtle gesture changes as seen in Fig. 4.19. Figure 4.19a and 4.19c denote the same hand posture 4.19b and 4.19d show the respective CSS images of Fig. 4.19a and 4.19c. The locations of the largest peaks which are related to finger directions are unstable in the CSS images.

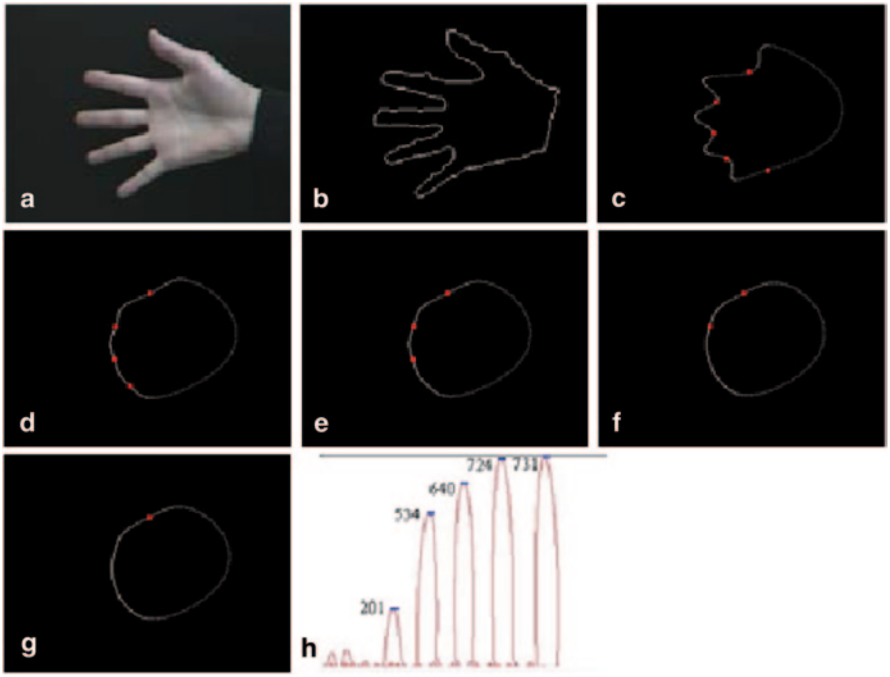


Fig. 4.18 **a** shows the input hand posture. **b** is the contour of the hand posture. **c** to **g** show the resulting contours of the hand pose contour iteratively low-pass filtered by performing a convolution with the (0.25, 0.5, 0.25) kernel for 201, 534, 640, 724 and 731 iterations, respectively. **h** shows the resulting CSS image. (Courtesy of [23])

As shown in Fig. 4.19, the locations of the maximal peaks in the CSS image approximately correspond to the deep concavities in original hand posture contour corresponding to five fingers [23]. Chang et al. extracted multiple sets of CSS features in order to overcome the above instability. They improved their recognition ability by confining their hand posture library to 6 as shown in Fig. 4.20. They reported a recognition rate of 98.3 %.

4.3 Features from Karhunen Loeve (K-L) Transform

K-L Transform is well-known for its ability compact data. It is known as the ideal transform for data compression. This ability is very useful in shape description as the shape can be described with minimum number of coefficients opposed to other approaches. The K-L transformation is also known as the principal component transformation, the eigenvector transformation or the Hotelling transformation. The advantages are that it eliminates the correlated data, reduces dimension keeping average square error minimum and provides good clustering characteristics. It establishes a new co-ordinate system whose origin will be at the centre of the object

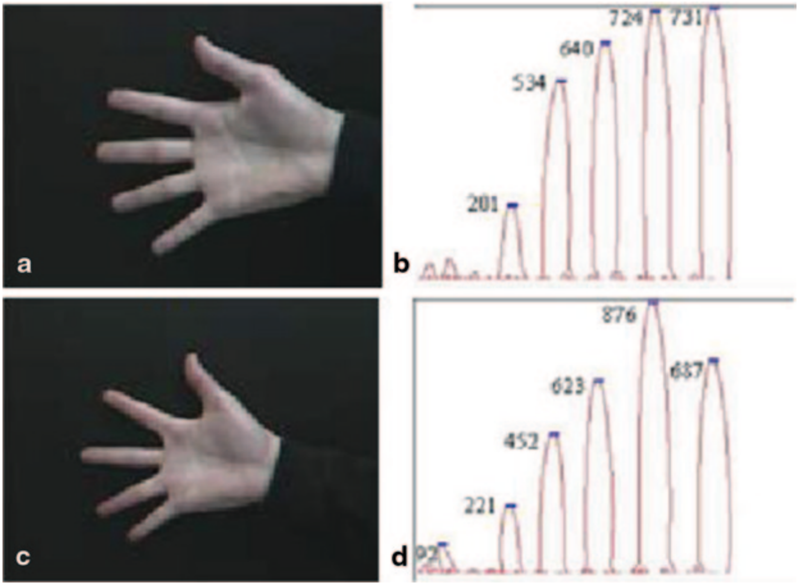
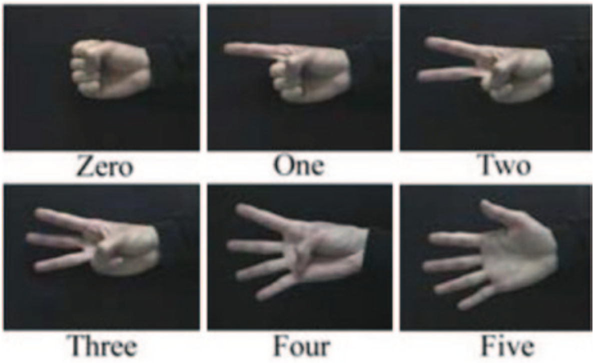


Fig. 4.19 **a** and **c** are the same hand postures. **b** and **d** are the CSS images of **a** and **c**, respectively and shows that the locations of the largest peaks are unstable in the CSS images. (Courtesy of [23])

Fig. 4.20 Hand posture library used by Chang et al. (Courtesy of [23])



and the axis of the new co-ordinate system will be parallel to the directions of the Eigen vectors. It is often used to remove random noise.

Singha and Das recently proposed a technique for hand gesture recognition based on K-L transform [26]. Their system composition is shown in Fig. 4.21 for feature extraction. When they extracted binary hand image after skin segmentation and successive cropping, Canny edge detection was used for edge extraction which is then used for K-L feature extraction. K-L provides a mechanism to extract unique features for each gesture which are independent of human hand size and light illumination which are uncorrelated with minimum entropy. As in the use of compression, K-L transform provides the best representation of a unique feature vector that

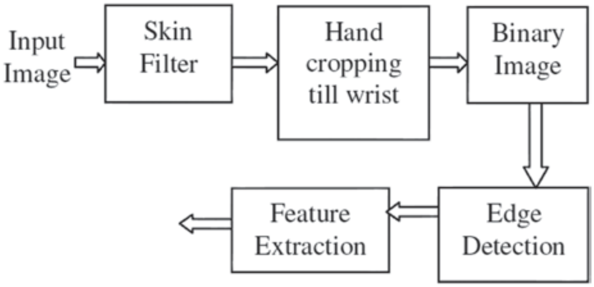


Fig. 4.21 K-L transform based feature extraction based on [23]

can be classified for gesture detection. Figure 4.22 shows hand gesture image along with the Eigen vectors obtained using K-L Transform. They managed to develop the system to recognize 10 different hand gestures with a recognition rate of 96 %.

4.4 Features Described by Histograms

Histogram of Oriented Gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs as it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy.

Dalal and Triggs were the researchers who first described Histogram of Oriented Gradient descriptors in 2005 [27]. In this work they focused their algorithm on the problem of pedestrian detection in static images, although since then they expanded their tests to include human detection in film and video, as well as to a variety of common animals and vehicles in static imagery. Figure 4.23 shows the use of histogram of oriented gradient descriptor used in human detection as described by Suard et al. [28]. Figure 4.24 shows the histograms with different bin resolution of the region shown in a square of Fig. 4.24. What is observed here is that gradient orientation around an edge should be more significant than the one of a point in a nearly uniform region. It also highlights that the larger the number of bins, the more detailed the histogram is.

In the context of object recognition, the use of edge orientation histogram has gained significant popularity [29–32]. However, the concept of dense and local histograms of oriented gradients (HOG) is a method introduced by Dalal et al. [27]. The aim of such a method was to describe an image by a set of local histograms. These histograms count occurrences of gradient orientation in a local part of the image [28].

Freeman and Roth were the pioneering researchers to test whether the use of histogram of local orientation would be useful as a feature in hand gesture recognition

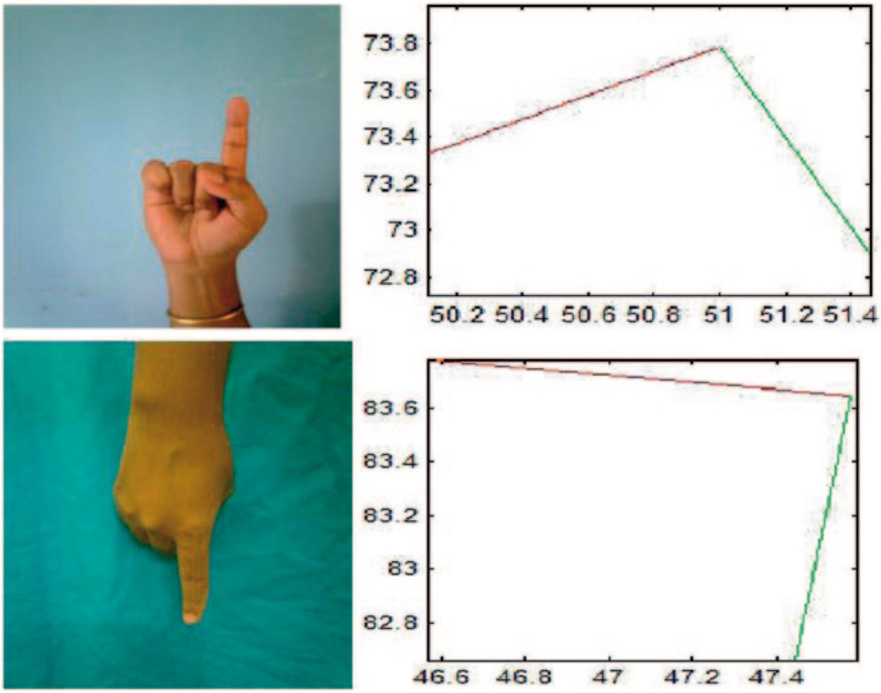


Fig. 4.22 Features extracted for gesture ‘UP’ and ‘DOWN’ and their Eigen vector plots (*Right*). (Courtesy of [26])

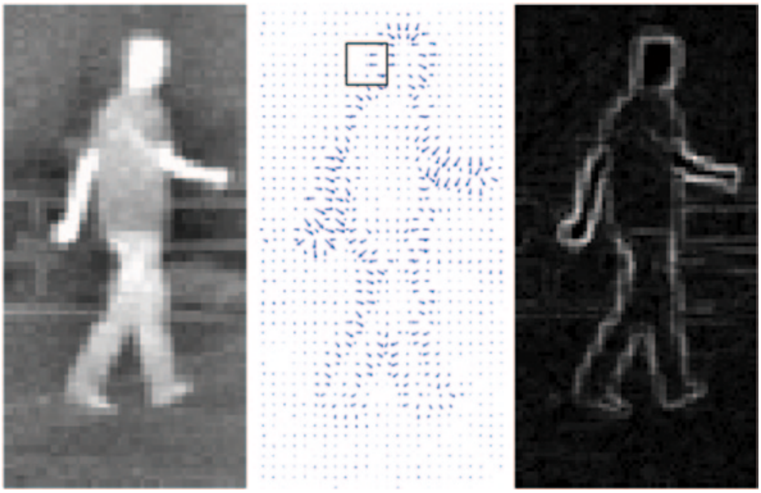


Fig. 4.23 The gradient computation of an image. (*left*) is the original image, (*middle*) shows the direction of the gradient, (*right*) depicts the original image according to the gradient norm [28]

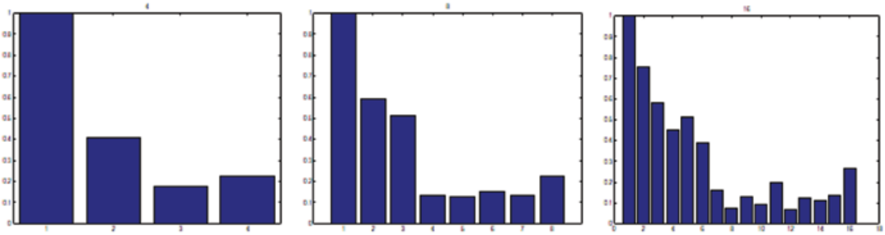


Fig. 4.24 This figure shows the histograms of gradient orientation for (*left*) 4 bins, (*middle*) 8 bins (*right*) 16 bins [28]

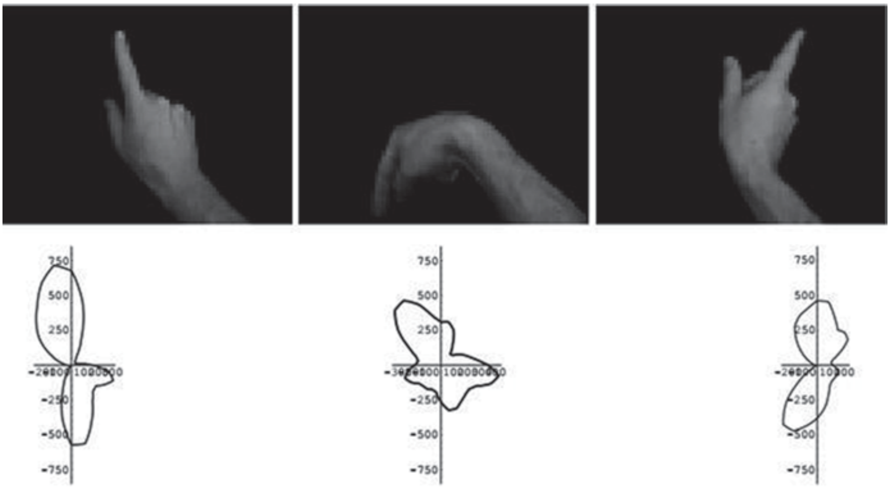


Fig. 4.25 *Top row*: *Up down* and *right* gestures and their orientation histograms shown on the bottom row. (Courtesy of [33])

[33]. They developed a training set that contained up to 15 histograms with their local orientation of various gestures. In their test phase, they compared another histogram of another gesture as shown in Figs. 4.25 and 4.26. The vector in the training database that was closest to the test vector was selected as the gesture was made. Even though their system was restricted to few gestures in today's standards, there goal was to develop a fast and a robust system that could be implemented on a desktop (in 1994) with invariability to moderate illumination changes. The selection of orientation histogram as a feature vector to represent hand gestures offered robustness to lighting changes and translational invariance of the hand position. Furthermore, the histogram can be calculated very quickly.

In 2004, Zhou et al. proposed a static hand gesture recognition system based on local orientation histogram features [34]. In general, orientation histograms cannot be directly applied to hand gestures as the hand does not provide sufficient texture

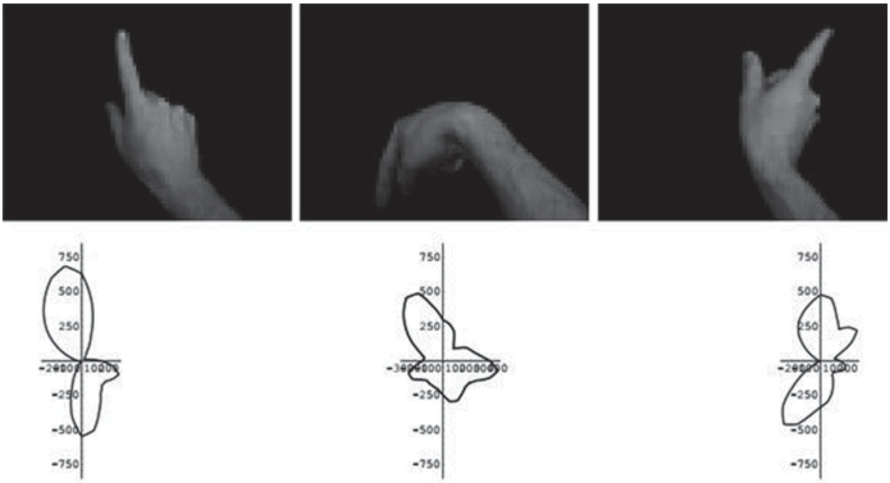


Fig. 4.26 Another instance of information similar to the ones shown in Fig. 4.25. The orientation histograms in this figure highlights that the gestures may be slightly different in each instance but their trajectory is unique to the gesture. (Courtesy of [33])

[35]. Since orientation histograms show the frequency of edges aligned in a certain angle, there might not be enough information available inside the hand area in order to uniquely describe a hand gesture. According to [33], the main problem that might arise is that hand gestures which look different to a human being, might have almost identical orientation histograms. Similar looking hand gestures due simply to rotation yield very different orientation histograms. However, in [34], it is found that the boundary of the hand shape contains enough information to uniquely describe the feature of a specific gesture. Therefore, the idea of local orientation histograms consists of creating overlapping subwindows, containing at least one pixel which lies inside the hand shape. For each of these subwindows, an orientation histogram is created, which is then added to the feature vector. Beside the local orientation histograms, subwindow positions are also added to the feature vector. These positions are measured relative to the median value of all pixel positions that were determined to be in the hand region. Clearly, the advantage of this technique lies in the improved robustness since using relative positions allow in-plane translations.

Misra et al. proposed a hand gesture recognition system that employed the techniques developed for pedestrian detection to recognize a small vocabulary of 7 hand gestures using Histogram of Oriented Gradients as the descriptors [36]. They claimed to use Partial Least Square (PLS) as a ‘class aware’ method of dimensionality reduction which performs better than Principle Component Analysis (PCA) and preserved significant discriminative information in the lower dimensions. Three sets of databases consisting of training as well as testing image sets with varying degree of positional variation were developed to analyse the importance of using multi-level HOG features for robust human hand gesture recognition. They demonstrated that using only low level HOG features were not adequate for high detection rate. They attained marginal degree of accuracy of detection of human hand gestures and the performance degraded due to the tradeoffs between the accuracy and positional variation of the hand. This was also due to the fact that simple brute-force implementation that they relied on using the *k*-nearest neighbor search algorithm to classify gestures was not effective. Their vocabulary of gestures were confined to only seven hand gestures as they were simply evaluating the feasibility of HOG descriptors and PLS reduction for human hand gesture recognition.

Many techniques exist that uses features derived from edge and gradient based descriptors for hand gesture recognition [37, 38]. Cluttered backgrounds with multiple users and skin-tone regions have hampered hand gesture recognition using such features as gradient based descriptors are only useful in simple uncluttered backgrounds. Dalal and Triggs [27] have demonstrated that for robust visual object recognition, Histogram of Gradients (HOG) descriptors can outperform many other gradient-based feature sets. The HOG descriptors are obtained using different block sizes on the same image and the blocks are contrast normalized to remove the illumination variance. These descriptors are then concatenated to realize the final image descriptors. The HOG features are computed several times for each block in the image, resulting in multiple contributions to the final descriptor, with each cell being normalized with respect to a different block [27].

The HOG based method by Misra et al. uses the edge and gradient based techniques developed for human detection for the problem of hand sign recognition. Similar features have been reported by other research [27, 37, and 38]. Some have used an array of moving spots [39], to recognize hand gestures, [40] presented a glove free solution to this problem.

The dimensionality of the final descriptors increases due to redundancy which needs to be curtailed for classical machine learning algorithms such as the *k*-nearest neighbor search algorithms to be discussed in the next chapter. Misra et al. used Partial Least Square regression technique for dimensionality reduction as it models relations between a set of observations by means of latent variables, and is aware of the classes into which the observations are classified [41]. They demonstrated that their PLS outperforms PCA in terms of classification of the training data into various hand gestures. They further demonstrated that PLS as the preferred method of dimensionality reduction. PLS is known to have a lower execution time than PCA which saves time in the learning phase [42]. HOG descriptors characterize the articulated gestures by the distributions of local intensity gradients. The feature extraction begins with the gradient computation for all the pixels of the image, with the largest of the gradient of three channels chosen as the gradient of the pixel. Each ‘cell’ in the image has a histogram which is constructed using the directions and the magnitudes of pixel gradients in the cell. The features are accumulated over a block and are then normalized.

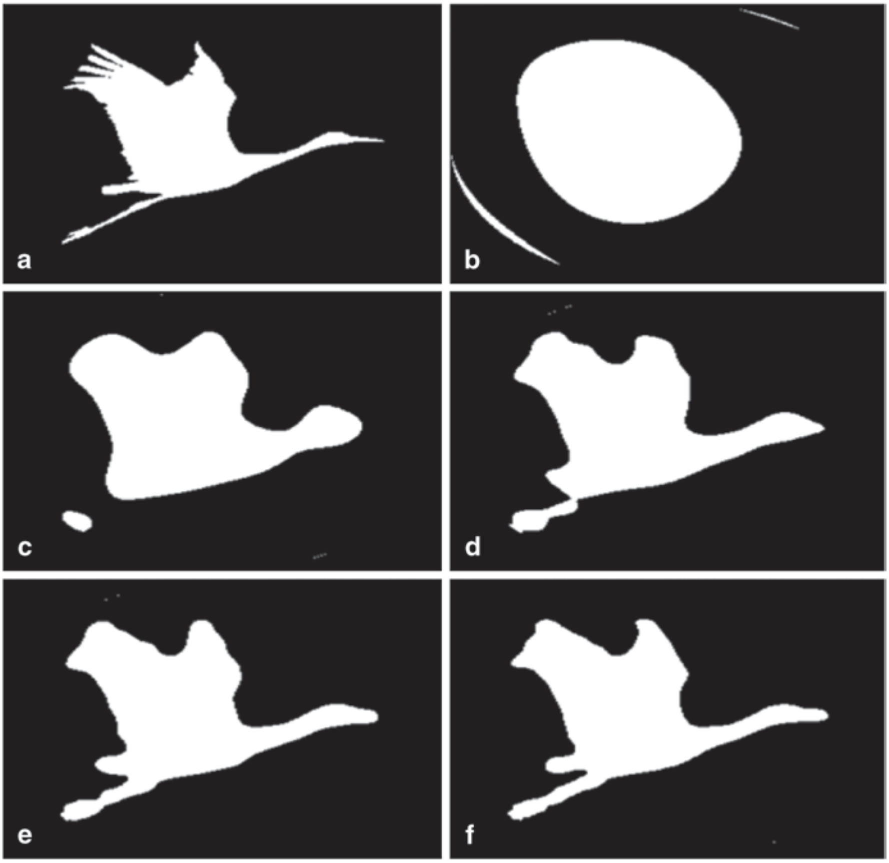


Fig. 4.27 Image reconstruction with Zernike moments. Starting with (**b**), image is reconstructed gradually using higher Zernike moments

4.5 Zernike Moments

Zernike polynomials are a sequence of polynomials developed by a Nobel laureate mathematician Frits Zernike in 1934 [43]. These sequences are orthogonal on the unit disk and play an important role in beam optics. Zernike moments have been used in image construction as shown in Fig. 4.27.

Moments have been used in image processing and classification type problems since Hu introduced them in his groundbreaking publication on moment invariants [44]. In 1962, Hu mathematically demonstrated that geometric moments can be made to be translation and scale invariant. Since then more powerful moment techniques have been developed. A notable example is Teague's work on Zernike Moments (ZM) as a pioneer to use the Zernike polynomials (ZP) as basis functions for the moments [45]. ZM's have been used in a multitude of applications with great success and some with 99 % classification accuracy [46].

The use of ZP's as a basis function is theoretically beneficial because they are orthogonal polynomials which allows for maximum separation of data points, given that it reduces information redundancy between the moments. Their orthogonal properties make them simpler to use during the reconstruction process as well. Furthermore, the magnitude of ZM's is rotationally invariant, which is crucial for certain image processing applications, such as classifying shapes that are not aligned.

4.5.1 Hu Moment Invariants

Hu demonstrated the utility of moment invariants through a simple pattern recognition experiment. The first two moment invariants were used to represent several known digitized patterns in a two-dimensional feature space [47]. An unknown pattern could be classified by computing its first two moment values and finding the minimum Euclidean distance between the unknown and the set of well-known pattern representations in feature space. If the minimum distance was not within a specified threshold, the unknown pattern was considered to be of a new class, given an identity, and added to the known patterns. A similar experiment was performed using a set of twenty-six capital letters as input patterns. When plotted in two-dimensional space, all the points representing each of the characters were distinct. It was observed, however, that some characters that were very different in image shape were close to each other in feature space. In addition, slight variations in the input images of the same character resulted in varying feature values that in turn lead to overlapping of closely spaced classes. Hu concluded that increased image resolution and a larger feature space would improve object distinction [47].

Moment invariants algorithm has been known as one of the most effective methods to extract descriptive feature for object recognition applications. The algorithm has been widely applied in classification of aircrafts, ships, ground targets, etc [48–56]. Essentially, the algorithm derives a number of self-characteristic properties from a binary image of an object. These properties are invariant to rotation, scale and translation. Let $f(i, j)$ be a point of a digital image of size $M \times N$ (i=1,2, …, M and j=1,2, …, N). The two dimensional moments and central moments of order $(p+q)$ of $f(i, j)$, are defined as:

$$m_{pq} = \sum_{i=1}^{M} \sum_{j=1}^{N} i^p j^q f(i, j)$$

$$U_{pq} = \sum_{i=1}^{M} \sum_{j=1}^{N} (i - \bar{i})^p (j - \bar{j})^q f(i, j)$$

Where

$$\bar{i} = \frac{m_{10}}{m_{00}} \quad \bar{j} = \frac{m_{01}}{m_{00}}$$

From the second order and third order moments, a set of seven moment invariants are derived as follows [44]:

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}{}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$\begin{aligned}\phi_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] \\ & + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]\end{aligned}$$

$$\phi_6 = (\eta_{20} - \eta_{02})\left[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$\begin{aligned}\phi_7 = {} & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] \\ & - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]\end{aligned}$$

Where η_{pq} is the normalised central moments defined by:

$$\eta_{pq} = {}^{U_{pq}}\!/_{U_{00}^{r}}$$

where $r = \left[(p+q)/2\right] + 1$ and $p + q = 2, 3, \ldots$

4.5.1.1 Example of Invariant Properties of Hu Moments

Figure 4.28 shows images containing letter ‘A’, rotated and scaled, translated and noisy versions of letter ‘A’ and Fig. 4.29 shows letter ‘L’. Their respective moment invariants calculated using the moment invariants are shown in Tables 4.1 and 4.2. It is obvious from Table 4.1 that the algorithm produces the same result for the first three orientations of letter ‘A’ despite the different transformations applied upon them. There is only one value, i.e. Φ_1 displays a small discrepancy of 5.7% due to the difference in scale. The other values of the three figures are effectively the same for Φ_2, Φ_3, Φ_4, Φ_5, Φ_6 and Φ_7. The last letter, however, reveals the drawback of the algorithm: it is susceptible to noise. Specifically, the added noisy spot in the letter has changed the entire moment invariants set. This drawback suggests that moment invariants can only be applied on noise-free images in order to achieve

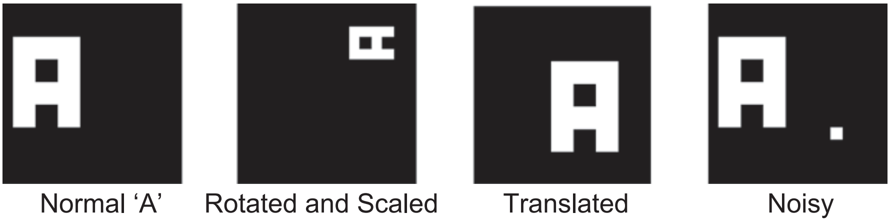


Fig. 4.28 Letter ‘A’ in different orientations

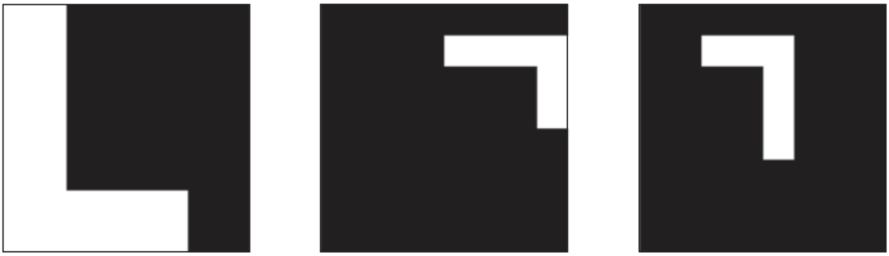


Fig. 4.29 Letter ‘L’ in different orientations

Table 4.1 Moment invariants of the different orientations of letter ‘A’

	A1	A2	A3	A4
Φ_1	0.2165	0.2165	0.204	0.25153
Φ_2	0.001936	0.001936	0.001936	0.002161
Φ_3	3.69×10^{-5}	3.69×10^{-5}	3.69×10^{-5}	0.004549
Φ_4	1.64×10^{-5}	1.64×10^{-5}	1.64×10^{-5}	0.002358
Φ_5	-4.03×10^{-10}	-4.03×10^{-10}	-4.03×10^{-10}	7.59×10^{-6}
Φ_6	7.21×10^{-7}	7.21×10^{-7}	7.21×10^{-7}	7.11×10^{-5}
Φ_7	0	0	0	1.43×10^{-6}

Table 4.2 Moment invariants of the different orientations of letter ‘L’

	L1	L2	L3
Φ_1	0.34028	0.31944	0.31944
Φ_2	0.043403	0.043403	0.043403
Φ_3	0.023148	0.023148	0.023148
Φ_4	0.002572	0.002572	0.002572
Φ_5	-5.56×10^{-6}	-5.56×10^{-6}	-5.56×10^{-6}
Φ_6	-0.00015	-0.00015	-0.00015
Φ_7	1.91×10^{-5}	1.91×10^{-5}	1.91×10^{-5}

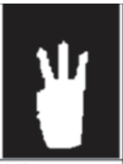
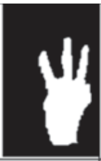
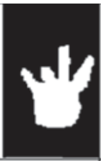
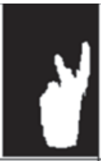
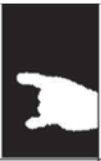
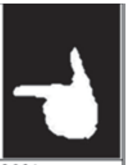
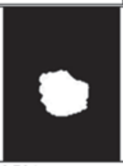
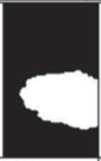
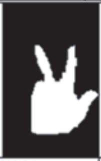
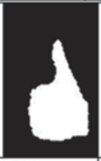
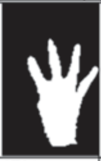
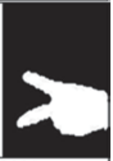
Hand Gesture						
Classification accuracy	44%	50%	74%	82%	97%	90%
Hand Gesture						
Classification accuracy	95%	96%	70%	95%	47%	89%

Table 4.3 Some hand gestures and their corresponding classification scores

the best results. Since the algorithm is firmly effective against transformations, a simple classifier can exploit these moment invariants values to differentiate as well as recognise the letter 'A' from other letters, such as the letter 'L'.

4.5.1.2 Application of Moment Invariants in Hand Gesture Recognition

The example in the previous section proved that moment invariants can be used for object recognition applications since it is rigidly invariant to scale, rotation and translation. The following account summarizes the advantages of moment invariants algorithm for gesture classification.

For each specific gesture, moment invariants always give a specific set of values. These values can be used to classify the gesture from a sample set. The set of chosen gestures have a set of unique moments.

- Moment invariants are invariant to translation, scaling and rotation. Therefore, the user can issue commands disregarding orientation of the hand.
- The algorithm is susceptible to noise. Most of this noise, however, is filtered at the gesture normalisation stage.
- The algorithm is moderately easy to implement and requires only an insignificant computational effort from the CPU. Feature extraction, as a result, can be progressed rapidly and efficiently.
- The first four moments, Φ_1, Φ_2, Φ_3, and Φ_4 are adequate to represent a gesture uniquely and hence result in a simple feature vector with only four values.

In 2005, the author successfully used moment invariants for classifying hand gestures to control consumer electronics with extremely high accuracy. This was partly due to the fact that selection of specific ten gestures resulted in a distinctive set of gestures which achieved good classification scores with Hu moments. The system was classified using a Neural Network approach [57]. Table 4.3 highlights the recognition accuracy for different hand gestures.

Feature extraction plays the most prominent role in any classification problem. Hand gesture recognition is no exception. Over the years, researchers have use d basic Fourier descriptor to exotic versions of Fourier descriptors such as Elliptic Fourier descriptors to modified Fourier descriptors to remove the limitations of feature extractions. Yet, poor results in classification further drove them to HOG to KL transform in an effort to robustly classify gestures. The authors personal involvement in developing a feature extraction method based on Hu moments improved the classification of hand postures significantly that resulted in a pioneering gesture controlled interface for home entertainment.

References

1. Zahn, C.T., Roskies, R.Z.: Fourier descriptors for plane close curves. IEEE Trans. Comput. **C-21**, 269–281 (1972)
2. Granlund, G.H.: Fourier preprocessing for hand print character recognition. IEEE Trans. Comput. **C-21**, 195–201 (1972)
3. Betrand, O., Queval, R., Maître, H.: Shape interpolation by Fourier descriptors with application to animation graphics. Signal Process. Chapter **4**, 53–58 (1981)
4. Persoon, E., Fu, K.: Shape discrimination using Fourier descriptors. IEEE Trans. Syst. Man Cybern. **7**(3), 170–179 (1977)
5. Zhang, D., Lu, G.: Shape-based image retrieval using generic Fourier descriptor. Signal Process. Image Commun. **17**(10), 825–848 (2002)
6. Bober, M.: MPEG-7 visual shape descriptors. IEEE Trans. Circuits Syst. Video Technol. **11**(6), 716–719 (2001)
7. Zhang, D., Lu, G: Generic Fourier descriptor for shape-based image retrieval. International Conference on Multimedia and Expo, 425–428 (2002)
8. Lin, C.S., Hwang, C.L.: New forms of shape invariants from elliptic Fourier descriptors. Pattern Recognit. **20**(5), 535–545 (1987)
9. Lin, C.S., Jungthirapanich, C.: Invariants of three-dimensional contours. Pattern Recognit **23**(8), 833–842 (1990)
10. Harding, P.R.G., Ellis, T.J.: Recognizing hand gesture using Fourier descriptors. Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004) 4, (2004)
11. Conseil, S., Bourennane, S., Martin, L.: Comparison of Fourier descriptors and Hu moments for hand posture recognition. European Signal Processing Conference, (2007)
12. http://www.idiap.ch/resource/gestures/
13. Utsumi, A., Miyasato, T., Kishino, F.: Multi-camera hand pose recognition system using skeleton image. IEEE International Workshop on Robot and Human Communication, 219–224 (1995)
14. Licsár, A., Szirányi, T.: Dynamic training of hand gesture recognition system. Proceedings of the 17th International Conference on Pattern Recognition vol. 4, (2004)
15. Rui, Y., She, A., Huang, T.S.: A modified Fourier descriptor for shape matching in MARS. Image Databases and Multimedia Search, 165–180 (1998)
16. Ng, C.W., Ranganath, S.: Real-time gesture recognition system and application. Image Vis. Comput. **20**, 993–1007 (2002)
17. Chen, F.S., Fu, C.M., Huang, C.L.: Hand gesture recognition using a real-time tracking method and hidden Markov models. Image Vis. Comput. **21**, 745–758 (2003)
18. Malima, A., Ozgur, E., Cetin, M.: A fast algorithm for vision-based hand gesture recognition for robot control. IEEE 14th Signal Processing and Communications Applications, (2006)
19. Hasan, M.M., Mishra, P.K.: HSV brightness factor matching for gesture recognition system. Int. J. Image Process. (IJIP). **4**(5), 456–467 (2010)

20. Li, X.: Gesture recognition based on fuzzy C-means clustering algorithm. Department of Computer Science, The University of Tennessee, Knoxville (2003)
21. Wachs, J., Kartoun, U., Stern, H., Edan, Y.: Real-time hand gesture telerobotic system. Proceedings of WAC, Florida (2002)
22. Segen, J., Kumar, S.: Fast and accurate 3d gesture recognition interface. Proceedings of the 14th International Conference on Pattern Recognition, vol. 1, 86 (1998)
23. Chang, C.C., Chen, I.-Y, Huang, Y.-S.: Hand pose recognition using curvature scale space. IEEE International Conference on Pattern Recognition (2002)
24. Mokhtarian, F., Mackworth, A.K.: Scale-based description and recognition of planar curves and two dimensional shapes. IEEE Trans. Pattern Anal. Machine Intell. **PAMI-8**(1), 34–43 (1986)
25. Mokhtarian, F., Mackworth, A.K.: A theory of multiscale, curvature-based shape representation for planar curves. IEEE Trans. Pattern Anal. Machine Intell. **14**(8), 789–805 (1992)
26. Singha, J., Das, K.: Hand gesture recognition based on Karhunen–Loeve transform. Mobile & Embedded Technology International Conference 2013, pp. 365–371 (2013)
27. Dalal, D., Triggs, B.: Histograms of oriented gradients for human detection. IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2, (2005)
28. Suard, F., Rakotomamonjy, A., Bensrhair, A., Broggi, A.: Pedestrian detection using infrared images and histograms of oriented gradients. IEEE Intelligent Vehicles Symposium, 206–212 (2006)
29. Bineesh, T.R., Simon, S.: Fast pedestrian detection using smart ROI separation and integral image based feature extraction. Int. J. Comput. Sci. Eng. **4**(11), 1771–1779 (2012)
30. Zhu, Q., Avidan, S., Yeh, M., Cheng, K.: Fast human detection using a cascade of histograms of oriented gradients. CVPR (2006)
31. Hu, B., Wang, S., Ding, X.: Pedestrian detection based on hybrid features. Second International Symposium on Intelligent Information Technology Application, 321–325 (2008)
32. Shashua, A., Gdalyahu, Y., Hayon, G.: Pedestrian detection for driving assistance systems: Single-frame classification and system level performance. Proceedings of IEEE Intelligent Vehicles Symposium, (2004)
33. Freeman, W., Roth, M.: Orientation histograms for hand gesture recognition. Mitsubishi Research Laboratory Report (1994)
34. Zhou, H., Lin, D.J., Huang, T.S.: Static hand gesture recognition based on local orientation histogram feature distribution model. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW'04) (2004)
35. Messer, T.: Static hand gesture recognition. University of Fribourg, Switzerland, (2009)
36. Misra, A., Okatani, T., Deguchi, K.: Hand gesture recognition using histogram of oriented gradients and partial least squares regression. MVA2011 IAPR Conference on Machine Vision Applications, 479–482 (2011)
37. Freeman, W.T., Roth, M.: Orientation histograms for hand gesture recognition. Intl. Workshop on Automatic Face- and Gesture- Recognition, IEEE Computer Society, Zurich, Switzerland, 296–300 (1995)
38. Lee, H.-J., Chung, J.-H.: Hand gesture recognition using orientation histogram. Proc. IEEE Region 10 Conf. **2**, 1355–1358 (1999)
39. Tartter, V.C., Knowlton, K.C.: Perception of sign language from an array of 27 moving spots. Nature **289**, 676–678 (1981)
40. Fukumoto, M., Mase, K., Suenaga, Y.: Real time detection of pointing actions for a glove free interface. Workshop on Machine Vision Applications, Tokyo, (1992)
41. Wold, H.: Partial least squares. In: Kotz S., Johnson, N. (eds.) Encyclopedia of Statistical Sciences, vol. 6, John Wiley: New York, pp. 581–591. (1985)
42. Schwartz, W.R., Kembhavi, A., Harwood, D., Davis, L.S.: Human detection using partial least squares analysis. Computer Vision, 2009 IEEE 12th International Conference on, (2009)
43. Zernike, F.: Beugungstheorie des Schneidenverfahrens und Seiner Verbesserten Form, der Phasenkontrastmethode. Physica **1**(8), 689–704 (1934)

44. Hu, M. K.: Visual pattern recognition by moment invariants. IRE Trans. Inf. Theory **8**(2), 179–187 (1962)
45. Teague, M. R.: Image analysis via the general theory of moments. Opt. Soc. Am. **70**(8), 920–930 (1979)
46. Khotanzad, A., Hong, Y. H.: Invariant image recognition by zernike moments. IEEE **12**(5), 489–497 (1990)
47. Prokop, R.J., Reeve, A.P.: A survey of moment-based techniques for unoccluded object representation and recognition. Survey Cornell University http://www.via.cornell.edu/ece547/text/survey.pdf. Accessed Oct. 29, 2013
48. Zhongliang, Q., Wenjun, W.: Automatic ship classification by superstructure moment invariants and two-stage classifier. ICCS/ISITA '92. 'Communications on the Move', 544–547 (1992)
49. Teague, M.R.: Image analysis via the general theory of moments. J. Opt. Soc. Am. **70**(8), 920–930 (1980)
50. Dudani, S.A., Breeding, K.J., McGhee, R.B.: Aircraft identification by moment invariants. IEEE Trans. Comput. **C-26**(1), 39–46 (1977)
51. Sluzek, A.: Using moment invariants to recognize and locate partially occluded 2D objects. Pattern Recognit. Lett. **7**(4), 253–257 (1988)
52. Gilmore, J.F., Boyd, W.W.: Building and bridge classification by invariant moments. SPIE 292, pp. 256–263 (1981)
53. Wong, R.Y., Hall, E.L.: Scene matching with invariant moments. Comput. Graphics Image Process. **8**, 16–24 (1978)
54. Smith, F.W., Wright, M.H.: Automatic ship photo interpretation by the method of moments. IEEE Trans. Comput. **C-20**, 1089–1095 (1971)
55. Boyce, J.F., Hossack, W.J.: Moment invariants for pattern recognition. Pattern Recognit. Lett. **1**(5, 6), 451–456 (1983)
56. Yin, B.H., Mack, H.: Target classification algorithms for video and FLIR imagery. SPIE 0302, pp. 134–141 (1981)
57. Premaratne, P., Safaei, F., Nguyen, Q.: Moment invariant based control system using hand gestures: Book intelligent computing in signal processing and pattern recognition. Book Series Lect. Notes Control Inf. Sci **345**, 322–333 (2006)

Chapter 5
Effective Hand Gesture Classification Approaches

Hand gestures recognition goals can only be fulfilled when gesture isolation is coupled with an effective feature extraction followed by highly efficient classification. In the context of machine vision, feature extraction and classification can be jointly called pattern recognition in which, previous known patterns are matched with a query gesture.

In hand gesture recognition, separating any input gesture into a pre-assigned class suggest that the classification problem is a multiclass classification. For any effort in classification to be effective, the feature extraction should have generated adequate conditions such that the classes cluster far apart from each other. There are two prominent ways of solving the classification problem: linear and nonlinear. Linear classification approach involving more than two classes is known as a multiclass linear classification which would be discussed in detail in this chapter.

It is quite difficult to identify any classification problem as linear or non-linear without observing the feature data. As an example, moment invariant features usually leads to non-linear classification as the features would visibly not cluster as separable. If some gestures are removed, it is always feasible to find a linear classifier that separates gestures. When assessing the features' likelihood of belonging to different clusters in classification, it is imperative to use metrics to ascertain the affinity of data to clusters. Given that many features are multi-dimensional in hand gesture recognition, simple thresholding will not be effective. Fortunately, there is a large number of metrics that handle multi-dimensional data offering different types of distance metrics that will also be visited in this chapter. This will be followed by an in-depth view of both linear and nonlinear techniques in classification especially related to hand gesture recognition.

One important aspect in classification is to concisely represent extracted features. This is due to the fact that features extracted usually contain correlation which is not possible to foresee. There are many methodologies out there that decouple this redundant information and represent the feature data more elegantly. This can be achieved by decorrelating data using Eigen vectors and Karhunen Loeve Transform which are used in Support Vector Machines (SVM) or Principal Component Analysis (PCA).

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_5,

In image processing, there is evidence of a long history in classification. Some of these classification approaches are related to classifying remote sensing data where images obtained by satellites or high-altitude aircrafts classifying data as vegetation, built-up areas or as waters. In hand gesture recognition, however, the gesture is fairly well-defined. The variability lies in size, the skin color, and lighting variation, movement of the hand and the other body movements. Therefore, even though the hand gesture is well-defined, the user may offer gestures which are quite similar to other gestures or some of the other variablities will result in capturing a hand gesture looking similar to a different gesture. This undoubtedly led to complications and limits the number of gestures a system can recognize with an acceptable accuracy.

Since all classification methods make use of one or multiple distance metrics for declaring class affinity, in depth knowledge of the leading distance metrics will enhance the understanding of the new researchers. Next few sections will describe most of the known distance metrics which are commonly associated with hand gesture recognition.

5.1 Distance Metrics

Classification is the process of finding a data point of set of data points to another set of cluster representative data points. Since the affinity can be thought of as a distance in multi-dimensional space, measurement of these distances has a prominent role in any type of classification. There are many different notions of distance measure based on the context of the problem from biological cell growth to tornado prediction. Some of the prominent and widely used metrics are described next.

5.1.1 Euclidean Distance

Historically, Euclidean distance which was famously conceived by Euclidean provides the best direct measure between two points in multiple dimensions. The original distance was calculated using the Pythagorean triads [1, 2]. Consider two vectors with n-dimension; $\boldsymbol{V}_1(\mathrm{a}_1, \mathrm{a}_2, \mathrm{a}_3, \mathrm{a}_4,\ldots \mathrm{a}_\mathrm{n})$, $\boldsymbol{V}_2(\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4,\ldots \mathrm{b}_\mathrm{n})$. There Euclidean distance is measured as:

$$D_{Euclid} = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_3 - b_3)^2 + (a_4 - b_4)^2 + \ldots + (a_n - b_n)^2}$$

and is the simplest form of distance measures used by a vast number of applications.

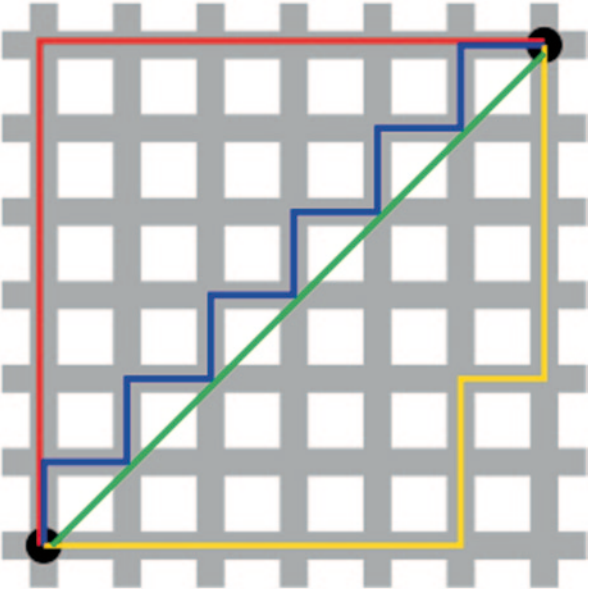


Fig. 5.1 Two dimensional view of Manhattan distance. Green line shows Euclidean distance where as *Red*, *Blue* and *Yellow* all show different Manhattan distance of the same value

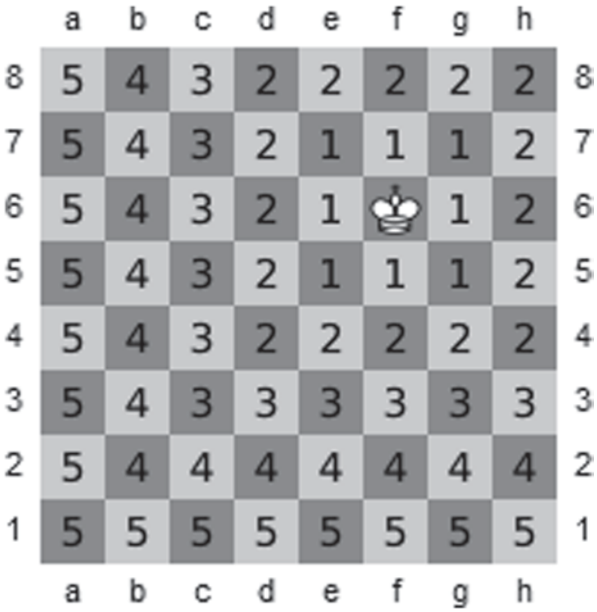
5.1.2 Manhattan Distance

This metric is also known as the *City Block Distance* assumes that in going from one vector to the other, it is only possible to travel directly along grid lines avoiding diagonal moves as illustrated in Fig. 5.1. Therefore the *Manhattan* distance is given by:

$$D_{\text{Manhattan}} = |a_1 - b_1| + |a_2 - b_2| + |a_3 - b_3| + |a_4 - b_4| + \ldots + |a_n - b_n|$$

5.1.3 Chebyshev Distance

Concieved by Pafnuty Chebyshev, Chebyshev distance (or Tchebychev distance), Maximum metric is a metric defined on a vector space where the distance between two vectors is the greatest of their differences along any coordinate dimension [3, 4]. It is also known as chessboard distance, since in the game of chess, the minimum number of moves needed by a king to go from one square on a chessboard to another equals the Chebyshev distance between the centers of the squares. If the squares have side length one, as represented in 2-D spatial coordinates with axes aligned to the edges of the board [5].



	a	b	c	d	e	f	g	h	
8	5	4	3	2	2	2	2	2	8
7	5	4	3	2	1	1	1	2	7
6	5	4	3	2	1	♔	1	2	6
5	5	4	3	2	1	1	1	2	5
4	5	4	3	2	2	2	2	2	4
3	5	4	3	3	3	3	3	3	3
2	5	4	4	4	4	4	4	4	2
1	5	5	5	5	5	5	5	5	1
	a	b	c	d	e	f	g	h	

Fig. 5.2 The Chebyshev distance between two spaces on a chess board gives the minimum number of moves a king requires to move between them. This is because a king can move diagonally, so that the jumps to cover the smaller distance parallel to a rank or column is effectively absorbed into the jumps covering the larger. Above are the Chebyshev distances of each square from the square f6

The metric is defined as:

$$D_{Chebyshev} = \max(|a_1 - b_1|,\ |a_2 - b_2|,\ |a_3 - b_3|,\ |a_4 - b_4|,\ \ldots |a_n - b_n|)$$

Figure 5.2 illustrates the *Chessboard Distance,* the metric assumes that moves can be made on the pixel grid as a ‘King’ making moves in chess, i.e. a diagonal move counts the same as a horizontal move.

5.1.4 Minkowski Distance

The *Minkowski distance* is a *superset* metric or a metric that can describe Euclidean, Manhattan and Chebyshev metrics as special cases. The definition is given by the following expression for a two vectors with dimension of n:

$$D_{Minkowski} = \left(|a_1 - b_1|^p + |a_2 - b_2|^p + |a_3 - b_3|^p + |a_4 - b_4|^p + \ldots + |a^n - b^n|^p\right)^{1/P}$$

Minkowski distance reduces to Euclidean distance when $p=2$ and Manhattan distance when $p=1$. It also reduces to Chebyshev distance when p reaches infinity as a limiting case described by:

$$D_{Chebyshev} = \max_{i=1}^{n} |a_i - b_i| = \lim_{p \to \infty} \left(\sum_{i=1}^{n} |a_i - b_i|^p \right)^{\frac{1}{p}}$$

It is interesting to notice that the Manhattan and Chebyshev metrics can be computed faster than the Euclidean metric so that applications relying on speed can make use of Chebyshev when the accuracy is not critically important.

5.1.5 Mahalanobis Distance

The Mahalanobis distance is a descriptive statistic that provides a relative measure of a data point's distance (residual) from a common point. It is a unitless measure introduced by P. C. Mahalanobis in 1936 [6]. The Mahalanobis distance is used to identify and gauge similarity of an unknown sample set to a known one. It differs from Euclidean distance in that it takes into account the correlations of the data set and is scale-invariant. The distance measure is given by:

$$D_{Mahalanobis} = (\mathbf{a}-\mathbf{b})^T C^{-1} (\mathbf{a}-\mathbf{b}).$$

Where *Covariance* Matrix of ***a*** and ***b*** defined by

$$C = \sum_{i=1}^{n} \frac{(a_i - \overline{\mathbf{a}})(b_i - \overline{\mathbf{b}})}{n-1}$$

The foundation of Mahalanobis measure can be understood by considering a classification problem. If given a point in n-dimensional space where its class affinity is in question, one would first estimate the average of the centre of mass of some given sample points whose class affinity is known. In determining the class affinity, the first logical step would be to the average or center of mass of the sample points. Intuitively, the closer the point in question is to this center of mass, the more likely it is to belong to the set.

However, it is important to know if the set is spread out over a large range or a small range. This would decide whether a given distance from the center is noteworthy or not. The simplistic approach is to estimate the standard deviation of the distances of the sample points from the center of mass. If the distance between the test point and the center of mass is less than one standard deviation, then it can be concluded that it is highly probable that the test point belongs to the set. The further away it is, the more likely that the test point should not be classified as belonging to the set. This intuitive approach can be made quantitative by defining the normalized distance between the test point and the set. This can be used in the normal distribution to derive the probability of the test point belonging to the set.

The drawback of the above approach was that it was assumed that the sample points are distributed about the center of mass in a spherical manner. Were the distribution to be decidedly non-spherical, for instance ellipsoidal, then the probability of the test point belonging to the set would depend not only on the distance from the center of mass, but also on the direction. In those directions where the ellipsoid has a short axis the test point must be closer, while in those where the axis is long the test point can be further away from the center.

Founding this on a mathematical basis, the ellipsoid that best represents the set's probability distribution can be estimated by building the covariance matrix of the samples. The Mahalanobis distance is simply the distance of the test point from the center of mass divided by the width of the ellipsoid in the direction of the test point.

In order to use the Mahalanobis distance to classify a test point as belonging to one of n classes, the covariance matrix has to be estimated for each class, usually based on samples known to belong to each class. This would be followed by computing the Mahalanobis distance from the test sample to each class. The minimum Mahalanobis distance would indicate the class affinity of the test sample.

As described in Chap. 4, the most important step in hand gesture recognition is the ability to capture unique features representing a unique gesture yet with enough robustness such that other users may offer the gesture with some variability. The system should be able to resolve the difference among different gestures yet be able to capture the common traits of the unique gesture offered by the same user at different times and the gestures offered by different users with different skin colors under different lighting conditions. Armed with the knowledge of various metrics for assessing the effectiveness of classification approaches, the next sections of this chapter will present in-depth discussion on linear and nonlinear classifications algorithms for developing a better insight for effective classification in hand gestures recognition.

5.2 Linear vs Nonlinear Classification Approaches

Shape classification problem in computer vision is known to be a more difficult problem than any other classification problem as the features which are extracted from shapes sometimes can result in very high dimensions. Humans are capable of visualizing up to three dimensions at a time however; very high dimensions usually leave the human user without any insight into high dimensionality of the classification problem. As shown in the example of Fig. 5.3, the red straight broken line can separate two data types with some error. The data encircled with the red broken circle will be misclassified. However, if this error is acceptable, the system can make use of a simple linear classifier. The common approach to classification problem is to start with some linear approaches and classify the data. If such classification results in error, then it can be assumed that the problem needs a non-linear classification approach to lower misclassification.

There are other instances where noise in the data (features) simply complicates the classification problem. Considering a simple hypothetical scenario of two linearly separable classes with added noise; it might be that a particular training set is not linearly separable due to the noise, but that no nonlinear classifier can generalize better than a linear one (e.g. one could simulate data such that a linear shift was the true underlying difference between the classes). If a system that is trained to classify with linear classifier fails to separate one class from the rest, then the system needs

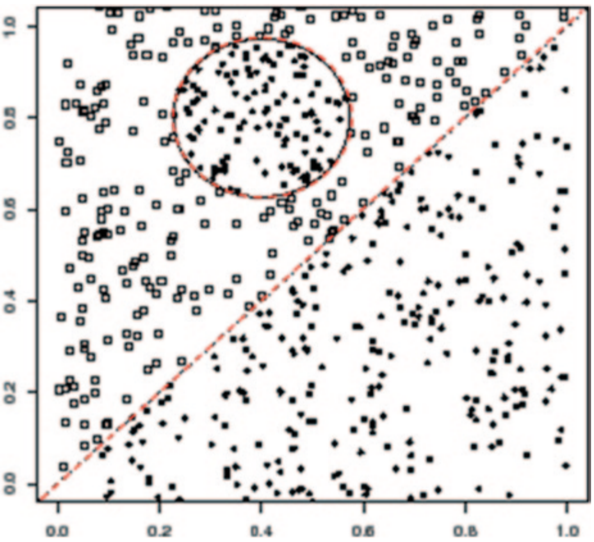


Fig. 5.3 *Red straight line* indicates a linear boundary that can separate two classes with some errors. If this linear classifier is used, data in the *circle* will be misclassified

to be non-linear in order to solve the problem. Alternatively, non-cross-validated measures that balance accuracy and complexity such as Bayesian model evidence could be used. When moving onto a non-linear solution, the improvement in accuracy should justify the use of a more complicated model that would also result in non-realtime classification especially in hand gesture recognition [7–14]. Figure 5.4 (left) indicates certain data distribution where linear classification whereas (right) indicates that a nonlinear approach is more suitable.

5.2.1 Linear Classifiers

Classification is the pivotal step in computer vision where a machine interprets the final outcome of any shape recognition. In applications such as face recognition,

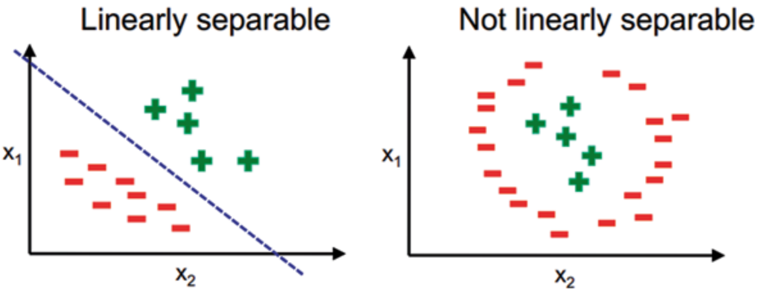


Fig. 5.4 Different data distribution requires different classification approaches [7]

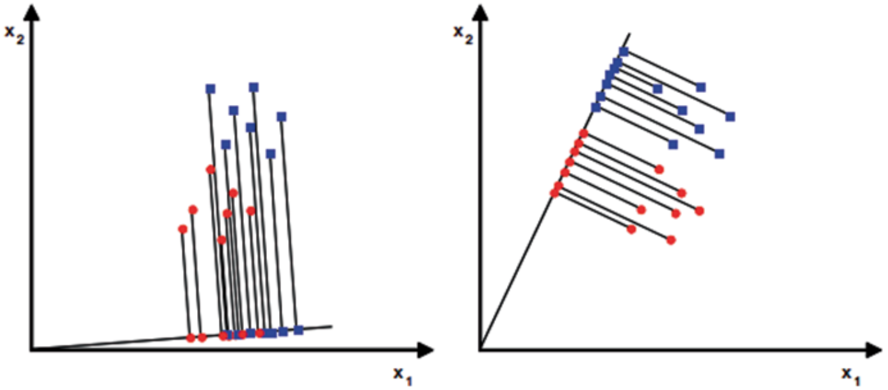


Fig. 5.5 LDA helps to classify data by maximizing the between class scatter and minimizing within class scatter when mapped to a different dimension

emotion detection or gesture recognition, the hurdle that limits success of any classification is the high dimensionality associated with large number of features needed to be extracted for improved classification. In order to handle data effectively and efficiently, its dimensionality needs to be reduced. Dimensionality reduction does not simply result in low dimensionality but leads to efficient, meaningful representation of reduced dimensionality. These lower dimensions now contain minimum number of features needed to represent the observed properties of the data. Dimension reduction techniques facilitate classification and compression of high-dimensional data. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are very powerful techniques for dimension reduction and feature extraction. Many applications use these techniques for recognition purposes especially in face recognition where LDA plays a very important roll but its performance get effected if there are less number of observation as compare to the dimensionality of the given samples [15]. PCA is being used so that LDA performance will not get degraded due to small sample size problem. As many recognition systems use PCA (Principal components analysis) [16, 17, 18]. PCA is an effective method which can reduce the dimensionality of the data and can effectively extract the required information of the image. It provides data which has no redundancy as Gabor filter wavelet are not orthogonal wavelets. In case of image processing the complexity of grouping the images can be reduced. As among other dimension reduction methods, PCA is the fastest algorithm which will reduce the time complexity [19]. PCA also has good performance for a small data set. Linear discriminant analysis (LDA) and the related Fisher's linear discriminant are methods used in statistics, pattern recognition and machine learning to find a linear combination of features which characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier or, more commonly, for dimensionality reduction before the final classification stage. LDA maps data from one dimension to another dimensional space wither the between class scatter is maximized while minimizing within class scatter as shown in Fig. 5.5.

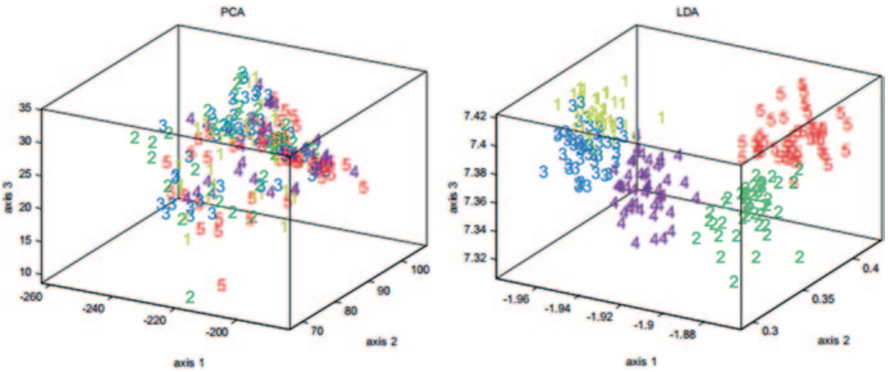


Fig. 5.6 From the 3D scatter plots it is clear that LDA outperforms PCA in terms of class discrimination, courtesy of [13]

LDA is also closely related to principal component analysis (PCA) and factor analysis in that they both look for linear combinations of variables which best explain the data [13]. LDA explicitly attempts to model the difference between the classes of data. PCA on the other hand does not take into account any difference in class, and factor analysis builds the feature combinations based on differences rather than similarities. Discriminant analysis is also different from factor analysis in that it is not an interdependence technique where a distinction between independent variables and dependent variables (also called criterion variables) must be made prior to classification. LDA is a parametric method based on unimodal Gaussian likelihoods. If LDA is used on distributions which are significantly non-Gaussian, the LDA projections may not preserve complex structure in the data needed for classification. Therefore, it is imperative that the user develops adequate knowledge about the feature distribution before any classification. This does not discourage a user from attempting to classify data using LDA involved approaches. In case the classification results are poor, non-Gaussian distributions may hold keys for its demise.

LDA's use 'can be better appreciated by looking at its performance in face recognition. In computerized face recognition, each face is represented by a large number of pixel values. Linear discriminant analysis is primarily used here to reduce the number of features to a more manageable number before classification. Each of the new dimensions is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher faces, while those obtained using the related principal component analysis are called Eigenfaces. The ability for LDA to outperform PCA is shown in Fig. 5.6 for a multi-class problem.

If the number of classes is more than two, then a natural extension of Fisher Linear discriminant exists using multiple discriminant analysis [20]. As in two-class case, the projection is from high dimensional space to a low dimensional space and the transformation suggested still maximize the ratio of intra-class scatter to the inter-class scatter [8]. But unlike the two-class case, the maximization should be done among several competing classes.

5.2.1.1 Fisher's Discriminant for Multiple Classes

Linear Discriminant Analysis, or simply LDA, is a well-known classification technique that has been used successfully in many statistical pattern recognition problems as was stated earlier. It was developed by Ronald Fisher, who was a professor of statistics at University College London, and is sometimes called Fisher Discriminant Analysis (FDA). The primary purpose of LDA is to separate samples of distinct groups by transforming the data to a different space that is optimal for distinguishing between the classes. In case of multiple classes, FDA can be extended to find a subspace which appears to contain all of the class variability. Assuming that each of C classes has mean μ_i and the same covariance Σ and the number of features is greater than the number of classes. Then the between class variability can be defined by the sample covariance of the class means such that

$$\Sigma_b = \frac{1}{C}\sum_{i=1}^{C}(\mu_i - \mu)(\mu_i - \mu)^T$$

Where μ is the means of the class means and *class mean* is defined as follows:

$$\mu_i = \frac{1}{M_i}\sum_{m=1}^{M_i}\vec{X}^i_m.$$

Here $\vec{Y} = \vec{D}^T\vec{X}$ signifies that $\vec{X}$ has been projected onto direction $\vec{D}$. The class separation or class scatter in a direction $\vec{D}$ will be given by the following expression:

$$s_k = \frac{\vec{D}^T\Sigma_b\vec{D}}{\vec{D}^T\Sigma\vec{D}}\sum_{m=1}^{M_i}\left(\vec{X}^k_m - \mu\right)\left(\vec{X}^k_{im} - \mu\right)^T.$$

This implies that $\vec{D}$ is an Eigenvector of $\Sigma^{-1}\Sigma_b$ where its separation will be equal to the corresponding Eigenvalue. If $\Sigma^{-1}\Sigma_b$ is diagonalizable, the variability between features will be contained in the subspace spanned by the Eigenvectors corresponding to the C-1largest Eigenvalues. These eigenvectors are primarily used in feature reduction, as in PCA. The eigenvectors corresponding to the smaller Eigenvalues will tend to be very sensitive to the exact choice of training data, and it is often necessary to use regularization.

In practice, the class means and covariances are not known but can be estimated from the training set. Either the maximum likelihood estimate or the maximum a posteriori estimate may be used in place of the exact value in the above equations. Although the estimates of the covariance may be considered optimal in some sense, this does not mean that the resulting discriminant obtained by substituting these values is optimal in any sense, even if the assumption of normally distributed classes is correct.

Another complication in applying LDA and Fisher's discriminant to real data occurs when the number of observations of each sample does not exceed the number

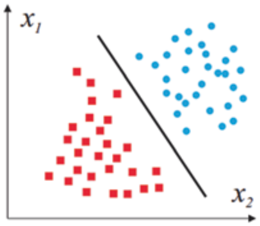


Fig. 5.7 A simple two class problem with clear class separation

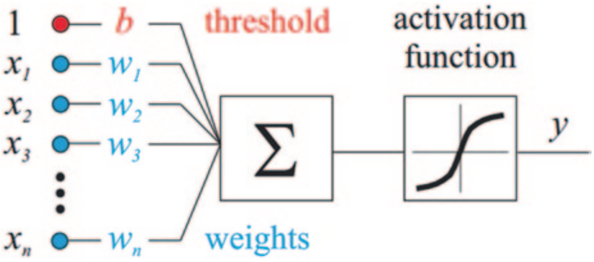


Fig. 5.8 Perceptron input/output relationship with weights, thresholds and activation functions

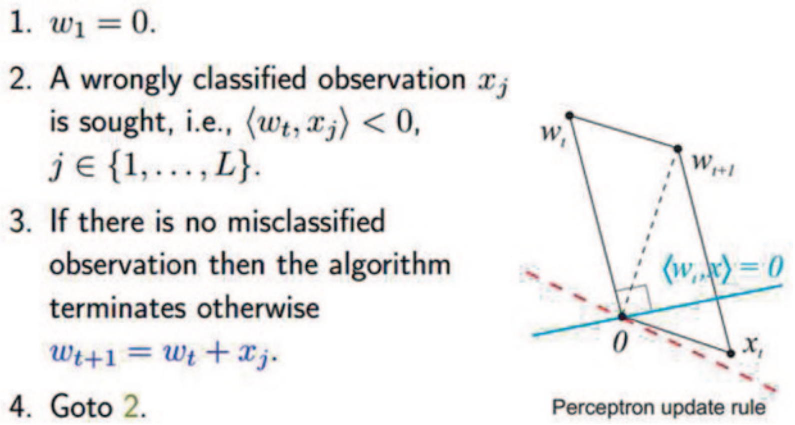
of samples [13]. In this case, the covariance estimates do not have full rank, and so cannot be inverted. There are a number of ways to deal with this. One is to use a pseudo inverse instead of the usual matrix inverse in the above formulae. However, better numeric stability may be achieved by first projecting the problem onto the subspace spanned by [21].

5.2.1.2 Multiclass Perceptron Classifier

In machine learning, the perceptron is an algorithm for supervised classification of an input into one of several possible non-binary outputs. It is known as a type of linear classifier as the classification algorithm makes its predictions based on a linear predictor function combining a set of weights with the feature vector describing a given input using the delta rule. The learning algorithm for perceptrons is an online algorithm, in that it processes elements in the training set one at a time. The perceptron algorithm was conceived in 1957 Frank Rosenblatt [22].

Even though, it was conceived as a two-class linear classifier, it has been extended to multiclass classification. Perceptron can be considered as the simplest example of Neural Network which would be discussed under nonlinear classification. The perceptron defines a hyperplane that split the representation space into two parts when handling two classes. Figure 5.7 shows a typical two-class problem with clear separation between classes and Fig. 5.8 shows the perceptron input-output relationship including all its intermediary functions.

Figure 5.9 depicts the online algorithm for two class-case where misclassifications are corrected continuously resulting in better classification results and Fig. 5.9 shows a commonly used activation function in perceptron classifier (Fig. 5.10).

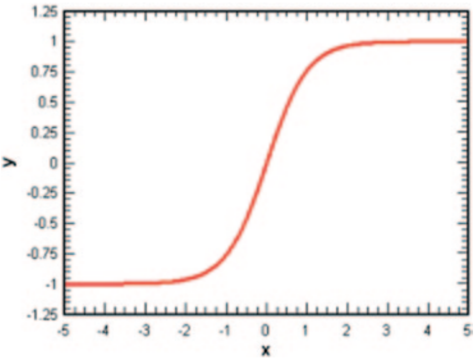


1. $w_1 = 0$.
2. A wrongly classified observation x_j is sought, i.e., $\langle w_t, x_j \rangle < 0$, $j \in \{1, \ldots, L\}$.
3. If there is no misclassified observation then the algorithm terminates otherwise $w_{t+1} = w_t + x_j$.
4. Goto 2.

Fig. 5.9 Rosenblatt perceptron algorithm for two class classification [22]

Fig. 5.10 A typical activation function: hyperbolic tangent function given by the expression

$$f(x) = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}$$



In multiclass perceptron algorithm, the classification starts with zero weights. Then each training vector is used one at a time and output is evaluated as follows:

$$y = \arg max_y \ \mathrm{w}_y.f(x)$$
$$= \arg max_y \sum_i \mathrm{w}_{y,i}.f_i(x).$$

If the output generates the correct classification, no adaptations of weights are carried out. However, if the classification is wrong, the score of the wrong weights are adjusted as follows:

$$w_y = w_y - f(x).$$

The weights of the correct answer are increased such that:

$$w_{y*} = w_{y*} + f(x).$$

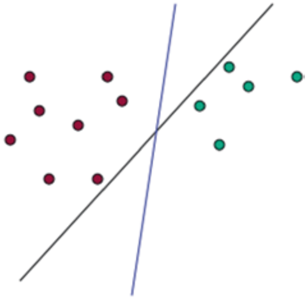


Fig. 5.11 When two classes can be separated using multiple decision boundaries, SVM determines the separation boundary in the most efficient way

The perceptron learning algorithm does not terminate if the learning set is not linearly separable. If the vectors are not linearly separable learning will never reach a point where all vectors are classified properly. The most famous example of the perceptron's inability to solve problems with linearly nonseparable vectors is the Boolean exclusive-OR problem. The solution spaces of decision boundaries for all binary functions and learning behaviors are studied in the reference [23].

In the context of artificial neural networks, a perceptron is an artificial neuron using the Heaviside step function as the activation function. The perceptron algorithm is also termed the single-layer perceptron, to distinguish it from a multilayer perceptron, which is a misnomer for a more complicated neural network. As a linear classifier, the single-layer perceptron is the simplest feedforward neural network.

5.2.1.3 Linear Support Vector Machine (SVM)

Linear support vector machines are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. The basic SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the output, making it a non-probabilistic binary linear classifier [24]. Given a set of training examples as shown in Fig. 5.11, each marked as belonging to one of two classes; an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.SVM are based on statistical learning theory and have the aim of determining the location of decision boundaries that produce the optimal separation of classes [25].

As shown in Fig. 5.11 and 5.12, SVMs maximize the margin around the separating hyperplane. The decision function is fully specified by a subset of training samples, the support vectors. SVM has been found attractive by researchers as it is known as a quadratic programming problem with application found in hand gesture recognition [26–32], and text classification methods [33–37]. These applications

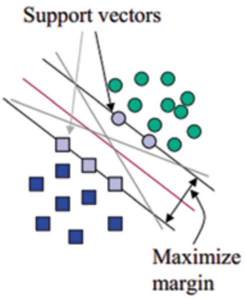


Fig. 5.12 Maximizing the separation between classes using support vectors [33]

are mainly feasible due to SVM's natural ability to handle large dimensional data with high accuracy and high flexibility with sparse representation of the solution using support vectors. They are also capable of making future predictions quickly without the need for probability estimates of data. Hence, they are also known as hard classifiers separated by Hyperplanes.

If the problem being investigated is separable and linear in two dimensions, a straight line can be found as discussed above. However, for higher dimensions, hyperplances need to be found by linear programming such as perceptrons. They can be expressed as straight lines given by $ax+by=c$. Support vectors are the elements of the training set that would change the position of the dividing hyper plane if removed. Support vectors are also the critical elements of the training set and the problem of finding the optimal hyper plane is an optimization problem and can be solved by optimization techniques (use Lagrange multipliers to get into a form that can be solved analytically).

In the case of a two-class pattern recognition problem in which the classes are linearly separable, the SVM selects from among the infinite number of linear decision boundaries the one that minimizes the generalization error. Thus, the selected decision boundary will be one that leaves the greatest margin between the two classes, where margin is defined as the sum of the distances to the hyperplane from the closest points of the two classes [38–54]. This problem of maximizing the margin can be solved using standard Quadratic Programming (QP) optimization techniques. The data points that are closest to the hyperplane are used to measure the margin; hence these data points are termed 'support vectors'. Consequently, the number of support vectors is small [25].

If the two classes are not linearly separable, the SVM tries to find the hyperplane that maximizes the margin while, at the same time, minimizing a quantity proportional to the number of misclassification errors. The trade-off between margin and misclassification error is controlled by a user-defined constant [38]. SVM can also be extended to handle non-linear decision surfaces.

SVM were initially designed for binary (two-class) problems. When dealing with multiple classes, an appropriate multi-class method is needed. Vapnik in 1995 suggested comparing one class with the others taken together [25]. This strategy generates n classifiers, where n is the number of classes. The final output is the class that corresponds to the SVM with the largest margin, as defined above. For multiclass problems one has to determine *n*-hyperplanes. Thus, this method requires the solution of *n* QP optimization problems, each of which separates one class from the remaining classes. This strategy can be described as ‘one against the rest’.

A second approach is to combine several classifiers (‘one against one’) to perform pair-wise comparisons between all *n* classes [55]. Thus, all possible two-class classifiers are evaluated from the training set of *n* classes, each classifier being trained on only two out of *n* classes, giving a total of *n*(*n*-1)/2 classifiers. Applying each classifier to the test data vectors gives one vote to the winning class. The data is assigned the label of the class with most votes. The results of a recent analysis of multi-class strategies are provided by Hsu and Lin [56].

SVM for Multiclass Classification Multiclass SVM aims to assign labels to instances by using support vector machines, where the labels are drawn from a finite set of several elements. The dominant approach for doing so is to reduce the single multiclass problem into multiple binary classification problems. Originally, SVMs were developed for binary classification. However, applications of binary classification are very limited especially in remote sensing land cover classification where most of the classification problems involve more than two classes. A number of methods to generate multiclass SVMs from binary SVMs have been proposed by researchers and is still a continuing research topic.

Instead of creating many binary classifiers to determine the class labels, this method attempts to directly solve a multiclass problem [48, 57, 58]. This is achieved by modifying the binary class objective function and adding a constraint to it for every class. The modified objective function allows simultaneous computation of multiclass classification and is given by [59] as shown next:

$$\min_{w,b,\xi}\left[\frac{1}{2}\sum_{i=1}^{M}\|w\|^{2}+C\sum_{i=1}^{k}\sum_{r\neq y_i}\xi_i^r\right]$$

Subject to the constraints,

$\mathbf{w}_{y_i}.\mathbf{x}_i+b_{y_i}\geq\mathbf{w}_r.\mathbf{x}_j+b_r+2-\xi_i^r$ and $\xi_i^r\geq 0$ for $i=1,\ldots\ k$.

Where y_{i} ∈ {1,…, M} are the multiclass labels of the data vectors and r ∈ {1,…, M}/y_i are multiclass labelling excluding y_i [25].

Lee et al. and Schölkopf and Smola showed that the results from this method and the one-against-the-rest are similar [57–59]. However, in this approach, the optimization algorithm has to consider all the support vectors at the same time. Therefore, it may be able to handle massive data sets but the memory requirement and thus, the computational time requirements may be very high. To summarize, it may be said that the choice of a multiclass method depends on the problem in hand. A user should consider the accuracy requirement, the computational time, the resources available and the nature of the problem. For example, the multiclass objective function approach may not be suitable for a problem that contains a large number of training samples and classes due to the requirement of large memory and extremely long computational time.

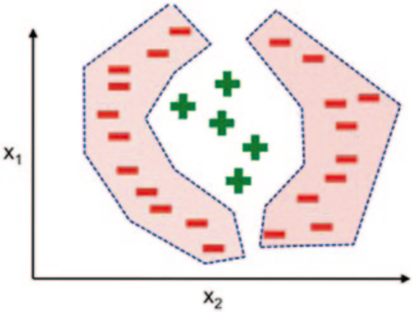


Fig. 5.13 Nonlinear classifiers have nonlinear and possibly discontinuous decision boundaries [77]

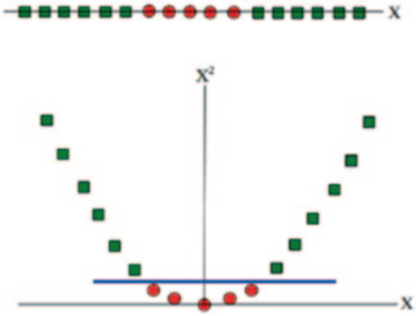


Fig. 5.14 Projecting data that is not linearly separable into a higher dimensional space can make it linearly separable [60]

5.2.2 NonLinear Classifiers

When class clusters cannot be separated effectively by a straight line, the problem becomes a nonlinear classification problem. This is illustrated in Fig. 5.13.

In some nonlinear classification approaches such as SVM (nonlinear SVM), nonlinear feature space is mapped to a linear space before classification. This is popularly known as the 'Kernal Trick'. Figure 5.14 shows that the 'red' dots can be separated from 'green' squares easily when the data is mapped into a different dimension using a single threshold.There are other techniques such as Artificial Neural Networks (ANN) where no such transformation is required. As would be discussed in details in this chapter, ANN has been shown to be very effective in classifying known objects or hand gestures when effective features are extracted and adequately trained with sample data. Hidden Markov Models (HMM) have been increasingly being used in hand gesture recognition for its ability to describe postures and gestures as incremental steps which is not possible with many other classification approaches. K Nearest Neighbor method has also been extremely popular as a general nonlinear classification approach when ample amount of sample data or feature data are available. However, this is also plagued by its tendency to result in long processing time when large amount of sample data is available.

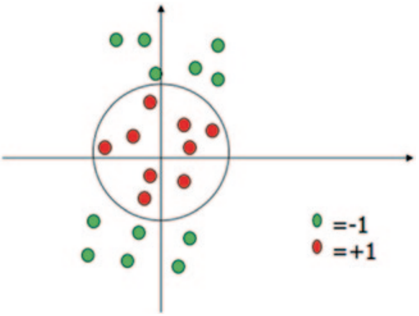


Fig. 5.15 Nonlinear Classification problem that can be attempted by nonlinear SVM [61]

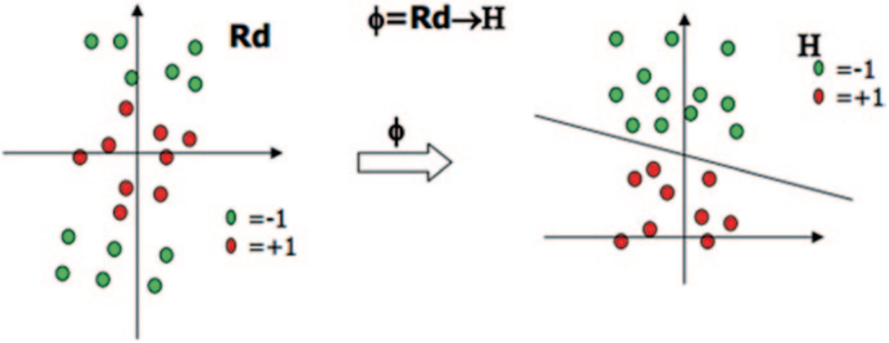


Fig. 5.16 Data can be mapped to a higher dimension for easy separation using SVM [59]

5.2.2.1 Non Linear SVM

The approach in nonlinear SVM is to map nonseparable data onto higher dimensions so that they are linearly separable as shown in Fig. 5.15. As depicted here, the red data cannot be effectively separated from the blue data in dimension x. When this data mapped to a higher dimension such that $x \mapsto \{x^2,\ x\}$, a linear separator can be found now to classify the data as shown on the bottom of the Fig 5.16. As would be obvious, such mapping could be extremely costly if the mapping for instance results in a quadratic space $x \mapsto \left\{x_1^2 x^2{}_2, \ldots x^2{}_D,\ x_1x_2x_1x_3 \ldots\right\}$. This leads to additional features and to inefficiency. The problem can be avoided by using Kernels approach which is extensively discussed in [62].

In many practical cases, it is not quite obvious as to what transform would result in a linear separable problem. SVM is a supervised learning method which is also a linear classifier that maximizes the distance between the decision lines. The main advantage of SVM is that it can use kernels for non-linear data transformation to cluster data into more separable ones in a new feature space as shown in Fig. 5.17. The main principle behind using SVM is to divide the given data into two distinct categories and then to get hyper-plane to separate the given classes.

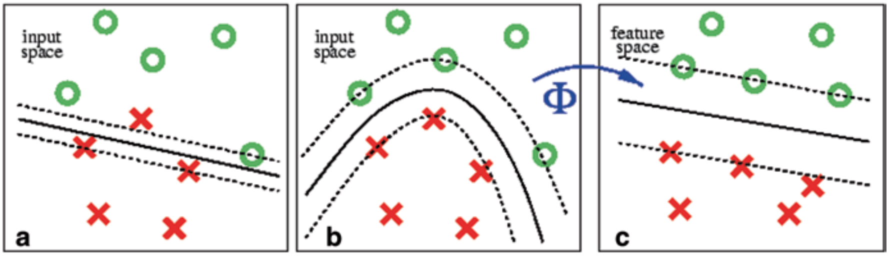


Fig. 5.17 Three different views on the same two class separation problem. **a** A linear separation of the input points is not possible without errors. Even allowing misclassification of one data point results in a small margin. **b** A better separation is provided by a non-linear surface in the input space. **c** This non-linear surface corresponds to a linear surface in a feature space. Data points are mapped from input space to feature space by the function Φ induced by the kernel function k [59]

Mathematically the well-known 'kernel trick' which brings a nonlinear problem to a linear one with ease can be describe as follows. Kernel-methods first preprocess the data by a certain non-linear mapping $\boldsymbol{\Phi}$ and then apply the same linear algorithm as before but in the image space of $\boldsymbol{\Phi}$. (cf. Fig. 5.17 for an illustration). More formally the following mapping $\boldsymbol{\Phi}: \mathbb{R}^N \rightarrow \varepsilon$ is performed.

$$x \mapsto \Phi(x)$$

to the data $x_1, \ldots x_M \in \aleph$ and consider our algorithms in ε instead of $\mathcal{X}$. The sample is preprocessed as

$$\{(\Phi(x_1), y_1), \ldots, (\Phi(x_M), y_M)\} \subseteq (\varepsilon \times \mathrm{y})^M.$$

In certain applications, there would be sufficient knowledge about problem in hand such that an appropriate Φ can be designed [63, 64]. The mapping should be simple enough so that ε will not be overly high dimensional leading to a feasible mapping. Similar notions are attempted in (single hidden layer) neural networks [65], radial basis networks [66] or Boosting algorithms [67], where the input data are mapped to some representation given by the hidden layer, the RBF bumps or the hypotheses space, respectively [68]. One of the key advantages of kernel-methods is that a suitably chosen Φ results in an algorithm that has powerful nonlinearities but is still very intuitive and retains most of the favorable properties of its linear input space version.

Besides being useful tools for the computation of dot-products in high- or infinite-dimensional spaces, kernels possess some additional properties that make them an interesting choice in algorithms. It was shown [69] that using a particular positive definite kernel correspond to an implicit choice of a regularization operator. For translation-invariant kernels, the regularization properties can be expressed conveniently in Fourier space in terms of the frequencies [70, 71]. For example, Gaussian kernels correspond to a general smoothness assumption in all k^{th} order derivatives [70]. Alternatively, using the property of correspondence, kernels matching a certain prior about the frequency content of the data can be constructed so as

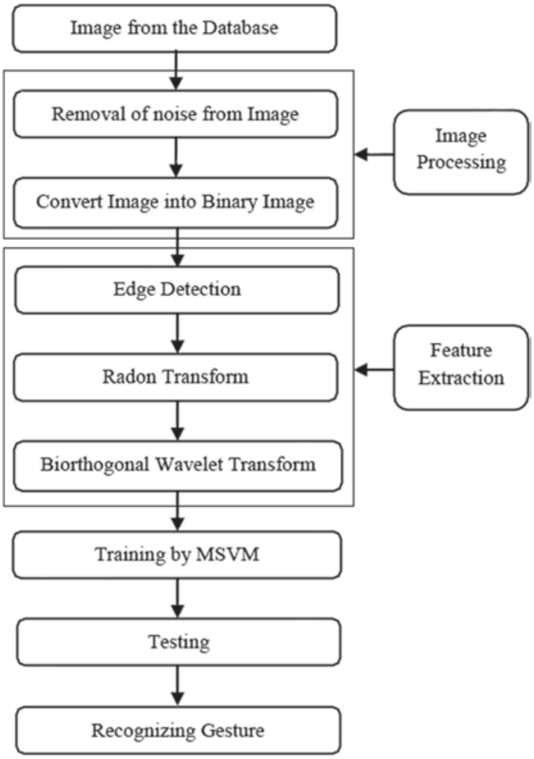


Fig. 5.18 SVM based hand gesture recognition, courtesy of Rahman et al. [83]

to reflect the known prior problem knowledge. Another particularly useful feature of kernel functions is that they are simply not restricted to kernels that operate on vectorial data, (e.g. $\chi = \mathbb{R}^N$). In principle, it is also possible to define positive kernels for e.g. strings or graphs, i.e. making it possible to embed discrete objects into a metric space and apply metric-based algorithms (e.g. [63, 72, 73]). Furthermore, many algorithms can be formulated using so called *conditionally positive definite kernels* [70, 74] which are a superclass of the positive definite kernels. They can be interpreted as generalized nonlinear dissimilarity measures as opposed mere scalar products and are also applicable in kernel PCA. More information on nonlinear SVM and Kernel methods can be found in the following [75–82].

A hand gesture recognition techniques using SVM is proposed by Rahman and Afrin [83] in which they use features extracted from the skin segmented hand postures. The hand postures are Radon transformed and Biorthogonal Transform coefficients were used for multiclass SVM classification. In this usage of SVM, they find the optimal separating hyper plane such that error for unseen patterns is minimized. They used classification for 10 gestures denoting the letters from the alphabet, A, B, C, D, G, H, I, L, V, and Y. They had moderate success for their approach which can be summarized using the Fig. 5.18.

Chen et al. [84] reported a 3 camera angle image capture system that used 3 SVM classifiers for each angle that voted and fused the classification results. Their

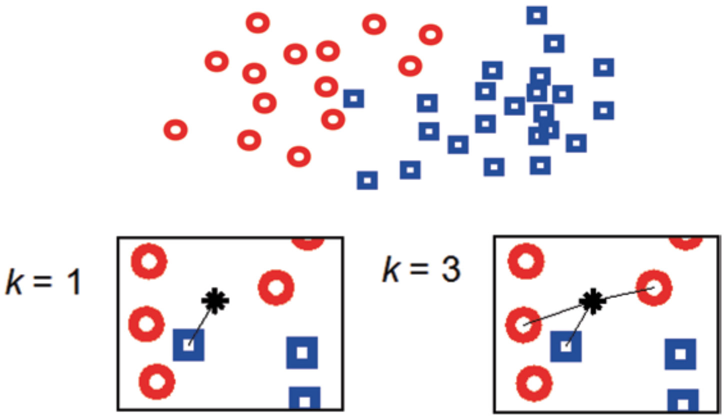


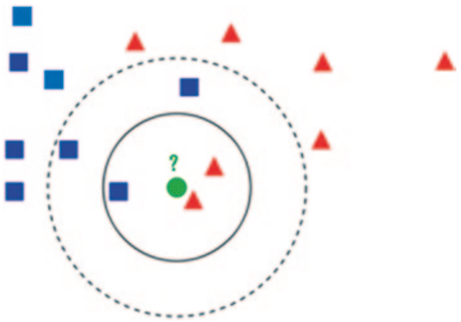
Fig. 5.19 KNN Algorithm. Two classes shown in *red circles* and *blue squares* on the *top* diagram. For a given query point *q* (shown in *), assign the class of the nearest neighbor. Compute the *k* nearest neighbors and assign the class by majority vote [86]

system was very sophisticated yet lacked proper feature extraction. They used a grayscale histogram equalized images and used a binary classification in the training phase with radial basis functions set as kernel functions. The reported performance accuracy was limited to be around 80 % with large misclassification in some gesture classes.

5.2.2.2 Nearest Neighbor Classification

K-nearest neighbors (KNN) is a classification (or regression) algorithm that in order to determine the classification of a point, group affiliations of the nearest neighbors are considered. It is a non-parametric method for classifying objects based on closest training examples in the feature space. It is also known as a supervised classification technique as the membership of the other class members (points) is known. KNN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. Many early work in pattern recognition relied on k-nearest neighbor algorithm as it is one of the simplest classification algorithms. In this classification, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor as shown in Fig. 5.19 [85]. Many early work on hand gesture recognition relied on KNN as it was highly effective inductive inference method for noisy training data and complex target functions. The target function or data points for a whole space may be described as a combination of less complex local approximations. The learning process is a simple and straight forward even though the classification is time consuming. Large number of data points could result in extremely high dimensions known as curse of dimensionality.

Fig. 5.20 KNN example



It can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. Usually, each neighbor is given a weight of 1/*d*, where *d* is the distance to the neighbor. This scheme is a generalization of linear interpolation [86]. The neighbors are taken from a set of objects for which the correct classification or membership is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. The *k*-nearest neighbor algorithm is sensitive to the local structure of the data [86]. Nearest neighbor rules in effect implicitly compute the decision boundary. It is also possible to compute the decision boundary explicitly, and to do so efficiently, so that the computational complexity is a function of the boundary complexity [87].

Algorithm The KNN algorithm can be better understood by following how the classification is carried out in Fig. 5.20. The test object shown in green circle should be classified either to the first class of dark blue squares, light blue squares or to the third class of red triangles. If $k=3$ (solid line circle) it is assigned to the third class because there are 2 triangles and only 1 dark blue square inside the inner circle. If $k=5$ (dashed line circle) it is assigned to the first class (3 dark blue squares vs. 2 triangles inside the outer circle).

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples.

In the classification phase, *k* is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the *k* training samples nearest to that query point. A commonly used distance metric for continuous variables is Euclidean distance which can be calculated using the following expression of Euclidean distance minimization:

$$\underset{S}{\arg\ \min} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \neq \mathbf{x}_i} \left\| \mathbf{x}_j - \mathbf{x}_i \right\|.$$

For discrete variables, such as for text classification, another metric can be used, such as the overlap metric (or Hamming distance). Often, the classification accu-

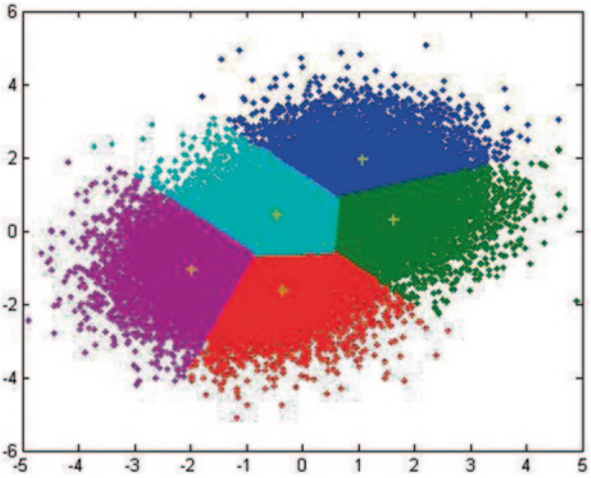


Fig. 5.21 K-means nearest neighbor representation, courtesy of [88]

racy of KNN can be improved significantly if the distance metric is learned with specialized algorithms such as Large Margin Nearest Neighbor or Neighborhood components analysis.

One of the major drawbacks in KNN is its inability to handle uneven class distribution of data. If one class tends to have more data than another class, the "majority voting" results in poor or misclassification. Samples of a more frequent class tend to dominate the prediction of the new sample, because they tend to be common among the *k* nearest neighbors due to their large number [87]. One way to overcome this problem is to weight the classification, taking into account the distance from the test point to each of its *k* nearest neighbors as was mentioned before. The class of each of the *k* nearest points is multiplied by a weight proportional to the inverse of the distance from that point to the test point as was mentioned before. K- means nearest neighbor can handle some of these drawbacks.

5.2.2.3 K-Means Nearest Neighbor Classification

K-means is a clustering algorithm that tries to partition a set of points into K clusters such that each point belongs to the cluster with the nearest mean, serving as a prototype of the cluster as shown in Fig. 5.21. It is an unsupervised classification technique unlike the KNN as the points have no external classification.

Given a set of observations ($\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_n$), where each observation is a *d*-dimensional real vector, *k*-means clustering aims to partition the *n* observations into *k* sets ($k \leq n$) $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster sum of squares (WCSS):

$$\underset{S}{\arg\ \min} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in \mathbf{s}_i} \left\| \mathbf{x}_j - \mu_i \right\|$$

Where $\boldsymbol{\mu}_i$ is the mean of points in S_i.

Since the clusters are not labeled, this technique is not commonly used in hand gesture recognition. However, KNN problem can be extended to incorporate cluster mean instead of the *k* neighbors which would not require the distance to all the points in a classification field. Yet, it is up to individual researcher to report the success of such an approach.

Usage of k-Nearest Neighbor in Hand Gesture Recognition There have been few attempts to use KNN as a classification means for hand gesture recognition. Some of these attempts used KNN in combination of probabilistic approaches. The main trend for using KNN is its simplicity and the intuitive sense over other techniques. However, when the feature vector had multiple dimensions and the clustering in each domain was not distinctive, the researchers resorted to other methods. However, data or feature decorrelation techniques such as Vector Quantization can be used to regenerate clusters in higher dimensional space so that KNN can be more successful as a pattern classification technique.

Ziaie et al. reported a classification scheme based on naïve Bayes classifier based on KNN with distance weighting for static hand gesture recognition [89, 90]. They reported a performance of 93 % accuracy for limited hand postures which were used in a human-robot dialog system. One of the major drawbacks of the KNN technique is the system is required to calculate the vector distance of the test gesture to all other neighbors in the system in order to select the *k* nearest neighbors. When a system has far more training data than another system, this results in unacceptable amount of computing time. This has resulted in discouraging many researchers from using KNN in pattern classification.

Vafadar and Berhad [91] reported a hand gesture recognition system based on spatio-temporal volume analysis technique in which hand contour extraction over time represented gesture paths for dynamic gesture recognition. In their approach, they used three types of classification methods: KNN, learning Vector Quantization Neural Network and back propagation neural networks. The reported that they had better recognition rate 96.6–99.6 % with a *k* value of 2 (two nearest neighbor). Kollorz et al. described a hand gesture recognition technique based on depth information from a Time of Flight camera [92]. The hand size information was extracted and projected on to x-y plane. The x-y data and the depth information were used in a KNN classification approach with 94 % accuracy for 12 gestures from 34 persons and a classification time of 30 ms.

5.2.2.4 Multi Layer Neural Networks

Artificial Neural Network or simply Neural Network as they are commonly known, models the behavior of neurons found in all animal life. These models effectively represent some of the fundamental attributes of neurons such as memory and outputs due one or many inputs exceeding pre-assigned thresholds. Scientists have used this notion to mimic the fundamental behavior of neurons using building blocks

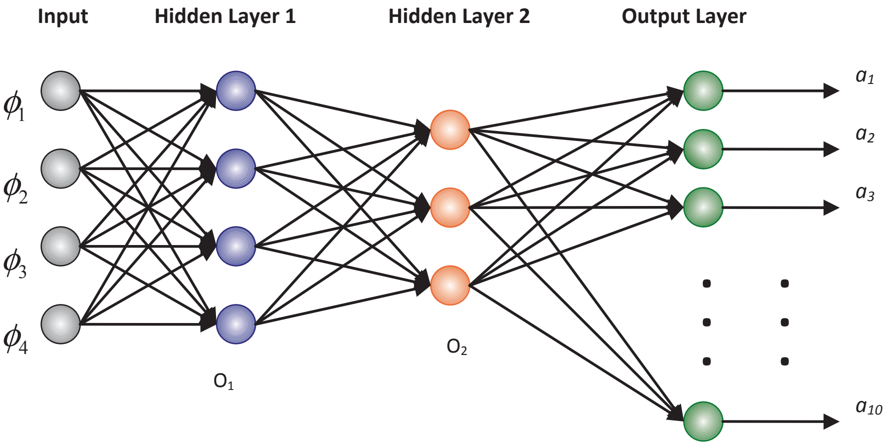


Fig. 5.22 Structure of the neural network classifier (conception perspective)

of mathematical functions with highly promising results. Today, there are myriad of scientific evidence to suggest that most non-linear classification problems have been attempted with Neural Networks. What has attracted the most interest in neural networks is the ability to memorize experience which is not available with many other classification approaches.

The artificial neural network refers to layers of inter connections between multiple inputs and multiple outputs. As shown in Fig. 5.22, many or one input may result in an output that would be used for many classification problems. The network nodes have the ability to remember their experience as a fundamental attribute of neurons due the weight each node is associated with. There are connections between input and output and some architectures have feedback and feed-forward from different layers. Each node executes a metric on incoming input to determine whether the input level exceeds a threshold to pass the communication of the signal to the next level. As it happens with human beings, for instance, some individuals may develop higher threshold for pain in which case their reaction to a level of pain will be different to a person experiencing the same level of pain.

It would be interesting to learn the historical developments of neural networks and their use in engineering problems. The first step toward artificial neural networks came in 1943 when Warren McCulloch, a neurophysiologist, and a young mathematician, Walter Pitts, wrote a paper on how neurons might work. They modeled a simple neural network with electrical circuits. Reinforcing this concept of neurons and how they work was a book written by Donald Hebb and the *The Organization of Behavior* was written in 1949 which pointed out that neural pathways are strengthened each time that they are used [93]. As computers advanced into their infancy of the 1950s, it became possible to begin to model the rudiments of these theories concerning human thought. Nathanial Rochester from the IBM research laboratories led the first effort to simulate a neural network. That first attempt failed however, their later attempts succeeded.

In 1956 the Dartmouth Summer Research Project on Artificial Intelligence provided a boost to both artificial intelligence and neural networks. One of the outcomes of this process was to stimulate research in both the intelligent side, AI, as it is known throughout the industry, and in the much lower level neural processing part of the brain [93]. In the years following the Dartmouth Project, John von Neumann suggested imitating simple neuron functions by using telegraph relays or vacuum tubes. Similarly, Frank Rosenblatt, a neuro-biologist of Cornell, began work on the Perceptron. He was intrigued with the operation of the eye of a fly as local level processing takes place in an eye of a fly which helps the fly to flee threatened. The Perceptron, which resulted from this research, was built in hardware and is the oldest neural network still in use today. A single-layer perceptron was found to be useful in classifying a continuous-valued set of inputs into one of two classes. The perceptron computes a weighted sum of the inputs, subtracts a threshold, and passes one of two possible values out as the result. Unfortunately, the perceptron was limited and was proven as such during the "disillusioned years" in Marvin Minsky and Seymour Papert's 1969 book Perceptrons [93].

In 1959, Bernard Widrow and Marcian Hoff of Stanford developed models they called ADALINE and MADALINE. These models were named for their use of Multiple ADAptive LINear Elements. MADALINE was the first neural network to be applied to a real world problem. It was an adaptive filter which eliminates echoes on phone lines which is still found commercial use. Unfortunately, these earlier successes caused unsubstantiated expectations on neural networks, particularly in light of the limitation in the electronics available then. These fears, combined with unfulfilled, outrageous claims, caused respected voices to critique the neural network research which resulted in funding cuts to neural network based research.

This period of stunted growth lasted through 1981. In 1982 several events caused a renewed interest in neural networks especially when John Hopfield of University of California Technology presented a paper to the national Academy of Sciences. Hopfield's approach was not to simply model brains but to create useful devices. With clarity and mathematical analysis, he showed how such networks could work and what they could do [93].

Pattern recognition is a powerful technique for harnessing information present in data for recognition purposes. Neural networks learn to recognize the patterns which exist in data sets. Neural network based system is developed through learning rather than programming. Programming needs a profound understanding of mechanisms when a system is developed through modeling. However, many pattern recognition problems are overly difficult to model as the pattern recognition problem can be thought of as in its infancy. Neural networks present the opportunity to classify data without in-depth knowledge of the pattern classification problem or accurate system modeling. Neural networks learn patterns that exist in data in a way which is still a mystery to the science and are also flexible in a changing environment. Rule based systems or programmed systems are limited to specific problems for which they were designed and when conditions change, they are no longer applicable. One of the disadvantageous of neural networks is that learning time or network training typically requires a long time when more training data is available. They are also adept at building informative models where more conventional approaches fail.

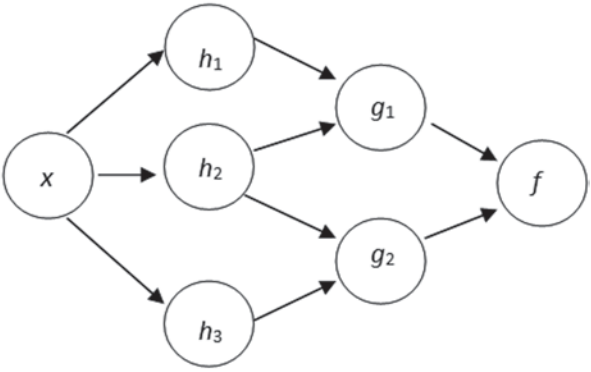


Fig. 5.23 Input, hidden layers and output relationships of a neural network

Even though their ability to model complex interactions a mystery, neural networks can easily model data which is too difficult to model with traditional approaches such as inferential statistics or programming logic. Performance of neural networks is typically outperforms classical statistical modeling. They also have the ability to build models that are more reflective of the structure of the data in significantly less time.

An ANN is typically defined by three types of parameters:

1. The interconnection pattern between different layers of neurons
2. The learning process for updating the weights of the interconnections
3. The activation function that converts a neuron's weighted input to its output activation.

Figure 5.23 shows a diagram neural network which can be used to describe the mathematical relationship of input and output. Neuron's network function or the output $f(x)$ can be mathematically described using the following expression:

$$f(x) = K\left(\sum_i w_i g_i(x)\right)$$

Here, w_i's are the weights of synapses g_i's are the output function of previous layer and K is known as the activation function. There are few activation functions or transfer functions commonly use such as hyperbolic tangent functions that was described in the 5.2.1.2.

Learning Paradigms There are three major learning paradigms in neural network learning corresponding to a particular abstract learning task. These are supervised learning, unsupervised learning and reinforcement learning.

Supervised Learning It can be stated that the neural networks enjoy the popularity that it enjoys today due to the ability to learn under supervision. In supervised learning, the system has ample pre-classified data for with corresponding inputs. The system uses the input and generates output and compare with the expected output. If the system output differs from the expected output, then the error is fed back to the system to adjust the weights until the outputs totally agree with the expected output which is known as training. This feeding error backwards through the network is known as *back-propagation*. Both the Multi-Layer Perceptron and the Radial Basis Function are supervised learning techniques. The Multi-Layer Perceptron uses the back-propagation while the Radial Basis Function is a feed-forward approach which trains on a single pass of the data.

Some systems may need many training cycles to adequately train the system so that system output matches exactly with the expected outputs. Different weight adjustment mechanisms are employed to optimize the weights so that the weights converge to their optimum values quickly [94–98].

Unsupervised Learning Unsupervised learning is used when the output is not expected to classify data but to group 'similar looking' entities together. The primary unsupervised technique is the Kohonen network which is used in cluster analysis. The advantage of the neural network for this type of analysis is that it requires no initial assumptions about what constitutes a group or how many groups there are. The system starts with a *clean slate* or *no prior knowledge* and is not biased about which factors should be most important [99–103]. Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering.

Reinforcement Learning Reinforcement learning is learning by interacting with an environment where input is not explicitly given as in supervised or unsupervised learning. The input is generated by the system by observation. The system learns from the consequences of its actions, rather than from being explicitly taught and it selects its actions on basis of its past experiences (exploitation) and also by new choices (exploration), which is essentially trial and error learning. The aim is to discover a *policy* for selecting actions that minimizes some measure of a long-term cost when correct decisions are rewarded and incorrect ones are penalized [104–115].

Advantages of Neural Computing Neural networks are increasingly applied in many types of classification problems in divergent fields such as crop classification, predicting weather patterns to cell abnormalities in mammograms. The prominent reason for such applications is that to use neural networks for classification, no knowledge of underlying reasons are required as mentioned in previous sections. For instance, if wind speed, humidity, angle of the sun rays are determined to be prominent factors affecting the temperature of the air, then a neural network can be easily setup to predict air temperature using recorded data for training. The mathematical relationship among the above four factors and how they affect the temperature is not required to be established to predict the temperature. If the system is able to predict the temperature fairly accurately (if no other factors are affecting the temperature significantly), then temperature can be predicted by simply measuring other data without ever establishing their relationship. This example clearly illustrates how useful neural networks are and the reasons for their wide use in classification in diverse fields. It also highlights the potential knowledge vacuum in using neural network. The key limitation of using neural networks is its inability to explain the model and their relationship in a useful way.

Learning Algorithms Training a neural network model refers to adjusting weights to memorize the experience using training inputs to predict the training outputs. These weights are modified using different models that minimize a cost function to optimize the weights quickly. This approach usually uses gradient descent type algorithm. Expectation-maximization, non-parametric methods and particle swarm optimization [116], Evolutionary methods [117], gene expression programming [118], simulated annealing [119] are some commonly used methods for training neural networks.

As was mentioned in the previous section, neural networks are increasingly found in classification approaches due to their ability to be used as an arbitrary function approximation mechanism that 'learns' from observed data. However, a thorough understanding of the underlying theory is required to develop a robust neural network that is not overly complex and optimize its weights or training rapidly. The utility of artificial neural network models lies in the fact that they can be used to infer a function from observations. This is particularly useful in applications where the complexity of the data or task makes the design of such a function by hand impractical. Some of their usage is highlighted below:

- Function approximation, or regression analysis, including time series prediction, fitness approximation and modeling.
- Classification, including pattern and sequence recognition, novelty detection and sequential decision making.
- Data processing, including filtering, clustering, blind source separation and compression.
- Robotics, including directing manipulators, prosthesis.
- Control, including Computer numerical control

Application areas of neural networks include system identification and control (vehicle control, process control, natural resources management), quantum chemistry [120], game-playing and decision making (backgammon, chess, poker), pattern recognition (radar systems, face identification, object recognition), sequence recognition (gesture, speech, handwritten text recognition), medical diagnosis, financial applications (automated trading systems), data mining (or knowledge discovery in databases), visualization and e-mail spam filtering.

Artificial neural networks have also been used to diagnose several cancers. A NN-based hybrid lung cancer detection system named HLND improves the accuracy of diagnosis and the speed of lung cancer radiology [121]. These networks have also been used to diagnose prostate cancer. The diagnoses can be used to make specific models taken from a large group of patients compared to information of one given patient. The models do not depend on assumptions about correlations of different variables. Colorectal cancer has also been predicted using the neural networks and are known to predict the outcome for a patient with colorectal cancer with a higher accuracy than many clinical methods [122].

Some of the important criteria that should be exercised in developing a robust neural network classification system are:

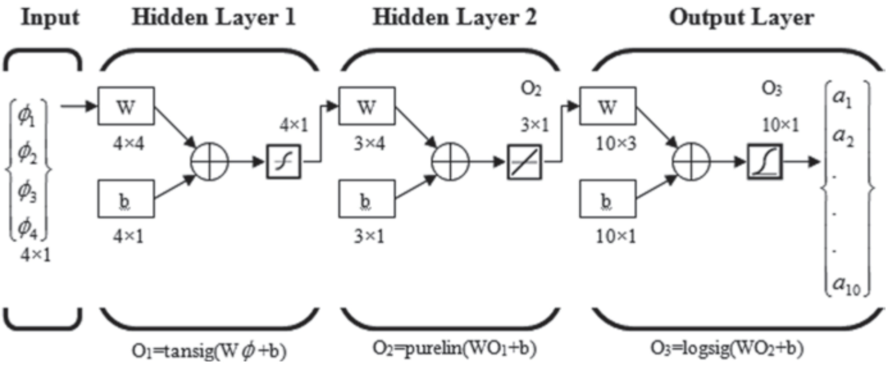


Fig. 5.24 Structure of the neural network classifier (implementation perspective)

1. Selecting an appropriate model: The application and the data (number of input parameters and output parameters) will determine this. Selecting complex models will take extremely long time to train and may never be optimized.
2. Learning algorithm: There are numerous trade-offs between learning algorithms. Almost any algorithm will work well with the *correct* hyper parameters for training on a particular fixed data set. However selecting and tuning an algorithm for training on unseen data requires a significant amount of experimentation.
3. System robustness: If the model, cost function and learning algorithm are selected appropriately the resulting neural network can be extremely robust.

Neural Network Classification for Hand Gesture Recognition In 2005, Premaratne et al. proposed a neural network based hand gesture recognition approach where invariant moments of binarized gesture images were used with extremely high accuracy for limited set of gestures [123]. The proposed neural network is a *backpropagation* network as shown in Fig. 5.24. In this particular network, the input vectors (the sample set of hand gestures) and the target vectors (the corresponding commands set) were used to train the network until it can approximate a function between the input and the output. There were a number of neuron layers between the input and the output. Each layer contains a number of nodes whose properties were characterized by a weight matrix W, a bias vector b and a transfer function f. Some popular transfer functions or activation functions used for testing were Log-Sigmoid transfer function, Tan-Sigmoid transfer function and linear transfer function [94].

In this design, there were only three layers due to the limited number of hand gestures to be classified. More complex network could possibly be designed and implemented, but it was not practical and necessary for this particular project. The first layer, i.e. the hidden layer 1, contained four nodes and it used the *tansig* transfer function. The second layer, i.e. the hidden layer 2, contained three nodes which used the *purelin* transfer function. This implied that the network could have had more hidden layers, but it was not necessary and practical for this particular project. The

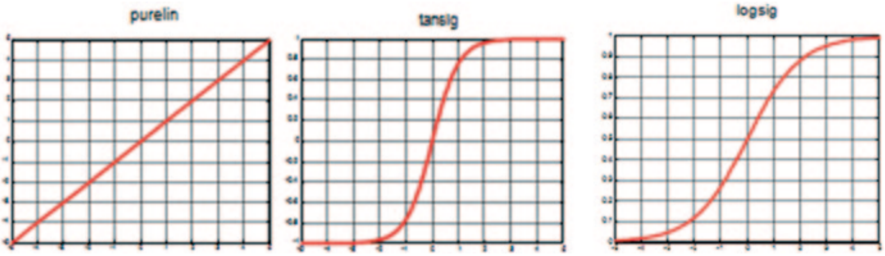


Fig. 5.25 Neural network activation (transfer) functions

last layer, i.e. the output layer, consisted of ten nodes and it used the *logsig* transfer function. Each of the above activation functions encompassed different properties and is illustrated in Fig. 5.25.

Purelin activation function is given by $f(x) = x$, and the tansig is given by $f(x) = \frac{2}{1+e^{-2x}} - 1$ whereas logsig is expressed as $f(x) = \frac{1}{1+e^{-x}}$.

The input was the set of moment invariant values derived from the sample set of hand gestures. The output was the target set of commands corresponding to each gesture. Some recent work on hand gesture recognition using neural networks can be found here [124–127].

5.2.2.5 HMM for Hand Gesture Classification

Hidden Markov Model (HMM) is a collection of finite states connected by transitions [128]. Each state is characterized by two sets of probabilities: a transition probability and either a discrete output probability distribution or continuous output probability density function which given the state, defines the condition probability of emitting each output symbol from a finite alphabet or a continuous random vector [129].

The use of Hidden Markov Model for hand gesture recognition stems from HMM's ability to successfully modeling speech recognition. The similarities between speech and gesture suggest that techniques effective for one problem may be effective for the other as well [129]. Hand gestures, like spoken language, vary according to the person, location or background, time, gender, age and social factors. There is common ground between speech and gestures which are known to have syntactic rules [128]. Since gesture is an expressive motion, it is natural to describe such a motion through sequential model [130].

HMMs can be used for object classification when given several models, it is possible to determine the model which will produce a given sequence of observations with the highest probability [131]. Thus, if for each class there is a model with the states, transitions and probabilities set appropriately, the Viterbi algorithm can be used to calculate the model that most probably resulted in the sequence of observations [132]. It is easy to see how this can be extended to a weak temporal classification system where a single model is constructed for each class, and given an unlabelled instance; the probability that that sequence of output symbols was generated by each HMM is calculated. The model with the highest probability is identifies as the predicted class [131].

However, it is not a trivial task to select the correct model and appropriate states and transitions. Knowledge in the problem domain and experience would play a significant role in succeeding with the application. Trial and error methods would identify the number of states and transitions when developing an efficient system.

The concept behind the use of HMM for gesture recognition is to use multi-dimensional HMM representing the parameters obtained from the training data using the chosen model. The trained models represent the most likely human performance and are used to evaluate the new incoming gestures. In using a HMM to classify in coming dynamic hand gestures, the first important step would be to develop a set of hand gestures which are distinct in terms of feature extraction. The HMM will use any type of parameters of features in classifying however, the success of the approach depends on the best choice of parameters or features. Therefore it is essential that the selection of the parameters of features are unique with properties such as rotation, scale, translation and illumination invariance as was outlined in the previous chapter on Feature Extraction. Such feature extraction step naturally to be preceded by preprocessing as already discussed in a previous chapter for the best classification results. Then these highly efficient features can be used to describe each gesture in terms of a multi-dimensional HMM. In other words, each gesture would have its own HMM with N distinct hidden states and r dimensional M distinct observable symbols. An HMM is characterized by a transition matrix A and r discrete output distribution matrices. Figure 5.26 shows a hand gesture recognition systems based on HMM.

Training the HMMs through training data is paramount to adjust the parameters in such as way that they can maximize the likelihood of known gestures for the given training data. The Baum-Welch [133] algorithm is commonly used to iteratively re-estimate model parameters to achieve the local maximum. Once the training is completed, any incoming dynamic gesture can be classified. The Forward-Backward algorithm or the Viterbi algorithm [132] can be used to classify both static and dynamic hand gestures.

Yang et al. [134] describes a dynamic gesture recognition method using state based spotting algorithm to split continuous gestures. Features using in the system include hand position, velocity, size and shape. They developed a data aligning algorithm to align feature vector sequences for training. Then a HMM is trained alone for each gesture with promising accurate recognition.

Chen et al. [135] used Fourier Descriptors (FD) (of dimension 10) to represent hand gesture boundary. When normalized, FDs are invariant to gesture size, rotation and translation. In their attempt to use HMM to recognize gestures, they used three parameters: the initial state probability vector, the state transition probability matrix and the observable symbol probability matrix. Since a dynamic gesture has more transitions, the likelihood of dynamic gesture being classified

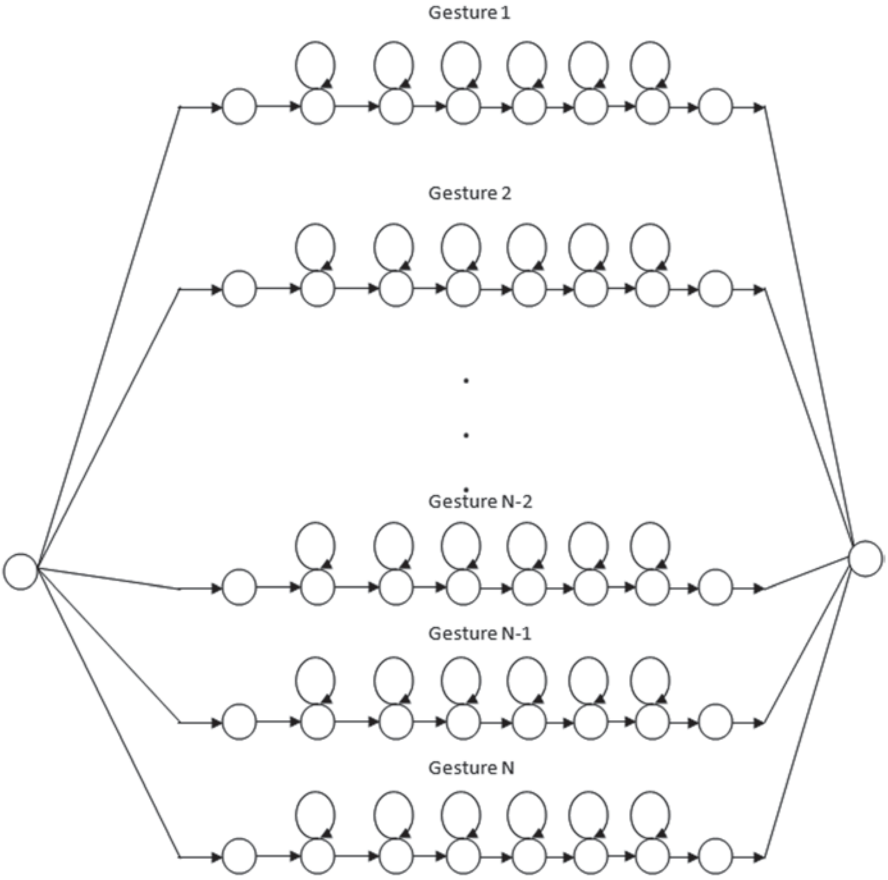


Fig. 5.26 A Typical HMM classification for dynamic hand gesture recognition, reproduced [129]

right is far higher than a single static gesture. To model various gesture expressions, they trained different HMMs to model different hand gestures. In order to remove redundancy in any gesture sequences, they applied vector quantization as preprocessing. They used a *M*-level VQ (referred to as a codebook with size M) to partition all k-dimensional training feature vectors into M clusters, whose centroid is the k-dimensional vector. They designed a technique where there would be a quantization error between the VQ and training data vector for each feature. The error decreases as the size of the codebook increases due to the availability of more training data.

Liu and Lovell [136] described a technique where they estimated a rotation angle of the hand direction after skin segmented hand region, and described a hand 'blob' using a ellipse model. They applied *k*-means Vector Quantization approach to determine the features using a HMM. In their quest to classify hand gestures using HMM, they made number of important conclusions:

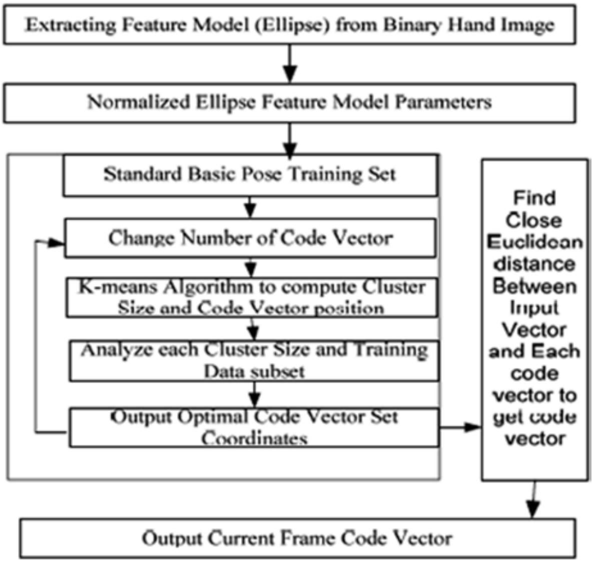

Extracting Feature Model (Ellipse) from Binary Hand Image
Normalized Ellipse Feature Model Parameters
Standard Basic Pose Training Set
Change Number of Code Vector
K-means Algorithm to compute Cluster Size and Code Vector position
Analyze each Cluster Size and Training Data subset
Output Optimal Code Vector Set Coordinates
Find Close Euclidean distance Between Input Vector and Each code vector to get code vector
Output Current Frame Code Vector

Fig. 5.27 Vector quantization method used by Liu and Lovell [136]

1. Baum Welch is the traditional algorithm for HMM learning, while Viterbi path accounting is the new method (VPC). VPC performed steadily in comparison with Baum Welch, and is demonstrated to be a reliable method [136].
2. Model Structure variation. Structure doesn't greatly affect the recognition ratio. For Baum Welch, Left-Right is a slightly better than full connection, while in Viterbi path accounting, Full connection has a slightly higher correct ratio [136].
3. The number of states doesn't affect the correct ratio significantly. The effect on Baum Welch is greater than on Viterbi path accounting [136].

They reported an accuracy rate of 90 % for 26 gestures. Their algorithm is denoted in Fig. 5.27 as a flowchart.

Classification completes the final link in associating hand gestures with their pre-assigned meanings in the context of hand gesture recognition. Yet, for any classification to be effective, feature extraction should provide the fundamental traits of any gesture by variety of users with many variables including, skin color, lighting condition, rotation and scale variations. This chapter discussed many techniques commonly used in linear and nonlinear classification while establishing there mathematical relationships providing in-depth insight into the problem of gesture recognition.

References

1. Chase, L.D.: Euclidean Distance (2008) http://www.warnercnr.colostate.edu/~ldchase/Melinda's%20Final%20writeup.doc. Accessed Oct. 12, 2013
2. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications. Quantitative biology quantitative methods (2012)

3. Cantrell, C.D.: Modern Mathematical Methods for Physicists and Engineers. Cambridge University Press (2000)
4. Abello, J.M., Pardalos, P.M., Resende, M.G.C.: Handbook of Massive Data Sets. Springer (2002)
5. Tax, D.M.J., Duin, R., De Ridder, D.: Classification, Parameter Estimation and State Estimation: An Engineering Approach Using MATLAB. John Wiley and Sons (2004)
6. Mahalanobis, P.C.: On the generalised distance in statistics. Proc. Natl. Inst. Sci. India 2(1), 49–55 (1936)
7. Li, T., Zhu, S., Ogihara, M.: Using discriminant analysis for multi-class classification: an experimental investigation. Knowl. Inf. Syst. 10(4): 453–472 (2013)
8. Li, T., Zhu, S., Ogihara, M.: Using discriminant analysis for multi-class classification: an experimental investigation. Knowl. Inf. Syst. 10(4), 453–472 (2006)
9. Su, Y., Shan, S., Cao, B., Chen, X., Gao, W.: Multiple fisher classifiers combination for face recognition based on grouping AdaBoosted Gabor features. Proceedings of the British Machine Vision Conference (2005)
10. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Ann. Eugen. 7(2), 179–188 (1936)
11. McLachlan, G.J.: Discriminant Analysis and Statistical Pattern Recognition. Wiley Interscience (2004)
12. Hendricks, D.: Analyzing Quantitative Data: An Introduction for Social Researchers, pp. 288–289 (2011)
13. Martinez, A.M., Kak, A.C.: PCA versus LDA. IEEE Trans. Pattern Anal. Mach. Intell. 23(2), 228–233 (2001)
14. Gupta, S., Jaafar, J., Ahmad, W.F.W.: Static hand gesture recognition using local gabor filter, international symposium on robotics and intelligent Sensors 2012. Procedia Eng. 41, 827–832 (2012)
15. Khan, A., Farooq, H.: Principal component analysis-linear discriminant analysis feature extractor for pattern recognition. IJCSI Int. J. Comput. Sci. 8(6), p276 (2011)
16. Suhas, S., Ajay, K., Khanale, P.: Face recognition using principal component analysis and linear discriminant analysis on holistic approach in facial images database. IOSR J. Eng. 2, 15–23 (2012)
17. Balakrishnama, S., Ganapathiraju, A.: Linear discriminant analysis- a brief tutorial. Institute for Signal and Information Processing. http://www.music.mcgill.ca/~ich/classes/mumt611/classifiers/lda_theory.pdf. Accessed Sept. 12, 2013
18. Khanale, P.B.: Face recognition against variation in pose and background. IEEE International Conference on Electro/Information Technology (2011)
19. Satonkar S.S., Kurhe A.B., Khanale P.B.: Face recognition methods & its applications. J Emerg. Technol. Appl. Eng., Technol. Sci. (IJ-ETA-ETS) 4(2), 294–297 (2011)
20. Johnson, R.A., Wichern, D.W.: Applied Multivariate Statistical Analysis. Prentice Hall (1998)
21. Yu, H., Yang, J.: A direct LDA algorithm for high-dimensional data—with application to face recognition. Pattern Recognit. 34(10), 2067–2069 (2001)
22. Rosenblatt, F.: The perceptron–a perceiving and recognizing automaton. Report 85–460-1, Cornell Aeronautical Laboratory (1957)
23. Liou, D.-R., Liou, J.-W., Liou, C.-Y.: Learning Behaviors of Perceptron. iConcept Press (2013)
24. Mahesh P.: Multiclass approaches for support vector machine based land cover classification. CORR 2008 (2008)
25. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag, New York (1995)
26. Chen, Y.T., Tseng, K.T.: Multiple-angle hand gesture recognition by fusing SVM classifiers, IEEE conference on Automation Science and Engineering, Scottsdale, AZ, USA, pp. 527–530 (2007)
27. Huang, D-Y., Hu, W-C., Chang, S-H.: Vision-based hand gesture recognition using PCA + Gabor filters and SVM. Fifth international conference on intelligent information hiding and multimedia signal processing (2009)

28. Bonansea, L.: 3D Hand gesture recognition using a ZCam and an SVM-SMO classifier. Graduate Theses and Dissertations Graduate College, Iowa State University (2009)
29. Liu, Y., Gan, Z., Sun, Y.: Static Hand Gesture Recognition and its Application based on Support Vector Machines, pp. 517–521 (2008)
30. Chen, Y-T., Tseng, K-T.: Multiple-angle Hand Gesture Recognition by Fusing SVM Classifiers, pp. 527–530 (2007)
31. Bonansea, L.: Demonstration video of the 3D gesture recognition system using Zcam and SVM. http://www.youtube.com/watch?v=VsM0a_3I1_Q (2009)
32. Ye, J., Yao, H., Jiang, F.: Based on HMM and SVM multilayer architecture classifier for Chinese sign language recognition with large vocabulary, pp. 377–380 (2004)
33. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features (1998) http://www.cs.cornell.edu/people/tj/publications/joachims_98a.pdf. Accessed April 18, 2013
34. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J Mach. Learn. Res. 2001, 45–66 (2001)
35. Sassano, M.: Virtual Examples for Text Classification with Support Vector Machines. Fujitsu Laboratories Ltd (2003)
36. Basu, A., Watters, C., Shepherd, M.: Support vector machines for text categorization. Proceedings of the 36th Hawaii International Conference on System Sciences (2003)
37. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML-00), pp. 287–295 (1998)
38. Cortes, C., Vapnik, V. N.: Support-Vector Networks, Machine Learning, p. 20 (1995)
39. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B. P.: Support Vector Machines. Numerical Recipes: The Art of Scientific Computing, 3rd edn. Cambridge University Press, New York (2007)
40. Aizerman, M.A., Braverman, E.M., Rozonoer, L.I.: Theoretical foundations of the potential function method in pattern recognition learning. Autom. Remote Control. 25, 821–837 (1964)
41. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Haussler, D (ed.) 5th Annual ACM Workshop on COLT, pp. 144–152 (1992)
42. Meyer, D., Leisch, F., Hornik, K.: The support vector machine under test, Neurocomputing 55(1–2), 169–186 (2003)
43. Hsu, C-W, Chang, C-C., Lin, C-J.: A Practical Guide to Support Vector Classification (Technical report). Department of Computer Science and Information Engineering, National Taiwan University (2003)
44. Duan, K-B., Keerthi, S. S.: Which is the best multiclass SVM method? An empirical study. Proceedings of the Sixth International Workshop on Multiple Classifier Systems. Lecture Notes in Computer Science vol. 3541, p. 278 (2005)
45. Hsu, C-W., Lin, C-J.: A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks (2002)
46. Platt, J., Cristiamanini, N., Shawe-Taylor, J.: Large margin DAGs for multiclass classification. In: Solla, S.A., Leen, T.K., Müller, K-R. (eds.) Advances in Neural Information Processing Systems, pp. 547–553. MIT Press (2000)
47. Dietterich, T.G., Bakiri, G.B.: Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. 2(2), 263–286 (1995)
48. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass Kernel-based vector machines. J. Mach. Learn. Res. 2, 265–292 (2001)
49. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines. Computing Science and Statistics, p. 33 (2001)
50. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. J. Am. Stat. Assoc. 99(465), 67–81 (2004)

51. Joachims, T.: Transductive inference for text classification using support vector machines. Proceedings of the 1999 International Conference on Machine Learning (ICML 1999), pp. 200–209 (1999)
52. Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A.J., Vapnik, V.N.: Support vector regression machines. In Advances in Neural Information Processing Systems 9, NIPS 1996, pp. 155–161 (1997)
53. Suykens, J.A.K., Vandewalle, J.P.L.: Least squares support vector machine classifiers. Neural Process. Lett. 9(3), 293–300 (1999)
54. Ferris, M. C. and Munson, T. S.: Interior-point methods for massive support vector machines. SIAM J Optim. 13(3), 783–804 (2002)
55. Knerr, S., Personnaz, L., Dreyfus, G.: Single-layer learning revisited: a stepwise procedure for building and training neural network. Neurocomputing: Algorithms, Architectures and Applications, NATO ASI. Springer-Verlag, Berlin (1990)
56. Hsu, C.-W., Lin, C.-J.: A comparison of methods for multi-class support vector machines, IEEE Trans. Neural Netw. 13, 415–425 (2002)
57. JAMES, G.: Majority vote classifiers: Theory and Applications. Ph. D. Thesis, Department of Statistics, Stanford University, Stanford, CA (1998)
58. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines Tech. Rep. 1043, Department of Statistics, University of Wisconsin, Madison, (2001)
59. Schölkopf, B., Smola, A. J.: Learning with Kernels—Support Vector Machines, Regularization, Optimization and Beyond. The MIT Press, Cambridge (2002)
60. Weston, J., Watkins, C.: Multi-class Support Vector Machines. Royal Holloway, University of London, U. K., Technical Report CSD-TR-98–04 (1998)
61. Piyush, R.: Kernel Methods and Nonlinear Classification CS5350/6350: Machine Learning (2011)
62. Berwick, R.: An Idiot's guide to Support vector machines (SVMs) http://www.web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf. Accessed Oct. 15, 2013
63. Scribe, M.I.J., Thibaux, R.: The kernel trick. Advanced Topics in Learning & Decision Making (2004) http://www.cs.berkeley.edu/~jordan/courses/281B-spring04/lectures/lec3.pdf. Accessed April 18, 2013
64. Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T., Müller, K.-R.: Engineering support vector machine kernels that recognize translation initiation sites. BioInformatics 16(9), 799–807 (2000)
65. Blankertz, B., Curio, G., Müller, K-R.: Classifying single trial EEG: towards brain computer interfacing. In: Diettrich, T.G., Becker, S., Ghahramani, Z., (eds.) Advances in Neural Information Processing Systems, vol. 14, pp. 157–164 (2002)
66. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
67. Moody, J., Darken, C.: Fast learning in networks of locally-tuned processing units. Neural Comput. 1(2), 281–294 (1998)
68. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comp. Syst. Sci. 55(1), 119–139 (1997)
69. Rätsch, G., Mika, S., Schölkopf, B., Müller, K.-R.: Constructing boosting algorithms from SVMs: an application to one-class classification. IEEE Patt. Anal. Mach. Intell. (IEEE PAMI) 24(9), 1184–1199 (2002)
70. Girosi, F., Jones, M., Poggio, T.: Priors, stabilizers and basis functions: from regularization to radial, tensor and additive splines. Technical Report A.I. Memo No. 1430, Massachusetts Institute of Technology (1993)
71. Smola, A.J., Schölkopf, B., Müller, K.-R.: The connection between regularization operators and support vector kernels. Neural Netw. 11, 637–649 (1998)
72. Girosi, F.: An equivalence between sparse approximation and support vector machines. Neural Comput. 10, 1455–1480 (1998)
73. Haussler, D.: Convolution kernels on discrete structures. Technical Report UCSC-CRL-99–10, UC Santa Cruz (1999)

74. Watkins, C.: Dynamic alignment kernels. In: Smola, A.J., Bartlett, P.L., Schölkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers, pp. 39–50 (2000)
75. Schölkopf, B.: The kernel trick for distances. In: Leen, T.K., Diettrich, T.G., Tresp, V. (eds.) Advances in Neural Information Processing Systems 13. MIT Press (2001)
76. Osuna, E., Freund, R., Girosi, F.: Training support vector machines: an application to face detection. In Proceedings CVPR'97 (1997)
77. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Schölkopf, B., Burges, C.J.C., Smola, A.J. (eds) Advances in Kernel Methods—Support Vector Learning, pp. 185–208 (1999)
78. Ralaivola, L., d'Alché Buc, F.: Incremental support vector machine learning: a local approach. Lect. Notes Comput. Sci. 2130, 322–329 (2001)
79. Schölkopf, B., Burges, C.J.C., Vapnik, V.N.: Extracting support data for a given task. In: Fayyad, U.M., Uthurusamy, R. (eds.) Proceedings, First International Conference on Knowledge Discovery & Data Mining (1995)
80. Schölkopf, B., Smola, A., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. Neural Comput. 12, 1207–1245 (2000)
81. Schölkopf, B., Smola, A.J: Learning with Kernels. MIT Press, Cambridge (2002)
82. Schölkopf, B., Smola, A.J., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. 10, 1299–1319 (1998)
83. Simard, P.Y., LeCun, Y.A., Denker, J.S., Victorri, B.: Transformation invariance in pattern recognition—tangent distance and tangent propagation. In: Orr, G., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade, LNCS 1524, pp. 239–274 (1998)
84. Afrin, M.H.RH.: Hand gesture recognition using multiclass support vector machine. Int. J. Comput. Appl. 74(1), 39–43 (2013)
85. Chen, Y-T., Tseng, K-T.: Multiple-angle hand gesture recognition by fusing SVM classifiers. Proceedings of the 3rd Annual IEEE Conference on Automation Science and Engineering, pp. 527–530 (2007)
86. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. Am. Stat. 46(3), 175–185 (1992)
87. Coomans, D., Massart, D.L.: Alternative k-nearest neighbor rules in supervised pattern recognition: Part 1. K-Nearest neighbor classification by using alternative voting rules. Anal. Chimi. Acta 136, 15–27 (1982)
88. Bremner D., Demaine E., Erickson J., Iacono J., Langerman S., Morin P., Toussaint G.: Output-sensitive algorithms for computing nearest-neighbor decision boundaries. Discret. Comput. Geom. 33(4), 593–604 (2005)
89. http://www.analyticbridge.com/forum/topics/clustering-idea-for-very-large-datasets
90. Pujan, Z., M¨uller, T., Foster, M.E., Knoll, A.: A Na¨ıve Bayes Classifier with Distance Weighting for Hand-Gesture Recognition. CSICC 2008, CCIS 6, 308–315 (2008)
91. Pujan, Z., M¨uller, T., Foster, M.E., Knoll, A.: Using a Naïve Bayes classifier based on k-nearest neighbors with distance weighting for static hand-gesture recognition in a human-robot dialog system. Adv. Comput. Sci. Eng. 6(1–8), 308–315 (2008)
92. Vafadar, M., Behrad, A.: Human hand gesture recognition using Spatio-temporal volumes for human-computer Interaction. International Symposium on Telecommunications, pp. 713–718 (2008)
93. Kollorz, E., Penne, J., Hornegger, J., Barke, A.: Gesture recognition with a time-of-flight camera. Int. J. Intell. Syst. Technol. Appl. 5–¾, 334–343 (2008)
94. http://www.byclb.com/TR/Tutorials/neural_networks/ch6_1.htm
95. Haykin, S.: Neural Network—a Comprehensive Foundation; a Computational Approach to Learning and Machine Intelligence, Macmillan (1994)
96. Zurada, J.M.: Introduction to Artificial Neural Networks System. Jaico Publishing House (1992)
97. Freeman: Artificial Neural Network Algorithm. Applications and Programming, Comp and Neural Systems Series, Addison-Wesley Pub (Sd) (1990)

98. Kulkarni, A.: Artificial Neural Network for Image Understanding. Reinhold, New York (1994)
99. Anderson, J.: An Introduction to Neural Network. A Bradford Book (1995)
100. Ranjan, A.: A New Approach for Blind Source Separation of Convolutive Sources (2008)
101. Carpenter, G.A., Grossberg, S.: The ART of adaptive pattern recognition by a self-organizing neural network. Computer **21**, 77–88 (1998)
102. Hinton, G., Sejnowski, T.J. (ed.): Unsupervised Learning: Foundations of Neural Computation, MIT Press (1999)
103. Duda, R.O., Hart, P.E., Stork, D.G.: Unsupervised Learning and Clustering, Chapter 10 in Pattern classification, 2nd edn. Wiley, New York, p. 571 (2001)
104. Ghahramani, Z.: Unsupervised Learning (2004) http://mlg.eng.cam.ac.uk/zoubin/papers/ul.pdf. Accessed April 18, 2013
105. Williams, R.J.: A class of gradient-estimating algorithms for reinforcement learning in neural networks. Proceedings of the IEEE First International Conference on Neural Networks (1987)
106. Sutton, R.S.: Learning to Predict by the Method of Temporal Differences. Machine Learning (Springer), vol. 3, pp. 9–44 (1998).
107. Bradtke, S.J., Barto, A.G.: Learning to Predict by the Method of Temporal Differences. Machine Learning (Springer), vol. 22, pp. 33–57 (1996)
108. Bertsekas, D.P., Tsitsiklis, D.: Neuro-Dynamic Programming. Athena Scientific, Nashua (1996)
109. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: a survey. J. Artif. Intell. Res. 4, 237–285 (1996)
110. Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement learning for humanoid robotics. IEEE-RAS International Conference on Humanoid Robots (2003)
111. Powell, W.: Approximate Dynamic Programming: Solving the Curses Of Dimensionality. Wiley-Interscience (2007)
112. Auer, P., Jaksch, T., Ortner, R.: Near-optimal regret bounds for reinforcement learning. J. Mach. Learn. Res. 11, 1563–1600 (2010)
113. Szita, I., Szepesvari, C.: Model-based Reinforcement Learning with Nearly Tight Exploration Complexity Bounds. ICML 2010, pp. 1031–1038 (2008)
114. Bertsekas, D.P.: Approximate Dynamic Programming. Dynamic Programming and Optimal Control II, 3rd edn. (2010)
115. Busoniu, L., Babuska, R., De Schutter, B., Ernst, D.: Reinforcement Learning and Dynamic Programming using Function Approximators. Taylor & Francis CRC Press (2010)
116. Tokic, M., Palm, G.: Value-difference based exploration: adaptive control between Epsilon-Greedy and Softmax. KI 2011: advances in Artificial intelligence. Lecture Notes in Computer Science, vol. 7006, pp. 335–346 (2011)
117. Wu, J., Chen, E., Wang, H., Shen, Y., Huang, T., Zeng, Z.: A Novel nonparametric regression ensemble for rainfall forecasting using particle swarm optimization technique coupled with artificial neural network. 6th International Symposium on Neural Networks (2009)
118. De Rigo, D., Castelletti, A., Rizzoli, A.E., Soncini-Sessa, R., Weber, E.: A selective improvement technique for fastening Neuro-Dynamic programming in water resources network management. In: Zítek, P. (ed.) Proceedings of the 16th IFAC World Congress (2005)
119. Ferreira, C.: Designing neural networks using gene expression programming. In: Abraham, A., de Baets, B., Köppen, M., Nickolay, B. (eds.) Applied Soft Computing Technologies: The Challenge of Complexity, pp. 517–536 (2006)
120. Da, Y., Xiurun, G., Villmann, T.: An improved PSO-based ANN with simulated annealing technique. New Aspects in Neurocomputing: 11th European Symposium on Artificial Neural Networks. Elsevier (2005)
121. Balabin, R.M., Lomakina, E.I.: Neural network approach to quantum-chemistry data: accurate prediction of density functional theory energies. J. Chem. Phys. 131(7), 1–8 (2009)

122. Ganesan, N. Venkatesh, K., Rama M.A.: Application of neural networks in diagnosing cancer disease using demographic data. International Journal of Computer Applications 1(26), 76–85 (2010)
123. Bottaci, L. Drew, P.J., Hartley, J.E., Hadfield, M.B., Farouk, R., Lee, P.W., Macintyre, I.M., Duthie, G.S., Monson, J.R.: Artificial Neural Networks Applied to Outcome Prediction for Colorectal Cancer Patients in Separate Institutions. The Lancet 350(9076), 469–472 (1997)
124. Premaratne, P., Safaei, F., Nguyen, Q.: Moment invariant based control system using hand gestures: book intelligent computing in signal processing and pattern recognition. Book Series Lecture Notes in Control and Information Sciences, vol. 345, pp. 322–333 (2006)
125. Gutta, S., Imam, I.F., Wechsler, H.: Hand gesture recognition using ensembles of radial basis functions (RBF) networks and decision trees. Int. J. Patt. Recognit. Artif. Intell. 11(6) (1997)
126. Murthy, G.R.S., Jadon, R.S.: Hand gesture recognition using neural networks. IEEE 2nd International Advance Computing Conference Artificial Intelligence, pp. 134–138 (2010)
127. Hasan, H., Abdul-Kareem, S.: Static hand gesture recognition using neural networks. Artificial Intelligence Review 41:147–181 (2012)
128. Zheng, X., Koenig, S.: A Project on Gesture Recognition with Neural Networks for Introduction to Artificial intelligence Classes (2010)
129. Min, B-W., Yoon, H-S., Soh, J., Yang, Y-M., Ejima, T.: Hand gesture recognition using hidden Markov models. 1997 IEEE International Conference on Systems, Man, and Cybernetics on Computational Cybernetics and Simulation, vol. 5, pp. 4232–4235 (1997)
130. Yang, L., Xu, Y., Chen, C.S.: Human action learning via hidden Markov model. IEEE Trans. Syst. Man. Cybern. 27(1), 34–44 (1997)
131. Yang, L., Xu, Y.: Hidden Markov model for gesture recognition.Thesis, The Robotics Institute Carnegie Mellon University (1994)
132. Kadous, W.: Machine learning is a subfield of artificial intelligence. PhD Thesis, University of New South Wales (2002)
133. Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Trans. Inf. Theory 13(2), 260–269 (1967)
134. Rabiner, L.: First Hand: The Hidden Markov Model. IEEE Global History Network. http://www.ieeeghn.org/wiki/index.php/First-Hand:The_Hidden_Markov_Model. Accessed Aug. 24, 2013
135. Yang, Z., Li, Y., Chen, W., Zheng, Y.: Dynamic hand gesture recognition using hidden Markov models. 7th International Conference on Computer Science & Education (ICCSE), pp. 360–365 (2012)
136. Chen, F.S., Fu, C.M., Huang, C.L.: Hand gesture recognition using a real-time tracking method and hidden Markov models. Image Vision Comput. 21(8), 745–758 (2003)

Chapter 6
Sign Languages of the World

Sign languages have been there since the start of the humanity and would have been the first means of communication among the primitive humans. Before people communicated with a vocabulary and using sounds, it is fair to assume that they communicated with various gestures using hand, face, mouth and body movements. However, today, the sign language is predominantly associated with disabilities from congenital to accidents. Most of the users are either hearing impaired or mute. There is also a subgroup of whom are children of such hearing impaired people whose senses are not affected but do use sign language because of the community needs in which they live.

Since the world population has surpassed 7 billion and the widespread use of technology, more and more work is done to advance the age-old sign languages which have developed from multitude of cultural backgrounds. Due to the advancement of societies and its welfare systems, more and more resources are allocated for the benefit of the disabled and their educational needs. Sign languages have seen the formal recognition of their use and are now making inroads to the new fields of usage such as in medicine so that disabled people will also benefit from the modern developments in the society.

Today the focus has shifted again from the mundane use of sign language to the more advanced human machine interaction. This would in effect advance the interactions that disabled people would have with technology as well as make sign languages easily understandable by ordinary users. The technology can also pave way for automatic translation to other languages in other parts of the world making a silent communication revolution for the disability. Yet, the challenges are enormous and the different approaches taken by researchers around the world have shed light on difficulties ahead as well as the progress made so far. This chapter would discuss various approaches the researchers including the author have attempted to decipher from Australian sign language as well as a broad account of research carried out on American Sign Language. The readers are encouraged to visit research performed by others on other sign languages in the world and some references are listed for their benefit.

Sign languages, like the spoken languages, emerge and evolve naturally within hearing-impaired and mute communities. Within each country or region, wherever

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_6,

such communities exist, sign languages develop, independently from the spoken language of the region. Each sign language has its own grammar and rules, with a common property that they are all visually perceived.

6.1 Australian Sign Language-Historical Perspective of Auslan

Sign languages in any part of the world have their inception in natural need to communicate especially for children with their parents. Auslan has its roots in British Sign Language (BSL) and Irish Sign Language (ISL) and is known to have been used in early 1800s in large residential schools for the deaf in Australia [1]. ISL was brought to Australia by Irish nuns who established the first school for Catholic deaf children in 1875 [1, 2]. The Irish one-handed alphabet and a tradition of Irish-based signs was kept alive well into the middle of the twentieth century through private Catholic schools that used many Irish signs and one-handed finger spelling, while public schools used Auslan signs (originally BSL) and two-handed finger spelling. Separate education systems aside, the two communities mixed freely, with British based signing being undoubtedly the dominant linguistic influence [1, 2].

Schools dedicated for deaf children were first established in Australia in the mid-nineteenth century. The Sydney school for the deaf was established in 1860 by Thomas Pattison. He had his education from the Edinburgh Deaf and Dumb Institution. At the same time another deaf person, Frederick Rose founded the Melbourne School for Deaf who had is formal education at Old Kent Road School in London [1, 2].

Even though Auslan is an offshoot of both BSL and ISL, it has developed some distinct gestures since its inception in Australia in the nineteenth century. New signs developed in the Australian deaf community, particularly in the residential schools for deaf children because signers may have had little contact with deaf communities in other parts of the country [1, 2].

6.1.1 Finger Spelling

A number of signs in modern Auslan clearly have their origins in ISL (and through ISL to the French and European signing tradition). Also as a consequence of this mixing and exposure to Irish-based signing, the one-handed alphabet (including its modern American form) does not feel quite so ‘alien’ to Auslan signers as one might expect. Initialized signs based on one-handed finger spelling have been and continue to be accepted by this linguistic community, even though finger spelling is regularly produced using the two-handed alphabet [2]. Auslan similar to every other sign language can represent numerals for signers. Numbers 0–9 can be made with simple postures as shown in Fig. 6.1 while larger numbers need dynamic gestures as shown in Fig. 6.1 (right).

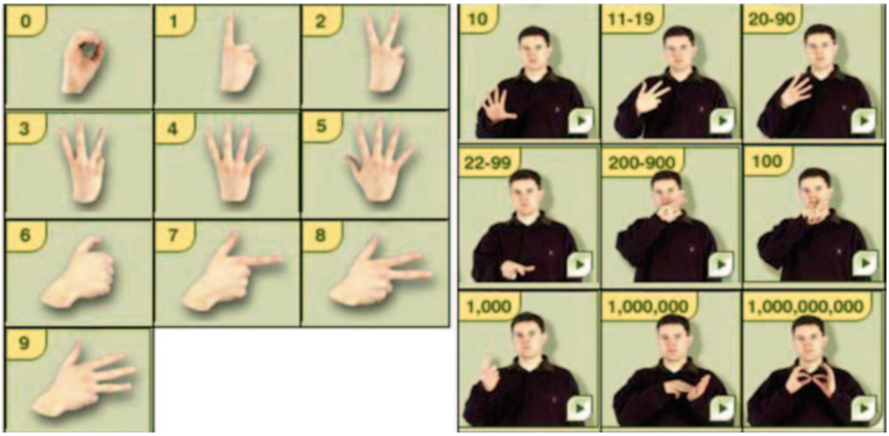


Fig. 6.1 Numerals of zero to nine and how numerals above 10 are calculated using a gesture sequence. (Courtesy of [1])

6.1.2 *Auslan Evolution*

Today Auslan seems to be undergoing a period of rapid change. The enormous expansion of sign language interpreter services, especially in the area of secondary and tertiary education and in the delivery of governmental, legal and medical services, has put great demands on the language by both interpreters and deaf people themselves. These developments have produced three main responses: (i) attempts to standardize usage, (ii) the development of new signs to meet new needs, (iii) the borrowing of signs from other sign languages, particularly from American Sign Language (ASL) [1, 3].

Most members of the deaf community have a personal and political preference for drawing on the internal resources of Auslan to expand and develop its vocabulary. However, some Auslan signers either do not object to ASL borrowings (sometimes they do not even realize that some signs are borrowed from ASL) or are actually willing borrowers (new signs are adopted because they are sometimes seen as more prestigious). The fact that ASL signers also have English as the language of the wider community, as do Auslan signers, may encourage this process. Many borrowed ASL signs are technical and deal with vocabulary used in education and in written English. Nevertheless, many Auslan signers reject any attempts to introduce borrowed ASL signs when a perfectly good and adequate Auslan sign already exists [1, 3].

Figure 6.2 depicts a chart containing Auslan alphabet. This alphabet is used when difficult or confusing words are pronounced including pronouncing names of people. Similar to number between 0 and 9, the alphabet can be thought of as a collection of hand postures opposed to hand gestures where the machine interpretation would be simple. Figure 6.3 shows two words being described by Auslan. These images merely show the key frames of words ‘stomach’ and ‘hands’ omitting many

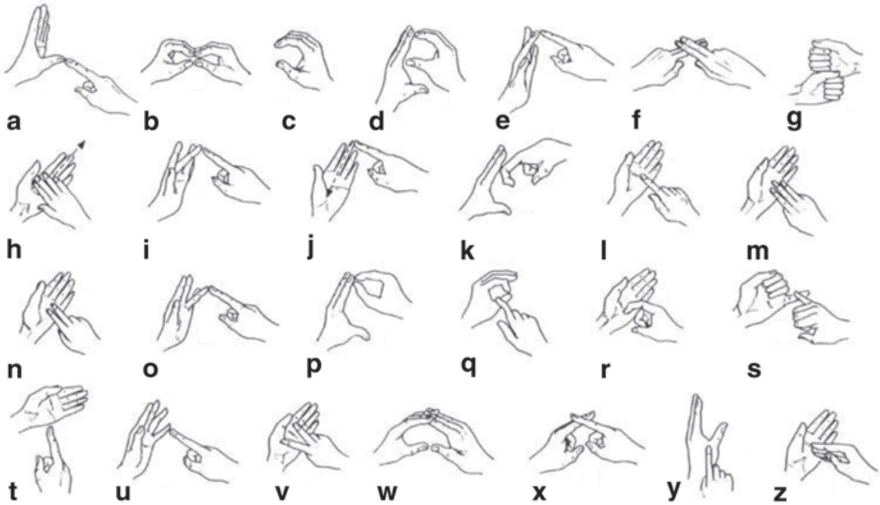


Fig. 6.2 Finger spelling alphabet of Auslan. [4]

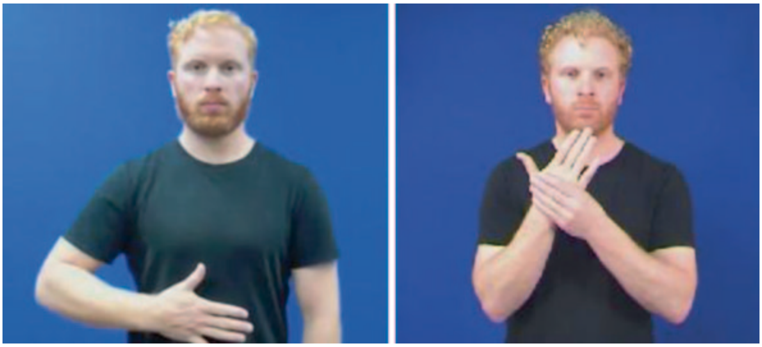


Fig. 6.3 Most of the words in Auslan are hand gestures opposed to postures. Key image frames of words 'stomach' (*left*) and 'hands' (*right*)

frames of a video sequence. This suggest that most of the Auslan words are hand gestures and need dynamic gesture recognition which is in its infancy today despite a continuous research activity of more than two decades.

6.2 American Sign Language

American Sign Language (ASL) is the predominant sign language of hearing-impaired and mute communities in the United States and English-speaking parts of Canada. Christian missionary activities over a century have spread dialects of

ASL and ASL-based creoles to many countries around the world, including much of West Africa and parts of Southeast Asia [5–7]. ASL is also widely learned as a second language, serving as a unifying language to bring many sign languages in the world together and is closely related to French Sign Language (FSL). It has been proposed that ASL is a Creole language, although ASL shows features atypical of Creole languages, such as agglutinative morphology [8].

Similar to the origin of Auslan, ASL originated in the early nineteenth century in the American School for the Deaf (ASD) in Hartford, Connecticut. Today it is estimated that there are approximately 250,000–500,000 persons using it including non-disabled children of hearing impaired and mute persons.

ASL is a language completely separate and distinct from English. It contains all the fundamental features of language such as its own rules for pronunciation, word order, and complex grammar. While every language has ways of signaling different functions, such as asking a question rather than making a statement, languages differ in how this is expressed. For example, English speakers ask a question by raising the pitch of their voice whereas an ASL users ask a question by raising their eyebrows, widening their eyes, and tilting their bodies forward [9–27].

Just as with other languages, specific ways of expressing ideas in ASL vary as much as ASL users do. In addition to individual differences in expression, ASL has regional accents and dialects. Just as certain English words are spoken differently in different parts of the country, ASL has regional variations in the rhythm of signing, form, and pronunciation. Ethnicity and age are a few more factors that affect ASL usage and contribute to its variety [12–20].

ASL signs have a number of phonemic components, including movement of the face and torso as well as the hands. ASL is not a form of pantomime, but iconicity does play a larger role in ASL than in spoken languages. Words derived from English are often borrowed through finger-spelling, although ASL grammar is unrelated to that of English [19, 24]. ASL has verbal agreement and aspectual marking, and has a productive system of forming agglutinative classifiers. Many linguists believe ASL to be a subject-verb-object (SVO) language, but there are several alternative proposals to account for ASL word order [24]. Figure 6.4 depicts the ASL alphabet where certain letters such as ‘J’ and ‘Z’ are dynamic gestures.

6.2.1 Historical Development of ASL

ASL today is considered a very structured language. However, prior to the birth in its present form, sign language had been used by various communities in the United States [15]. In the United States and an other parts of the world, hearing families with hearing-impaired or mute children have historically employed hand signs as a way of communicating sophisticated information unlike the gestures used by non-disabled people in spoken conversations [15]. As early as 1541 at first contact by Francisco Vásquez de Coronado, there were reports that the Red Indians had developed a sign language to communicate between tribes of different languages [26].

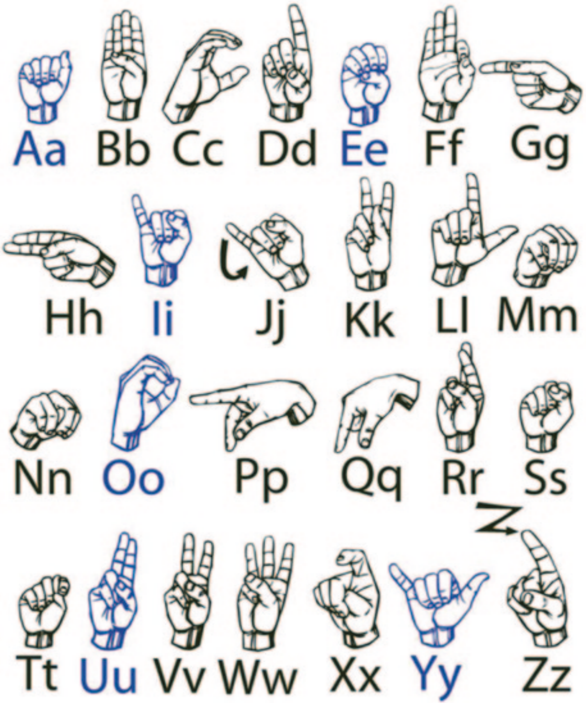


Fig. 6.4 ASL finger spelling alphabet. [28]

Martha's Vineyard Sign Language (MVSL), is known to be the first sign language used by settlers in the USA due to their predisposal to genetic deafness due to intermarriages [15]. MVSL was used even by hearing residents whenever a deaf person was present [15].

American Sign Language has its origins in the American School for the Deaf (ASD), founded in Hartford, Connecticut in 1817 [15]. Originally, it was known as 'The American Asylum' located at Hartford. It was formally known as 'The Education and Instruction of The Deaf and Dumb'; the school was founded by the Yale graduate and divinity student Thomas Hopkins Gallaudet [29, 30]. Gallaudet was known to be inspired by his success in demonstrating the learning abilities of a young deaf girl he helped. He traveled to Europe in order to learn deaf pedagogy from European institutions [30] which he adopted in ASD in the USA in 1817 [30].

Due to the congenital nature of the hearing impaired in Martha' Vineyard, the majority of the students enrolled in the ASD for the first 70 years came from Martha's Vineyard. Since their early exposure to MVSL, they brought that knowledge to the ASD [6]. There were other students who brought their knowledge of home signs to ARD [6]. The first teacher as ASD was Laurent Clerc who taught using French Sign Language (FSL), which itself had developed in the Parisian school for the deaf established in 1755 [15]. The culmination of knowledge from MVSL, home signs and European practice resulted in a new sign language known as ASL [15].

Due to the success of ASD in addressing and formalizing sign language norms, more schools for hearing-impaired sprang up across the USA [15]. These schools contributed to the growth of ASL by using it and spreading it across the USA [15]. Societies such as the National Association of the Deaf and the National Fraternal Society of the Deaf held national conventions that attracted signers from across the country [6]. This all contributed to ASL's wide use over a large geographical area such as the USA and Canada initially and later to the western parts of Africa [6].

Until the early part of 1950, the predominant method in hearing-impaired education was known to be 'oralism' in which acquiring oral language comprehension and production was emphasized over sign languages [14]. Up to that point, Linguists in the USA did not consider sign language to be a genuine *language*, but merely an accessory to convey certain oral communication [14]. The formal acceptance and legitimacy for sign language was achieved due to the untiring work of Stokoe, who arrived from Gallaudet University in France in 1955 [23]. He spearheaded the movement coupled with US civil rights movement of the 1960s, Stokoe argued for manualism, the use of sign language in deaf education [14, 20]. Stokoe noted that sign language shares the important features that oral languages have as a means of communication, and even devised a writing system for ASL [14]. His contribution lead to a revolution in both deaf education and linguistics and also created opportunities for the 'mute' who could now rely on sign language instead of 'oralism' [14]. Figure 6.5 shows the ASL alphabet sometimes signed using two hands.

6.3 Machine Recognition of Sign Language using Computer Vision

The research on hand gesture recognition has two main categories such as static gesture (hand posture) recognition and gesture recognition (dynamic hand movement). The posture recognition refers to an isolated single hand pose or hand configuration and the gesture recognition or dynamic gesture recognition uses a variety of static gestures and hand movements sometimes accompanied by body or facial movements. In a continuous gesture (dynamic), there are many hand poses that changes in communicating a gesture as the gesture used in letter 'z' in ASL as shown in Fig. 6.6. In hand gesture controlled environments, the problem can be considered as a gesture spotting problem, where the task is to differentiate the meaningful gestures of the user from the unrelated ones. In sign language recognition, the continuous recognition problem includes the co-articulation problem. The preceding sign affects the succeeding one, which complicates the recognition task as the transitions between the signs should be explicitly modeled and incorporated to the recognition system. Moreover, language models are used to be able to perform on large-vocabulary databases.

Human Computer Interaction is geared towards seamless human machine integration without the need for LCDs, Keyboards or Gloves. Systems have already been developed to react to limited hand gestures especially in gaming and

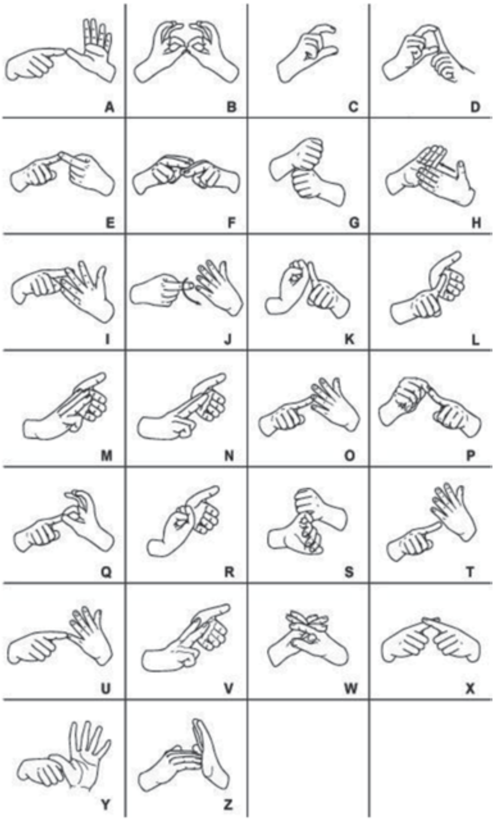


Fig. 6.5 Two handed alphabet

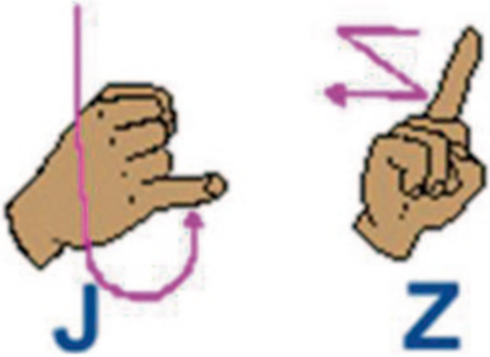


Fig. 6.6 Gesture sequence for letter J and Z

in consumer electronics control. Yet, it is a monumental task in bridging the well-developed sign languages in different parts of the world with a machine to interpret the meaning. One reason is the sheer extent of the vocabulary used in sign language and the sequence of gestures needed to communicate different words and phrases. Auslan the Australian Sign Language is comprised of numbers, finger spelling for

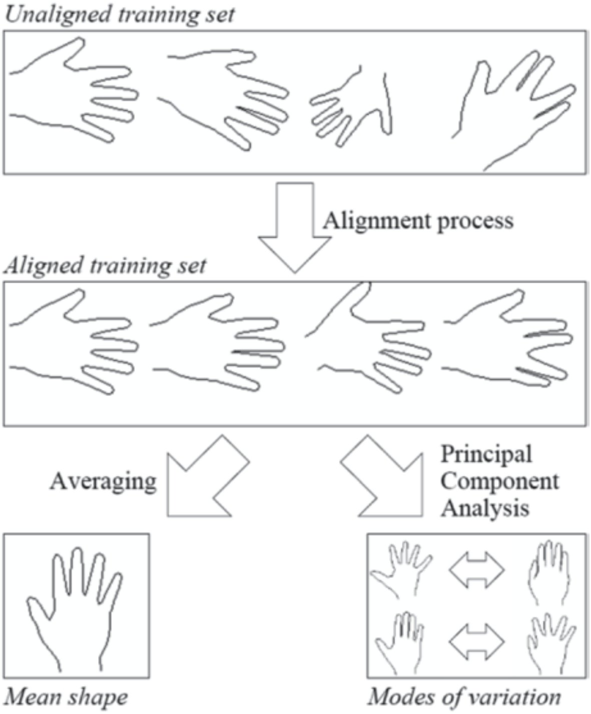
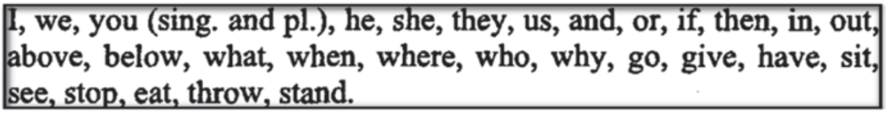


Fig. 6.7 Motion patterns of the sign language system proposed by [38]

words-used-in-common practice and a medical dictionary. There are 7,415 words listed in Auslan website. This research article tries to implement recognition of numerals using a computer using the static hand gesture recognition system developed for consumer electronics control at the University of Wollongong in Australia [31–33]. The experimental results indicated that the numbers, zero to nine can be accurately recognized with occasional errors in few gestures. The system can be further enhanced to include larger numerals using a dynamic gesture recognition system.

One of the first attempts to interpret sign languages using a machine is reported by Kawai and Tamura [34] who developed a system which can recognize 20 Japanese hand gestures. This was the first instance of using image processing techniques to recognize a sign language. They detected the motion of the hands in realtime by comparing a grey scale intensity of two consecutive image frames. They used temporal intensity changes, hand location tracking and classification of each sign using stop positions, simple shapes of trajectory and hand shape at each stop position. Few years prior to this development in 1984, they demonstrated a system that interpreted speech into computer generated hand gestures for machine interaction with deaf and mute [35, 36].

Heap and Samaria used deformable active shape models called 'smart snakes' developed by [37] for hand tracking [38]. Their research clearly shows how their tracking algorithm work but their disclosure on gesture recognition was very much limited. Figure 6.7 shows 'smart snakes' approach for hand gesture recognition.



I, we, you (sing. and pl.), he, she, they, us, and, or, if, then, in, out, above, below, what, when, where, who, why, go, give, have, sit, see, stop, eat, throw, stand.

Fig. 6.8 A subset of eighty words that can represent 2,000 words or phrases in ASL. [45]

In 1995, Starner and Pentland published their research on dynamic gesture recognition and classification based on colored gloves and Hidden Makov Model classifier [39]. Their research highlighted that the finer details (pixel clarity) of gestures were not required and simple coarse outline along with their trajectory obtained using camera tracking color glove was sufficient to interpret gestures. They used 395 simple sentences as training set and separate 99 simple sentences for testing in which recorded a recognition accuracy of 92 % with a limited 40 hand postures.

Grobel and Assan reported that vision based system using HMM was able to recognize 262 isolated hand signs (non sentences) from Netherland sign language with an accuracy of 94 % [40, 41]. They used both hands with cotton gloves. One hand which was the dominant hand making the signs used five colored fingers, separate colors for palm and another for back of the dominant hand whereas the non-dominant hand had a different color glove.

Vogler and Metasas of University of Pennsylvania devised both static gesture and dynamic gesture recognition system for ASL based on HMM and 3D motion analysis [42, 43]. They used a 3D image-based shape and motion tracking of a human's arm using deformable models. The output of these models consists of the three dimensional motion parameters of the subject's arms which can be used for recognizing ASL gestures. They argued that since the arm movements indicated the hand motion, their article did not include the use of hand signals.

Imagawa et al. argued that for accurate sign language recognition, both global and local information should be recognized [44]. Since the hand or arm movement conveys global information such as the hand and arm location and the path, static information such as the configuration of the hand and fingers should be combined with the information conveyed by the hand and arm movement. Their system first interprets the appropriate word from a dictionary using global features and then narrows word down to one by using detected local features. They recorded accuracy around 94 % which was a significant achievement given that they relied on very low resolution images.

Even though, many hand gestures need dynamic tracking for meaningful interpretation, posture estimation results in very discrete recognition as discussed in the previous section of Auslan recognition. Isaacs and Foo developed a similar hand pose recognition system that relied on wavelet decomposition for feature extraction and neural networks for classification [45]. They developed the system to represent around 2,000 sign language sentences and phrases using a set of eighty words. A subset of their eighty words is shown in Fig. 6.8. One of the main limiting factors of their system was to remove the gestures that required facial gestures, since such signs represented a more complex problem.

Isaacs and Foo successfully decomposed all static ASL alphabet images into two set of feature vectors with different dimensions in order to develop more insight into the classification problem. The first approach produced feature vectors by normalizing the energy and entropy measures of the resultant approximation matrix after two-level Daubechies 4 decomposition. Another set of feature vectors were obtained using decomposing each image to its lowest decomposition level of the particular wavelet used. The second set of feature vectors had an added advantage of longer length which represented a higher dimensional space that allowed for more variability when a large ASL vocabulary was incorporated. The ASL recognition system developed in this project had the ability to distinguish the reduced vocabulary set feature vectors from each other and from non-words. For classification purposes, neural networks were used.

The classification system that was used had a two layer feed-forward neural network that recognized the 24 static letters in the American Sign Language (ASL) alphabet using still input images. Both wavelet- based decomposition methods discussed above were used for classification. The first produced an 8-element real-valued feature vector and the second 18-element feature vector. Each set of feature vectors were used to train a feed-forward neural network using Levenberg-Marquardt training. The system was capable of recognizing instances of static ASL finger spelling with 99.9 % accuracy with an SNR as low as 2.

In 2012, Cooper et al. presented one of the most sophisticated sign language recognition system based on Kinect [46]. It consisted of several approaches to sub-unit based Sign Language Recognition (SLR). Their approach deviated from traditional classification of gesture to recognizing large lexicons of signs such as citation-form and dictionary look-up. Sub-unit based SLR uses a two stage recognition system where sign linguistic sub-units are identified followed by combining of sub-units to create a sign level classifier. The conceptual framework of their approach is shown in Fig. 6.9.

The concept of using sub-units for SLR was first reported by Kim and Waldron, whom were among the first adopters. They worked on a limited vocabulary of 13–16 signs using data gloves to get accurate input information [47, 46]. Using the work of Stokoe [48] as a base, and their previous work in telecommunications [49], they noted the need to break signs into their component sub-units for efficiency [46]. They continued this throughout the remainder of their work, where they used phonemic recognition modules for hand shape, orientation, position and movement recognition [50, 46]. They made note of the dependency of position, orientation and motion on one another and removed the motion aspect allowing the other sub-units to compensate (on a small vocabulary, a dynamic representation of position is equivalent to motion) [51, 46].

The sub-unit based approach relies in the early work of Vogler and Metaxas [52] who borrowed heavily from the studies of sign language by Liddell and Johnson [53], splitting signs into motion and pause sections [46]. Their later work [54], used parallel HMMs on both hand shape and motion sub-units, similar to those proposed by the linguist Stokoe [48, 46]. Kadir et al. [55] took this further by combining head, hand and torso positions, as well as hand shape, to create a system based on

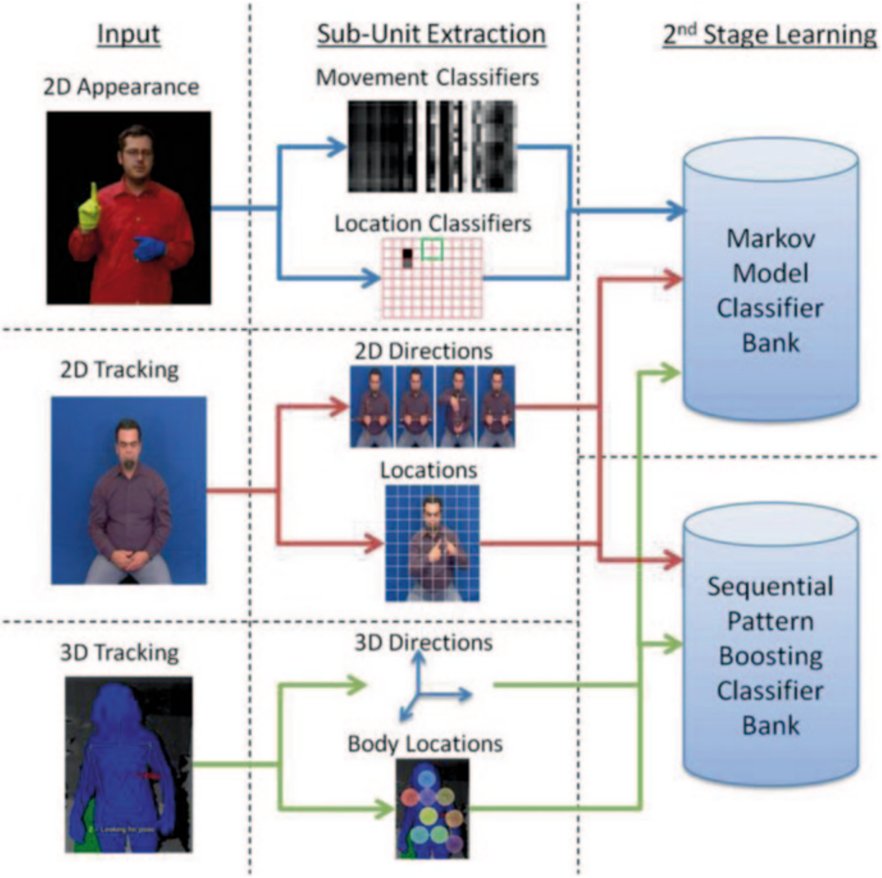
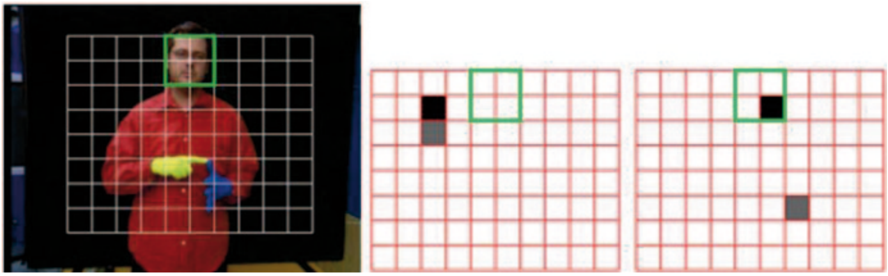


Fig. 6.9 Overview of the 3 types of sub-units extracted and the 2 different sign level classifiers used. (Courtesy of [46])

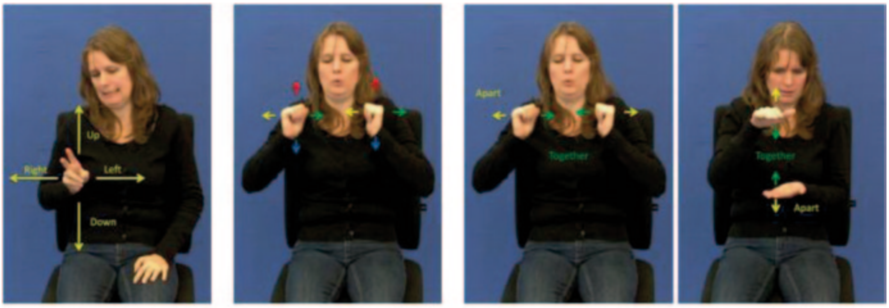
hard coded sub-unit classifiers that could be trained on as little as a single example [46]. Alternative methods looked at data driven approaches to defining sub-units. Yin et al. [56] used an accelerometer glove to gather information about a sign, they then applied discriminative feature extraction and 'similar state tying' algorithms, to decide sub-unit level segmentation of the data [46]. Kong and Ranganath [57] and Han et al. [58] developed automatic segmentation of sign motion into sub-units, using discontinuities in the trajectory and acceleration to indicate where segments begin and end [46]. These were then clustered into a code book of possible exemplar trajectories using either Dynamic Time Warping (DTW) distance measures attempted by Han *et al.* or Principal Component Analysis (PCA) approach by Kong and Ranganath [46].

Appearance based sub-units learns a subset of each type of sub-unit using AdaBoost from hand labeled data. Not all types of sub-units can be detected using the



The grid applied over the signer On Right Shoulder Lower Face/Chin

Fig. 6.10 Grid features for two stage classification. (*left*) shows an example of the grid produced from the face dimensions while (*middle*) and (*right*) show grid features chosen by boosting for two of the 18 Location sub-units. The highlighted box shows the face location and the first and second features chosen, are shown in *black* and *grey* respectively. (Courtesy of [46])



a Single handed **b** Bimanual: Synchronous **c** Bimanual: Together/Apart

Fig. 6.11 Motions detected from tracking. (Courtesy of [46])

same type of classifier. For Location sub-units, there needs to be correlation between where the motion is happening and where the person is; to this end spatial grid features centered on the face of the signer are employed. For Motion sub-units, the salient information indicates the type of motion occurring, often regardless of its position, orientation or size. Salient information is extracted using moment features and Binary Patterns (BPs) and additive classifiers based on their changes over time. This process is shown in Fig. 6.10. Hand-Arrangement sub-units look at where the hands are in relation to each other when both hands are used for signing as shown in Fig. 6.11. This is achieved using the same moment features as for Motion but over a single frame, as there is no temporal information required. All of these sub-unit level classifiers are learnt using AdaBoost [59]. The features used in this section require segmentation of the hands and knowledge of where the face is. The Viola Jones face detector [60] is used to locate the face. Skin segmentation could be used to segment the hands, but since sub-unit labels are required this work uses the data set from the work of Kadir et al. [55] for which there is an in-house set of sub-unit labels for a portion of the data. Cooper et al. created this data set using a gloved signer and as such a color segmentation algorithm is used in place of skin segmentation.

Cooper et al.'s method can be described as the most comprehensive effort to date for gesture recognition based on vision which combined many aspects of successful approaches over two decades. It described a combined gesture classification system using facial and bi-hand gestures known as linguistic sub-units were attempted by the best available vision technology available with Microsoft Kinect. They presented three types of sub-units for consideration; those learnt from appearance data as well as those inferred from both 2D and 3D tracking data. These sub-units were then combined using a sign level classifier which used a Markov Models to encode the temporal changes between sub-units and a Sequential Pattern Boosting to apply discriminative feature selection at the same time as encoding temporal information. This resulted in a more robust to noise and performed well in signer independent tests, improving results from the 54 % achieved by the Markov Chains to 76 %.

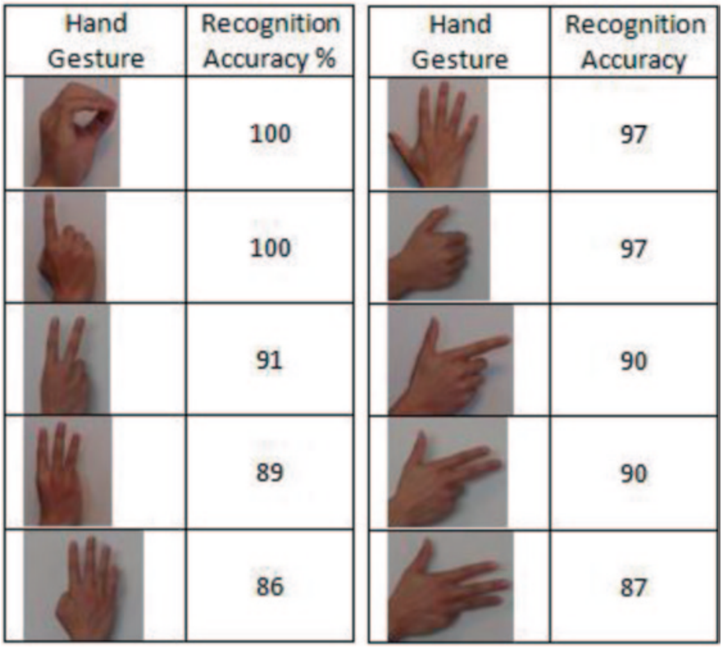
6.3.1 Machine Recognition of Auslan

Currently the Auslan website list 7,415 words in its data base. Looking at this figure, it would be almost impossible for a machine to see the subtle variation from one word to the other [1–4]. With our experience in Hand Gesture Recognition systems for over 7 years, interpreting sign language can be meaningfully attempted only for numerals [61]. Applying hand gesture recognition techniques developed for consumer electronics control will be sufficient to accurately decipher hand gestures used for numerals between 0 and 9. Tens, hundreds, thousands may still need a dynamic hand gesture recognition developed by the authors [62–65]. Overall, this research will discuss attempt to recognize numerals from 0 to 9 and will ascertain their accuracy.

The moment invariants algorithm has been recognized as one of the most effective methods to extract descriptive feature for object recognition applications and has been widely applied in classification of subjects such as aircrafts, ships, and ground targets [66, 67]. Essentially, the algorithm derives a number of self-characteristic properties from a binary image of an object. These properties are invariant to rotation, scale and translation.

The above four features have achieved accurate gesture recognition for a limited number of gestures in our previous research [62–65]. These moment invariants can be used to train a Support Vector Machine (SVM) approach or Neural Network for Classification.

We developed a database containing 10 images for each gesture and used features extracted using moment invariants for classification. Only the first four moments were used as features similar to our approach in static gesture recognition system [4–6]. The classification was carried out using Neural Network similar to our previous approach for gesture classification as shown in Fig. 6.12 which shows



Hand Gesture	Recognition Accuracy %	Hand Gesture	Recognition Accuracy
	100		97
	100		97
	91		90
	89		90
	86		87

Fig. 6.12 Classification scores for numerals from zero to nine

the classification scores for the 10 numerals. The results indicate that this approach does have some strength in accurately interpreting 6 gestures and 4 gestures might misclassify 5–15 % occasionally. Looking at these figures, we conclude that it would almost be an impossible task to decipher sign language using machine learning. Current approaches into gesture recognition and face recognition will never be suitable for accurate interpretation of sign language as it already has 7,415 different words and phrases.

Why does many vision based classification systems have low accuracy? There are few obvious reasons behind the failure of many vision based classification system to interpret sign languages over the years [40]. Some of the more prominent ones can be listed as follows:

1. Signs vary in time and space. Even if a sign is repeated by the same user, slight changes of speed and position of hands will occur.
2. Lack of depth information complicates the problem
3. Some fingers are occluded behind hand or arms
4. The enormity of the number of gestures available and their similarity to many others and the users inability to sign them as in (1) above makes the machines unreliable with the present feature extraction and classification approaches.

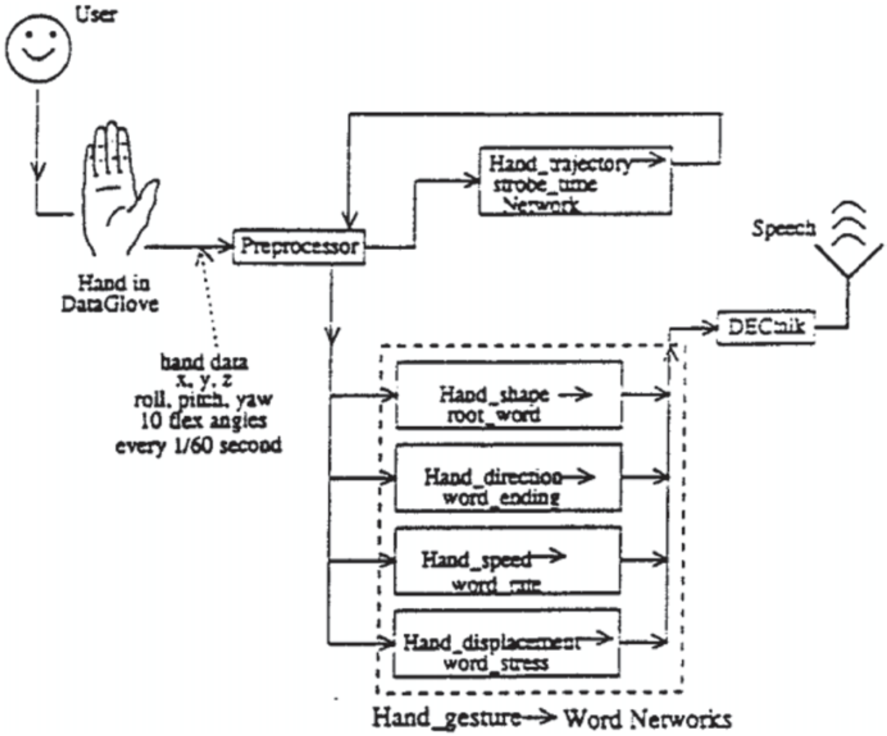


Fig. 6.13 GloveTalk System. [68]

6.4 Glove Based Sign Language Recognition

Even with advancements of computer vision, glove based sign language recognition offers the widest vocabulary and the best possible recognition accuracy. This is not alarming given that all vision based systems suffer from occlusion to misclassification due to vast number of signs which are so subtle to understand even for highly trained humans. Glove based systems that could measure every bend of fingers and the location of hand in 3D can accurately detect both global features and local features that distinguish words. However, no recent such system has been reported with very high accuracy as researchers are more focused on vision based systems due to advancements in depth cameras and the flexibility and naturalness offered by vision based systems.

One of the first more advances system to convert gesture to speech was demonstrated by Fels and Hinton [68] who used a VPL Data-Glove in 1992 to convert hand gestures to speech via DECtalk speech synthesizer. Their Glove-Talk vocabulary consisted of 66 root words, each with up to six different endings. The total size of the vocabulary was 203 words. Their GloveTalk system is shown in Fig. 6.13. Most of these hand shapes represent ASL alphabet. They also utilized orientation

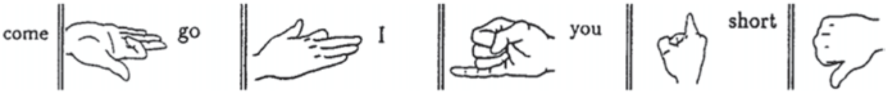


Fig. 6.14 Example of GloveTalk Language. [68]

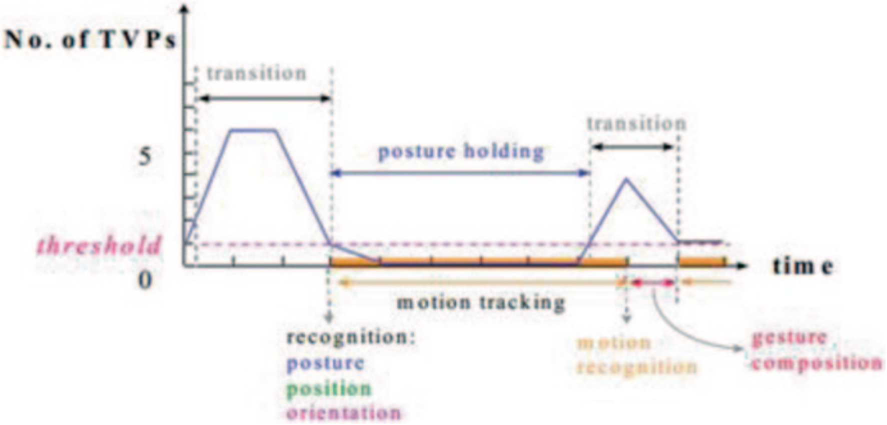


Fig. 6.15 The number of TVPs of a gesture input stream varies along the time axis. [69]

differences in the hand shapes for semantically opposite words such as 'come' and 'go' which have a 180 degree orientation difference as shown in Fig. 6.14. Here 'come' is performed with palm up and 'go' with palm down. Various ending of words were formed by different directions of the hand movement.

They demonstrated that the system was capable of fairly rapid intelligible speech with only 1 % incorrect spoken words and about 5 % of attempts resulting in no words being spoken due to failure to detect the gesture or failure to accurately identify the root word. The system classification was based on neural network classifier with five feedforward neural networks with one hidden layer. The system was trained with backpropagation. Many limitations in the data glove halted further progress in accuracy or the expansion of the vocabulary.

Liang and Ouhyoung also used the DataGlove to develop a Taiwanese sign language recognition based on HMM and integrated statistical approach used in computational linguistics [69]. They utilized specific cues used in Taiwanese Sign Language in order to develop the system. There are 51 fundamental postures in Taiwanese Sign Language [69]. Most gestures mainly contain only one posture, for example, 'I', 'you', 'who', etc., while gestures with multiple postures are also used, such as, 'originally', 'father', 'mother', 'thank' and 'good-bye'.

To determine end points in a sequence of gesture input, discontinuities are detected for segmentation. The discontinuity detection is done by time-varying parameter (TVP) detection as shown in Fig. 6.15. Whenever the number of TVPs of

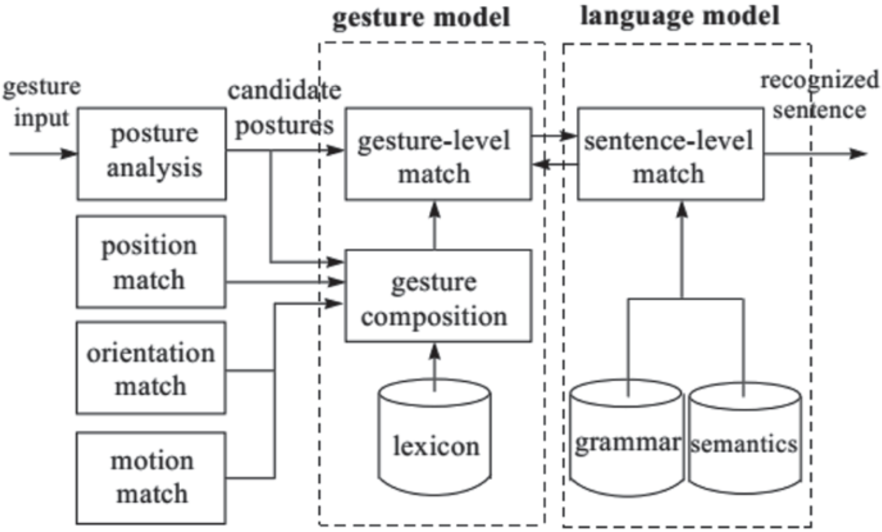


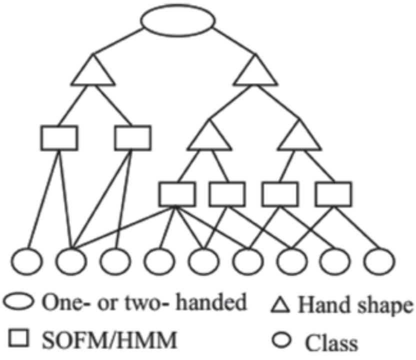
Fig. 6.16 Block diagram of posture-based continuous gesture recognition. (Courtesy of [69])

the hand flexion begins to reduce to below a threshold, the motion of posturing is thought to be quasi-stationary, and its corresponding frame of data is attempted to be recognized. A filter is used to tolerate the jittering of sensed data. They argued that a gesture input stream can be likened to repeated patterns of the following states: transition and posture holding as shown if Fig. 6.15. On detecting the beginning of posture holding, the system extracted features, including position, orientation, and posture, while tracking the motion trajectory. The motion trajectory was observed until the beginning of the next transition. This allowed the system to use all four parameters le to perform a higher level gesture match, which is depicted in Fig. 6.16.

This system intended to recognize large set of vocabularies in a sign language by recognizing constructive postures and context information. They had a 250 word vocabulary. Their system could classify 51 static gestures in 6 orientations. They reported that a user dependent system classifying in real-time with an accuracy of 80 %.

One of the major difficulties in accurate recognition of hand gestures is due to the enormity of the sign language vocabulary. Many feature extraction based methods rely on searching for matches with these large vocabulary databases for query matches. Fang et al. developed a hierarchical decision tree based method that drastically reduced this searching time increasing the efficiency of the classification system [70]. This method simplifies the complexity of gestures using ‘divide and conquer’ approach. A dynamic gesture is a sequence of many postures that is issued in a temporal fashion. Once few initial postures are identified, many hand gestures can be ruled out as not probable as every gesture has basic starting pos-

Fig. 6.17 The hierarchical decision tree for sign language recognition. [70]



tures. Hand shapes are one of the primitives of sign language and reflect the information of hand configuration. They are known to very stable and can be used to distinguish most signs. In the Chinese sign language dictionary, there are 75 basic hand shapes extracted by the sign language expert [70]. The orientation of the hand can be described in terms of two orthogonal directions—the facing of the palm, and the direction to which the hand is pointing. If we consider only six possible directions (up, down, left, right, towards the signer, away from the signer), then there are 15 different orientations used in Chinese sign language (CSL). The position of the hand is usually partitioned in terms of the signer's hand relative to the defined three parts of his body: head, chest and below chest. In each part, the position can be further subdivided into body's left, right and middle. In total, there are 12 positions defined in CSL according to the hand with respect to the body part [70]. Motion differs from the other features in that it is inherently temporal in nature. It is difficult to enumerate the complete range of possible categories used within CSL, as many signs involve unique tracing motions which indicate the shape of an object. Fang et al. defined 13 most commonly used motions for simplicity.

Their hierarchical classification was further helped by certain unique characteristics of Chinese sign language. In the Chinese sign language dictionary, one-handed signs are always performed by right hand except for one sign "luo ma ni ya" by left hand [70]. The difference between one-handed sign and two-handed sign is whether signer's left hand performs the action. In the one-handed sign performance, signer's left hand usually puts on the left knee and remains motionless. However, in the two-handed sign, left hand may either stay a fixed posture or perform a movement trajectory. The position and orientation information of left hand plays a dominant part in determining one-or two- handed signs.

As was described in Chap. 5 under classification, HMM can be effectively used for classifying sequences as shown in Fig. 6.17. Fang et al. reported that their system used a sign language vocabulary of 5,113 with both single hand and two-handed gestures with a accuracy of almost 95 %. The approach also provided 11 fold increase in speed and can be considered as a very robust system for its large vocabulary.

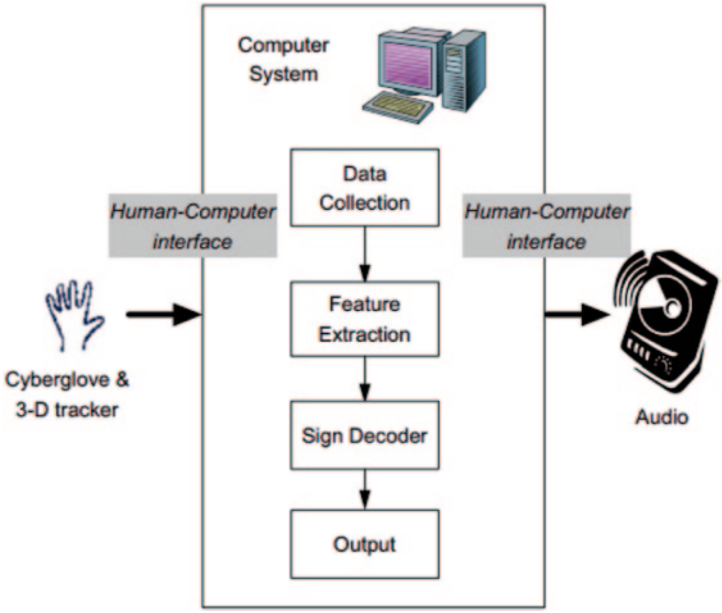


Fig. 6.18 The ASL to English conversion system. (Courtesy of [71])

In 2011, Oz and Leu developed an American Sign Language (ASL) recognition system based on Cyberglove™ sensor glove and artificial neural networks (ANNs) to translate ASL words into English [71]. They used the Cyberglove™ in conjunction with Flock of Birds® 3-D motion tracker to extract gesture features. The data related to finger joint angles obtained from strain gauges in the sensory glove defined the hand shape, while the data from the tracker described the trajectory of hand movements. The data from these devices were processed by a velocity network with noise reduction and feature extraction followed by a word recognition network. Global and local feature extracted for each ASL word were used as feature vectors in a neural network classifier for recognition.

Figure 6.18 shows the overall structure of the proposed system which consists of sensory glove and the Flock of Birds motion tracker as shown in Fig. 6.19. The data stream from these devices is received and segmented by a data collection program. The gesture features extracted from the raw data are then sent to a decoder of the recognition system. The final outcome of the system is to produce voice of the recognized ASL words with a speech synthesizer.

The system relies on identifying the words correctly in terms of their durations of signing from the measured signals. In this case, distinguishing words requires additional calculations due to noise and missed movements. A filter is used for noise reduction, and a velocity network is used to determine whether the sign is an ASL word or not. Figure 6.20 shows an example of the hand velocity profile during a typical signing movement [71].

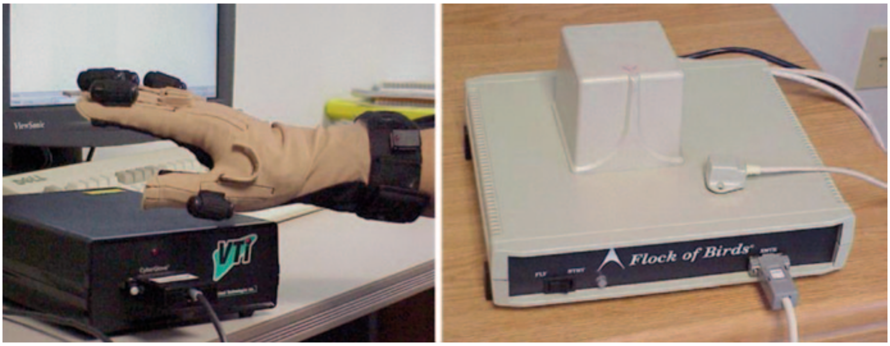


Fig. 6.19 Cyberglove™ and Flock of Birds® 3-D motion tracker. (Courtesy of [71])

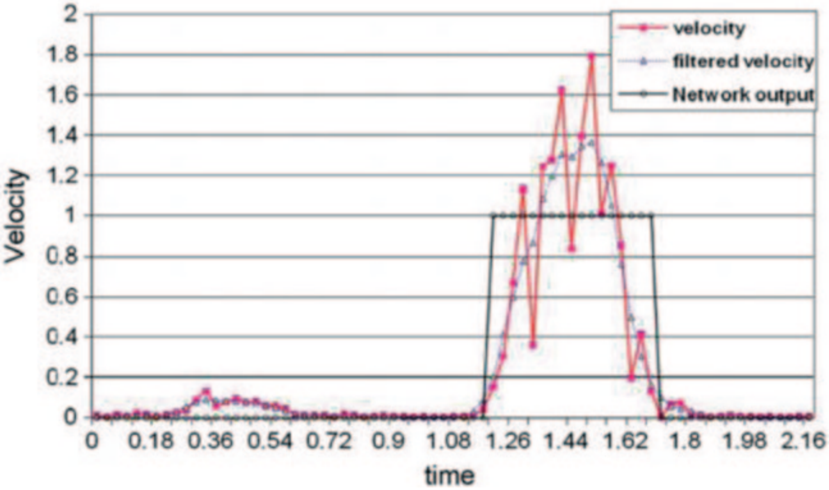


Fig. 6.20 Velocity network performance for sign *BELL*. [71]

During this movement, the hand velocity can increase or decrease momentarily due to momentary starting or stopping of the hand. Hence, the use of a threshold value of velocity might not give a good solution for classification of hand movements. The ideal output for the velocity graph shown in Fig. 6.20 should be 1 from the time the sudden change in velocity is first seen until the time the velocity graph shows a series of low velocities. Hence, the hand velocity is filtered and then input to the velocity neural network. The velocity neural network used in their approach was a three-stage network with two input neurons, ten hidden layer neurons, and two output neurons. The target vectors were created manually by observing the velocity graphs for different, randomly-selected signs. Twenty training samples of different signs were found to be Sufficient to achieve the desired accuracy. The network was trained using the Levenberg–Marquardt algorithm. A hyperbolic tangent sigmoid function was used in both layers.

Their goal was to continuously recognize ASL signs using the glove and developed system in real time. They trained the ANN model for 50 ASL words with a different number of samples for every word and the classification results achieved 90 % accuracy which demonstrated that their used successfully for isolated word recognition. They also concluded that some gestures in ASL required that both the right and left hands be manipulated simultaneously by applying the proposed model but with two data gloves and more motion trackers.

Today, ASL and many sign languages around the world see continuously being interpreted using either vision (with and without markers) or glove based systems as new technology develops. This trend will continue until it results in a highly reliable system in which, the mute and deaf would feel more natural to express their feelings like their able counterparts.

References

1. http://www.auslan.org.au/about/history. Accessed Aug. 20, 2013
2. Johnston, T.: Signs of Australia: A New Dictionary of Auslan. North Rocks Press, NSW (1998)
3. Johnston, T., Schembri, A.: Australian Sign Language: An Introduction To Sign Language Linguistics. Cambridge University Press (2007)
4. http://www.deafcando.com.au/Services/SignLanguagecourses/AboutAuslan.aspx. Accessed Aug. 20, 2013
5. Nyst, V.: Sign Languages in West Africa. Sign Languages, pp. 405–432. Cambridge University Press (2010)
6. Padden, C.: Sign Language Geography. Mathur, Gaurav; Napoli, Donna, Deaf Around The World, pp. 19–37. Oxford University Press, New York (2010)
7. Hurlbut, H.: A Preliminary Survey of the Signed Languages of Malaysia. Cross-Linguistic Perspectives in Sign Language Research: Selected papers from TISLR, pp. 31–46. Signum Verlag, Hamburg (2003)
8. Kegl, J., Kouwenberg, S., Singler, J.: The Case of Signed Languages in the Context of Pidgin and Creole Studies. The Handbook of Pidgin and Creole Studies. Blackwell Publishing (2008)
9. http://www.nidcd.nih.gov/health/hearing/pages/asl.aspx. Accessed Aug. 20, 2013
10. Petitto, L.A.: On the autonomy of language and gesture: Evidence from the acquisition of personal pronouns in American sign language. Cognition **27**(1), 1–52 (1987)
11. Lillo-Martin, D.: Two kinds of null arguments in American sign language. Nat. Lang. Linguist. Theory **4**(4), 415 (1986)
12. Valli, C.: Linguistics of American Sign Language: An Introduction, pp. 85–86. Clerc Books, Washington D.C. (2005)
13. Neidle, C.: The Syntax of American Sign Language: Functional Categories and Hierarchical Structures, p. 59. Cambridge, The MIT Press (2000)
14. Armstrong, D., Karchmer, M., William C.: Stokoe and the study of signed languages. In Armstrong, D., Karchmer, M., Van Cleve, J. (eds.) The Study of Signed Languages, pp. 11–14. Gallaudet University (2002)
15. Bahan, B.: Non-Manual Realization of Agreement in American Sign Language. Boston University (1996)

16. Bailey, C., Dolby, K.: The Canadian Dictionary of ASL. The University of Alberta Press, Edmonton (2002)
17. Bishop, M., Hicks, S.: Orange eyes: Bimodal bilingualism in hearing adults from deaf families. Sign Language Studies (Gallaudet University Press) **5**(2), 188–230 (2005)
18. Collins, S.: Adverbial morphemes in tactile American sign language. Union Institute & University (2004)
19. Costello, E.: American sign language dictionary. Random House (2008)
20. Stokoe, W.C.: Sign language structure: An outline of the visual communication systems of the American deaf, studies in linguistics: Occasional papers (No. 8). Buffalo: Department of Anthropology and Linguistics, University of Buffalo, (1960)
21. Charlotte, B.-S., Cokely, D.: American Sign Language: A student text units 10–18. Washington, DC: Gallaudet University Press. (1991) [1981]
22. Nakamura, K.: About ASL. Deaf resource library (2008)
23. Supalla, S.J., Cripps, J.H.: ASL gloss as an intermediary writing system (2011) http://www.towson.edu/asld/documents/SupallaCripps_FNL_000.pdf. Accessed Sept. 30, 2013
24. Stokoe, W.C., Dorothy C.C., Croneberg, C.G.: A Dictionary of American Sign Languages On Linguistic Principles. Gallaudet College Press, Washington D.C. (1965)
25. Lane, H., Pillard, R., French, M.: Origins of the American Deaf-World. Sign Language Studies (Gallaudet University Press) **1**(1), 17–44 (2000)
26. Lucas, C., Bayley, R., Valli, C.: What's Your Sign for Pizza?: An Introduction to Variation in American Sign Language. Gallaudet University Press, Washington (2003)
27. Mitchell, R., Young, T., Bachleda, B., Karchmer, M.: How many people use ASL in the United States?: Why estimates need updating. Sign Language Studies (Gallaudet University) (2006)
28. http://en.wikipedia.org/wiki/Fingerspelling. Accessed Oct. 11, 2013
29. A Brief History of ASD. American School for the Deaf. http://www.asd-1817.org/page.cfm?p=429. Accessed Oct. 19, 2013
30. A Brief History of The American Asylum, at Hartford, For The Education and Instruction of the Deaf And Dumb http://www.disabilitymuseum.org/dhm/lib/detail.html?id=1371&page=all. Accessed Oct. 15, 2013
31. Premaratne, P., Nguyen, Q.: Consumer electronics control system based on hand gesture moment invariants. IET Computer Vis. **1**(1), 35–41 (2007)
32. Premaratne, P., Ajaz, S., Premaratne, M.: Hand gesture tracking and recognition system using Lucas-Kanade algorithm for control of consumer electronics. Neurocomputing J. (2012)
33. Premaratne, P., Ajaz, S., Premaratne, M.: Hand gesture tracking and recognition system for control of consumer electronics. Springer Lect. Notes Artif. Intel. (LNAI) **6839**, 588–593 (2011)
34. Kawai, H., Tamura, S.: Recognition of sign language motion images. Pattern Recogn. **21**(4), 343–353 (1988)
35. Kawai, H., Tamura, S.: Deaf-and-mute sign language generation system. Proceedings Medical Images and Icons SPIE 0515 (1984)
36. Kawai, H., Tamura, S.: Deaf-and-mute sign language generation system. Pattern Recogn. **18**(3/4), 199–205 (1985)
37. Cootes, T.F., Taylor, C.J.: Active shape models—'smart snakes'. Proceedings of the British Machine Vision Conference, 266–275 (1992)
38. Heap, T., Samaria, F.: Real-time hand tracking and gesture recognition usingsmart snakes. Proceedings of Interface to Real and Virtual Worlds, pp. 1–13 (1995)
39. Starner, T., Pentland, A.: Real-time american sign language recognition from video using hidden markov models, Technical Report 375, MIT Media Lab (1995)
40. Assan, M., Grobel, K.: Video-based sign language recognition using hidden Markov models. Gesture and Sign Language in Human-Computer Interaction, pp. 97–109 (1997)
41. Assan, M., Grobel, K.: Isolated sign language recognition using hidden markov models. IEEE International Conference on Computational Cybernetics and Simulation, pp. 162–167 (1997)

42. Vogler, C., Metaxas, D.: Handshapes and movements: Multiple-channel ASL recognition. Lect. Notes Artif. Intel. (LNAI) **2915**, 247–258 (2004)
43. Vogler, C., Metaxas, D.: ASL Recognition based on a coupling between HMMs and 3D Motion Analysis. Technical Reports (CIS), Department of Computer and Information Science, University of Pennsylvania (1998)
44. Imagawa, K., Matsuo, H., Taniguchi, R., Arita, D.: Recognition of local features for camera-based sign language recognition system. Proceedings 15th International Conference on Pattern Recognition, 849–853 (2000)
45. Isaacs, J., Foo, S.: Hand pose estimation for American sign language recognition. Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory, pp. 132–136 (2004)
46. Cooper, H., Ong, E., Pugeau, N., Bowden, R.: Sign language recognition using sub-units. J. Mach. Lear. Res. **13**, 2205–2231 (2012)
47. Kim, S., Waldron, M.B.: Adaptation of self organizing network for ASL recognition. In Proceedings of the Annual International Conference of the IEEE Engineering in Engineering in Medicine and Biology Society, pp. 254–254 (1993)
48. Stokoe, W.C.: Sign language structure: An outline of the visual communication systems of the american deaf. Studies in Linguistics: Occasional Papers, 8, pp. 3–37 (1960)
49. Waldron, M.B., Simon, D.: Parsing method for signed telecommunication. In Proceedings of the Annual International Conference of the IEEE Engineering in Engineering in Medicine and Biology Society: Images of the Twenty-First Century 6, pp. 1798–1799 (1989)
50. Waldron, M.B., Kim, S.: Increasing manual sign recognition vocabulary through relabelling. In Proceedings of the IEEE International Conference on Neural Networks IEEE World Congress on Computational Intelligence 5, 2885–2889, (1994)
51. Waldron, M.B., Kim, S.: Isolated ASL sign recognition system for deaf persons. IEEE Trans. Rehabil. Eng. **3**(3), 261–271 (1995)
52. Vogler, C., Metaxas, D.: Adapting hidden markov models for ASL recognition by using three-dimensional computer vision methods. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, volume 1, pp. 156–161 (1997)
53. Liddell, S.K., Johnson, R.E.: American sign language: The phonological base. Sign Lang. Stud. **64**, 195–278 (1989)
54. Vogler, C., Metaxas, D.: Parallel hidden markov models for American sign language recognition. In Proceedings of the IEEE International Conference on Computer Vision 1, pp. 116–122 (1999)
55. Kadir, T., Bowden, R., Ong, E.J., Zisserman, A.: Minimal training, large lexicon, unconstrained sign language recognition. In Proceedings of the BMVA British Machine Vision Conference 2, pp. 939–948 (2004)
56. Yin, P., Starner, T., Hamilton, H., Essa, I., Rehg, J.M.: Learning the basic units in american sign language using discriminative segmental feature selection. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4757–4760 (2009)
57. Kong, W.W., Ranganath, S.: Automatic hand trajectory segmentation and phoneme transcription for sign language. In Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, pp. 1–6 (2008)
58. Han, J.W., Awad, G., Sutherland, A.: Modelling and segmenting subunits for sign language recognition based on hand motion analysis. Pattern Recogn. Lett. **30**(6), 623–633 (2009)
59. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In Proceedings of the European Conference on Computational Learning Theory, pp. 23–37 (1995)
60. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1, pp. 511–518 (2001)
61. Premaratne, P., Yang, S., Zou, Z., Vial, P.: Australian sign language recognition using moment invariants. Lect. Notes Artif. Int. **7996**, 509–514 (2013)

62. Premaratne, P., Ajaz, S., Premaratne, M.: Hand gesture tracking and recognition system for control of consumer electronics. Springer Lect. Notes Artif. Int. (LNAI) **6839**, 588–593 (2011)
63. Premaratne, P., Nguyen, Q., Premaratne, M.: Human computer interaction using hand gestures. In Advanced Intelligent Computing-Theories and Applications. Communications in Computer and Information Science vol. 93, pp. 381–386 (2010)
64. Premaratne, P., Ajaz, S., Premaratne, M.: Hand gesture tracking and recognition system using Lucas-Kanade algorithm for control of consumer electronics. Neurocomputing J. **116**(20), 242–249 (2013)
65. Zou, Z., Premaratne, P., Premaratne, M., Monaragala, R., Bandara, N.: Dynamic hand gesture recognition system using moment invariants'. In Dias, D. (ed.) ICIAfS 2010: 5th International Conference on Information and Automation for Sustainability, IEEE Computational Intelligence Society, Colombo, Sri Lanka, pp. 108–113 (2010)
66. Zhongliang, Q., Wenjun, W.: Automatic ship classification by superstructure moment invariants and two-stage classifier, ICCS/ISITA '92 Communications on the Move (1992)
67. Hu, M.K.: Visual pattern recognition by moment invariants. IRE Trans. Info. Theory, IT8 179–187 (1962)
68. Fels, S.S., Hinton, G.: GloveTalk: A neural network inteface between a DataGlove and a speech synthesiser. IEEE Trans. Neural Netw. **4**(2–8), (1993)
69. Liang, R., Ouhyoung, M.: A realtime continuous gesture recognition system for sign language. Third IEEE International Conference on Automatic Face and Gesture Recognition (1998)
70. Fang, G., Gao, W., Zhao, D.: Large vocabulary sign language recognition based on hierarchical decision trees. Proceedings of the 5th International Conference on Multimodal Interfaces, pp. 125–131 (2003)
71. Oz, C., Leu, M.C.: American sign language word recognition with a sensory glove using artificial neural networks. Eng. Appl. Artif. Intel. **24**(7), 1204–1213 (2011)

Chapter 7
Future Trends in Hand Gesture Recognition

In 2005, the author developed a comprehensive hand gesture recognition system that was capable of controlling many consumer electronics control devices. The publicity around this development was echoed in 2007 after the publication of this research in IET Computer Vision journal [1]. The media frenzy that was generated around the world due to this invention was unprecedented. He was contacted by Microsoft Australia, Logitech USA and NDA the world's largest settop box manufacturer to discuss future trends emanating from the development. Six months later, Samsung patented similar technology for their mobile phones. By 2009, Toshiba and Samsung developed digital Television with a built-in hand gesture interface. In 2013, there were 20 consumer electronics devices with gesture control were added to the gadgets world.

Today, hand gesture interfaces are common place in variety of consumer electronics devices. It is no longer have to be speculated what the future would stand. It is already here and the future trends are certain. Figure 7.1 shows a recent development of a company acquisition by Google to control their Gmail and Youtube with more gesture friendly interfaces.

Following LG and TCL, Hisense, the world's fifth largest smart TV brand manufacturer, adopted Hillcrest Labs' Freespace technology for hand gesture recognition. The model that they developed has in-air pointing, gesture control and motion control via a remote control for it smart Televisions and set-top boxes as shown in Fig. 7.2 [3].

Recently, Google has been investing heavily on futuristic technology such as driverless cars and all types of robots. Keeping up with this trend, they recently applied for a patent that uses hand motion to control the car itself. Its proposed system relies on both a ceiling-mounted depth camera and a laser scanner to trigger actions based on an occupant's hand positions and movements. Swipe near the window will move the car forward while pointing to the radio and you'll turn the volume up [4]. Figure 7.3 highlights this futuristic trend envisioned by google.

At IDF, one of the largest technological exhibitions in Beijing in 2013, Intel promoted a development platform aimed at Creative Interactive Gesture Camera for a perceptual computing by way of voice recognition, gesture control and

P. Premaratne, *Human Computer Interaction Using Hand Gestures,*
Cognitive Science and Technology, DOI 10.1007/978-981-4585-69-9_7,

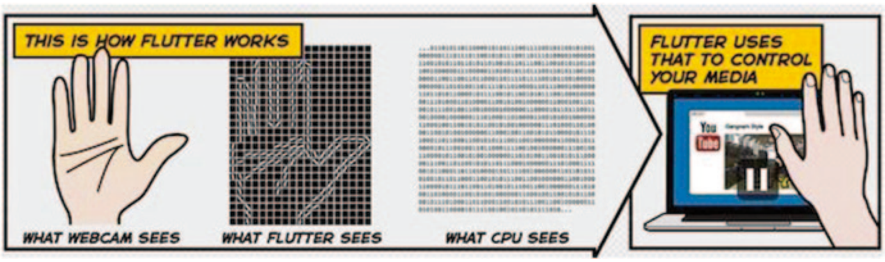

THIS IS HOW FLUTTER WORKS
WHAT WEBCAM SEES
WHAT FLUTTER SEES
WHAT CPU SEES
FLUTTER USES THAT TO CONTROL YOUR MEDIA

Fig. 7.1 Gesture control startup Flutter acquired by Google, could make Gmail Motion a reality, courtesy of [2]

Fig. 7.2 Hisense smart television with in-air point and gesture control

face recognition. They made complementary SDK for realizing gesture and voice control when using the Creative interactive camera. Intel has been offering developers a Creative Interactive Gesture Camera for $ 149 to promote the growth of future proof user interfaces. The camera consists of a time-of-flight depth camera is very similar to Microsoft Kinect II that was discussed in Chap. 3. However, the main difference being that the new camera is designed for a closer proximity and can therefore also pick up the movement of each finger as shown in Fig. 7.4 [5].

At the start of 2013, Samsung launched their latest smart television that was capable of hand tracking and function control as shown in their promotion in Fig. 7.5. It also supported voice control and face recognition showing the future trends in user interfaces [6].

Hand gesture recognition will continue to grow in consumer electronics control arena for the foreseeable future as it is the only user interface that provides larger vocabulary of control features without the mismatch of a keyboard in a living room. Yet, as was highlighted in Chaps. 3, 4, 5, and 6, true potential of hand gesture recognition only can be realized when system is developed with multiple cameras and depth measurements that would be able to identify gestures as a human does. The

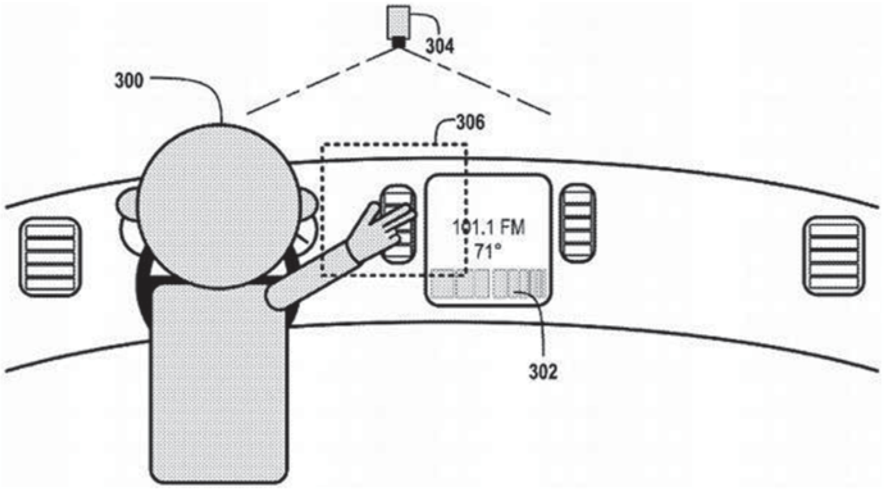


Fig. 7.3 Patented car interface by Google using hand gestures. [4]

Fig. 7.4 Creative interactive gesture camera

Fig. 7.5 Samsung smart television with face recognition and hand-tracking

research will take many more decades to reach such maturity yet, it will herald a milestone in man-machine interface.

References

1. Premaratne, P., Nguyen, Q.: Consumer electronics control system based on hand gesture moment invariants. IET Comput. Vision. **1**(1), 35–41 (2007)
2. http://www.engadget.com/2013/10/02/google-acquires-flutter-gesture-control/
3. http://www.engadget.com/2013/09/04/hisense-hillcrest-labs-freespace-gesture-and-motion-control/
4. http://www.engadget.com/2013/10/03/google-applies-for-patent-on-gesture-based-car-controls/
5. http://www.engadget.com/2013/04/11/creative-interactive-gesture-camera-intel/
6. http://www.engadget.com/2012/01/09/live-from-samsungs-ces-2012-press-event/