Real-time Hand Gesture Detection and Recognition for Human Computer Interaction

By

Nasser Dardas

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa
Ottawa, Ontario

Abstract

This thesis focuses on bare hand gesture recognition by proposing a new architecture to solve the problem of real-time vision-based hand detection, tracking, and gesture recognition for interaction with an application via hand gestures. The first stage of our system allows detecting and tracking a bare hand in a cluttered background using face subtraction, skin detection and contour comparison. The second stage allows recognizing hand gestures using bag-of-features and multi-class Support Vector Machine (SVM) algorithms. Finally, a grammar has been developed to generate gesture commands for application control.

Our hand gesture recognition system consists of two steps: offline training and online testing. In the training stage, after extracting the keypoints for every training image using the Scale Invariance Feature Transform (SIFT), a vector quantization technique will map keypoints from every training image into a unified dimensional histogram vector (bag-of-words) after K-means clustering. This histogram is treated as an input vector for a multi-class SVM to build the classifier. In the testing stage, for every frame captured from a webcam, the hand is detected using my algorithm. Then, the keypoints are extracted for every small image that contains the detected hand posture and fed into the cluster model to map them into a bag-of-words vector, which is fed into the multi-class SVM classifier to recognize the hand gesture.

Another hand gesture recognition system was proposed using Principle Components Analysis (PCA). The most eigenvectors and weights of training images are determined. In the testing stage, the hand posture is detected for every frame using my algorithm. Then, the small image that contains the detected hand is projected onto the most eigenvectors of training images to form its test weights. Finally, the minimum Euclidean distance is determined among the test weights and the training weights of each training image to recognize the hand gesture.

Two application of gesture-based interaction with a 3D gaming virtual environment were implemented. The exertion videogame makes use of a stationary bicycle as one of the main inputs for game playing. The user can control and direct left-right movement and shooting actions in the game by a set of hand gesture commands, while in the second game, the user can control and direct a helicopter over the city by a set of hand gesture commands.

Résumé

Cette thèse propose une solution pour le problème de reconnaissance des gestes de la main nue (sans marqueurs) en proposant une nouvelle architecture pour résoudre le problème de détection en temps réel d'une main dans une image, du suivi et de la reconnaissance des gestes effectués. La première étape dans notre système consiste à détecter et suivre la main nue dans un environnement quelconque par soustraction du visage, détection de la peau et par comparaison de contours. La deuxième étape permet de reconnaitre les mouvements de la main en utilisant un "sac de caractéristiques" et des machines à vecteurs de support (SVM) multi-classes. Finalement, un vocabulaire est proposé pour générer des commandes par gestes pour différentes applications.

Notre système de reconnaissance de gestes comprend deux phases: Un phase d'apprentissage hors-ligne et une phase de test en ligne. Durant la phase d'apprentissage les points clés sont extraits pour chaque image dans l'ensemble d'apprentissage en utilisant les transformés SIFT (Scale Invariance Feature Transform); Ensuite on associe chaque image à un vecteur d'histogramme de dimension unifiée (un sac de mots) par une quantification vectorielle et un partitionnement de type K-means. Cet histogramme est traité comme un vecteur d'entrée pour le classifieur SVM multi-classes. Durant la phase de test, on utilise notre algorithme pour détecter la main dans toutes les images acquises par une webcam. Les points clés sont ensuite extraits pour toute région contenant la pose de la main détectée. Ces points clés sont fournis au modèle de partionnement pour ainsi les associer à un vecteur de type sac de mots, qui est lui même utilisé pour reconnaitre le geste par le classifieur SVM multi-classes.

Un autre système de reconnaissance de gestes utilisant une analyse en composantes principales (PCA) est aussi proposée. Les axes et les nouveaux poids des images d'apprentissage sont ainsi calculés. Durant la phase de test, la main est détectée et le poids de la région de l'image la contenant est calculé suivant les nouveaux axes. Finalement, on reconnait le geste de la main en trouvant le geste dans l'ensemble d'apprentissage avec la distance euclidienne minimale à l'image de test.

Deux applications de l'interaction par gestes étaient aussi implémentée pour un environnement virtuel de jeu 3D. Ce jeu, dit d'épuisement, utilise un vélo stationnaire comme l'une des entrées et permet à l'utilisateur de contrôler les mouvements gauche et droit, et de tirer en utilisant un vocabulaire de gestes. Dans le deuxième match, l'utilisateur peut contrôler et de diriger un hélicoptère au-dessus de la ville par un ensemble de commandes geste de la main.

Acknowledgements

First of all, I thank God for giving me the wisdom and strength for accomplishing this research.

I would like to express my deep gratitude, appreciation, and respect for my first supervisor Professor Nicolas D. Georganas and my current supervisors Professor Emil M. Petriu and Professor Abdulmotaleb El Saddik for their endless support, encouragement, advice, patience, interest, and guidance throughout the pursuit of my PhD's degree. I have gained enormously from their ideas, technical insights and profound thinking. They gave me the opportunity to shape all my future.

I wish to thank Dr. Qing Chen for his ideas, suggestions and cooperation in my first project. I also wish to thank Juan Silva for his cooperation in the project of hand gestures interaction with exertion videogame.

Finally, great thanks and love to my wife for her constant understanding, encouraging, and support to fulfill my goals. Great thanks for my lovely sons Majid, Mohammad, and Hashim for being patients while I'm busy and away from them.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of the Acronyms

ANN. . . . . Artificial Neural Network

ASL . . . . . American Sign Language

BoG . . . . . Bag of Features

BoW . . . . . Bag of Words

DOF. . . . . .Degrees Of Freedom

FEMD . . . . . .Finger-Earth Mover's Distance

FPS. . . . . .Frames Per Second

FSM . . . . . Finite State Machine

HCI . . . . . .Human Computer Interaction

HMI. . . . . .Human Machine Interface

HMM . . . . . Hidden Markov Model

HSV. . . . . .Hue, Saturation, Value

LiDAR. . . . . .Light Detection and Ranging

PCA . . . . . Principle Component Analysis

PGM. . . . . Protable Gray Map

RPM. . . . . .Revolutions Per Minute

SIFT . . . . . Scale-invariant Feature Transform

SURF. . . . . .Speeded Up Robust Features

SVM . . . . . .Support Vector Machine

TAM. . . . . Technology Acceptance Model

TDNN . . . . . .Time-Delay Neural Network

VE. . . . . Virtual Environment

VQ. . . . . Vector Quintization

Chapter 1

Introduction

1.1 Background

Traditional human-computer interaction devices such as the keyboard and mouse became ineffective for an effective interaction with the virtual environment applications because the 3D applications need a new interaction device. An efficient human interaction with the modern virtual environments requires more natural devices. Among them the "Hand Gesture" human-computer interaction modality has recently become of major interest. The main objective of gesture recognition research is to build a system which can recognize human gestures and utilize them to control an application. Hand gestures are a collection of movements of the hand and arm that vary from the static posture of pointing at something to the dynamic ones used to communicate with others. Recognizing these hand movements needs modeling them in the spatial and temporal domains. Hand posture is the static structure of the hand while its dynamic movement is called hand gesture and both are particularly crucial for human-computer interaction. The methods used for understanding these structures and movements are among the most classifying researches that still in progress.

The information related with hand gestures during a talk has a spatial as well as a temporal structure. The gesture recognition methods are primarily divided into Data-Glove Based and Vision Based methods. The Data-Glove based approaches use mechanical or optical sensors connected to a glove that converts finger flexions into electrical signals for recognizing the hand posture. This method obstructs the ease and naturalness of the user interaction because it compels the user to carry a load of cables which are attached to the computer. However, the equipment is relatively expensive and awkward. On the contrary, the Vision Based techniques need only a camera, consequently achieving a natural interaction between humans and computers without the need for any extra devices. These systems are inclined to complement biological vision by implementing artificial vision systems in software and/or hardware. This creates a challenging problem because these systems require user and camera to be independent and

invariant against background changes, transformations, and lighting conditions to attain real-time performance. Besides, since the human hand is a deformable and articulated object and, this will increase the difficulty in the segmentation process and the shape recognition stage.

Vision-based approaches to hand gesture detection are most likely the most natural way of building a human-computer interface. However, it is also the most difficult one to build in a satisfactory manner because of the drawbacks in machine vision today. More specifically, the vision-based approach is executed by using one or more cameras. The visual information about the user in some visual environment is obtained, and the required gesture is extracted. However, there are some challenges: the detection of the moving hand from a cluttered background, the analysis of hand motions, the tracking of hand positions relative to the environment, and finally the recognition of hand postures.

Vision-based methods differ among themselves in:

- Number of cameras. If there are more than one, are they combined early (stereo) or late (multi-view)?
- Speed and latency. Is the system real-time (i.e., fast enough, with low enough latency interaction)?
- Structured environment. Are there constraints on the background, lighting, speed of movement, etc.?
- User requirements. If the person has to wear anything particular such as markers, gloves, long sleeves, rings or anything on the face such as glasses, and beard.
- Primary features. What low-level features are extracted such as color, shape, edges, regions, silhouettes, moments, histograms, etc.
- 2D or 3D (dimensional) representation.
- Representation of time and space: How are the temporal and spatial aspects of the gesture represented and used in recognition?

There is a loss in information whenever a 3D image is projected to a 2D plane. Moreover, elaborate 3D models include prohibitive high dimensional parameter spaces. Besides, a tracker should be invariant against sizes and shapes changing of hand gesture, other moving objects in the background, and noise.

A new vision-based framework is presented in this thesis, which permits humans to interact with computers through hand postures. The proposed system is invariant against

different lighting conditions and cluttered backgrounds. The proposed system is appropriate for real-time applications because of its speed and accuracy. Our technique concentrates on different stages including hand posture detection, tracking and recognition for frames captured from a video file or webcam to their final classification.

The general technique to tackle the gesture recognition problem is to deal with it as a pattern recognition problem where a set of features are extracted from images captured from a video file or webcam that in turn are matched (i.e., Classification) to a predefined representation of the gesture (i.e., pattern).

1.2 Motivation

Vision-based hand gesture recognition recently became a highly active research area with motivating applications such as sign language recognition [Fang07], socially assistive robotics [Baklou08], directional indication through pointing [Nickel07], control through facial gestures [Baklou08], human computer interaction (HCI), immersive game technology, virtual controllers, affective computing and remote control. Within the broad range of application scenarios, hand gestures can be categorized into at least four classes [Wu99]: controlling gestures, conversational gestures, communicative gestures, and manipulative gestures. Hand gestures are powerful human interface components. However, their fluency and intuitiveness have not been utilized as computer interface. Recently, hand gesture applications have begun to emerge, but they are still not robust and are unable to recognize the gestures in a convenient and easily accessible manner by the human. Several advanced techniques are still either too fragile or too coarse grained to be of any universal use for hand gesture recognition. Especially, techniques for hand gesture interfaces should be developed beyond current performance in terms of speed and robustness to attain the needed interactivity and usability.

It is a difficult task to recognize hand gestures automatically from camera input. It usually includes numerous phases such as signal processing, detection, tracking, shape description, motion analysis, and pattern recognition. The general problem is quite challenging because of several problems such as the complex nature of static and dynamic hand gestures, cluttered backgrounds, transformations, lighting changes, and occlusions. Trying to solve the problem in its generality needs elaborate techniques that require high performance against these issues.

Hand gesture recognition from video frames is one of the most main challenges in image processing and computer vision because it provides the computer the capability of detecting, tracking, recognizing and interpreting the hand gestures to control various devices or to interact with several human machine interfaces (HMI).

The objective of this thesis is to develop a novel approach to the current problem of hand gesture recognition. By proposing an entirely new algorithm to an existing problem, it is anticipated that this technique is a step forward in practical applications of hand gesture recognition for everyday use. The algorithm should satisfy numerous conditions:

- The first requirement is real-time performance. This is critical for complete interactivity and intuitiveness of the interface. This is measured by frames per second (fps) which essentially gives information about the refresh rate of the application. If the refresh rate is long, then there is a delay between the real event and the recognition. If gestures are carried out in an extremely fast sequence, the event may not be recognized at all.
- The second required condition is flexibility, and how well it combines with new applications and existing applications. The algorithm should be able to accommodate external programs easily to be a candidate for practical applications. This will be an advantage for the application developer and the user.
- Third, the approach should be practically precise enough to be used. The approach should correctly recognize the defined gestures 90%-100% of time to be successful and of practical use.
- The fourth needed condition is robustness in which the system should be able to detect, track, and recognize different hand gestures successfully under different lighting conditions and cluttered backgrounds. The system should also be robust against scale and rotations.
- The fifth needed condition is scalability. A large gesture vocabulary can be involved with a small number of primitives. The user interacts easily with an application by building different gesture commands.
- Finally, the approach should be user-independent in which the system must be able to work for various persons rather than a particular person. The system must recognize hand gestures for different human hands of different scales and colors.

This thesis proposes a novel human computer interaction (HCI) system, which uses hand

gestures as input for communication and interaction. The system is starts with capturing images from a webcam or a pre-recorded video file. Several systems in the literature have strict restrictions such as using particular gloves, uniform background, long-sleeved user arm, being in particular lightning conditions and using particular camera parameters. These restrictions destroy the recognition rate and naturalness of a hand gesture recognition system. The performances of those systems are not strong enough to be used on a real-time HCI system. This thesis aims to design a vision based hand gesture recognition system with a high recognition rate along with real-time performance. The system is invariant against previous strict restrictions on the human environment and can be used for real-time HCI systems.

Usually, these interaction systems have two challenges: hand detection and hand gesture recognition. Hand detection must be done before gesture recognition. Once the hand is detected clearly in the current image, the gesture recognition process is started around the detected hand.

Scale Invariance Feature Transform (SIFT) features, proposed by Lowe [Lowe04], are features (keypoints) extracted from images for helping in reliable matching between different views of the same object, image classification, and object recognition. The extracted keypoints are invariant to scale, orientation and partially invariant to illumination changes, and are highly distinctive of the image. Therefore, the SIFT is adopted in this thesis for the bare hand gesture recognition. However, SIFT features are too high dimensionality to be used efficiently. We propose to solve this problem by the bag-of-features approach [Lazebn06, Jiang07] to reduce the dimensionality of the feature space.

A hand gesture is an action, which consists of a sequence of hand postures. Gestures are restricted to a discrete set of postures recognized in static form. In this thesis, we will highlight the dynamic aspect of hand gestures and how to recognize different hand gestures that differ solely in their timing and space aspects. Our goal is to recognize dynamic gestures in real-time with high recognition rate.

1.3 Thesis Organization

This thesis includes 6 chapters:

- Chapter 1 introduces the background, vision-based hand gesture processing stages, and motivations of this work. The objectives to be accomplished by the work are also given.

- Chapter 2 provides a comprehensive literature review based on two categories for vision-based hand detection, tracking and gesture recognition methods: appearance-based methods versus 3D hand model-based methods. Feature extraction and classification methods are also reviewed.
- Chapter 3 presents our algorithm to detect hand posture, which includes detecting and tracking the bare hand in a cluttered background using skin detection and a hand posture contour comparison algorithm after face detection and subtraction have been performed. Our algorithm will be used in Chapter 4, 5, and 6 for hand posture detection before recognition.
- Chapter 4 proposes the overall architecture for the system: a two-stage architecture which decouples the hand gesture recognition system into a training stage to build a k-means clustering model and multi-class support vector machine (SVM) classifier model to use them in the testing stage to recognize hand posture after detecting bare hand using our algorithm in Chapter 3. A grammar is built for our system using a finite state machine (FSM). Based on the grammar, our system can generate gesture commands, which are used to control an exertion videogame. The user in the exergame will control and direct left-right movement and shooting actions by a set of hand gesture commands.
- Chapter 5 presents a method for recognizing hand postures using Principle Components Analysis (PCA) after detecting the bare hand using our algorithm. A game that employs gesture-based interaction with a 3D gaming virtual environment (VE) is implemented. The user in the game will control and direct the movements of a helicopter over a city by a set of hand gesture commands.
- Chapter 6 provides conclusions and outlines future work.

1.4 Contributions

Detecting and tracking hand gestures in a sequence of images help in extracting hand region. Thus, processing time will be reduced and accuracy will be increased as the features of that region will represent the hand gesture only. Skin color is a significant image feature to detect and track human hands. However, color-based methods face the challenge of removing other objects with similar color such as the face and human arm. To solve this problem, we proposed a

new technique to detect hand postures only using face subtraction, skin detection and a hand posture template contour comparison algorithm as we will discuss in Chapter 3.

Recently, bag-of-features representations have shown outstanding performance for action and gesture recognition. They permit to recognize a rich set of actions ranging from basic periodic motion (waving, running) to interactions (kissing, shaking hands). However, "bag-of-features" approaches exclusively rely on the dense local motion features. They do not have the relations between the features in the spatial and the temporal domains. These are significant correlations, which are useful for recognition. One of our main contributions is the ability to use spatiotemporal (space-time) correlation between every two consecutive frames in a video sequence in terms of the transition among recognized postures and their locations to allow our system for real-time and accurate dynamic hand gesture recognition as we will discuss in Chapter 4. Techniques and algorithms with accuracy measures over 90% can generally be considered accurate [Chen08]. The statistical analysis based on bag-of-features model and multi-class SVM.

The major contributions of this thesis are as follows:

1) We proposed a new real-time and accurate approach for bare hand postures detection and tracking using face subtraction, skin detection and hand postures contours comparison algorithm. [Dardas11].
2) We achieved real-time performance and accurate recognition for the detected hand postures using a bag-of-features model and multi-class SVM. [Dardas10, Dardas11].
3) We developed our system into real-time dynamic hand gesture recognition with high recognition rate by using spatiotemporal (space-time) correlation between every two consecutive frames in a video sequence in terms of the transition between recognized postures and their locations. [Dardas11].
4) We built a grammar that generates a large number of gesture commands used for controlling an application or videogame by tracking and monitoring the size or scale of the detected hand posture, its movement direction and the transitions among postures. [Dardas11].
5) We achieved real-time performance and accurate recognition for the bare hand postures using Principal Component Analysis (PCA) after using our approach for detecting and tracking hand posture. [Dardas11b].

1.5 Publications Arising from the Thesis

1. **[Dardas10] Nasser Dardas**, Qing Chen, Nicolas Georganas, Emil Petriu, "Hand Gesture Recognition Using Bag-of-Features and Multi-Class Support Vector Machine", HAVE 2010, 9th IEEE International Workshop on Haptic Audio Visual Environments and Games, Oct. 16-17, 2010, Phoenix, AZ, USA.
2. **[Dardas11] Nasser Dardas**, Nicolas Georganas, "Real-time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques", IEEE Transactions on Instrumentation and Measurement, Vol. 60, No. 11, pp. 3592 - 3607, Nov 2011.
3. **[Dardas11b] Nasser Dardas**, Emil Petriu, "Hand Gesture Detection and Recognition Using Principal Component Analysis", CIMSA 2011, 4th IEEE International Conference on Computational Intelligence for Measurement Systems and applications, Sep. 19-21, 2011, Ottawa, ON, Canada.
4. **[Dardas11c] Nasser Dardas**, M. Alhaj, "Hand Gesture Interaction with a 3D Virtual Environment", The International Journal of ACM JORDAN, Vol.2, No.3, September 2011.
5. **Nasser Dardas**, Juan M. Silva, Abdulmotaleb El Saddik, "Target-Shooting Exergame with a Hand Gesture Control", International Journal of Multimedia Tools and Applications, 2012.

Chapter 2

Literature Review

2.1 Introduction

Recently, there has been a surge of interest on hand detection, tracking, and gesture recognition. While a comprehensive literature review of this field would be a daunting challenge because of the huge number of publications, we will concentrate in this chapter on a review of the most representative developments that are pertinent to my thesis research topics.

Human hands are the parts that people use the most and with the most ease to interact with the world. Hands are highly articulated structures with some 27 degrees of freedom (DOF) [Wu01b]. Because of these high DOF of the human hand, hand gesture recognition is indeed an extremely challenging task.

Even though hand postures and gestures are frequently considered as being identical, there are actually differences as explained in [Corrad01, Liang98]. While the hand posture is a static motionless pose, such as making a palm posture and holding it in a certain position, the hand gesture is a dynamic process consisting of a sequence of changing hand postures over a short duration, as for instance waving the hand.

Because of the inherent complexity of the hand gesture, its recognition can be divided into two stages: the low stage hand posture detection and the high stage hand gesture recognition. For a computer to recognize a hand gesture, first the hand should be detected from the captured image, and then the recognition of the gesture made by that hand should be done in a similar way as humans do. However, this is a more challenging technique to be implemented due to the limitations of such a natural system. Segmentation of the hand image is the first stage of all these systems. In [Aran06, Tokatl05], gloves or finger marks have been used to extract the hand posture information from the image and facilitate the hand segmentation process by removing the varying skin color issue of the problem. This approach permits the system to detect hands easily, and it is invariant against lightning conditions which may change. Another way for solving the problem is by implementing a hand gesture recognition system with a uniform

background such as a black curtain [Ionescu05]. Using a uniform background will facilitate the segmentation of the hand region. However, including gloves or uniform background reduce the natural (transparent) interaction feeling of gesture-recognition applications by imposing artificial constraints on the user's environment. Since the scope of the thesis is detecting hands in normal backgrounds, our system must be able to detect and recognize hand gestures in a cluttered background using face subtraction, skin detection, and contour comparison with hand postures templates.

The well known "come as you are" expression [Triesch98] stands for a Human Computer Interaction (HCI) system without constraints for the user to wear markers, gloves, or long sleeves, uniform background, or select a particular illumination. These constraints hinder the user's ability to interactively track and recognize gestures. In order to avoid all these problems, the computer vision technique needs to meet the challenge to recognize in real-time hand gestures without requiring the user to wear any additional aids or to be connected to a special hand tracking or haptic device.

Many researchers have proposed various techniques for hand gesture recognition systems. Usually, these systems are divided into two techniques, namely the glove-based and respectively the vision-based methods [Garg09]. The Data-Glove based approaches utilize sensor devices for digitizing hand and finger movements into multi-parametric data. The additional sensors facilitate detection of hand configuration and movement. However, the sensor devices are quite expensive and wearing gloves or trackers is uncomfortable and enlarges the "time-to interface," or setup time. On the other hand, vision-based methods are more natural and useful for real-time applications. Besides, computer-vision-based interfaces provide many outstanding advantages:

- Computer vision is nonintrusive;
- Sensing is passive and silent;
- Installed camera systems can be used for other jobs;
- Sensing and processing hardware is commercially available at low cost.

On the other hand, vision based systems need application-specific image processing algorithms, programming, and machine learning. The need for all these to work in a variety of environments causes several issues because these systems require user and camera independent and invariant against the background and lighting changes to attain real-time performance.

A reliable set of characteristic features and relevant information of how they correlate in representing hand gestures are needed in order to successfully recognize hand gestures. A taxonomy for gesture types and a complete gesture modeling framework is presented in [Pavlov97]. It provides a characterization of gestures' spatial properties that divides gesture interpretation into two groups 3D hand model-based and appearance-based.

Appearance-based methods were developed for hand posture and gesture recognition directly from images using visual features such as hand contours and fingertips positions.

Other methods, called 3D model-based, provide a geometrical representation of the hand configuration using the joint angles of a 3D hand's articulated structure recovered from the sequence of images

2.2 Vision Based Hand Gesture Recognition

In a vision based hand gesture recognition system shown in Figure 2.1, the motion of the hand is captured by a camera [Garg09]. A set of features is extracted from every frame captured. A classifier uses the extracted features to recognize different postures for every frame. Since gestures are a dynamic sequence of hand postures connected though continuous movements, the classifier can be trained against a possible grammar. There are mainly two categories for vision based hand gesture recognition, namely the three dimensional (3D) hand model-based methods and, respectively, the appearance-based methods [Zhou03].

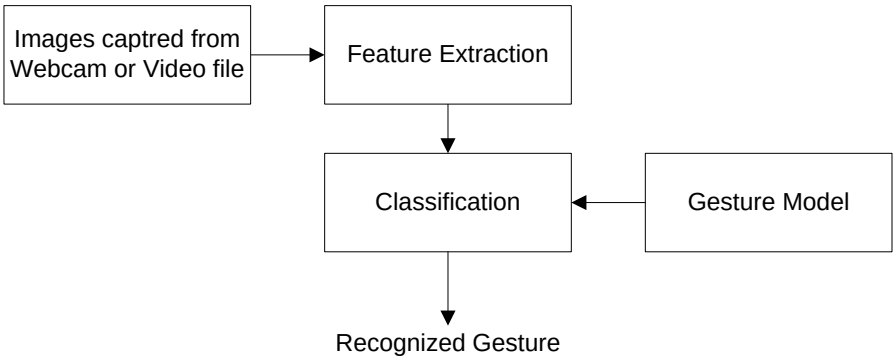


Figure 2.1: Vision-Based Hand Gesture Processing Stages.

2.2.1 3D Hand Model Based Approach

Many approaches that use the 3D hand model based technique [Rehg94, Heap96, Wu01, Stenger01] depend on the 3D kinematic hand model with significant degrees of freedom (DOF),

and calculate the hand parameters by comparing the input frames and the 2D appearance projected by the 3D hand model. This will be suitable for realistic interactions in virtual environments. The 3D hand model based technique provides a rich description that permits a wide class of hand gestures. However, since the 3D hand models are articulated deformable objects with many DOF's, a large image database is required to deal with the entire characteristic shapes under several views. This method has many drawbacks that have prevented it from practical use. First, for every image the initial parameters must be close to the solution; otherwise, the method is likely to find a suboptimal solution such as local minima. Second, the fitting process is sensitive to noise such as lens aberrations and sensor noise in the imaging process. The final drawback is the difficulty of feature extraction and inability to handle singularities that occur from unclear views.

2.2.2 Appearance Based Approach

Appearance based techniques extract image features to model the visual appearance of the hand and compare these features with the extracted features from the video frames using a pattern classification module [Iwai99, Lement04] similar to our approach that will be described in Chapter 4. Appearance-based techniques are generally known as an instance of the general problem of pattern recognition, which includes three tasks:

1. Feature Extraction.
2. Classifier (learning to use training samples (known label)).
3. Classification of unknown samples.

A large number of gesture recognition methods proposed in the literature fall in the system of the block diagram given in Figure 2.1, such as [Travie03, Chen03]. These systems use a module for feature extraction such as a feature vector and a module for the classification of the set of N buffered feature vectors.

Appearance based approaches have real-time performance as with our approaches described in Chapter 4 and 5 because of the simpler 2D image features that are used. A simple method consisting of searching for skin coloured regions in the image, was used in [Stenger06]. However, this method has some limitations; first, it is highly sensitive to lighting conditions. Secondly, it is assumed that there are no other skin like objects within the image.

In [Bretzn02], scale-space color features were used to recognize hand gestures, which are based on feature detection and user independence. However, the system shows real-time performance only when no other skin coloured objects exist in the image.

The authors of [Argyros06] obtained a clear-cut and integrated hand contour to recognize hand gestures, and then computed the curvature of each point on the contour. Due to noise and unstable illumination in the cluttered background, the segmentation of the integrated hand contour had some difficulty.

Eigenspace approaches can provide an effective illustration of a large group of high-dimensional points using a small group of basis vectors. However, eigenspace methods are not invariant to translation, scaling, and rotation. In Chapter 5, we will discuss how to solve this problem by making the system invariant to scaling and rotation using an eigenspace approach.

There have been a number of research efforts recently focusing on local invariant features [Wang08, Barczak05, Chen07]. In [Wang08], Adaboost learning algorithm and SIFT features were used to achieve in-plane rotation invariant hand detection. In addition, a sharing feature concept was used to speed up the testing process and increase the recognition accuracy. Therefore, efficiency of 97.8% was achieved. However, several features such as a contrasted context histogram had to be used to achieve hand gesture recognition in real-time.

2.3 Feature-extraction

Feature-extraction techniques are intended for collecting data about gesture position, orientation, posture, and temporal progression. Feature extraction refers to the processing and analysis of lower-level data (pixel values), which are required to generate higher-level data such as objects contours. Theoretically, feature extraction is a method of reducing data dimensionality by encoding related information in a compressed representation and removing less discriminative data. Obviously, related or discriminative information differs with the object of interest and objectives of the application.

Feature extraction is vital to gesture recognition performance. Therefore, the selection of which features to deal with and the extraction method are probably the most significant design decisions in hand motion and gesture recognition development.

Appearance-based methods depend on direct registration of hand gestures with 2D image features. The well known image features used to detect human hands and recognize gestures are

hand colors [Wu02, Mckenna04, Yao02, Bretzn02], shapes [Ramamo03, Ong04, Chen03], local hand features [Oka02, Zhang01, Malik02], optical flow [Cutler98, Lu03] and others.

Color property has been most likely the most exploited for feature extraction [Sigal04, Yao04]. However, texture, luminosity, and motion defined by two or more frames have been tackled also [Wu00, Cutler98].

In the following subsections, many famous feature types and their computation techniques will be discussed such as SIFT features and will be used in Chapter 4.

2.3.1 Motion

A frame-to-frame comparison in a learned background model is an efficient and computationally effective method for finding foreground objects and for detecting their position and motion. This comparison makes many assumptions such as a stationary camera or image pre-processing to stabilize the video and a static background. In [Binh05], the Kalman filter was used to predict hand location in one frame based on its location detected in the previous frame by tracking the motion of the hand from frame to frame to search for a skin color region.

2.3.2 Depth

Depth information from a calibrated camera pair [Rausch02] or direct depth sensors such as LiDAR (Light Detection and Ranging) or Microsoft Kinect [Takaha09, Lee09] is a good indication that the person is expected to face the camera(s) and the hands are considered the closest objects. However, depth from stereo is coarse-grain and relatively noisy and therefore, it is frequently combined with other image cues such as color [Grange04, Nickel07].

2.3.3 Color

Skin color [Bretzn02, Mckenna04, Imagawa00] is a significant image feature to detect and track human hands. However, color-based methods face the challenge of removing other objects with similar color such as the face and arm. In order to solve this problem, persons are regularly required to wear long-sleeve shirts and constraints are enforced on the colors of other objects in the view. According to our approach in Chapter 3, we eliminate these restrictions by proposing a new technique to detect hand postures only using face subtraction, skin detection

and hand postures contours comparison algorithm. Most color-based methods are not invariant against lighting changes, while our approach in Chapter 3 solves this problem.

Skin color is the vital invariant feature in this thesis as we will discuss in Chapter 3. Human skin color does not depend on human race and the wavelength of the exposed light. The threshold of the skin color is a wide range to extract current skin color in an input image accurately. Since shadows, illumination and pigmentation of human skin conditions can differ, it is logical to adjust this common threshold and contract the skin color for the current conditions. Skin color is assumed to contain all the skin pixels in an image with some other skin like pixels. Those false positive pixels should be removed by a fine skin segmentation like our approach in Chapter 3 by using contours comparison with hand postures template.

Several efforts in the literature use various skin color thresholds for different color spaces to detect faces or hands in images. The authors of [Kim06] begin with an RGB image and by using a dimension reduction method to propose its own skin color in two dimensions to compare its performance with HSV skin color. In [Sabeti07, Terril99], a comparison was discussed among HSV/HSI, RGB, TSL and YCbCr skin color space performances. In [Askar04], skin color was rapidly segmented in YUV color space and then tuned the present skin color with a quasi-automatic technique which requires some data from user. In [Albiol01], chrominance with luminance data in YCbCr color space was utilized to segment the skin color for existing situations and histogram clustering is used for fine skin segmentation.

2.3.4 Multi-cue

Instead of depending on a single image cue, several methods can combine information from multiple cues. Appearance- based hand, face, or body detection together with motion-based region-of-interest designation will increase the accuracy and speed. Appearance and color for detection and motion combined with color were applied for hand-gesture interfaces as in [Rausch02] and as shown in Figure 2.2 from [Kölsch04]. Elimination of any of these cues will reduce the performance. Techniques that detect a gesture in an image depending on color and then classify the shape will not be considered as multi-cue methods because the cues are applied successively not cooperatively. To illustrate, if the first cue does not succeed, the second cue is worthless. Actual multi-cue systems have similar problems with information combination such

as classic sensor fusion. According to our approach in Chapter 3, skin color detection combined with contour comparison was applied for hand posture detection.

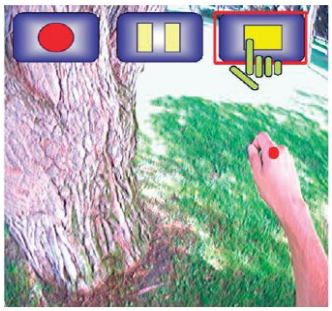


Figure 2.2: Multi-cue hand tracking and posture recognition [Kölsch04].

2.3.5 Hu Invariant Moments

Moment invariants are a group of nonlinear functions. Hu [Hu62] first introduced the geometric moment invariants which are invariant under change of scale, translation, and rotation defined on normalized geometrical central moments up to the third order. Moments and functions of moments describe characteristics of an object that uniquely represent its shape and have been widely utilized as the invariant global features of an image in pattern recognition and image classification. Hu's moments have a major problem because of the large values of geometric moments, which cause numerical instabilities and noise sensitivity. In [Yun09, Ren10], Hu invariant moments were used to recognize hand gestures.

2.3.6 SIFT Features

In [Lowe04], a method called Scale Invariant Feature Transform (SIFT) was introduced to extract distinctive invariant features from images that can be used to carry out reliable matching between various views of an object or a scene as shown in Figure 2.3.

The SIFT features utilize scale-space extrema in the difference-of-Gaussian (DoG) function convolved with the image as the keypoints which are invariant to image scaling. By assigning a consistent orientation for every keypoint depending on local image features, the keypoint descriptor can be represented in relation to this rotation and thus attain invariance to image rotation as shown in Figure 2.3 for the rotated toy train in the image. In addition to the invariance against image scaling and rotation, the SIFT features are also partially invariant against changes in lighting conditions and 3D camera viewpoint. Besides, in [Lowe04] a method

was presented for object recognition using these features. The recognition is achieved by comparing individual features to a database of features from known objects using a fast nearest-neighbour method, followed by a Hough transform to discover clusters belonging to a single object, and finally executing verification by using least-squares solution for consistent pose parameters. This recognition method can robustly recognize objects with cluttered backgrounds and occlusion, and the method can achieve real-time performance for low resolution images [Lowe04]. Therefore, the SIFT is adopted in this thesis for the bare hand gesture recognition.

Figure 2.3: Training images are on the left. These can be recognized in a cluttered image with extensive occlusion, in the middle. Results of recognition are on the right [Lowe04].

The four detection stages for SIFT features are [Lowe04]:

1. Scale-space extrema detection.
2. Keypoint localization.
3. Orientation assignment.
4. Generation of keypoint descriptors.

In [Lowe04, Estrada04], detection stages for SIFT features were described as the following:

Interest points for SIFT features stand for local extrema of difference-of-Gaussian filters at different scales. Given a Gaussian-blurred image:

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \qquad (2.1)$$

where

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \qquad (2.2)$$

is a variable scale Gaussian, the result of convolving an image with a difference-of-Gaussian (DoG) filter: $G(x,y,k\sigma) - G(x,y,\sigma)$ is given by

$$D(x,y,\sigma) = L(x,y,k\sigma) - L(x,y,\sigma) \qquad (2.3)$$

Which is the difference of the Gaussian-blurred images at scales σ and $k\sigma$.

The first stage to detect interest points is the convolution of the image with Gaussian filters at different scales, and the production of difference-of- Gaussian (DoG) images from the difference of adjacent blurred images.

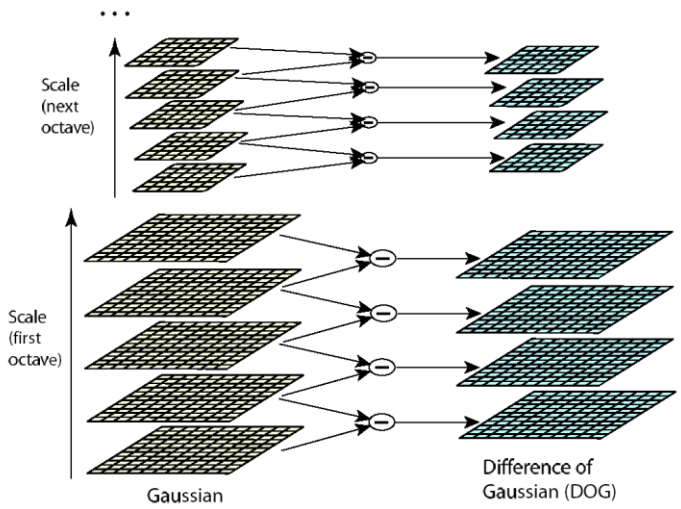


Figure 2.4: Diagram showing the blurred images at different scales, and the computation of the difference-of-Gaussian (DoG) images [Lowe04].

An efficient approach to construction of $D(x, y, \sigma)$ is shown in Figure 2.4. The convolved images were grouped by octaves (an octave stands for doubling the value of σ), and the value of k was chosen so that a fixed number of blurred images per octave was obtained. This also ensures that the same number of difference-of-Gaussian images per octave was obtained.

Note: The difference-of-Gaussian (DoG) filter gives an approximation to the scale-normalized Laplacian of Gaussian $\sigma^2 \nabla^2 G$. The difference-of-Gaussian filter is in effect a tunable bandpass filter.

Interest points (called keypoints in the SIFT framework) are recognized as local maxima or minima of the difference-of-Gaussian (DoG) images across scales. Every pixel in the DoG images is compared to its 8 neighbours at the same scale, plus the 9 corresponding neighbours at neighbouring scales as shown in Figure 2.5. If the pixel is a local maximum or minimum, it is chosen as a candidate keypoint. For every candidate keypoint:

- Interpolation of nearby data is used to determine its position accurately.
- Keypoints with low contrast are eliminated.
- Responses along edges are removed.
- The keypoint is assigned an orientation.

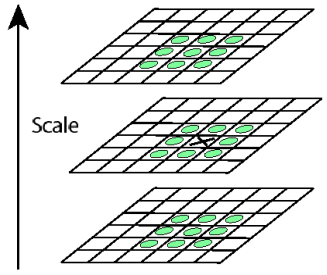


Figure 2.5: Local extrema detection, the pixel marked × is compared against its 26 neighbours in a 3 × 3 × 3 neighbourhood that spans adjacent DoG images [Lowe04].

Figure 2.6 illustrates the computation of the keypoint descriptor. In order to determine the keypoint orientation, a gradient orientation histogram is computed in the neighbourhood of the keypoint (using the Gaussian image at the closest scale to the keypoint's scale). The contribution of each neighbouring pixel is weighted by the gradient magnitude and a Gaussian window with a σ that is 1:5 times the scale of the keypoint [Lowe04].

Peaks in the histogram correspond to dominant orientations. A separate keypoint is created for the direction corresponding to the histogram maximum, and any other direction within 80% of the maximum value. All of the properties of the keypoint are measured relative to the keypoint orientation, and this provides invariance to rotation [Lowe04].

Once a keypoint orientation has been chosen, the feature descriptor is computed as a set of orientation histograms on 4 × 4 pixel neighborhoods. The orientation histograms are relative to the keypoint orientation, and the orientation data comes from the Gaussian image closest in scale to the keypoint's scale. Just like before, the contribution of each pixel is weighted by the gradient magnitude, and by a Gaussian with σ 1:5 times the scale of the keypoint [Lowe04]. This is illustrated with a circular window on the left side of Figure 2.6.

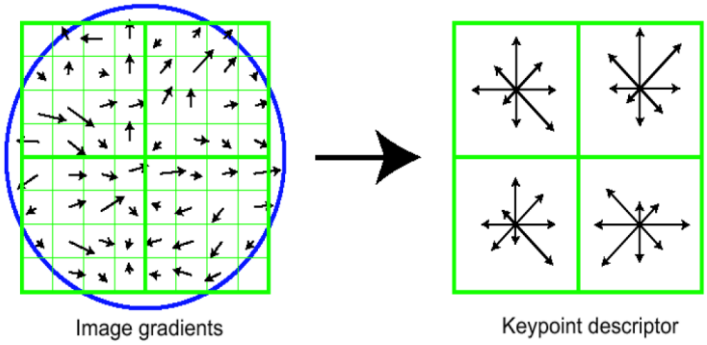


Figure 2.6: SIFT feature descriptor [Lowe04].

Histograms contain 8 bins each, and each descriptor contains an array of 4 histograms around the keypoint. This leads to a SIFT feature vector with $4 \times 4 \times 8 = 128$ elements. This vector is normalized to improve invariance to variations in illumination [Lowe04].

It has been found that the keypoints provide an efficient image representation for tasks varying from image classification to object recognition. Keypoints are salient image patches that include rich local information of an image. Keypoints are detected by robust feature detection methods such as SIFT [Lowe04], its variant Principal Component Analysis (PCA)-SIFT [Ke04], and Speeded Up Robust Features (SURF) [Bay08]. In [Lowe04], SIFT was used for extracting robust features from images that are invariant to scale and rotation. Then, it was broadly applied in image mosaic, recognition, and retrieval. Then, in [Ke04], PCA was used to normalize a gradient patch instead of histograms. It turned out that PCA-SIFT based local descriptors were also distinctive and invariant to scale and rotation. In [Bay08], robust features (SURF) were speeded up, and integral images were used for image convolutions and a Fast-Hessian detector. Experiments showed that it was remarkably fast and worked properly. In [Juan09], many experiments were conducted to evaluate performance of SIFT, PCA-SIFT, and SURF. Table 2.1 summarizes their results.

Table 2.1: Performance of robust feature detection methods: SIFT, PCA-SIFT & SURF [Juan09]

Method	Time	Scale	Rotation	Illumination
SIFT	Good	Best	Best	Common
PCA-SIFT	Good	Common	Good	Good
SURF	Best	Good	Common	Best

Table 2.1 shows that SIFT is fast for low resolution images and invariant to scale and rotation and partially to illumination changes. SURF is the fastest and has good performance like SIFT, but it is not invariant to rotation changes. PCA-SIFT has advantages with respect to illumination changes, while not in scale changes.

In Chapter 4, the hand gesture recognition system using SIFT feature will be proposed. Since the features, which represent the detected hand postures only, were extracted in real-time using the SIFT algorithm, and were invariant to scale and orientation, and the training images had been captured under different lighting conditions, our system has real-time performance and is robust against scale, rotation, illumination changes and cluttered backgrounds.

2.3.7 Bag-of-Features

A Bag of Features technique is one that represents images as orderless groups of local features. Keypoint features extracted from SIFT algorithm can be used in their raw format for direct image matching [Zhao06], or vector-quantized keypoint features into a representation like the bag-of-words representation of text documents. Since SIFT features are too high dimensionality to be used efficiently, we will propose to solve this problem in Chapter 4 by the bag-of-features approach [Lazebn06, Jiang07] to reduce the dimensionality of the feature space. Many others have used this vector-quantized, or bag-of-features representation, for image classification [Zhao06, Lazebn06, Jiang07].

Recently, bag-of-features representations have shown outstanding performance for action recognition [Jhuang07, Niebles08, Schüldt04]. They permit the recognition of a rich set of actions ranging from basic periodic motion (waving, running) to interactions (kissing, shaking hands) [Gilbert09, Laptev08, Marsza09, Yeffet09]. However, "bag-of- features" approaches exclusively rely on the dense local motion features. They do not have the relations between the features in the spatial and the temporal domains. These are significant correlation, which are useful for recognition. "Bag-of- features" has been expanded to include the spatial relation in the context of object categorization [Savare06, Agarwal06, Marsza06, Grauman05, Lazebn06]. The pyramid match kernel [Grauman05] utilized the weighted multi resolution histogram intersection as a kernel function for classification with groups of image features. Moreover, the spatial pyramid method [Lazebn06] introduced a spatial pyramid structure to discover approximate correspondence at different levels between two sets. In [Zhao08], the spatiotemporal pyramid was employed as a representation and at the same time integrated the spatiotemporal relation among visual features with their appearance information. In [Dollár05], a spatio-temporal interest point detector was proposed based on 1D Gabor filters. In [Willems08], a detector was proposed based on the determinant of the space-time Hessian matrix. In [Junejo08], local features were calculated from temporal self-similarity matrices of actions. In [Kläser08], a local space-time descriptor was proposed based on space-time gradients.

There are extensions for "bag-of-words" in the context of activity recognition [Ikizler07, Wang07, Wong07]. In [Ikizler07], spatial orientation information was captured in the local features. In [Wang07, Wong07], the latent semantic model was used to discover the activity types as topics in the hidden layer between the visual features and the video sequence. Different

from them, our approach in Chapter 4 uses spatiotemporal (space-time) correlation between every two consecutive frames in a video sequence in terms of the transition among recognized postures and their locations to develop our system into dynamic hand gesture recognition. This part of our contributions will be discussed thoroughly in Chapter 4.

2.3.8 Principal Component Analysis (PCA)

Feature extraction converts the data in the high-dimensional space into a low-dimensional space. The data transformation may be linear such as principal component analysis (PCA). The principal linear method for dimensionality reduction is the principal component analysis, which carries out a linear mapping of the information to a lower dimensional space to maximize variance of the information in the low-dimensional representation [Turk91]. The correlation matrix of the data is built, and the eigenvectors on this matrix are calculated. The eigenvectors that are related to the largest eigenvalues (the principal components) will be utilized to reconstruct a large fraction of the variance of the original information. Furthermore, the first few largest eigenvectors can usually be interpreted in terms of the large-scale physical behaviour of the system. The original space with dimension of the number of points has been decreased (with information loss, but ideally keeping the most significant variance) to the space spanned by a few eigenvectors.

PCA was first used in recognition in [Sirovi87] and later expanded in [Turk91] and [Murase95]. The PCA approach begins with a training stage, in which a set of training images of similar content is processed. Usually, the intensity values of each image are values of a 1D vector, whose dimensionality is equal to the number of pixels in the image. All the training and testing images have to be equal size. For each such set, some basis vectors are built to represent any of the training images in the set. In the case of gesture recognition as we will discuss in Chapter 5, the training set includes images of hands for different postures alone without any other objects in the background. In this way, the PCA features will represent the hand postures only. This procedure is done for every hand posture in the vocabulary. The system will match these PCA features later with PCA features in the testing stage to recognize a hand posture. A problem of eigenspace reconstruction approaches is that they are not invariant to image transformations such as translation, scaling, and rotation. We will discuss in Chapter 5 how this problem is solved by making our system invariant against translation, scaling, and rotation.

2.3.9 The Viola-Jones Algorithm

The object detection algorithm presented by Viola and Jones [Viola04] is one of the greatest achievements in computer vision technology to date. It was designed to be real-time speed and reliable for many objects in many difficult conditions. Viola and Jones designed their work on a novel image representation named Integral Images, which permits the detector of features to detect them very rapidly. In Chapter 3, we will discuss how to use the Viola and Jones algorithm in our approach to detect the face first and then subtract it to detect hand postures only.

The integral image at location (x, y) in Figure 2.7 includes the sum of the pixels values above and to the left of (x, y):

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x,y) \qquad (2.4)$$

where $ii(x,y)$ is the integral image and $i(x,y)$ is the original image as shown in Figure 2.7.

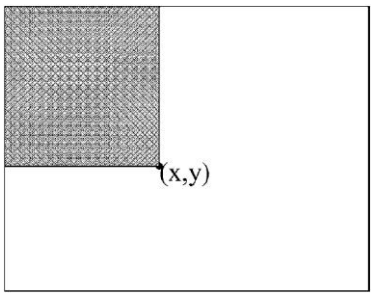


Figure 2.7: The value of the integral image at point (x; y) is the sum of all the pixels above and to the left [Viola04].

Using the following pair of repetitions:

$$s(x,y) = s(x,y \ldots 1) + i(x,y) \qquad (2.5)$$

$$ii(x,y) = ii(x \ldots 1,y) + s(x,y) \qquad (2.6)$$

where $s(x,y)$ is the cumulative row sum, $s(x, \ldots 1) = 0$, and $ii(\ldots 1, y) = 0$. The integral image can be calculated in one pass over the original image.

The features used in the Viola & Jones algorithm are known as Haar-like features or rectangular Haar-like features. Their name is derived from the similarities they share with the Haar wavelets basis functions [Chui92]. Usually, Haar-like features exploit simple patterns of color and luminosity changes in the image decreasing the effect of noise.

These features are calculated from the image pixels by scanning the image with particular rectangular patches that are divided into white and black sub rectangles as shown in Figure 2.8.

The feature at every point is calculated by subtracting the weighted sum of the pixels in the white areas from the weighted sum of the pixels in the dark areas.

Using the integral image representation, the calculation of that difference of sums appears like an ordinary table look-up process. Only four array references to the Integral Image are required to calculate any of the required sums.

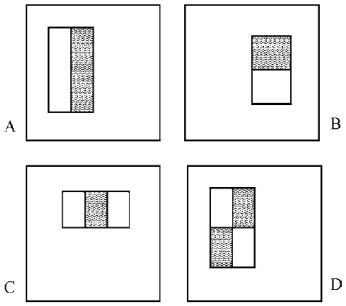


Figure 2.8: Original Haar-like features used by Viola and Jones [Viola04].

In [Lienhart02], the original set of rectangular features was expanded to contain tilted Haar-like features (as shown in Figure 2.9), which are fast to be calculated like the original ones, but help to enhance performance. In [Messom06], the result was generalized using the concept of rotated Integral Images.

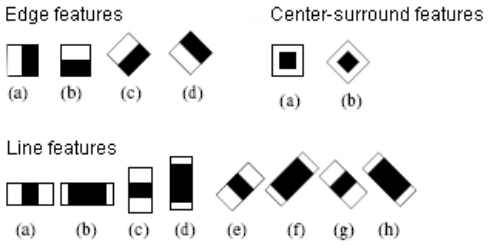


Figure 2.9: The extended Set of Haar-like features.

In [Barczak05, Chen07], Haar like features were applied for hand detection. Haar like features concentrate more on the information within a certain area of the image rather than each single pixel. To enhance classification accuracy and attain real-time performance, the AdaBoost learning algorithm, which can adaptively choose the best features in each step and join them into a strong classifier, can be used. In [Kölsch04b], view-specific hand posture detection with the Viola-Jones algorithm was explored. A frequency analysis-based technique was introduced for instantaneous evaluation of the "detectability" of various hand postures without the need for exhaustive training. Experimental results demonstrated that the classification accuracy improves

with a more expressive frequency-based feature types such as a closed hand palm. In [Kölsch04c], the in-plane rotational robustness of the Viola-Jones method was studied for hand detection. The results showed that hand detection with the Viola-Jones detector can be accomplished with about 15 degree in-plane rotations compared to 30 degree in-plane rotation to detect the face.

2.3.10 Haar-Wavelet

The Haar-Wavelet approach is utilized to extract image features by decomposing every pixel value of the image. In the scanning procedure of an image, the image features being scanned are compared to the images in the database having similar features under certain threshold. In [Chung09], the hand gestures were recognized using Haar-Wavelet approach.

2.4 Classification

The extracted features are then exposed to different classifiers, from generic Support Vector Machines (SVMs) invented by Vladimir Vapnik [Chen07b] to highly customized shape classifiers such as in [Yin06]. Some features carry out classification implicitly such as the Lucas-Kanade-based tracker, which removes "unreliable" patches, and Camshift [Bradski98], which determines a decision boundary in space and color histograms.

Classification can be integrated with feature extraction such as the boosting method including a combination of weak detectors [Viola04]. Other approaches include a distinct translation step into feature space and subsequent classification. To illustrate, consider tracking a hand gesture, with its spatial location over time providing feature vector and a hidden Markov model classifying hand trajectory into different temporal/ dynamic gestures [Nickel07, Rausch02, Schlöm08]. Hand gesture classification approaches can be divided into two main categories: rule based approaches and machine learning based approaches.

2.4.1 Rule Based Approaches

In these methods, features of the gesture are compared to manually encoded rules. If any of the features or feature sets agree with a rule, the related gesture will be provided as output. In [Cutler98], a rule based technique was employed to recognize an action depending on a set of conditions in their view based approach to gesture recognition.

2.4.2 Machine Learning Based Approaches

Machine learning based methods use a set of exemplars to deduce models of gestures. The rule based methods rely on the capability of persons for discovering rules to classify the gestures. Learning based methods are other solutions to this problem when discovering that rules between features is not applicable. In these methods, mappings between high dimensional feature sets and gestures are carried out by machine learning methods. The most well known approach for these methods is using hidden markov models (HMMs) in which gestures are dealt with as the output of a stochastic process. Much work in [Nair02, Marcel00, Starner97] has concentrated on HMMs for gesture recognition.

In a machine learning based methods, two sets are needed: a training set and a test set. A training set is used by an automatic classifier for learning the differentiating features of gestures, and a test set is used to test the accuracy of the automatic classifier. The machine learning based method concentrates on optimising either a group of parameter values with regard to a group of features or a group of induced rules with regard to a group of attribute-value pairs.

One of the main contributions of our work as will be discussed in Chapter 4 is the innovative way in which machine learning algorithm, which is multiclass SVM, is integrated into the bag-of-features model. Machine learning is also used for posture and gesture classification. Therefore, this section is devoted for discussing the literature regarding machine learning theory and finally different methods such as artificial neural networks (ANNs) and SVM.

Machine learning methods have been used effectively in many applications such as internet search engines, medical diagnoses, credit card fraud detection, bioinformatics, handwritten recognition, speech recognition, classification of DNA sequences, stock market analysis.

The main idea of machine learning is to make computers capable of learning patterns from available information and then later recognize these patterns.

The theory of perceptron learning and artificial neural networks (ANN) [Anders95] gave the first successful machine learning paradigm, which is still one of the most employed today. Two of the major benefits of ANNs are the existence of a well tested implementation and a relatively quick classification speed in comparison to training time. ANNs have been employed to gesture recognition frequently in the last few decades [Kim96, Liang98, Yuan05, Malric08].

Support vector machines (SVMs) are the most notorious product of the modern statistical learning theory started in [Vapnik98]. Except a few parameters to be set, a great advantage of SVMs over ANNs is that there are no concerns about the definition of neuron topology like the case with ANNs. The SVM learning procedure will discover an optimal number of support vectors, in terms of structural risk. SVM had been assessed in a lot of applications in late years. It had been found that SVM outperform and generalize better than ANNs.

The Hidden Markov Model (HMM) is possibly the most well known machine learning tool used in gesture recognition literature [Iwai99, Stenger01b, Fei04, Rajko07]. An HMM used in dynamic gesture recognition provides many advantages compared with other classifiers while on the contrary to static posture recognition. This is due to temporal gesture segmentation and recognition that happen simultaneously, and HMMs can be trained by a number of training samples like Neural Networks [Eickel98]. The first well known application of HMM technology was speech recognition [Rabiner86, Juang91]. Other machine learning approaches have been also used in hand tracking and gesture recognition such as decision trees [Gutta96, Hang04], syntactic grammars [Chen08], Bayesian networks [Hua04, Ketteb05, ElSawah08], AdaBoost [Ong04], and Fuzzy models [Su00, Rao06, Habib06].

2.4.2.1 Support Vector Machine (SVM)

SVM, which was developed by Vladimir Vapnik, is a group of associated supervised learning techniques applied for classification and regression. We will use it in Chapter 4 to classify the detected hand postures. It carries out classification by creating an N-dimensional hyperplane that optimally divides the data into two groups. SVM classifiers are closely related to neural networks. Actually, SVM classifier using a sigmoid kernel function is the same as the two-layer, perceptron neural network.

In the SVM literature, a predictor variable is known as an attribute, and a transformed attribute that is applied to identify the hyperplane is known as a feature. The operation of selecting the most appropriate representation is called as feature selection. A group of features that describe one case is known as a vector. Therefore, the objective of SVM modeling is to find the optimal hyperplane that divides clusters of vectors in such a way that cases with one class of the target variable are on one side of the plane and cases with the other classes are on the other side of the plane. The vectors near the hyperplane are the support vectors.

Even though SVMs were initially intended as binary classifiers, other methods that deal with a multi-class problem as a single "all-together" optimization problem exist [Weston99], but are computationally much more costly than solving several binary problems.

A variety of approaches for decomposition of the multiclass problem into several binary problems using SVMs as binary classifiers have been proposed. In our implementation, multi-class SVM training and testing are performed using the library for SVM described in [Chang01]. This library supports multi-class classification and uses a one-against-one (OAO) approach for multi-class classification in SVM [Hsu02]. For the M-class problems (M being greater than 2), the OAO approach creates M(M-1)/2 two-class classifiers, using all the binary pair-wise combinations of the M classes. Each classifier is trained using the samples of the first class as positive examples and samples of the second class as negative examples. To combine these classifiers, the Max Wins method is used to find the resultant class by selecting the class voted by the majority of the classifiers [Friedm97].

2.4.2.2 Artificial Neural Networks (ANN)

Artificial neural networks (ANNs) offer a new group of nonlinear algorithms for feature extraction using hidden layers, and classification using multilayer perceptrons. An ANN is an information processing model that is motivated by the biological nervous systems, like the brain, to process information. The key element of this model is the novel configuration of the information processing system. It consists of a large number of highly interconnected processing elements (neurons) working in unity to solve particular problems. An ANN is structured for a particular application, such as pattern recognition and data classification using a learning process. In [Marcel99], a neural network model was used to recognize a hand posture in an image.

2.5 Hand Gesture Applications

In this section, we will discuss hand gesture applications [Wachs11]. We will start with medical systems and assistive technologies, which gives the person sterility required to prevent the spread of infection. The next application is gaming, which requires naturalness of the interface for the user experience. The final application is human-robot interaction, which have to

be natural and intuitive for the personal robot of the future. In addition, we will discuss the hand-gesture interaction interfaces for each type and discuss the associated design considerations.

2.5.1 Medical Systems and Assistive Technologies

Hand gestures can be utilized to interact with visualization screens, control medical devices, and assist handicapped persons as part of their rehabilitation treatment [Nishik03, Wachs08]. Some of these ideas have been used to enhance medical procedures and systems. In [Graetz04], methods were discussed to integrate hand gestures with physician-computer interfaces, illustrating a computer-vision system that allows doctors to carry out standard mouse functions such as pointer movement and button presses with hand gestures that meet with the "intuitiveness" requirement. In [Wachs11], a hand-gesture-tracking machine named Gestix was designed, which permits doctors to explore MRI images in an operating room via a natural interface to meet with both "come as you are" and "intuitiveness" as shown in Figure 2.10.

Figure 2.10: Surgeon using Gestix to browse medical images [Wachs11].

2.5.2 Gaming (Entertainment)

Computer games are a technologically promising and commercially rewarding field for innovative interfaces because of the entertaining nature of the interaction like our proposed systems in Chapter 4 and Chapter 5. People are willing to try new interface technologies because they have the chance to be immersed in a challenging game-like environment [Starner00]. In computer-vision-based hand gesture- controlled games, the system has to react fast to user gestures, must be robust and efficient in contrast to other applications such as inspection systems, which do not need real-time requirement. An additional issue is the "gesture spotting and

immersion syndrome," intending to recognize actual gestures from accidental motion. A method to solve this issue is by choosing a specific gesture to mark the "beginning" of a sequence of gestures.

In the Mind-Warping augmented reality fighting game [Starner00], where players interact with virtual opponents by hand gestures, gesture spotting is tackled by voice recognition. The initial and end of a temporal gesture are "marked" by voice. These games employ contextual data like gesture velocity and curvature to extract useful gestures from a video frames. In [Bannach07], gesture spotting was tackled by a sliding window and bottom-up method in a mixed-reality parking game. In [Schlöm08], accelerometer-based gesture recognition was utilized for interaction with a computer game. Gesture spotting in several Nintendo Wii games is solved by pushing a key on the WiiMote control by the "push to talk" analogy.

Intuitiveness is another significant requirement in gaming systems. The intuitive feature of hand-gesture vocabulary is tackled in a children's action game named QuiQui's Giant Bounce [Höysni05] where control gestures are chosen by a "Wizard of Oz" paradigm in which a user interacts with a computer application controlled by a hidden subject, with five full-body gestures detected by a low-cost USB Web camera.

"User adaptability and feedback" is the most necessity tackled in gaming applications. In gaming systems, players benefit from having to learn the gesture vocabularies used by the games. A training session is needed to train them with respect to how the gestures should be carried out, involving speed, trajectory, finger configuration, and body posture. While novices require time to learn the gesture-related tasks, skilled players move through the games at least as fast as if they were using a mechanical-control device or connected sensors [Brashe06, Pentla97].

2.5.2.1 Microsoft's Kinect

The apparition of reasonably inexpensive image and depth sensors has encouraged researchers in the area of object detection, tracking and gesture recognition. A popular device is the Kinect [Takaha09, Bramwe09, Lee09], shown in Figure 2.11, developed by Microsoft for the Xbox 360 video game console. It employs cameras and microphones to detect and track the motions of a user standing in front of it and permits voice control of games and video content. Using a webcam-style add-on peripheral for the Xbox 360 console, it permits players to control

and communicate with the Xbox 360 by a natural user interface using gestures and voice commands without the requirement to touch a game controller. The Kinect is capable of concurrently tracking up to six players, involving two active users for movement analysis with a feature extraction of 20 joints per user. Although the Kinect can detect hand gestures, it does not come with a hand gesture recognition application software [Bergh11]. Its software permits advanced human body tracking, facial recognition and voice recognition [Dudley09].

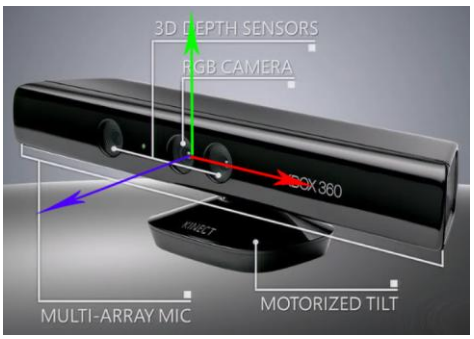


Figure 2.11: The Kinect device. The z–axis is pointing out of the camera [Frati2011]

The block-diagram of Kinect is shown in Figure 2.12. It has an infrared camera and a PrimeSense sensor to calculate the depth of the object, and an RGB camera to capture frames. The depth images and RGB image of the object could be got simultaneously. This 3D scanner system named Light Coding uses a variant of image-based 3D reconstruction [Raheja11].

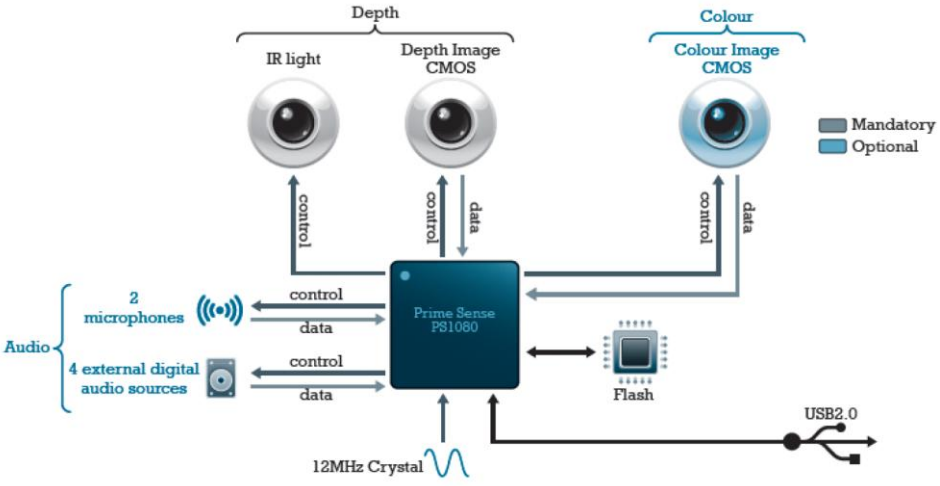


Figure 2.12: Kinect block-diagram [Raheja11]

Despite many achievements in using the Kinect sensor for face recognition and the human body tracking, it is still a challenge to employ Kinect for hand gesture recognition

because of the low-resolution of the Kinect depth map of only 640×480 pixel [Ren11]. While Kinect can track a large object such as the human body, it has problems when detecting and recognizing smaller objects such as a complex articulated human hand which occupies a very small region of the image [Ren11]. As a result, the segmentation of the hand is inherently imprecise, and this may considerably affect its recognition.

In [Bergh11], the Kinect, a depth sensor combined with color camera are used to detect hand gestures, which are then translated into interaction commands and 3D pointing directions. A specialized Kinect-based system, for hand detection, shown in Figure 2.13, is described in [Ren11b]. The system uses both the depth and color information from the Kinect sensor to detect the shape of the hand. Besides, providing robustness to the input changes or distortions due to the low resolution of Kinect sensor, it produces a shape distance metric, named Finger-Earth Mover's Distance (FEMD) for hand gesture recognition. Another Kinect based technique was proposed in [Raheja11] for fingertip and centre of palm detection for both hands in the 3D reference frame. The two hands are recognized using the depth vector and middle of palms and a distance transformation on inverse image. A Kinect-based method is presented in [Frati11] allowing to compensate for the lack of position sensing in wearable haptics. The location of the fingertips is calculated by a hand tracker specially optimized for the Kinect. The rendering method calculates the contact forces for wearable haptic display.

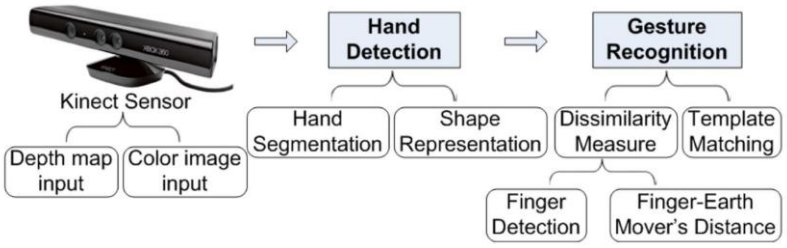


Figure 2.13: Kinect-based hand detection system [Ren11b]

2.5.3 Human-Robot Interaction

Another traditional topic of application for gesture interaction systems is human-robot interaction with a huge number of published papers [Al-Amin06, Bhuiyan07, Brooks06, Hong06, Rao06]. Hand gesture recognition is a significant feature for fixed and portable robots [Korten96]. In [Kawara04], robotic manipulators and human users were integrated with hand gesture commands for recognizing four postures such as when a user points at an object, the

robot can detect it. In [Chen07b], a human-machine interaction system for game playing allows the visual recognition of three postures with different rotation angles and scales with 95% accuracy. In [Rogalla02], a robotic- assistant interaction application was designed by combining gesture recognition and voice to generate commands. Once the hand posture is detected, six gestures are trained using a hand contour as the major feature of each gesture. In [Nickel07], a human-robot system was designed to recognize dynamic gestures by tracking head and arm direction. The system employs Hidden Markov Model (HMM) to recognize trajectories of the detected hands. In [Yin06], a programming-by-demonstration method allows the robot to learn a human user's gestures. The system employs eight hand postures to control a hybrid service robot system. In [Calinon07], a programming-by-demonstration idea was also utilized to allow users to help the robot make gestures by kinesthetic teaching. The user trains the robot to make 10 dynamic gestures captured by sensors connected to the user's torso, upper- and lower-arm.

2.6 Summary

This chapter presents a literature review of hand posture, hand gesture, and hand gesture recognition techniques, which includes feature extraction, and classification. According to this survey, the existing methods for hand gesture recognition systems can be classified into two classes, namely the glove-based and respectively the vision-based methods. When data gloves are used to track hand gestures, the user has to wear burdensome and expensive data gloves to track hand and finger movements. Vision-based methods can detect, track and recognize hand gestures more naturally and efficiently. However, they are sensitive to operating environment parameters such as users's position, orientation and scale, background texture, color, and lighting conditions. Vision based methods could be classified into 3D hand model-based methods and appearance-based methods.

The 3D hand model-based techniques provide a rich description that permits recognizing a wide class of hand gestures. However, they require a massive image database in order to deal with the entire characteristic shapes under several views. Another drawback is the difficulty of feature extraction algorithms to deal with singularities that occur from unclear views. Appearance-based techniques extract image features to model the visual appearance of the hand and compare these features with the extracted features from the video frames. They offer better real-time performance because of the easier 2D image features that are used.

Chapter 3

Hand Posture Detection

3.1 Introduction

Object detection is a challenging stage in most image processing applications. In order to analyze and extract relevant data about an object of interest from an image, one needs to first detect that object in the image. Hand detection refers to finding the place and size of hand within a sequence of images. It is an interesting computer vision problem and has many applications, such as sign language recognition, computer games, and human computer interaction.

This chapter will propose hand detection and position recovery in a sequence of images, and then the recognition of the gestures made by hand as described further in Chapter 4 and 5. A segmentation algorithm is used to separate the detected hand area from the image background, before processing it in the next, tracking and recognition, stages.

Skin color [Bretzn02, Mckenna04, Imagawa00] is a significant feature used for hand detection and tracking. However, color-based methods face the challenge of removing other objects with similar color such as face and human arm. To solve this problem, we proposed a new technique in [Dardas11] to detect hand postures only using face detection and subtraction, skin detection, and hand postures contours detection and comparison algorithm. The face was subtracted because the skin detection will detect the face and the face's contours are very close to the fist hand gesture contours. We used Viola-Jones method [Viola04] to detect the face and this method is considered the fastest and most accurate learning-based method. The detected face was subtracted by replacing the face area with a black circle. After subtracting the face, we detected the skin area using the hue, saturation, value (HSV) color model since it has real-time performance and it is robust against rotations, scaling and lighting conditions. Then, the contours of skin area were compared with all the loaded hand posture template contours to eliminate other skin like objects existing in the image. The hand posture area only was saved in a small image. The small image that contains the detected hand posture only will be used for recognition in our proposed systems in Chapter 4 and 5. In Chapter 4, the SIFT algorithm [Lowe04], discussed in

section 2.3.6, will be used for extracting the keypoints of the small image (50×50 pixels) for hand gesture recognition. In Chapter 5, the PCA will be used for extracting the features of the small image (160×120 pixels) for hand gesture recognition.

3.2 Hand Detection Approaches

Several approaches for hand detection have been introduced in the literature that employ different visual features and, in many cases, their combination. These features are motion, skin color, shape, and 3D models of hands. Hand detection approaches were discussed in [Zabulis09] and will be discussed in this section. In [Cote06], a comparative revision on the performance of some hand segmentation methods was discussed.

3.2.1 Color

Skin color detection has been employed in many hand detection applications. A main decision towards giving a model of skin color is the choice of the color space to be utilized. A number of color spaces have been introduced such as RGB, normalized RGB, HSV, YCrCb, and YUV. Color spaces that efficiently split the chromaticity from the luminance parts of color are typically regarded as preferable as we did in our approach by discarding the Value (V) component in HSV model. This is because of utilizing chromaticity-dependent components of color only. Thus, some degree of robustness against illumination variations can be attained.

Usually, skin color detection can be confused by background objects that have a skin color distribution similar to human skin color. Some research has found a solution to this problem by using background subtraction [Rehg94b, Gavrila96]. On the other hand, it was assumed that background subtraction normally depends on the camera system that does not move with respect to a static background. Another solution [Utsumi98, Blake99] has utilized the dynamic correction of background models and/or background compensation techniques.

3.2.2 Shape

Hand shape features have been employed to detect it in frames. More information can be acquired by extracting the contours of objects in the frame. If the contour is accurately detected, it provides good representation of the hand shape which is not directly related to viewpoint, skin color, and illumination. However, the 2D shape extraction can be obstructed by occlusions or

degenerate viewpoints. Usually, contour extraction based on edge detection uses a large number of edges belonging to the hand image area but also to unrelated background objects. Consequently, sophisticated post-processing techniques are needed to enhance the performance of this method such as our approach in [Dardas11] by combining skin color detection with contours detection and comparison after face subtraction.

Some methods exploit the particular morphology of the hand and try to locate it using characteristic hand shape features like fingertips. The method proposed by [Argyros06, Maggio95] uses curvature as a sign to fingertip detection. A second method that has been utilized in fingertip detection is template matching. Templates can be images of fingertips [Crowley95] or fingers [Rehg95] or generic 3D cylindrical models [Davis94]. These pattern matching methods can be improved by using additional image features such as contours [Rehg94b].

3.2.3 Learning Detectors from Pixel Values

More recently, techniques that use a machine learning method named boosting have showed remarkably robust results in face detection and good results in hand detection [Chen07]. In [Viola01], an object detection method was proposed in which a weak classifier may be a simple detector that uses basic image block differences efficiently computed using an integral image. On the other hand, this technique may provide an excessive number of weak classifiers. The AdaBoost method, discussed in section 2.3.9, has a drawback because it does not consider the elimination of chosen weak classifiers that no longer participate in the detection procedure. Besides, there are some issues to detect the hand using the Viola-Jones method [Viola01, Viola04] related to rotation and cluttered background as it will be discussed in Chapter 4.

3.2.4 3D Model-Based Detection

One major benefit of a 3D model-Based approach is that it can allow for view-independent detection. The utilized 3D models should have sufficient degrees of freedom to adapt to the dimensions of the hand that exist in an image. Different models use different image features to build feature-model correspondences. Point and line features are utilized in kinematic hand models for recovering angles created at the joints of the hand [Wu01, Shimada98, Wu99b]. Hand postures are then evaluated based on the relations between the 3D model and the observed image features.

3.2.5 Motion

Motion is a dynamic feature employed by some methods for hand detection. Motion-based detection requires a highly controlled setup, and it assumes that the only movement in the image resulted from hand motion. In more recent applications, movement information is integrated with additional visual cues as discussed in section 2.3.4. In the case of static cameras, the issue of movement evaluation is solved with background maintenance and successive subtraction. This solution is employed in [Cutler98, Martin98] for identifying the hand from other skin-coloured objects, and for dealing with lighting conditions resulting from coloured lights. The difference in pixel intensity between two consecutive frames is close to zero for the background pixels. Moving objects are located by selecting and maintaining a suitable threshold.

3.3 Our Approach for Hand Detection

We propose a unified system for detection, segmentation and tracking of the hand in a gesture recognition system using a single webcam. Different from other approaches that utilize color gloves [Aran06, Tokatl05], our method is able to detect the bare hand posture by integrating two useful features: skin color detection and contour matching. Our proposed hand posture detection algorithm has real-time performance and is robust against rotations, scaling, a cluttered background, and lighting conditions. Section 4.4.2 shows the robustness of our proposed hand posture detection algorithm based on comparison with other approaches.

Detecting and tracking the human hand in a cluttered background will enhance the performance of hand gesture recognition systems used in Chapter 4 and 5 in terms of accuracy and speed. The extracted SIFT features used in Chapter 4 and extracted PCA features used in Chapter 5 of the small image will represent the detected hand posture only. In addition, we will not be confined with the frame resolution size captured from a webcam or video file, because the SIFT features and PCA features will be extracted from the small image that contains the detected hand posture area only. In this way, the speed and accuracy of recognition will be the same for any frame size captured from a webcam such as 640×480, 320×240 or 160×120 and the system will be also robust against cluttered background because we process the detected hand posture area only. The small image size that contains the detected hand posture area only has to be matched with the training images size of training stage as we will discuss the training and testing recognition systems stages in Chapter 4 and 5.

In order to detect the hand posture in the image, a four steps system was designed according to our approach [Dardas11] and as shown in Figure 3.1. First, the templates of hand postures were loaded before capturing images from a webcam or video file. Second, the face was detected using the Viola&Jones algorithm [Viola04] and then subtracted with a black circle. Third, the skin color locus for the image was extracted for the user's skin color after face subtraction. Then as the fourth step, the hand posture was detected by eliminating false positive skin pixels and identifying hand posture and other real skin color regions using contours matching with the loaded hand postures templates contours.

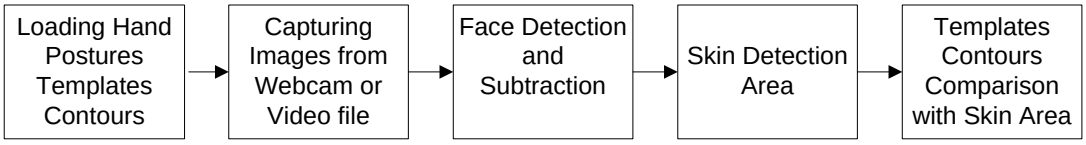


Figure 3.1: Hand posture detection stages.

3.3.1 Loading Templates of Hand Postures

For detecting hand gesture using skin color, there are different methods including skin color based methods. In our case, after detecting and subtracting the face, skin detection and a contour comparison algorithm were used to search for the hand and discard other skin coloured objects for every frame captured from a webcam or video file. Before capturing the frames from a webcam, we loaded the templates of hand gestures as shown in Figure 3.2: fist, index, little and palm to extract their contours and saved the four for comparison with the contours of the skin detected area of every frame. After detecting the skin area for every frame captured, we compared the contours of the detected areas with the previously loaded hand posture template contours to eliminate other skin like objects existing in the image. If the contour comparison of the detected skin area complies with any one of the stored hand postures contours, a small image will enclose the hand posture area only. As shown in Chapter 4, a small image (50×50 pixels) will be used to extract the keypoints using the SIFT algorithm for hand posture recognition. In Chapter 5, PCA will be used for extracting the features of the small image (160×120 pixels) to recognize the hand posture.

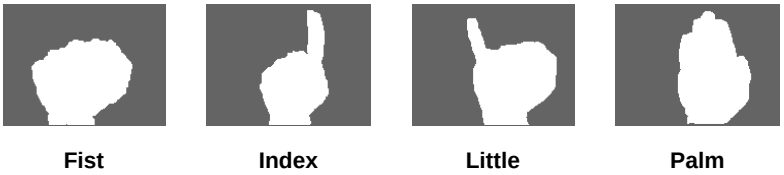


Figure 3.2: Templates of hand postures.

3.3.2 Face Detection and Subtraction

The skin detection and contours comparison algorithms used to detect the hand posture, can also be used to detect the face. To subtract the face area, we detected the face using the Viola and Jones method [Viola04] and then subtracted the face and replaced the face area with a black circle in each captured image frame before actually proceeding to hand gesture recovery. We are using the Viola and Jones algorithm [Viola04] which offers high accuracy and real-time performance.

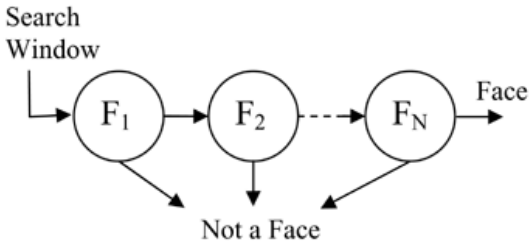


Figure 3.3: Cascade of classifiers.

We are using a cascade of classifiers. First, we load a statistical model, which is the XML file classifier for detecting frontal faces from the frames captured from a webcam during the testing stage. The XML file classifier, which is a cascade of classifiers working with Haar-like features, was trained with a few hundred sample views of the human face, called positive examples that were scaled to the same size such as 30×30 pixels and negative examples, which are random images of the same size. The term "cascade" in this classifier indicates that the resultant classifier includes numerous simpler classifiers that are applied later to an area of interest until at some stage the candidate is discarded for image regions least likely to be a face or all stages are accepted for those that represent a face as shown in Figure 3.3. These classifiers use simple rectangular features, called Haar-like features as in Figure 3.4.

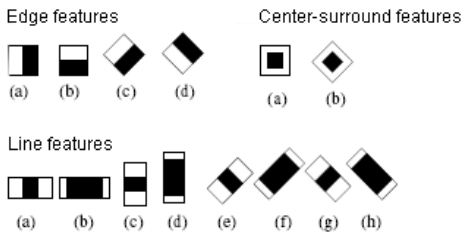


Figure 3.4: Set of Haar-like features.

The Haar feature used in a particular classifier is determined by its shape, position within the area of interest, and scale. The existence of a Haar feature is found by subtracting the average dark-region pixel value from the average light-region pixel value. If the difference is above a threshold, which is set during learning, that feature will be existed. The Haar-like features make use of the following information as distinctive features that can be used to detect the face as shown in Figure 3.5.:

- Eyes are dark (eyebrows and shadows).
- Cheeks and forehead are bright.
- Nose is bright.

Figure 3.5: Detecting a human face using Haar-like features.

After a classifier is trained, a set of features are extracted and distinctive features that can be used for recognizing the face are selected. The classifier can be applied to an area of interest in an input image. The classifier outputs a "1" if the region is probably showing the face, and "0" otherwise. In order to search for the face in the entire image, the search window can be moved across the image and check every pixel using the classifier. The classifier can be "resized" simply to allow finding the objects of interest in various scales, which is more efficient than resizing the image itself. Therefore, to locate an object of an unknown size in the image, the search process should be repeated several times at different sizes.

Once the face had been detected by the XML file classifier for every frame captured, we replace the detected face area with a black circle to remove the face from the skin area, ensuring that the skin detection is applied for detecting hand posture only as shown in Figure 3.6.

Figure 3.6: Face detection and subtraction.

3.3.3 Skin Detection

Skin detection is a useful technique for many computer vision applications such as face detection, recognition, tracking and facial expression extraction, or hand tracking and gesture recognition. There are well-known methods for skin color modeling and recognition that will allow to differentiate between skin and non-skin pixels based on their color. In order to get appropriate distinction between skin and non-skin regions, a color transformation is needed to separate luminance from chrominance [Zhu04].

The input images generally are in RGB format, which has the disadvantage of having components dependent on the lighting conditions. The confusion between skin and non-skin pixels can be decreased using color space transformation. Many skin detection methods use color components in other color spaces, such as HSV, YCbCr, TSL or YIQ to provide more robust parameter recovery under changes in lighting condition.

Experiments have shown that skin colors of individuals cluster closely in the color space for all people from different ethnicities, for example, color appearances in human faces and hands vary more in intensity than in chrominance [Jun08, Kelly08, Yang98]. Thus, removing the intensity V of the original color space and working in the chromatic color space (H,S) provides invariance against illumination conditions. In [Zhu04], it had been noted that discarding the Value (V) component and only using the Hue and Saturation components, can still permit for the detection 96.83% of the skin pixels.

In our implementation, we use the *hue, saturation, value* (HSV) color model since it has shown to be one of the most adapted to skin-color detection [Zarit99]. It is also compatible with the human color perception. Besides, it has real-time performance and it is more robust in cases

of rotations, scaling, cluttered background, and changes in lighting condition. Therefore, our proposed hand posture detection algorithm is real-time and robust against the mentioned previous changes. The other skin like objects existing in the image are eliminated by contour comparison with the loaded hand postures template contours.

The HSV color space is obtained by a non-linear transformation of the fundamental RGB color space. The transformation between RGB and HSV was described in [Ford98]. Hue (H) is a component that represents pure color such as pure yellow, orange or red, whereas saturation (S) provides a measure of the degree to which a pure color diluted by white light [Gonzal04]. Value (V) attempts to represent brightness along the grey axis such as white to black, but since brightness is subjective, it is thus difficult to measure [Gonzal04].

According to [Foley96] and Figure 3.7, *Hue* is measured in HSV color space by an angle with Red starting at 0 degrees, Green at 120 degrees and Blue at 240 degrees. The black line in the diagram at the lower left on the screen demonstrates the hue angle.

Saturation, S, is a ratio that ranges from 0.0 along the center line of the cone (the V axis) to 1 on the edge of the cone. *Value*, V, ranges from 0.0 (dark) to 1.0 (bright).

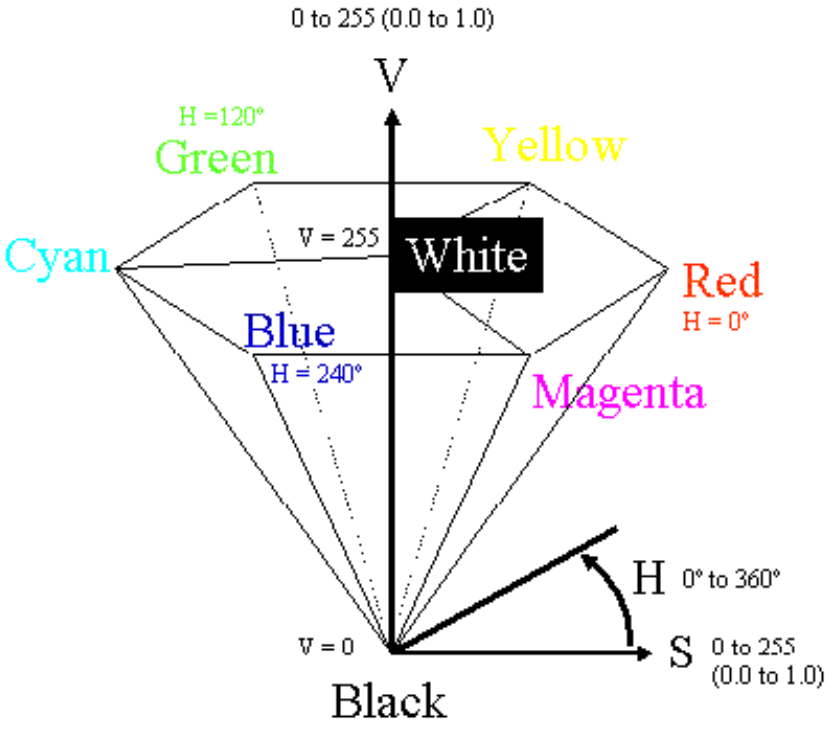


Figure 3.7: HSV color space [Foley96]

According to [Jun08], the HSV model can be derived from non-linear transformation from an RGB model according to the following equations.

$$H = \begin{cases} \theta, G \geq B \\ 2\pi - \theta, G < B \end{cases} \quad (3.1)$$

$$S = \frac{\max(R,G,B) - \min(R,G,B)}{\max(R,G,B)} \quad (3.2)$$

$$V = \frac{\max(R,G,B)}{255} \quad (3.3)$$

$$\theta = \arccos\left\{\frac{[(R-G)+(R-B)]/2}{[(R-G)^2 + (R-G)(G-B)]^{1/2}}\right\} \quad (3.4)$$

From a classification point of view, skin-color detection can be considered as a two class problem: skin-pixel vs non-skin-pixel classification. There are many known classification techniques such as thresholding, Gaussian classifier, and multilayer perceptron [Nallap07, Greens01, Phung01].

In our implementation, we used a thresholding method that allows for higher computation speed when compared with other techniques, given our real-time requirements. The basis of this thresholding classification is to find the range of two components H and S in the HSV model as we discarded the Value (V) component. Usually a pixel can be viewed as being a skin-pixel when the following threshold ranges are simultaneous satisfied: $0° < H < 20°$ and $75° < S < 190°$.

3.3.4 Contour Comparisons

Once the skin area is detected, the contours of the detected areas are recovered and then compared them with the contours of the hand posture templates. If the recovered contours are recognized as belonging to the hand posture contour templates, that area will be identified as a region of interest (by enclosing the detected hand posture with a rectangle) which will then be used for tracking the hand movements and saving the hand posture in a PGM-format small images as shown in Figure 3.7. These images will further be used to extract the features needed to recognize the hand postures in the testing stage as discussed in Chapters 4 and 5. The total time needed to detect the hand posture is around 15 milliseconds for every frame captured.

Figure 3.8: Small images of detected hand postures.

If there are two hand postures in the image, our system will alternate in detecting one of the two hands for every frame captured because the OpenCV function cvBoundingRect will enclose one rectangle only around the detected hand, which has the largest matching contours with the loaded hand posture templates contours. The single rectangle will enclose the detected hand posture for one frame and may enclose the other hand posture for the next frame if it has a larger matching contour.

3.4 Summary

In this chapter, we described the use of skin detection and the contour comparison algorithm to detect the hand posture. This approach can also be used to detect the face because the face has a skin color and its contours are similar to the hand fist posture contours. To eliminate the face area, we detected the face using the Viola and Jones method and then subtracted the face before applying the skin detection algorithm to detect the hand posture only by replacing the face area with a black circle for every frame captured.

Skin detection and the contour comparison algorithms were used for detecting the human hands and discarding other skin-coloured objects after face detection and subtraction. Before capturing the frames from a webcam, we load the templates of hand postures: fist, index, little, and palm to extract their contours, and save them for comparison with the contours of the skin area of every frame captured. After detecting the skin area using the HSV color model, we compared the contours of that area with the loaded hand posture template contours to eliminate other skin-like objects existing in the image. If the contours of the detected skin area comply with any one of the loaded hand posture templates contours, a small image will enclose the hand posture area only. The small image will be used for extracting features to recognize the detected hand posture. Our approach will be used in Chapter 4, 5, and 6 for hand posture detection before recognition.

Chapter 4

Hand Gesture Recognition Using Bag-of-Features and SVM Classifier Techniques

4.1 Introduction

Vision-based hand gesture recognition is a challenging task because of the complexity of hand gestures due to the high number of the degrees of freedom (DOF) of the human hand. Hand gesture recognition systems for practical applications have to meet tough requirements in terms of real-time performance, recognition accuracy, and robustness against transformations and a cluttered background. In order to meet these requirements, many gesture recognition systems use the help of coloured markers or data gloves to make the task easier [ElSawah08]. However, using markers and gloves sacrifices the user's convenience. In this thesis, we focus on the study of a more user-friendly bare hand gesture recognition system without the help of any markers and gloves.

The novel bare hand gesture recognition system proposed in this thesis uses the Scale Invariant Feature Transform (SIFT) features proposed by Lowe [Lowe04]. These visual features/keypoints allow for reliable matching between different views of the same object, image classification, and object recognition. The SIFT features/keypoints are invariant to scale, orientation, and partially invariant to illumination changes, and are highly distinctive of the image. However, SIFT features are too high dimensionality to be used efficiently. We propose to solve this problem by using the bag-of-features approach [Lazebn06, Jiang07] to reduce the dimensionality of the feature space.

In the training stage, hand gesture training images can be represented by sets of keypoint descriptors, but the number of these keypoints differs from image to image and they lack meaningful ordering. This creates difficulties for machine learning methods such as the multiclass Support Vector Machine (SVM) classifier that require feature vectors of fixed dimension as input. To address this problem, we use the bag-of-features approach, which has several steps. The first step is extracting the features/keypoints of hand posture training images

using the SIFT algorithm. The next step is using the vector quantization (VQ) method [Bosch07] to cluster the keypoint descriptors in their feature space into a large number of clusters using the K-means clustering algorithm, proposed by Lloyd [Lloyd82], and to encode each keypoint by the index of the cluster (codevector) to which it belongs. This VQ maps keypoints of every training image into a unified dimensional histogram vector after K-means clustering as shown in Figure 4.1. Finally, each cluster is considered as a visual word (codevector) that stands for a particular local pattern shared by the keypoints in that cluster. Therefore, the clustering algorithm constructs a visual word vocabulary (codebook) representing several local patterns in the training images. The size of the vocabulary is determined by the number of clusters (codebook size), which can vary from hundreds to tens of thousands. Each training image can be described as a "bag-of-words" vector by mapping the keypoints to a visual words vector. With the described feature representation, we can train a multiclass SVM classifier.

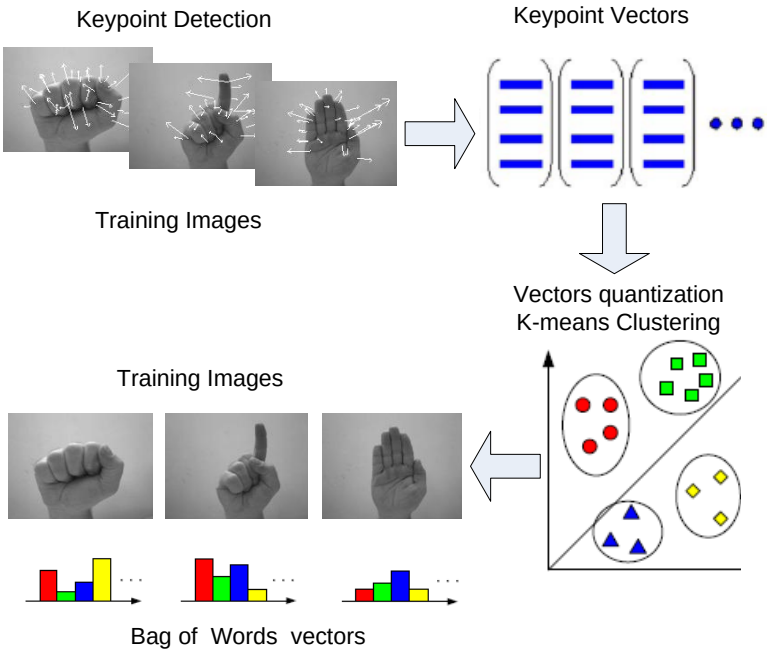


Figure 4.1: Generating bag-of-words.

After the training stage, the testing stage can be executed to recognize hand gestures in the captured webcam images regardless of their resolution. First, we substract the face and detect the hand using the skin detection, and contour comparison algorithms presented in Chapter 3 [Dardas11]. Then, a small image (50×50 pixels) that contains only the detected hand will be used to extract the keypoints in order to reduce processing time and increase the accuracy of

recognition. The extracted keypoints will represent the hand only and will be used as the input for the cluster and multiclass SVM classifier models, which were built in the training stage, to recognize the hand gesture. In this way, the system will be robust against a cluttered background. After hand posture detection and recognition, our system will build a grammar that generates gesture commands, which will be used in sections 4.7, 4.8, 4.9, and 4.10 to control a videogame.

This chapter is organized as follows: Section Two describes our proposed hand gesture recognition system in details, including the training stage to build the cluster and the SVM classifier models and the testing stage for recognition, Section Three presents experimental results, Section Four compares the performance of our approach with other approaches, Section Five discusses building a grammar for our system using a finite state machine (FSM), Section Six explains how our system generates gesture commands, Sections Seven, Eight, Nine, and Ten describe how the generated gesture commands of our system control an exertion videogame, and finally Section Eleven gives the conclusion of our approach.

4.2 System Overview

The proposed hand gesture recognition system consists of two stages: the offline training stage and the online testing stage, which is extended from our system in [Dardas10] by using face detection and subtraction and hand posture detection. The cluster and multiclass SVM classifier models will be built in the training stage, and will be used in the testing stage for recognizing hand gestures captured from a webcam. Three critical factors affect the accuracy of the system: the quality of the webcam in the training and testing stages, the number of the images used for training, and choosing the number of clusters to build the cluster model.

4.2.1 Training Stage

The training stage model is shown in Figure 4.2. Before building the bag-of-features model, we captured 100 training images for each hand posture, which are the fist, index, palm, and little finger postures, for 10 people with different scales, rotations and lighting conditions to increase the robustness of the multiclass SVM classifier and the cluster model. The system can also recognize any other posture, such as two, three, and five fingers as we will discuss in Section 4.3.2. All of the training images contain only hand postures without any other objects,

and the background has no texture or objects (white wall). This makes sure that all the keypoints extracted from training images using the SIFT algorithm will refer to the hand posture only.

SIFT allows for real-time performance for low resolution protable gray map (PGM) images [Poskan91]. The processing time for extracting keypoints using SIFT can be reduced when the image resolution is reduced and converted into PGM format as we will discuss in section 4.3.1.2 and according to Table 4.4. The image processing time is not too critical in the training stage as it is in the testing stage.

The training stage was conducted twice to build two multiclass SVM classifiers and two cluster models that will be used in the testing stage for two cases. The first case was without hand posture detection, while for the second case was with hand posture detection using our algorithm in Chapter 3.

For the first case of the testing stage that we did not use the hand detection algorithm, as we will discuss in section 4.3.1.1, the size of training images was reduced to 320×240 pixels and converted them into PGM format to coincide with the size of images captured from the video file in the testing stage. We repeated the training phase with another set of 160×120 pixels training images to build another cluster and multi-class SVM classifier models.

In the second case of the testing stage, our hand detection algorithm was used as we will discuss in section 4.3.1.2. The size of training images was reduced to 50×50 pixels and converted into PGM format to coincide with the size of the small image (50×50 pixels) that contains the detected hand posture for every frame captured from the video file in the testing stage.

The bag-of-features model was built using feature extraction, learning a "visual vocabulary" by k-means clustering, quantizing features using visual vocabulary, and finally representing images by frequencies of "visual words," as will be discussed further.

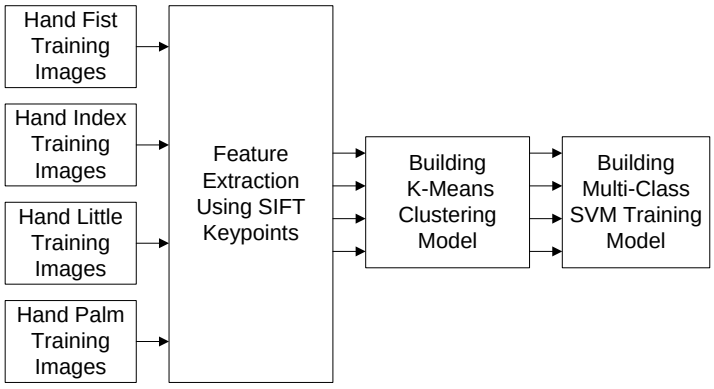
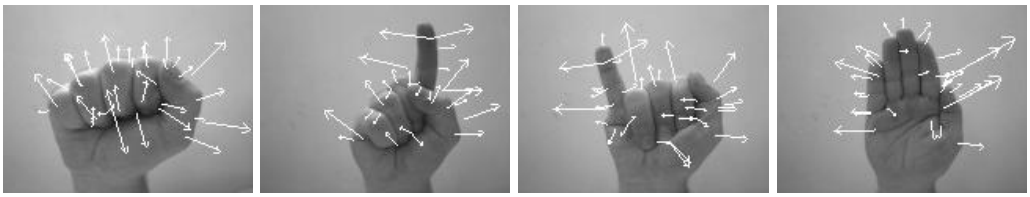


Figure 4.2: Training stage.

4.2.1.1 Features Extraction Using SIFT

Features based on the SIFT algorithm are invariant to scale and rotation and can be extracted in real-time for low resolution images. They are extracted in four stages. The first step finds the locations of potential interest points in the image by detecting the maxima and minima of a set of difference of Gaussian filters applied at different scales all over the image. Then, these locations are refined by eliminating points of low contrast. An orientation is then assigned to each keypoint based on local image features. Finally, a local feature descriptor is computed at each keypoint. This descriptor is based on the local image gradient transformed according to the orientation of the keypoint for providing orientation invariance. The size of the feature vector depends on the number of histograms and the number of bins in each histogram. In Lowe's original implementation [Lowe04] a 4-by-4 patch of histograms with 8 bins each is used, generating a 128-dimensional feature vector.

We use the SIFT algorithm to extract the keypoints (vectors) for each training image. Figure 4.3 shows some training images with their keypoints. The number of keypoints decreases as the hand moves away from the camera and increases when the hand moves closer to the camera, because the area of the hand increases. For the same distance from the camera, we notice that the palm gesture has the maximum number of keypoints as it has the largest area. We can increase the number of training images to train the system as we wish for all the hand postures for different people with different scales, orientations, and illumination conditions. The more training images used with different illumination conditions, the more accurate in building k-means cluster and SVM models since extracted features for training images using SIFT are invariant to scale, orientation, and partially to illumination changes [Lowe04]. Therefore, the time will increase for building the cluster model in the training stage [Dardas11]. However, this will not affect the testing stage speed because the cluster and SVM models will be built in the training stage and after that will be used in the testing stage. The time needed for building the cluster and SVM models in the training stage using 100 training images was around 3 hours. In our case we did not perform calibration by increasing the number of training images more than 100 images to build more accurate k-means cluster and SVM models because our proposed system achieved high classification accuracy more than 90% using 100 images as we will see the results in section 4.3.



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

Figure 4.3: Training images features (keypoints). (a) Fist with 35 features. (b) Index with 41 features. (c) Little finger with 38 features. (d) Palm with 75 features.

4.2.1.2 K-Means Clustering

Clustering divides a group into subgroups, called clusters, so that the elements in the same cluster are similar. It is implemented using an unsupervised learning algorithm and an ordinary method for statistical data analysis applied in several fields, such as machine learning, pattern recognition, image analysis, data mining, and bioinformatics.

Among many different types of clustering, we selected to use the k-means clustering algorithm [MacKay03]. The number of clusters (the codebook size) depends on the structure of the data. There will be a compromise for how to choose the vocabulary size or number of clusters. If it is too small, then each bag-of-words vector will not represent all of the keypoints extracted from its related image. If it is too large, it will lead to overfitting because of insufficient samples of the keypoints extracted from the training image.

During the training phase, when the training images contain only hand postures on a white background, the keypoints that are extracted will represent the hand posture only, and this will not exceed 75 keypoints for the palm posture, which has the largest number of keypoints. From this information, we know that the number of clusters must be larger than 75. Therefore, the training stage provides the minimum number of clusters that we can use. In the testing stage, the webcam will capture other objects besides the hand such as the face and background. There will be around 400 keypoints for all the objects in the image. We chose the value 750 as the number of clusters (visual vocabularies or codebook) to build our cluster model. This number provides the most accurate recognition rate by trying different values [Dardas11].

The first step in k-means clustering is to divide the vector space (128-dimensional feature vector) into k clusters. K-means clustering starts with k randomly located centroids (points in

space that represent the center of the cluster) and assigns every keypoint to the nearest one. After the assignment, the centroids (codevectors) are shifted to the average location of all the keypoints assigned to them, and assignments are redone. This procedure repeats until the assignments stop changing. Figure 4.4 shows this process in action for five keypoints: A, B, C, D, and E and two clusters.

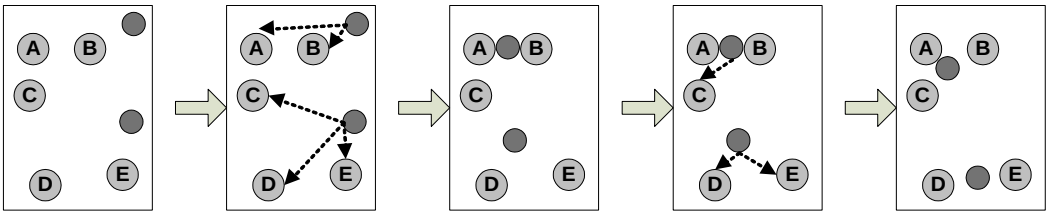


Figure 4.4: K-means clustering with two clusters.

Once this is done, each feature vector (keypoint) is assigned to one and only one cluster center that is in the nearest distance with respect to the Euclidean distance metric in the 128-dimensional feature vectors. The keypoints that are assigned to the same cluster center will be in the same subgroup so that after clustering, we have k disjoint subgroups of keypoints. Therefore, k-means clustering decreases the dimensionality for every training image with n keypoints (n×128) to 1×k, where k is the number of clusters.

The keypoint vectors for each training image will be used to build the cluster model using k-means clustering. The number of clusters (codebook) will represent the number of centroid in the cluster model. Finally, the cluster model will build codevectors equal to the number of clusters assigned (k), and each codevector will have 128 components, which is equal to the length of each keypoint. Then, the keypoints of each training image will be fed into the k-means clustering model for reducing its dimensionality into one bag-of-words vector with components equal to the number of clusters (k). In this way, each keypoint, extracted from a training image, will be represented by one component in the generated bag-of-words vector with a value equal to the index of the centroid in the cluster model with the nearest Euclidean distance. The generated bag-of-words vector, which represents the training image, will be grouped with all of the generated vectors of other training images that have the same hand posture, and labeled with the same number. This label will represent the class number. For example, label or class 1 for the fist training images, class 2 for the index training images, class 3 for the little finger training images, and class 4 for the palm training images.

4.2.1.3 Building the Training Classifier Using Multiclass SVM

All the keypoints recovered in the training images are mapped with its generated bag-of-words vector using k-means clustering. Then, all bag-of-words vectors with their related class or label numbers are fed into a multiclass SVM classifier to build a training classifier model. The SVM classifier will be used in the testing stage to classify the detected hand posture using the bag-of-words vector of the small image (50×50 pixels) that contains the detect hand posture only for every frame captured.

4.2.2 Testing Stage

Figure 4.5 shows the testing stage, which is extended from our previous work in [Dardas10] to detect the hand posture using face detection and subtraction, skin detection, and contour comparison of the detected area with the loaded hand posture template contours as discussed in Chapter 3. We are detecting and subtracting the face before detecting the hand because the face's contours are close to the fist hand posture contour. In order to eliminate other similar skin-colored objects within the image, the contours of the detected skin areas have to comply with the contours of the loaded hand posture template contours. The detected hand posture within a small image (50×50 pixels) is used to extract the keypoints. These keypoints will be fed into the cluster model to map them into a "bag-of-words" vector. Finally, this vector will be fed into multiclass SVM training classifier model for recognizing the hand gesture.

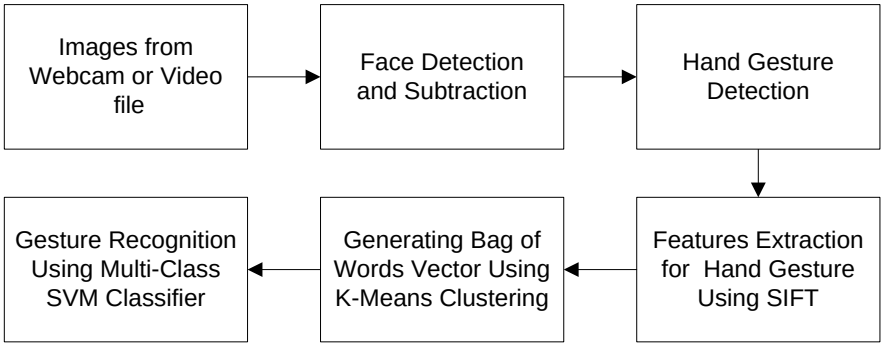


Figure 4.5: Testing Stage.

We converted the small image (50×50 pixels) that contains the detected hand posture only for every frame captured into a PGM format for reducing the time needed in extracting the keypoints. For every small PGM image, we extracted the keypoints (vectors) using the SIFT

algorithm. The keypoints were fed into the k-means clustering model that was built in the training stage to map all the keypoints with one generated vector (bag-of-words) with components (k), which is equal to the number of clusters (k) used in the training stage. Each feature vector in the keypoints will be represented by one component in the generated vector with the value equal to the index of the centroid in the cluster model with the nearest Euclidean distance. Finally, the generated bag-of-words vector will be fed into the multiclass SVM training classifier model that was built in the training stage for classifying and recognizing the hand gesture.

4.3 Experimental Results

In this section, we will discuss results of experiments conducted on our image data set and on a public image data set.

4.3.1 Results on Our Image Data Set

The four hand postures: the fist posture, the index posture, the little finger posture, and the palm posture were tested by the multiclass SVM training classifier. The camera used for recording video files in our experiment was a low-cost webcam that provides video capture with different resolutions such as 640×480, 320×240, and 160×120, at 15 frames-per second, which is adequate for real-time speed image recognition. In order to maintain the real-time performance during the testing stage, the frame rate for webcam capturing was set by using the following OpenCV function:

```
cvSetCaptureProperty(capture, CV_CAP_PROP_FPS, 15);
```

The frame rate per second (FPS) was set to 15.

We conducted the experiment to recognize hand postures for two cases in the testing stage. First case was without hand posture detection [Dardas10], and the other case was with hand posture detection [Dardas11] using the algorithm descriped in Chapter 3. There were two restrictions for the first case. The first restriction is that the resolution of video frames in the testing stage must not exceed 320×240 pixels since the SIFT algorithm is real-time for low resolution images only, while the second restriction was the use of a white wall background because the extracted keypoints will increase dramatically with a cluttered background.

Therefore, it becomes difficult to recognize the keypoints of hand gestures from the other keypoints of cluttered background.

For the second case, there were no restrictions in the testing stage to run it against the resolution of video frame and cluttered background because the region of interest will be determined by hand posture detection using the algorithm described in Chapter 3. The region of interest, which contains the hand posture only, will be saved in a small image and used for extracting the keypoints of hand postures only for every frame captured.

4.3.1.1 Hand Gesture Recognition without Face Subtraction and Hand Posture Detection

Ten video files with the resolution of 320×240 had been recorded for each hand posture, which are fist, index, little finger, and palm, using the Logitech webcam. The length of each video file was 100 images. The hand postures were recorded with different scales and rotations and without any object in the background. The 40 video files were recorded under different illuminations conditions. Fifty images of each video file were recorded with fluorescent light. While for others, an extra incandescent light bulb was installed. The test had been run for the 40 video files for evaluating the performance of the multiclass SVM classifier model for each posture.

Table 4.1: Performance of the multiclass SVM classifier for hand posture recognition without other objects (320×240 pixels frame size)

Gesture Name	Number of frames	Correct Recognition	Incorrect Recognition	Recognition Time (Second/frame)
Fist	1000	972	28	0.0633
Index	1000	989	11	0.0633
Little	1000	963	37	0.0633
Palm	1000	995	5	0.0633

Table 4.1 shows the performance of the multiclass SVM classifier for a number of postures under different scale, rotation, and illumination conditions, when there are no other objects in the image. We repeated the experiment with another ten video files for each hand posture with the resolution of 160×120. Only about 0.0492 second was required to recognize the hand posture for every frame with size of 160×120 pixels. The accuracy for 160×120 pixels

videos was the same as 320×240 pixels videos results. During the real-time testing with live input from each video file, there was no detectable pause and latency to recognize the hand postures. It appears that the system showed acceptable results in terms of accuracy and speed because the features of hand postures were extracted in real-time using the SIFT algorithm, and invariant to scale and orientation. In addition, since the training images had been captured under different lighting conditions, the multiclass SVM classifier is robust against illumination changes. The system wrote the posture name for each frame captured in the above left of the image as shown in Figure 4.6.

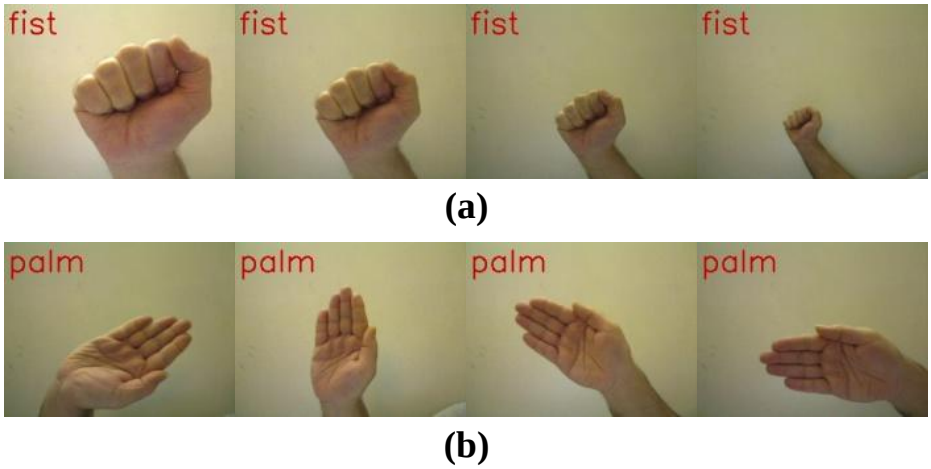


Figure 4.6: Hand posture recognition without any object having different (a) scale and (b) rotation.

Another set of ten video files with the resolution of 320×240 had been recorded for each hand posture, which are fist, index, little finger, and palm, using the webcam. The length of each video file was 100 images. The hand postures were recorded under different scale, rotation, and illuminations conditions and other objects in the background such as the face. The test was run for the 40 video files in order to evaluate the performance of the multiclass SVM classifier model for each posture. The trained multiclass SVM classifier showed an acceptable degree of robustness against different scale, rotation, and illuminations conditions according to Table 4.2.

Table 4.2 shows the performance of the multiclass SVM classifier for a number of postures under different scale, rotation, and illumination conditions, when there are other objects in the image. The recognition time increased because the extracted keypoints increased from other objects. We repeated the experiment with another ten video files for each hand posture with the resolution of 160×120. Only about 0.0512 s was required to recognize a hand posture for

every frame with the same accuracy as 320×240 pixels videos results. Figure 4.7 shows some correct samples for hand posture recognition using multiclass SVM classifier for the little and index gestures with testing against different scale, rotation, and illumination conditions.

Table 4.2: Performance of the multiclass SVM classifier for hand posture recognition with other objects (320×240 pixels frame size)

Gesture Name	Number of frames	Correct Recognition	Incorrect Recognition	Recognition Time (Second/frame)
Fist	1000	951	49	0.0699
Index	1000	979	21	0.0699
Little	1000	935	65	0.0699
Palm	1000	991	9	0.0699

(a)

(b)

Figure 4.7: Hand posture recognition with other objects in the image, having different (a) scale and (b) rotation

4.3.1.2 Hand Gesture Recognition with Face Subtraction and Hand Posture Detection

Ten video files with a resolution of 640×480 were recorded for each hand posture: fist, index, little finger, and palm using a commercial grade webcam. The length of each video file was 100 images. The hand gestures were recorded under different scales, rotations, and illumination conditions and a cluttered background. The test was run for the 40 of video files for evaluating the performance of the multiclass SVM classifier model for each gesture. The trained multiclass SVM classifier showed an acceptable degree of robustness against different scale, rotation, and illumination conditions, and cluttered background according to Table 4.3.

Table 4.3: Performance of the multiclass SVM classifier for cluttered background (640×480 pixels frame size)

Posture Name	Number of frames	Correct Recognition	Incorrect Recognition	Recognition Time (Second/frame)
Fist	1000	967	33	0.017
Index	1000	990	10	0.017
Little	1000	956	44	0.017
Palm	1000	994	6	0.017

Table 4.3 shows the performance of the multiclass SVM classifier for a number of postures under different scale, rotation, and illumination conditions, and a cluttered background. The recognition time did not increase in case of the cluttered background or with the increase in the video file resolution because the keypoints were extracted each time from the small image (50×50 pixels) that contains the detected hand posture only. We repeated the experiment with another ten video files for each hand posture with the resolution of 320×240 pixels; the same time was needed to recognize every frame with the same accuracy as 640×480 videos results because the keypoints were extracted in both cases from the small image that contained the hand posture.

As shown in Table 4.3, the recognition time for every frame had been reduced considerably to 0.017 s for any video resolution size used because the time spent for extracting the keypoints was only for the small image (50×50 pixels) that contains the detected hand gesture. Moreover, the recognition accuracy increased because the keypoints extracted represent the hand posture only. This is a notable improvement compared with the case without hand gesture detection [Dardas10], which requires more time because the keypoints were extracted for the whole objects in the image.

Figure 4.8 shows some correct samples for hand posture recognition using the multiclass SVM classifier for the four hand postures with testing against scale, rotation, illumination, and a cluttered background.

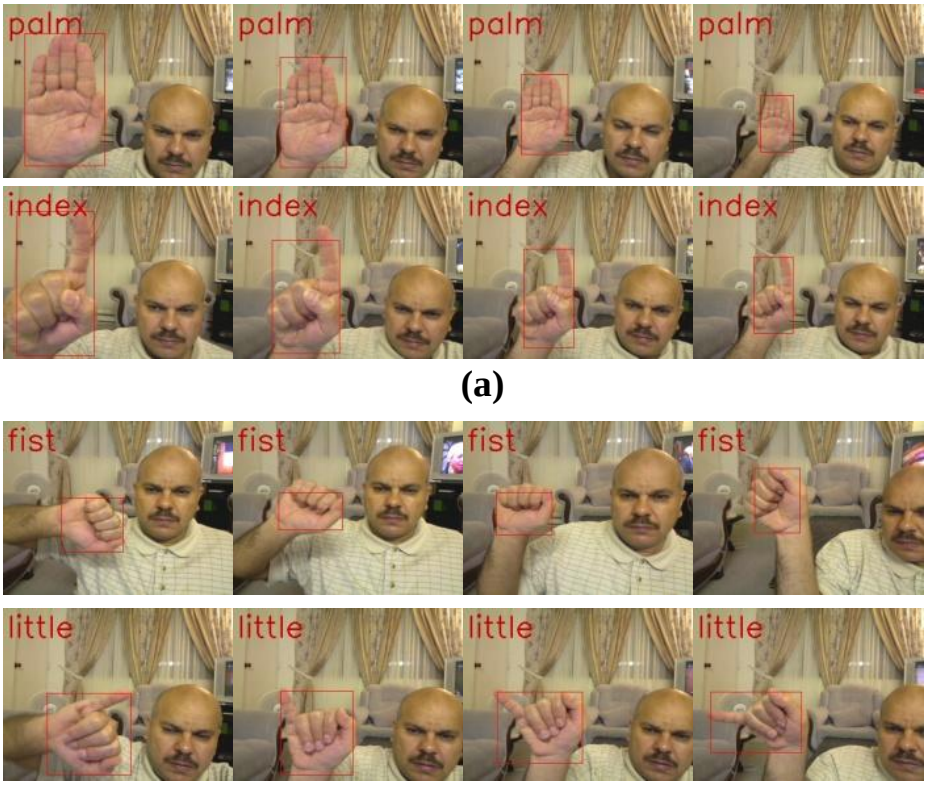


Figure 4.8: Hand posture detection and recognition with cluttered background having different (a) scale and (b) rotation

Table 4.4: Recognition time needed with cluttered background for different sizes of the small image containing the detected hand posture only

Size of small image (Pixels)	Recognition Time (Second/frame) (640×480 Pixels frame size)
50×50	0.017
80×80	0.017
90×90	0.034
130×130	0.034
140×140	0.049
170×170	0.049
180×180	0.063
210×210	0.063

We repeated the same experiment for the case of face subtraction and hand posture detection with different resolution sizes for the small image that captures the detected hand posture only. Table 4.4 shows the recognition time needed for every small image size, which contains the detected hand gesture only. We notice from Table 4.4 that if we increase the size of the small image that captures the detected hand only, the recognition time will increase. Therefore, we chose 50×50 pixels for the size of the small image to reduce the processing time.

To further test the robustness of our system, we added three new hand postures defined in ASL (American Sign Language) to our system, namely l (L), W (three), and Y as shown in Figure 4.9.

l (L) **W (three)** **Y**

Figure 4.9: The three new hand postures added to our system.

In the training stage, we captured 100 images with a size of 50×50 for each of these three new hand postures and added them to the four previous hand postures training images. The 700 training images for seven hand postures were used to build new cluster and multiclass SVM classifier models that can classify seven hand postures, which are fist, index, little, palm, l(L), W and Y. For every new hand posture, we recorded ten video files of 100 image length with different scales, rotations, and illuminations conditions and a cluttered background to test them with the previous 40 video files of fist, index, little, and palm postures.

The seven hand postures contours were loaded to detect hand posture for every frame of 70 video files after face subtraction and skin detection. Then, the test was run for the 70 video files for evaluating the performance of the new multiclass SVM classifier model for each hand posture. The multiclass SVM classifier produced excellent recognition results with recognition accuracy of 97% as shown in Table 4.5.

Figure 4.10 shows a set of images used for the detection and recognition of the l (L), W (three), and Y hand postures.

Table 4.5: Performance of the multiclass SVM classifier with cluttered background for seven hand postures (640×480 pixels frame size)

Posture Name	Number of frames	Correct Recognition	Incorrect Recognition	Recognition Time (Second/frame)
Fist	1000	964	36	0.017
Index	1000	988	12	0.017
Little	1000	951	49	0.017
Palm	1000	993	7	0.017
l (L)	1000	958	42	0.017
W(Three)	1000	975	25	0.017
Y	1000	962	38	0.017

Figure 4.10: Images used for the detection and recognition of the l (L), W (three), and Y hand postures.

4.3.2 Results on Public Image Data Set

The public database images used for testing in this chapter were from the Marcel database [Marcel99], which is a benchmark database for hand gesture recognition. This database contains (100×100 pixels) color images of six hand postures in ASL, namely the C, A (fist), Five, Point (index), B (palm), and V (two) as shown in Figure 4.11, performed by different

people against uniform and complex backgrounds. We had to add three new postures to our system, which are C, Five, and V (two) postures.

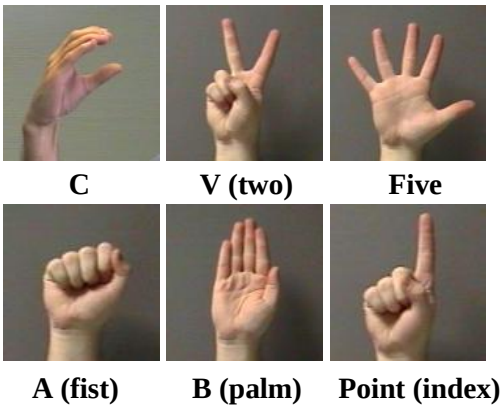


Figure 4.11: Hand postures used in Marcel database images.

In the training stage, we used 100 training images with a size of 50×50 pixels for every new hand posture, which are C, Five, and V (two), and added them to the three previous hand posture training images to build new cluster and multi-class SVM classifier models that can classify six postures. The 600 training images for six postures contain only hand postures on a white background. Therefore, the extracted keypoints will represent the hand posture only.

Table 4.6: Performance of the multiclass SVM classifier with public image data set (100×100 pixels image size)

Posture Name	Number of Images	Correct Recognition	Incorrect Recognition	Recognition Accuracy	Recognition Time (Second/frame)
Fist	97	92	5	94.85%	0.017
Palm	102	100	2	98.04%	0.017
C	112	108	4	96.43%	0.017
V(Two)	95	91	4	95.79%	0.017
Five	134	127	7	94.78%	0.017
Index	119	116	3	97.48%	0.017

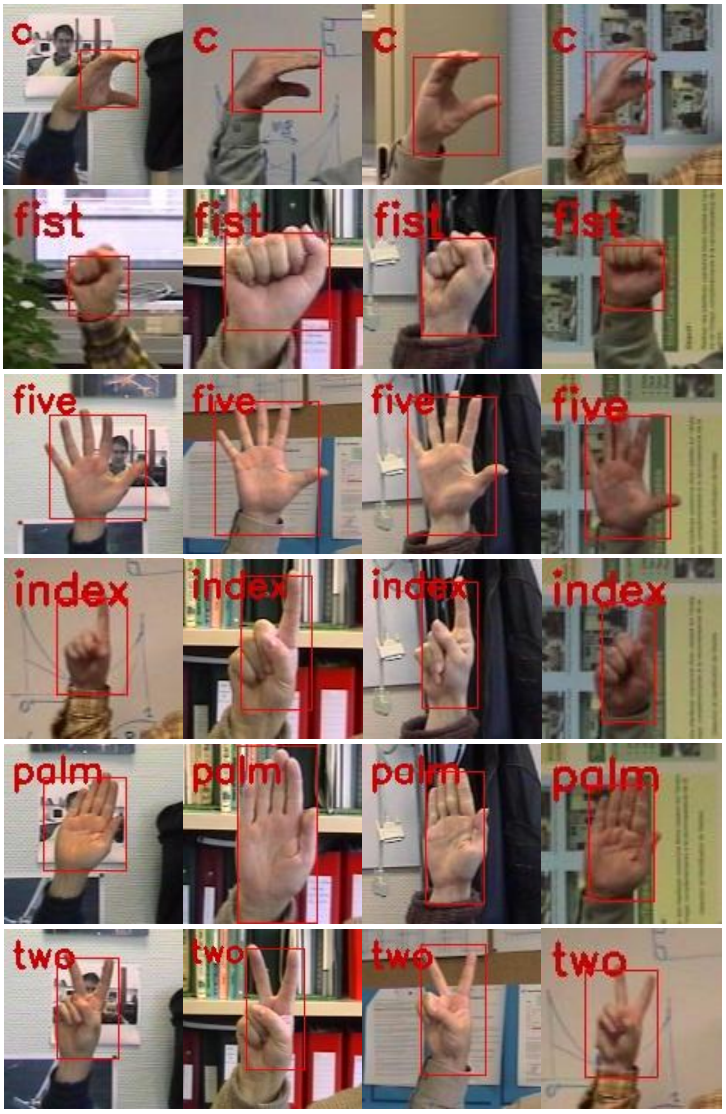


Figure 4.12: Hand posture detection and recognition for the six hand postures of Marcel database images using our approach.

In the testing stage, we used the new cluster and multiclass SVM classifier models that were built in the training stage for classifying the six hand postures provided by Marcel database. The six hand posture contours were loaded to detect hand postures in test images after face subtraction and skin detection. The overall recognition accuracy of the multiclass SVM classifier on this benchmark database was 96.23% as shown in Table 4.6. As we mentioned before, the recognition time has always been 0.017 s for each frame regardless of the frame size used because the keypoints were extracted from the small image (50×50 pixels) that contains the detected hand posture only.

As previously mentioned, regardless of the test image resolution (such as 100×100 pixels for the database images), the OpenCV function *cvBoundingRect* will enclose the detected hand with a rectangle, which has the largest matching contours with the loaded hand posture template contours. We can resize and save the rectangle in a small image using the OpenCV function *cvResize*. As shown in Table 4.4, the recognition time decreases as the size of the small image that contains the detected hand decreases because the time needed to extract keypoints using the SIFT algorithm decreases when the image size decreases. Therefore, we chose the smallest size of 50×50 pixels to reduce the processing time. Furthermore, since the extracted features using SIFT are invariant against scale and rotation, choosing any size for the small image that contains the detected hand will not affect the recognition accuracy.

Figure 4.12 shows some correct samples for the six hand postures from the Marcel database using our approach.

4.4 Performance Comparison with other Approaches for Hand Posture Recognition

In this section, we compare the performance of our approach with the Marcel [Marcel99] approach and other approaches for hand posture recognition. We selected a number of recognition methods [Chen07, Marcel99, Chung09, Fang07b, Yun09, Ren10] that have good real-time performance and discussed their recognition time and accuracy to compare their performance with our approach in [Dardas11]. In [Chung09], the hand gesture was detected using a skin color approach. Features for all of the detected hand postures were extracted based on a Haar Wavelet Representation and stored in a database. Recognition was performed through a measurement metric between the features of a test image and those in the database. In [Fang07b], hand detection with Adaboost was used to trigger hand posture tracking and recognition. Then, adaptive hand segmentation using motion and color cues was executed during detection and tracking. Finally, scale-space feature detection was applied to find palm-like and finger-like structures. Hand gesture type was determined by palm–finger configuration using blob and ridge structures detection. In [Yun09], hand posture was detected based on the Viola–Jones method. Then, the Hu invariant moment feature vectors of the detected hand posture were extracted, and a SVM classifier was trained for final recognition. In [Chen07], the posture was detected using Haar-like features and the AdaBoost classifier. Based on the cascade classifiers, a

parallel cascade structure was implemented to classify different hand postures. In [Ren10], a gesture recognition system acquired and preprocessed video image frames from the camera, then extracted the normalized moment of inertia features and Hu invariant moments of gestures to constitute a feature vector, which is input into SVM to achieve classification results. In [Marcel99], a neural network model (constrained generative model) was used to recognize a hand posture in an image. A space discretization based on face location and body anthropometry was used to segment hand postures.

4.4.1 Performance Comparison with Other Approaches Based on Their Own Image Database

Table 4.7 shows the performance of our approach comparatively with that of all other approaches mentioned in section 4.4 based on their test results using their own database test images in terms of recognition time, accuracy, resolution, number of tested images, background, and invariance against scale, rotation, and different lighting conditions.

Table 4.7: Performance comparison with other hand posture recognition approaches

Reference	Number of Postures	Recognition Time (Second/Frame)	Recognition Accuracy	Frame Resolution	Number of Test Images/Posture	Scale	Rotation	Lighting Changes	Background
[Chung09]	15	0.4	94.89%	160×120	30	Not discussed	Invariant	Not discussed	Wall
[Fang07b]	6	0.09 – 0.11	93.8%	320x240	195-221	Not discussed	Not discussed	Not discussed	Cluttered
[Yun09]	3	0.1333	96.2%	640x480	130	Invariant	±30°	Variant	Different
[Chen07]	4	0.03	90.0%	320x240	100	Invariant	±15°	Invariant	White wall
[Ren10]	8	0.066667	96.9%.	640x480	300	Variant	Invariant	Not discussed	Not discussed
[Marcel99]	6	Not discussed	93.4 %	100×100	57-76	Invariant	Not discussed	Invariant	Wall
[Marcel99]	6	Not discussed	76.1 %	100×100	38-58	Invariant	Not discussed	Not discussed	Cluttered
Our Method [Dardas11]	10	0.017	96.23%	640x480 or any size	1000	Invariant	Invariant	Invariant	Cluttered

4.4.2 Performance Comparison with Other Approaches while Using the Same Public Image Database

To make comparison fair, this section discusses how the other algorithms from the literature in Table 4.7 were retested using the same public image database to compare their performances based on the same public image database.

[Chen07, Fang07b, Yun09] used Viola and Jones's method [Viola04] to detect the hand posture before recognition. Therefore, we created six Haar classifiers to test the detection algorithms for six postures in the Marcel database [Marcel99] before recognition. We used *OpenCV* to train and create six classifiers for C, A (fist), Five, Point (index), B (palm), and V (two) postures using Haar-like features for training. Each classifier was trained with 1000 positive images for each hand posture and 3500 negative images. The result of training was six "XML" classifier files. The "XML" classifiers were used to detect the six hand postures in the Marcel database images. Around 40% of those database images had complex or cluttered backgrounds while others have a wall background. The detection results were not remarkably successful. The classifiers did not detect any posture for all of the test images that had cluttered background. While the classifiers detected six postures successfully for around 50% of the images that had a wall background, with some difficulty to detect the palm posture. The classifiers failed for another 50% because the rotations for hand postures were more than ±15° [Kölsch04c]. The remaining of the total images was around 30% for recognition after detection, which will be the maximum recognition rate for algorithms used in [Chen07, Fang07b, Yun09]. To validate the performance of the six Haar classifiers, we tested them to detect our hand postures in images captured from a webcam. The Haar classifiers detected our six hand postures successfully if the hand posture's rotation was less than ±15°, and if the background consisted of a wall as shown in Figure 4.6 and Figure 4.7. It has been observed that the palm classifier had difficulty to detect the palm posture. Also, the six Haar classifiers failed to detect all of the postures if the background was cluttered as shown in Figure 4.8 and Figure 4.10.

Haar-like features have previously been used successfully in face classification [Viola04]. However, their use for hand detection and tracking [Kölsch04c, Barczak05b, Micilo04] has not been remarkably successful. The major reason for this is that the Haar-like features are not invariant against rotation [Messom06]. The Haar-like features that describe faces are insensitive to small angle changes as much as 30° from the vertical [Kölsch04c]. In

[Kölsch04c], the in-plane rotational robustness of the Viola-Jones method was studied for hand detection. The results showed that hand detection with the Viola-Jones detector can be done with about 15 degree in-plane rotations compared to 30 degree on face. The person's head is normally aligned vertically with respect to gravity. Thus, rotational sensitivity is not a serious problem for faces. However, hands are not naturally aligned with the horizontal or vertical axes. Therefore, it is difficult to model them with traditional Haar-like features. As we discussed in Chapter 2, the face has strong Haar-like features related to shading, which facilitates the face detection in cluttered background. However, Haar-like features are not sufficient to describe the hand postures because the most significant feature of a hand posture are the shape of the hand rather than the hand surface shade and texture [Shi10], which makes the hand posture detection difficult in a cluttered background. In order to avoid the background's negative effect [Chen07] used a white wall background while detecting hand postures using Haar classifiers.

In [Chung09, Ren10], a skin color detection method was used to detect hand postures before recognition. There were many conditions for their methods to detect hand postures successfully. For example, the face should not appear in the image, skin-like objects should not exist in the background, and the front arm of the user had to be covered with clothes [Chung09]. The images used in [Chung09, Ren10] contained only the hand postures without any other skin-like objects or the face. The background used in [Chung09] is a wall only without any other objects. In [Ren10], the algorithm for skin detection was not discussed at all. Therefore, we used the algorithm proposed in [Chung09] to detect the skin color only in the Marcel database images [Marcel99]. We found that the performance of the algorithm used in [Chung09] to detect skin color only is the same as the performance of our algorithm. The skin detection algorithm detected six postures with other skin-like objects for around 70% of the public test images in the Marcel database images that have cluttered background as shown in Figure 4.13. While the skin detection algorithm detected six postures alone without other skin-like objects, for around 50% of the images that have wall background. The remaining was around 42% of the total images for recognition after detection, which will be the maximum recognition rate for algorithms used in [Chung09, Ren10]. Figure 4.14 shows successful skin detection for hand postures alone for the public test images that have a wall and cluttered background. Figure 4.15 shows unsuccessful skin detection for hand postures alone for the public test images that have wall background.

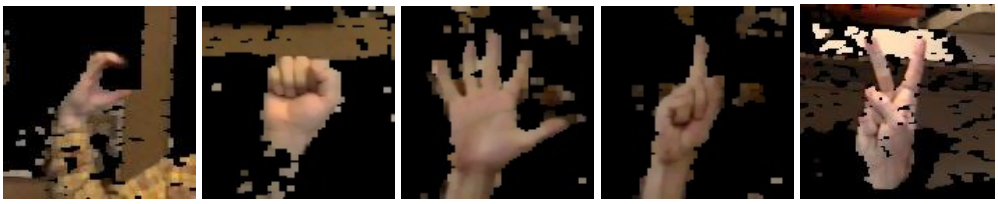


Figure 4.13: Skin detection for public database images with cluttered background.

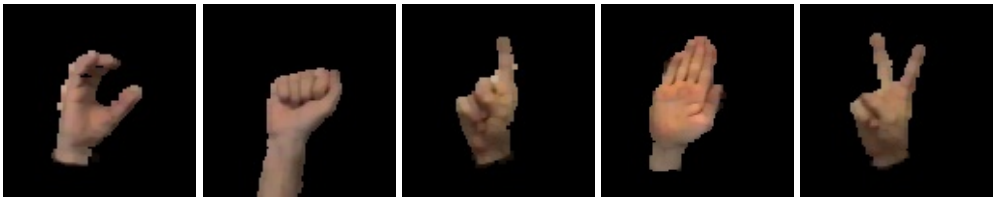


Figure 4.14: Skin detection for postures alone for public database images with wall and cluttered background.

Figure 4.15: Skin detection for public database images with wall background.

4.5 Building Grammar Using a Finite State Machine (FSM)

Dynamic hand gesture recognition is defined by the direction of the hand and the sequence of hand postures in a sequence of images. Moreover, there are two issues to deal with dynamic hand gesture recognition: spotting or gesture segmentation and classification. Gesture segmentation is one of the challenging aspects in the area of gesture recognition, where it is needed to detect the start point and the end point of a gesture pattern. While classification is the task of matching the segmented gestures against reference patterns or templates of predefined gestures, to determine which class it belongs to. Therefore, the classifier can be trained against a possible grammar.

Behaviour classification has developed to be an active research area in the computer vision field. There are a number of methods to classifying high-level behaviours of vision based information. In [Goshorn08], the behaviour classification was carried out in two stages: (1) a low-level event classifier based on features of raw video information, and (2) a high-level

behaviour classifier based on sequences of events that were generated from the first stage. Therefore, a behaviour classifier can be considered as a classifier of sequences of events [Iglesi10, Goshorn08, Lee06, Remagn01].

Behavior classification has been well-known for using state-space models, such as Hidden Markov Models (HMMs). The authors of [Ivanov97] considered that using syntactic (grammar-based) structural methods for behavior modeling are better than HMMs when some sequences of low-level information inherently fall into meaningful behaviours. We recognize patterns of behavior as meaningful in terms of patterns we are already familiar with. In [Ivanov00], a stochastic context-free grammar was utilized for hand gesture in car parking. In [Chen08b], a hand gesture recognition system was proposed using Haar-like features and a stochastic context-free grammar. In [Licsar04], more complex gestures were synthesized by requiring users perform a sequence of static hand postures to step through a finite state machine (FSM). In this logic, a gesture is any sequence of static hand postures that comply with the grammar defined by the FSM.

We integrated the spatiotemporal (space-time) relation between every two consecutive frames in a video sequence in terms of the transition among recognized postures and their locations to develop our system into real-time dynamic hand gesture recognition. The timing relation for transition among recognized postures is monitored for every two consecutive frames of a video sequence by saving every two recognized consecutive postures in two states queues, where the recognized previous posture is saved in the old state queue, while the recognized current posture is saved on the new state queue. The movement direction or space relation between every two recognized postures from every two consecutive frames is tracked by monitoring the difference between the two locations of the recognized previous and current hand postures.

A series of events or features, over time and space, create behaviour. In our proposed hand gesture recognition system, the events are based on monitoring the transitions among detected hand postures and their locations for every two consecutive frames. Low-level classifications are based on features of raw video information resulting from the real-time posture recognition classifier discussed in section 4.2. At a high-level, the detected hand postures for every two consecutive frames are concatenated in a temporal sequence for forming behaviour, which is a hand gesture. A hand gesture is an action, which consists of a sequence of

hand postures. The rules for the composition of hand postures into various hand gestures can be described by a grammar [Chen08b]. A method is required to monitor the sequences of detected hand postures and classify them such as a syntactical grammar-based method, which was presented in [Goshorn08]. The different hand gesture compositions have to be defined before classifying sequences. Every hand gesture behaviour will have a different temporal composition of the detected hand postures. These sequence of compositions are defined with syntax rules such as regular grammars [Sipser97]. A set of syntax rules, describing hand gesture behaviour, is known as a grammar. The grammar is implemented through a finite state machine (FSM). The FSM monitors the sequence of the hand postures. If a sequence of hand postures goes with a specific hand gesture behaviour, its related FSM will accept this sequence. Thus, the sequence of hand postures is recognized as hand gesture behaviour if its related FSM agrees to the sequence.

The gesture classifier recognizes the detected hand posture and the result of the recognition are fed into a FSM. The grammar of the FSM decides the real tasks. These tasks can be commands or events to be executed by the keyboard. The FSM of the grammar for the palm posture is shown in Figure 4.16. The FSM of the grammar for the other postures such as fist, index, and little will be the same as the palm posture.

The non-terminal symbols of the grammar are the states = {Palm, Little, Index, Fist}.

The terminals symbols of the grammar = {Turn right/Left, Move Up/Down,
Zoom In/Out, Hold, Send Ctrl key}.

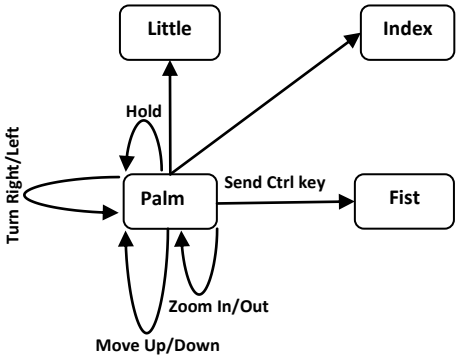


Figure 4.16: FSM of the grammar for the palm posture.

The grammar of the FSM was implemented to generate gesture commands sent to the keyboard for interacting with a videogame that requires a novel input mechanism as we will discuss in sections 4.7, 4.8, 4.9, and 4.10. In our implementation, we wanted to minimize the requirements for modifying the codebase of the videogame to interface with our proposed hand-gesture recognition system developed in this chapter. No significant modifications will be made to the videogame to be compatible with the gesture interface as it simply responded to the keyboard input events received from the operating system.

To implement the grammar of the FSM, we mapped these gesture commands sent to keyboard by correlating the transitions between recognized hand postures and hand motion directions for every two consecutive frames for the palm posture case according to Table 4.8. This spatiotemporal (space-time) correlation develops our system into real-time dynamic hand gesture recognition.

Table 4.8: Interaction commands performed by palm posture

Current State	Next State	Direction	Send Key	Command
Palm	Palm	Up	{UP}	Move Up
Palm	Palm	Down	{DOWN}	Move Down
Palm	Palm	Right	{RIGHT}	Move Right
Palm	Palm	Left	{LEFT}	Move Left
Palm	Palm	Forth	^{+}	Zoom In
Palm	Palm	Back	^{-}	Zoom Out
Palm	Fist	Hold	^	Send Ctrl key

4.6 Generating Gesture Commands

Based on the grammar outlined in the previous section, our hand gesture recognition system can generate gesture commands, which can be used to control or interact with an application or a videogame instead of keyboard or mouse, by sending events to be executed such as double click, close, open, go left, right, up, or down, and so on. In sections 4.7, 4.8, 4.9, and 4.10, the gesture commands will be used to control an exergame. The expression ‘exergame’ comes from the words ‘exercise’ and ‘game’ and relates to computer games that entail deliberate strong physical activity [Silva11]. We will discuss how the user can control and direct left-right

movement and shooting actions by a set of hand gesture commands in an exertion videogame that makes use of a stationary bicycle as one of the main inputs for game playing.

The newly developed depth sensors such as the Microsoft Kinect, provide great opportunities for human-computer interaction. Despite a lot of recent achievements in using the Kinect sensor for human body tracking and body gesture recognition, it is still a challenge to employ the Kinect for hand gesture recognition because of the low-resolution of the Kinect depth map of only 640×480 pixel [Ren11, Ren11b]. Although the Kinect can detect hand gestures, it does not come with a hand gesture recognition application software [Bergh11]. Many popular technologies such as the Kinect, Sony PSP, and Nintendo DS and Wii contain gesture recognition with their consoles. Unfortunately, only dynamic gestures such as waving and fist hitting are recognized so far [Wachs11].

We define 9 postures: C, fist, index, (l) L, little, palm, Two, W and Y. The gesture commands can be generated by three ways. First, by observing the gesture transitions from posture to posture, such as from fist to index, fist to little, etc as shown in Figure 4.17. For each posture, we have ten transitions to other states. Second, by observing the direction of movement for each posture: up, down, left, or right. We have four directions or gesture commands for each posture. The case of no movement was not considered for each posture, because it is already counted in the first way when the transition occurred from fist to fist or palm to palm and so on. Finally, by observing the hand posture size or scale: when it comes close (zoom in) or far away (zoom out) from the camera. We have two cases for each posture. Thus, from the three ways, we have large gesture commands that can be sent to interact with an application or video game. In the following, we will explain how our system can generate gesture commands, which will be used in the following sections to control an exertion videogame.

4.6.1 Transitions Among Postures

Transitions among postures depend on saving every two recognized postures from every two consecutive frames of a video sequence in two states queues: the new state queue, which holds the recognized current posture and the old state queue, which holds the recognized previous posture. The recognized current posture will be saved in the new state queue after transferring its posture state to the old state queue. Thus, for every frame captured, the posture state of the old state queue is emptied. Then, the posture state of the current state queue is

transferred to the old state queue. Finally, the recognized current posture from the frame will be saved in the current state queue.

Figure 4.17: Transitions from fist to other postures.

The system will keep observing the two states queues for every two consecutive captured frames. In addition, it will keep monitoring the two states queues transitions for every two consecutive recognized postures to generate a specific gesture command for a specific transition among recognized postures. Figure 4.17 shows all the transition states of fist posture with all other postures.

4.6.2 Movement Direction for Each Posture

Movement direction for each posture depends on tracking the movement direction of the detected posture using rectangles, which captures the detected hand posture. Once a posture is detected, the coordinates of the middle of the rectangle X and Y are recorded. The system will always monitor the absolute difference of distance between the two points of the middle of the rectangle in the X and Y coordinates for every two successive frames that have the same posture. If the absolute difference of distance in the X direction is larger than the absolute difference of distance in the Y direction and the absolute difference is larger than 1 cm, then the hand posture is moved left or right. If the difference of distance in the X direction is positive, then the hand posture is moved right, and if it is negative, then the hand posture is moved left. If the absolute difference of distance in the Y direction is larger than absolute difference of distance in the X direction and the absolute difference is larger than 1 cm, then the hand posture is moved up or down. If the difference of distance in the Y direction is positive, then the hand posture is moved down, and if it is negative, then the hand is moved up. Figure 4.18 shows all the motion direction cases of palm posture.

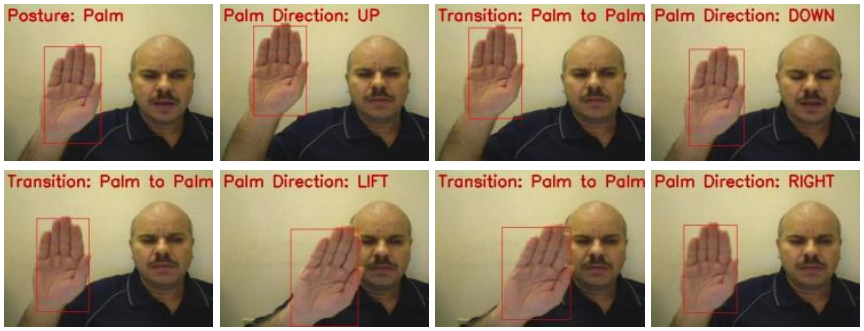


Figure 4.18: Movement direction cases of palm posture.

4.6.3 Distance from the Camera for Each Posture

Distance from the camera for each posture depends on tracking the size of height for the rectangle, which captures the detected hand posture only, and the transition of the recognized posture still the same. Once the hand posture is detected from each frame by the rectangle and the transition of recognized posture still the same such as little to little, the height of the rectangle is recorded. The system will always monitor the difference between the heights of two rectangles for every two successive frames that have the same posture. If the difference of rectangle height between the new frame and the previous frame is positive and larger than 1 cm, then the posture gets closer to the camera (zoom in), and if the difference is negative and larger than 1 cm, then the posture gets away from the camera (zoom out). Figure 4.19 shows all the zoom cases of the little posture. The threshold for detecting hand movement away from or close to the camera depends on the distance difference between the detected hand postures from the camera for every two successive frames that have the same posture, which is 1 cm, regardless of the speed of motion.

Figure 4.19: Zoom cases of little posture.

4.7 Hand Gesture Interaction with Gaming

To show the effectiveness of our hand gesture recognition system described in this chapter for human-computer interaction (HCI), a gesture-based interaction with a 3D gaming virtual environment (VE) was implemented and described in the following sections. With this system, the player will explore, manipulate, and control the 3D gaming world objects by a set of hand gestures commands. Our system demonstrates the hand gesture-based interface can attain an enhanced and more intuitive and flexible interaction for the player than other HCI devices.

We will present an exertion videogame that can be controlled with a bare hand using our proposed hand gesture recognition system described in this chapter to provide natural, user-independent, user-friendly, and real-time interaction between human and computers. Exertion games are computer videogames that require the player to be physically active while playing the

game in order to achieve her/his goal [Silva11]. The user in the exergame will control and direct left-right movement and shooting actions by a set of hand gesture commands in an exertion videogame that makes use of a stationary bicycle as one of the main inputs for game playing.

4.8 Exertion Game

With the purpose of testing our hand gesture recognition system described in this chapter in the context of an interactive game application, we have integrated the recognition engine into a target shooting exertion game. Given the physical nature of this type of game, players are faced with the difficulty of interacting with the game using traditional input controllers like gamepads. Hence, exertion games present a good opportunity for using body movement tracking techniques as an alternative to more traditional input controllers such as keyboards and mice. It is in this context that we have decided to combine a cycling exertion game with our hand gesture recognition algorithm. A hand gesture interface is designed for additional exciting interactions since it does not incur any conflicts with the main activity of cycling. With this interface, the player can control and direct left-right movement of crosshairs and shooting actions by a set of hand gesture commands while the vertical movement of the crosshairs is controlled by a stationary bicycle. The exergame offers a fully interactive and immersive experience because our gesture recognition system recognizes the player's hand gestures and incorporates them into the flow of actions displayed upon a screen. The exergame can be a useful tool in different areas such as tackling obesity and for rehabilitation.

The game implementation simply takes the RPM (revolutions per minute) readings from a stationary bike to determine the height to which the player is pointing in a target-shooting video game. We then enhance the interactivity by controlling left-right movement and shooting actions in the game with hand gesture recognition using an off-the-shelf web cam. This way, the player does not need to use the computer keyboard to perform such actions while exercising on the bike.

We will describe in detail each of the components of the exertion game (without hand gesture control) and how they interact with each other and how they are used by the player to interact with the game.

The exertion game is based on a typical target-shooting scenario and is fully described in previous work [Silva11]. The user interface shows a 3D island environment, where the camera

viewport has a first person shooter perspective showing the direction where the player looks at. The player's character is standing on a platform facing a field, and in her/his line of view, there are three targets located at different distances, which the player must knock down by throwing coconuts at them as shown in Figure 4.20. The player plays the game while exercising on a stationary bicycle which acts as the input controller for the game.

The game engine handles the rules of the game as follows: points are gained when the player knocks down targets by hitting them with a coconut. Bonus points are gained when all three targets are knocked down within a 10 second time frame. After 10 seconds of being knocked down, a target automatically rises again. Targets are located at various distances from the player. The player receives feedback about the general direction of the throw by a visual crosshair icon on the screen as shown in Figure 4.20.

Figure 4.20: Exergame interface screenshot, showing island environment, targets & crosshairs pointer.

The vertical movement of the crosshairs is controlled by the stationary bicycle used as an input controller. The RPM (revolutions per minute) reading from the stationary bike's pedal speed is mapped onto the Y coordinates that drive the vertical displacement of the crosshairs, which affects the trajectory and distance (depth) to which a coconut may be thrown by the player. The higher the RPM reading is, the higher the throw and vice versa. As the game is played, the player must adjust slightly the speed of his/her pedaling to reach the targets of varying distance.

The player controls the horizontal movement of the crosshairs by using the left and right arrows of the keyboard. The control is limited to changing the direction on which the crosshairs displaces (left or right), but the speed and movement of the crosshairs are preset. This adds a

level of difficulty and exertion to the game, but also requires the player to make frequent changes of direction. Throwing the coconuts is triggered by pressing the 'control' key on the keyboard. The player may shoot as often as she/he wants.

Figure 4.21 depicts the three input methods to control the game. Shooting and changing direction left and right are controlled by the keyboard while changing direction up and down is controlled by the bike. This is an excellent example of a game that requires a novel way of interaction, since handling the keyboard while exercising on the bicycle is an awkward challenge. Thus, in the next section the functionality of the keyboard will be replaced with hand gesture control to provide more natural and fast movement and entertainment for the user.

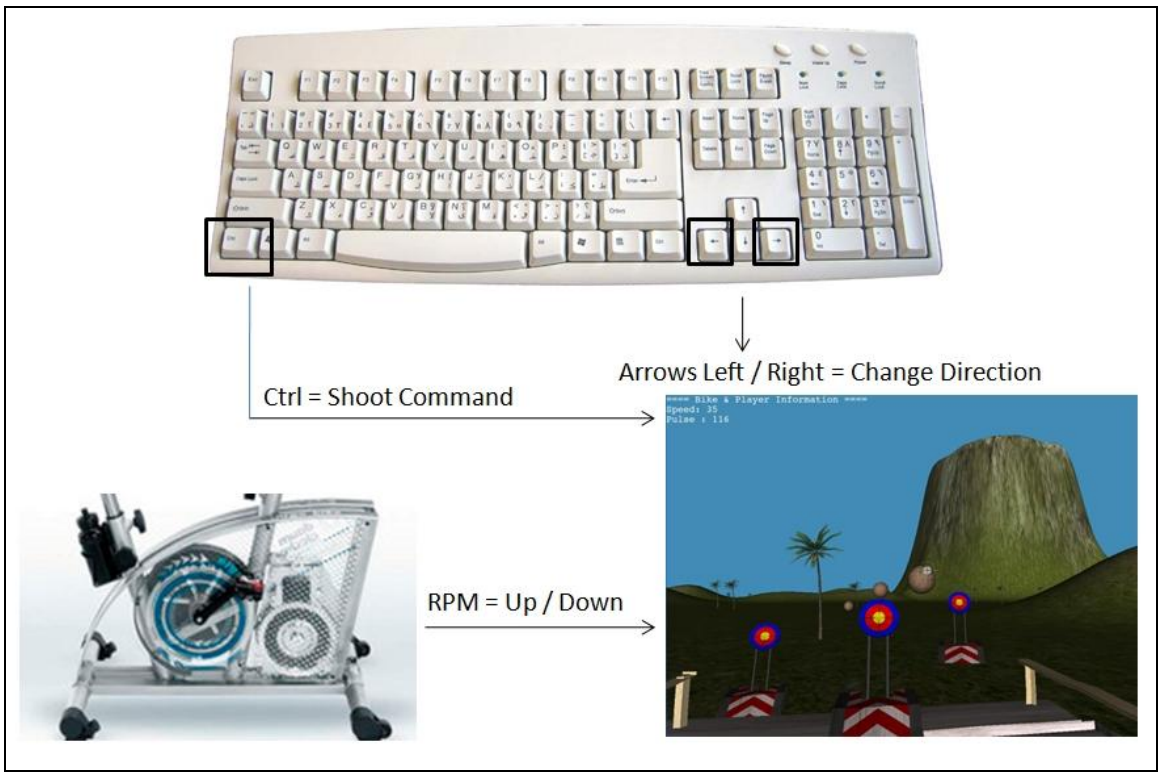


Figure 4.21: Game input methods (without hand gesture control).

4.9 Interaction with 3D Exertion Game Using Hand Gestures

In order to address the problem of moving the crosshairs left/right and throwing coconuts while pedaling the bike, we used three hand gestures that generate commands to allow the player to change direction (left/right) and for shooting a coconut. In our implementation, we wanted to minimize the requirements for modifying the codebase of the videogame to interface with our proposed hand-gesture recognition system built in this chapter. For this purpose, we decided to

have both applications (the game and gesture recognition process) run independently but at the same time on the same host computer. Since the game is already coded to respond to keyboard events for left/right and shooting, the gesture recognition process will make a hardware interrupt to the underlying operating system to set those keyboard events manually. In this fashion, for each in-game action, we mapped a specific hand-gesture to the related keyboard input event as we will describe in the next section. This was done to investigate using the designed hand gesture system built in this chapter with existing games without modifying their code. With the hand gesture recognition replacing the functionality of the keyboard, we aimed at having the player more engaged in the gaming experience using a more natural interface. To implement this functionality, we mapped these events sent to the keyboard by integrating the transition between recognized hand postures and hand motion directions for every two successive frames for the palm posture case according to the FSM as shown in Figure 4.22. The derived interaction commands for the game are given in Table 4.9.

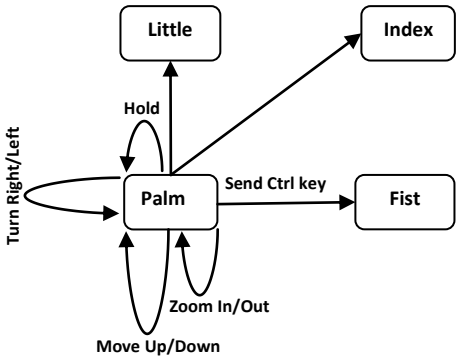


Figure 4.22: FSM of the grammar for the palm posture.

Table 4.9: Interaction commands performed by hand gestures.

Current State	Next State	Direction	Send Key	Event
Palm	Palm	Right	{RIGHT}	Move Right
Palm	Palm	Left	{LEFT}	Move Left
Palm	Fist	Hold	^	Shoot

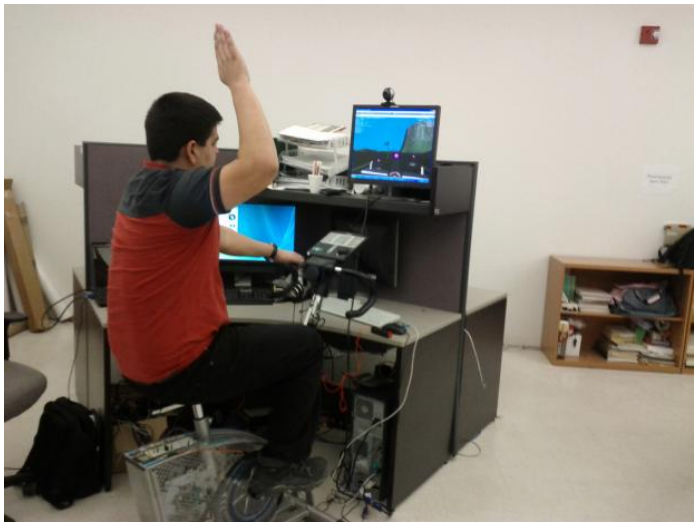


Figure 4.23: Game input methods with a hand gesture recognition.

Figure 4.23 depicts the input methods to control the game with the hand gesture interface while the player is on the bike making hand gestures and playing the game.

4.9.1 Experimental Set Up

For a host computer, we used a PC running the Windows XP operating system. The configuration of the system was a 2.0 GHz processor and 2 GB RAM. The exercise bike was an Ergo-bike 8008 TRS 3 model connected to the host computer by a serial cable. The web cam was a low-cost Logitech QuickCam connected to the host computer by a USB port. The game was developed using Unity Engine and programmed using C#. No significant modifications were made to the game to be compatible with the gesture interface as it simply responded to the keyboard input events received from the operating system.

C# was used to integrate our DLL file generated from the C-based gesture recognition component. With this framework, the C# methods can call applications and libraries in C/C++. The program in C# was used to send events to the keyboard using the SendKeys.SendWait method. Those events or commands are generated using our hand gestures.

4.9.2 Shooting Action Using Hand Gesture

The first custom gesture represents a “fire button” action. When this gesture is detected by the gesture recognition process, it generates a Key-pressed event with the “Ctrl” key as an identifier. This event is then picked up by the game, which responds by throwing a coconut. This

gesture can be observed in Figure 4.24, when the player changes her/his hand posture from a palm posture to a fist posture to shoot the target by throwing a coconut. The palm posture of the player is shown in the small screenshot in the top left corner of Figure 4.24 (1), while in the next frame, the player changes her/his hand posture to fist posture to shoot the target as shown in the small screenshot in the top left corner of Figure 4.24 (2).

(1) **(2)**

Figure 4.24: "Shooting" action using hand gesture for the exertion game. In the top left: (1) Player with palm posture (2) Player changes to fist posture for shooting.

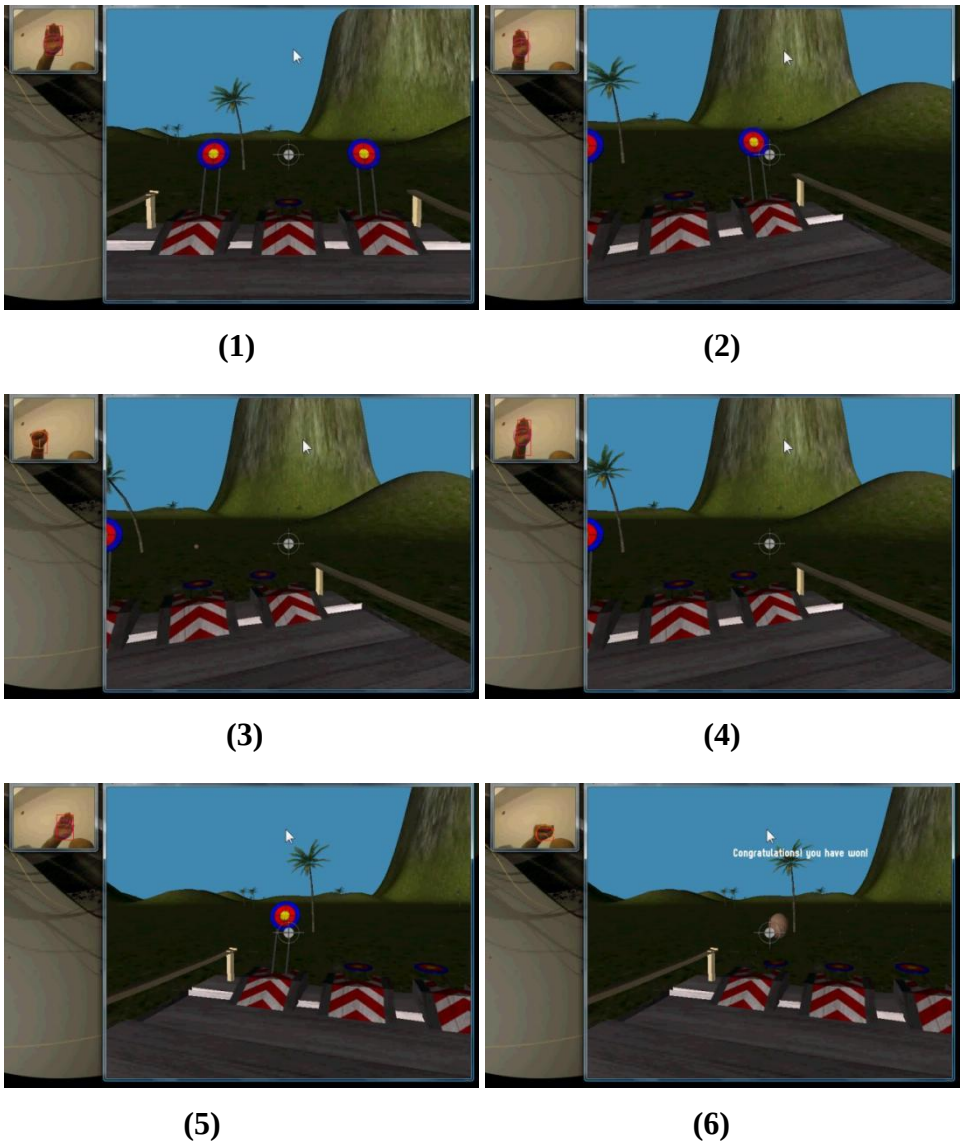
4.9.3 Changing Direction Action Using Hand Gesture

The other two gestures represent the action of moving the crosshairs left or right. Again, when either of these two gestures is detected, a key-pressed event is generated with the "Left arrow" or "Right arrow" identifiers, accordingly. These gestures can be observed in Figure 4.25 when the player moves her/his hand palm posture left or right.

4.10 User Study

In order to test our proposed hand gesture recognition system developed in this chapter for user-independence, we tested interaction with the exertion game using our proposed hand gesture system for five users. With the purpose of obtaining feedback from actual users about our interface implementation, we conducted a user study, where 15 participants played the game using the hand gesture interface. At this stage of our research we wanted to assess the level of technology acceptance from users to this particular interface implementation. For this, the scope of our present user study is to gather the opinions and impressions from users in a structured way by using the Technology Acceptance Model [Davis86]. At the end of the game sessions participants answered a questionnaire and were interviewed to collect their impressions and their

suggestions regarding how natural it was for them to play the game using hand gestures to control the actions in the game.



(1) **(2)**

(3) **(4)**

(5) **(6)**

Figure 4.25: "Left/Right" movement actions with shooting using hand gesture for the exertion game. In the top left: (1) Player with palm posture (2) Player moves her/his palm posture right to move crosshairs right (3) Player changes to fist posture for shooting (4) Player changes to palm posture (5) Player moves her/his palm posture left to move crosshairs left (6) Player changes to fist posture for shooting.

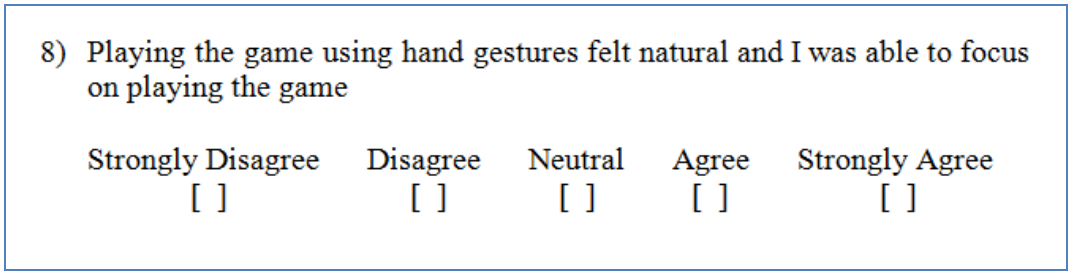
There were 12 male and 3 female participants. All of them were students in the computer science department at the University of Ottawa at the undergraduate level, and none of them worked in the same laboratory than the authors. Their ages ranged from 21 to 25. None of them

reported having any physical or mental disabilities and although all of them reported having played video games at some point, none of them were particularly avid gamers.

The sessions were carried out with one participant at a time. At the beginning of the session, the participant was introduced to the overall procedure of the experiment, and a demonstration of the game and the general instructions for playing the game were presented. Then the participant was given five minutes to do some warm up in the bicycle alone without having the game on. Once the warm-up period ended, the game was started and the player was left to play the game for five minutes. At the end of the five minute period, the game would be over and the participant was allowed to cool down pedaling on the bike.

After the gaming session participants were asked to answer a questionnaire. The questionnaire consisted of a 5-point Likert scale with 10 items including reversal statements [Likert32]. Likert scales are typically used to measure psychometric response to assess the attitude that a user has towards some topic or experience she/he had, and are applied often as the items in a questionnaire. It has been suggested that Likert scales are among the most common methods to estimate usability [Dumas98].

In our study, player perception regarding the ease of use, accuracy and overall performance of the exergame's hand gesture interface was measured using Likert scale. Particularly we wanted to assess how natural the interactions were to them and if they perceived any performance issues such as difficulty to perform changes of direction or perceived delays in those changes. Figure 4.26 shows a sample question from the full questionnaire (Appendix 1).


8) Playing the game using hand gestures felt natural and I was able to focus on playing the game

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

Figure 4.26: Sample Likert item from the questionnaire.

The interviews were guided by open questions that covered similar topics to those in the questionnaire except that they were meant to encourage the participants to elaborate verbally about their overall experience.

4.10.1 Results and Discussion

In this section we present the results of the questionnaire and the interviews carried out during the evaluation. Responses to the individual items of the questionnaire are considered as ordinal data, which is a common practice with Likert scales. Hence, we present the results in terms of descriptive statistics that when analyzed can help to draw a picture of the perception that the users received for each topic covered by the items of the questionnaire. Tables 4.10 and 4.11 show such descriptive statistics of the data collected.

The items of the questionnaire were divided into four main areas. Questions 1 to 3 were all related to the 'Left/Right' hand gestures and each one addressed a specific topic: ease of use, perception of delay, and accuracy to perform the task. Questions 4 to 6 similarly addressed the same topics but for the "Shoot" hand gesture. Questions 7 and 8 were in general about the whole hand gesture interface to the game, in order to assess if it was comfortable and felt natural and intuitive. Finally questions 9 and 10 were aimed at assessing the willingness of the participants to use such a system in their daily lives. The structure of the questionnaire was defined following the precepts of the Technology Acceptance Model (TAM) [Davis86].

Table 4.10: Median Mode and Range for each Likert item.

Question:	Median	Mode	Range
1.- Left/Right Gesture – Easy	4	4	3
2.- Left/Right Gesture – No Delay	3	3	2
3.- Left/Right Gesture – Accurate	4	4	3
4.- Shoot Gesture – Easy	5	5	2
5.- Shoot Gesture – No Delay	3	3	3
6.- Shoot Gesture – Accurate	4	4	2
7.- Hand Gestures - Comfortable	3	2	2
8.- Hand Gestures -Natural	5	5	3
9.- Use Intention	4	5	2
10.- Use Intention (reversed)	5	5	1

Table 4.10 shows a summary of the median, mode and range on the data set pertaining to each of the items of the questionnaire.

From this data summary it can be seen that the lowest value for the median across all questions was 3 ('neutral') while the highest was 5 ('strongly agree') with a similar tendency for

the most repeated responses (mode) with values between 2 and 5. The range on the data sets was never beyond 3 points, which shows a cohesive response pattern.

Table 4.11 shows an aggregation of responses for each of the Likert levels in each question. For each question we see the tally of responses and the second row shows the same values expressed in percentage.

Table 4.11: Aggregation of responses of each Likert level across all 10 questions.

Question:	Strongly Disagree[1]	Disagree [2]	Neutral [3]	Agree [4]	Strongly Agree [5]
1.- Left/Right Gesture - Easy	0	1	4	5	5
		7%	27%	33%	33%
2.- Left/Right Gesture - No Delay	0	5	8	2	0
		33%	53%	14%	
3.- Left/Right Gesture - Accurate	0	4	0	10	1
		27%		67%	6%
4.- Shoot Gesture - Easy	0	0	3	4	8
			20%	27%	53%
5.- Shoot Gesture - No Delay	1	4	6	4	0
	6%	27%	40%	27%	
6.- Shoot Gesture - Accurate	0	0	3	9	3
			20%	60%	20%
7.- Hand Gestures - Comfortable	0	7	6	2	0
		47%	40%	13%	
8.- Hand Gestures -Natural	0	1	2	4	8
		7%	13%	27%	53%
9.- Use Intention	0	0	1	7	7
			6%	47%	47%
10.- Use Intention (reversed)	0	0	0	7	8
				47%	53%

From the details of the distribution of responses in Tables 4.10 and 4.11, it can be seen that both of the modalities of hand gestures (‘Left/Right’ and ‘Shoot’) had a low rating (3 ‘neutral’) in response to the statement of ‘not perceived delay’. According to the responses, users did not generally agree, and presented a ‘neutral’ perspective to the statement that said that there was no-delay. By observation we could measure an occasional noticeable delay between the execution of the hand gesture by the player and the recognition of it by the system. When it occurred, this delay could vary in a range from 0.5 to 1.0 second. However, players soon adjusted their game strategy to cope with the delay in those cases and were able to perform well

in the game. There were no significant false positive detections, and when the system would interpret the natural movement of the hand (resulting from exercising on the bike) as a ‘move left/right’ command, players would simply correct the action to aim for the desired target. This adjustment from the subjects is consistent with their responses in the questionnaire, where users did not seem to be affected in terms of their perceived accuracy to knock down the targets (the goal of the game). In this case, they rated with values mostly on or above the ‘agree’ level with 73% of the responses to the ‘left/right’ gesture and with 80% of the responses for the ‘shoot’ gesture.

In terms of ease of use, both types of hand gestures were rated generally in the same order, between ‘agree’ and ‘strongly agree’. This trend was reinforced during the interviews by the users, who expressed that the method of interaction was simple and intuitive.

When judging if the use of the hand gestures was comfortable for directing the actions on the exertion game, users expressed some reservations. The median value was at 3 with a mode of 2, and 47% of the responses expressing a ‘disagree’ position and 40% expressing ‘neutral’. During the interviews, users expressed that, even for an exertion game, they think that keeping the hand raised at all times for a longer game session would be quite uncomfortable, which is consistent with their responses on the questionnaire. Still, in terms of how natural the interaction was to them, the median and mode values were at 5 with a low range of 1, which shows that the mode of interaction was intuitive.

Overall from the interviews we could see that users were highly motivated to use the hand gesture control and this can also be seen on the answers to the last two items of the questionnaire, which show an intention of use with a rating of 4 and 5 in the Likert scale.

The proposed system demonstrated acceptable interaction with 15 users in terms of perceived accuracy and speed because the features of hand gestures were extracted in real-time for using the SIFT algorithm, and were invariant to scale and orientation.

4.11 Discussion

This chapter presents a novel real-time system for hand gesture recognition. Our system includes detecting and tracking a bare hand in a cluttered background using skin detection and hand posture contour comparison after face subtraction, recognizing hand gestures via bag-of-features and multiclass support vector machine (SVM) and building a grammar that generates

gesture commands sent to keyboard for interacting with an exertion videogame. In the training stage, after extracting the keypoints for every training image using the scale invariance feature transform (SIFT), a vector quantization technique mapped keypoints from every training image into a unified dimensional histogram vector (bag-of-words) after K-means clustering. This histogram was treated as an input vector for a multiclass SVM to build the training classifier. In the testing stage, for every frame captured from a webcam, the hand was detected using our algorithm described in Chapter 3 [Dardas11]. Then, the keypoints were extracted for every small image that contained the detected hand posture only, and fed into the cluster model to map them into a bag-of-words vector. Finally, the generated bag-of-words vector was fed into the multiclass SVM training classifier for recognizing the hand posture. Three critical factors affect the accuracy of the system: the quality of the webcam in the training and testing stages, the number of the images used for training, and choosing the number of clusters to build the cluster model. The grammar was built using a finite state machine (FSM) by integrating the spatiotemporal (space-time) relation between every two consecutive frames in a video sequence in terms of the transition among recognized postures and their locations.

We showed how a game running on a computer can be controlled with hand gestures using our system, which can allow for natural interaction between a human and a computer. A vision-based hand gesture interface for interacting with objects in a 3D VE was designed and implemented. With this interface, the user of the exergame can control and direct left-right movement and shooting actions by a set of hand gesture commands in an exertion videogame that makes use of a stationary bicycle as one of the main inputs for game playing. We carried out a user study to evaluate the interaction between our proposed hand gesture recognition system described in this chapter with the exertion game. Our preliminary results show the high level of interest from users to make use of multimedia systems that implement natural ways of interaction as the one presented here. Although some concerns in terms of comfort, users had a positive experience using our exertion game and they expressed their positive intention to use a system like this in their daily lives.

The exergame presented in [Silva11] has three input methods to control the game. Shooting and changing direction left and right are controlled by the keyboard while changing direction up and down is controlled by the bike. The 3 input methods can be replaced by our hand gesture system to control the game. However, the vocabulary of gestures recognized by the

system was reduced to keep the input from the bike, which will provide exercise or fitness for game players, while we replaced the keyboard input by our hand gesture system. Gestural interactions make exergame more interesting and immersive to manipulate objects. The key to our interactive exergame is providing the players with accurate visual feedback as to their hand gestures and the effects of their hand gestures upon objects in the virtual environment. During the exergaming program, a user obtains real-time and accurate feedback and encouragement on her/his progress, while the visual cue of the game encourages the user to do more, work harder, and achieve personal goals.

Despite a lot of recent achievements in using the Kinect sensor for human body tracking, it is still a challenge to employ Kinect for hand gesture recognition because of the low-resolution of the Kinect depth map of only 640×480 pixel [Ren11, Ren11b]. In [Ren11, Ren11b], hand postures were detected first by Kinect and then Finger-Earth Mover's Distance (FEMD) was used for hand gesture recognition. Although the Kinect can detect the hand, it does not come with a hand gesture recognition system [Bergh11]. Many popular technologies such as Microsoft Kinect, Sony PSP, and Nintendo DS and Wii contain gesture recognition in their consoles. Unluckily, only dynamic gestures such as waving and fist hitting are recognized by them so far [Wachs11]. Kinect tracks full body movement [Seatovic11, xbox12]. It can only track a player who is standing [xbox12]. Thus, it has difficulty recognizing a human's structure when sitting down. It is designed to detect and interpret motions [Seatovic11, xbox12]. Detection must be done within the range from the camera of Kinect. It doesn't work when the player gets too close or gets too far away [xbox12]. It has difficulty to set up because it needs a large empty space to detect the player [xbox12]. One player should stand six feet away from the Kinect sensor [xbox12]. It is also sensitive to lighting conditions [xbox12]. Our system can recognize static and dynamic hand gestures in real-time with high recognition rate without problems against the issues mentioned above.

Chapter 5

Hand Posture Recognition Using Principal Component Analysis

5.1 Introduction

Principal Component Analysis (PCA) [Pearson901] has been successfully applied in human face recognition using the concept of "eigenface" in [Turk91]. PCA can also be used for data compression, image compression, data dimension reduction, feature extraction, and facial expression classification [Jolliffe02, Cor08, Zha09]. The input image is projected to a new coordinate system consisting of a group of ordered eigenvectors. The M highest eigenvectors retain most of the variation presented in the original image set. The objective of PCA is to minimize the dimensionality of the image data to enhance efficiency by expressing the large 1-D vector of pixels constructed from 2-D hand image into the compact principal components of the feature space referred to as eigenspace projection. Eigenspace is computed by finding the eigenvectors of the covariance matrix obtained from a group of hand postures training images (vectors). These eigenvectors stand for the principal components of the training images [Conde03] and are frequently orthonormal to each other.

PCA based approaches normally contain two stages: training and testing. In the training stage, an eigenspace is established from the hand posture training images using PCA and the hand postures training images are mapped to the eigenspace for classification. In the testing stage, a hand posture test image is projected to the same eigenspace and classified by an appropriate classifier based on Euclidean distance. However, PCA lacks the detection ability. In addition, its performance degrades with scale, orientation and changes in lighting [Turk91].

In [Zhao09], a part-based PCA was proposed for facial feature extraction and a modified PCA reconstruction method was used for expression classification. The features were detected automatically by multi-step integral projection curves without being manually chosen and PCA was applied in the detected area instead of the whole face. The training set was divided into seven classes and PCA reconstruction was performed on each class independently. The

expression class can be recognized by measuring the distance between the input image and the reconstructed image.

A 3-D model-based tracking algorithm presented in [Cordea08] allows real-time recovery of 3-D position, orientation, and facial expressions of a moving head. The first stage of the training process consists of applying PCA in collected shape and texture vectors for the faces. The advantage of formulating eigenspace of combined modes for shape and texture of faces is that the anthropometric characteristics and facial expressions are directly obtained during the synthesis phase. The appearance model parameters control the shape (facial types and expressions) and texture in the normalized model frame according to eigenspace of combined modes (c). Additionally, the 3-D model projection in the image plane also depends on the 3-D pose transformations, namely, translations, rotations, and scaling $\gamma = (r_x\ r_y\ r_z\ t_x\ t_y\ z)$. During fitting, the pixels in the region of the image defined by (c, γ) were sampled and projected into the texture of the model normalized frame. The model fitting is treated as an optimization problem that minimizes the difference vectors between the new image and the one synthesized by the 3-D appearance model.

In this chapter, different hand postures were detected and tracked using our approach in Chapter 3 and recognized using principal component analysis (PCA) according to our approach described in [Dardas11b], which was used to achieve real-time performance and accurate classification of hand postures. With this approach, we can detect, track and recognize a set of bare hand postures. Since all of the training images were used with different scales, rotations and lighting conditions, the system showed reasonable robustness against scale, rotation, different illuminations and cluttered backgrounds. Therefore, the system can work in different environments.

The major contribution of the approach described in this chapter is that we have achieved real-time performance and accurate recognition for the bare hand postures with different scales, rotations and lighting conditions using Principal Component Analysis (PCA).

In [Zhao09], the performance of PCA with scale, orientation and lighting changes was not addressed because the training and testing stages were performed on The Japanese Female Facial Expression (JAFFE) Database, which have the same scale, orientation and lighting conditions. However, in our approach we addressed with scale, orientation and light changes to make our system invariant against them.

The algorithm in [Cordea08] aimed to recover the 3-D pose and facial expressions of a moving head. This method has drawbacks. First, extensive image processing is needed since a huge image database is required to deal with the entire characteristic expressions under several views. Second, the fitting process is sensitive to noise and variations in pose and expressions. However, in our approach a 2-D appearance based model was used. Accurate recognition and real-time performance have been achieved because of the simpler 2-D image features that are used.

This chapter is organized as follows: Section Two presents the mathematical modeling for hand posture recognition using a PCA approach based on the approach described in [Turk91]; Section Three describes our system in detail, and more specifically, the training stage and the testing stage including detection and tracking hand postures using our approach in [Dardas11] and recognition using PCA using our approach in [Dardas11b]; Section Four provides experimental results; Section Five compares between PCA method and Bag-of-Features method for hand posture recognition; Sections Six, Seven, and Eight describe how our system controls a videogame; the last section gives the summary of our method.

5.2 Mathematical Modeling of PCA for Hand Posture Recognition

This sections extends the mathematical modeling method used in [Turk91] for face recognition to the hand posture recognition.

Let f (x, y) be a two-dimensional function, where f is the intensity of the image point at that coordinates. Each image I can be denoted as a matrix of f (x, y) of dimension N × N.

Let F be the set of M training images of the same dimension denoted as an array of dimension: (N × N) × M

$$F = (I_1\ I_2\ I_3 \ldots\ldots\ldots\ldots I_M) \qquad (5.1)$$

These M images are transformed as vectors X_i, $1 \le i \le M$ of dimension N^2 where X_i is an $N^2 \times 1$ vector corresponding to the image I_i in the image space F. Now F becomes:

$$F = (X_1\ X_2 \ldots\ldots\ldots X_M) \qquad (5.2)$$

The mean image Ψ is computed by summing all the training images and dividing by the number of images with dimension $(N^2 \times 1)$ as below:

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} X_i \qquad (5.3)$$

The difference image is obtained by subtracting the mean image from all the training images X_i and storing in a vector Φ_i

$$\Phi_i = X_i - \Psi \qquad (5.4)$$

The main idea behind the eigenspace method in [Turk91] is to utilize the similarities between the different images. For this purpose, the covariance matrix C_v with dimension $N^2 \times N^2$ is calculated as:

$$C_v = \frac{1}{M} \sum_{i=1}^{M} \Phi_i \Phi_i^T = AA^T \qquad (5.5)$$

Where $A = [\Phi_1 \ \Phi_2 \ \Phi_3 \ldots\ldots\ldots\ldots \Phi_M]$ is of dimension $N^2 \times M$.

This covariance matrix AA^T dimension ($N^2 \times N^2$) will normally be a huge matrix, and full eigenvector computation is impractical.

In our case, we used 40 training images with a size of 160×120 pixels for each hand posture: fist, index, little and palm. The number of training images used was 4 × 40 = 160 images. Every training image is a point in a 160×120 = 19,200 dimensional space and the covariance matrix C_v is a matrix of 19,200 × 19,200 = 3.6864 × 10^8 elements. The covariance matrix $C_v = AA^T$ can have up to 160×120 = 19,200 eigenvalues and eigenvectors. However the rank of the covariance matrix C_v is limited by the number of training images. If there are M = 160 training images, there will be at most M =160 eigenvectors with non-zero eigenvalues.

5.2.1 Computation of the Eigenspace

To reduce the dimensionality of covariance matrix AA^T, let us compute the eigenvector ν_i of $B = A^T A$ of dimension M × M.

$$B = A^T A = \frac{1}{M} \sum_{i=1}^{M} \Phi_i^T \Phi_i \qquad (5.6)$$

The eigenvectors of the covariance matrix C_v are computed by using the matrix B. Then the eigenvector ν_i and the eigenvalue λ_i of B are obtained by solving the characteristic eigenvalue problem $| B - \lambda I | = 0$

$$B.\nu_i = \lambda_i . \nu_i \quad (5.7)$$

Substituting the value of B in equation (5.7):

$$A^T A.\nu_i = \lambda_i . \nu_i \quad (5.8)$$

Multiplying both sides with A in equation (5.8), the result is:

$$AA^T . A\nu_i = \lambda_i . A\nu_i \quad (5.9)$$

Substituting the value of $C_v = AA^T$ in equation (5.9), the result is:

$$C_v . A\nu_i = \lambda_i . A\nu_i \quad (5.10)$$

Assume $u_i = A\nu_i$ (5.11) , then

$$C_v . u_i = \lambda_i . u_i \quad , \quad (5.12)$$

Thus, AA^T and $A^T A$ have the same eigenvalues and their eigenvectors are related as follows:

$$u_i = A\nu_i \quad (5.11)$$

AA^T can have N^2 eigenvalues and eigenvectors.

$A^T A$ can have up to M eigenvalues and eigenvectors.

The M eigenvalues of $A^T A$ (along with their corresponding eigenvectors) are related to the M largest eigenvalues of AA^T (along with their corresponding eigenvectors).

Computation the M best eigenvectors of AA^T is by solving: $u_i = A\nu_i$

Then only M eigenvectors are kept (corresponding to the M largest eigenvalues).

5.2.2 Representing Training Images Using Eigenspace

Now, the training images are projected into the eigenspace and the weights (w_k) of every training image in the eigenspace are calculated. The weights of each training image are simply the dot product of each normalized training image with the M eigenvectors.

$$w_k = u_k^T . \Phi_i = u_k^T . (X_i - \Psi) \quad (5.12)$$

where k = 1,2,…….M.

All weights are converted in the form of a vector with dimension M × 1 as the following:

$$\Omega = [w_1, w_2, w_3 \ldots\ldots\ldots w_k]^T \quad (5.13)$$

Therefore, each normalized training image X_i is represented in this basis by a vector:

$$\Omega_i = [w_1^i, w_2^i, w_3^i \ldots\ldots\ldots w_M^i]^T \quad (5.14)$$

5.2.3 Hand Posture Recognition Using Eigenspace

To recognize an unknown hand posture image Γ in the testing stage, its weight (wi) is calculated by multiplying the eigenvector (u_i) of the covariance matrix (C_v) with difference image $(\Gamma - \Psi) = \Phi$

$$w_i = u_i^T . (\Gamma - \Psi) \quad (5.15)$$

Now the weight vector of the unknown image becomes

$$\Omega = [w_1, w_2, w_3 \ldots\ldots\ldots w_M]^T \quad (5.16)$$

The minimum Euclidean distance $\in_k$ between test image and each training image is defined by

$$\in_k = \text{minimum} \parallel \Omega - \Omega_k \parallel; k = 1,2,\ldots\ldots,M \quad (5.17)$$

where M is the number of training images.

Then, Γ is recognized as hand posture *k* from the training images set.

5.3 System Overview

Our hand posture recognition system consists of an offline training stage, which is shown in Figure 5.1 and an online testing stage, which is shown in Figure 5.3. The file that contains the mapping of every training image to the eigenspace was built in the training stage, and loaded in the testing stage for recognizing hand postures captured from a webcam.

5.3.1 Training Stage

The training stage is shown in Figure 5.1. PCA decreases the dimensionality of the training images set, leaving only those features that are essential for hand posture recognition as we discussed in the previous section. It transforms each N × N training image into a vector of length $N^2 \times 1$. The average hand posture vector Y is computed by adding all the training images and then dividing the result by the number of all the training images. Each training image is normalized by subtracting it from the mean Y. The main idea of PCA is based on the eigenvectors of the covariance matrix of the group of hand posture training images. These eigenvectors can be thought of as a group of features which characterize the variation between hand posture training images. Hand posture training images are projected into the subspace spanned by a hand posture space. The hand posture space is defined by the eigenspaces which are the eigenvectors of the set of hand postures. Each training image corresponds to each

eigenvector. An eigenvector can be shown as an eigenspace. Then in the testing stage, every detected hand posture is classified by comparing its position in hand posture space with the positions of every hand posture training images. Each hand posture image in the training set is represented a linear combination of the eigenspaces, which indicate a feature space that spans the variations among the known hand posture training images. The number of eigenspaces is equal to the number of hand posture images in the training set. The hand postures can also be approximated using only the best eigenspaces, which have the largest eigenvalues.

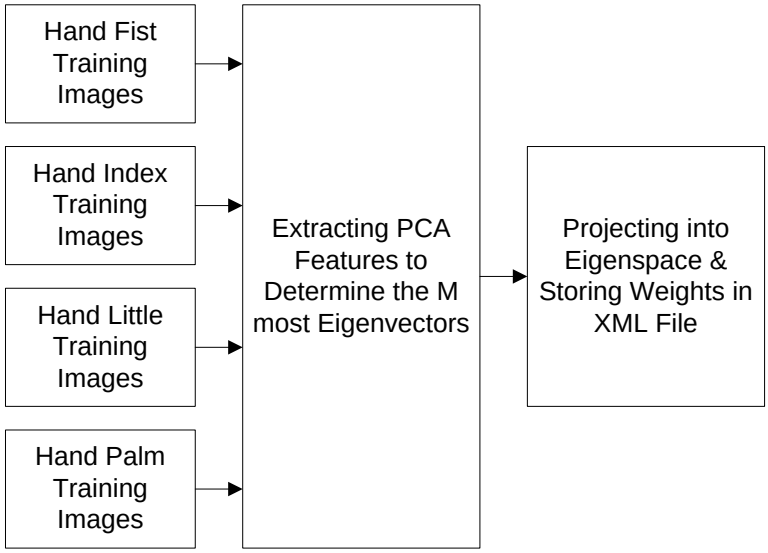


Figure 5.1: Training Stage.

Since PCA performance decreases with scale, orientation and light changes [Turk91], we used 40 training images with size of 160×120 pixels for each hand posture: fist, index, little and palm as shown in Figure 5.2 with different scales, rotations and lighting conditions to ensure invariance against these changes. The eigenvectors and eigenvalues were computed for the covariance matrix of the training images for all hand postures. The M highest eigenvectors were kept. Finally, the hand posture training images were projected, or mapped, into the eigenspace, and their weights were stored in XML file which is used in the testing stage. The weights of each training image are simply the dot product of each normalized training image with the M eigenvectors.

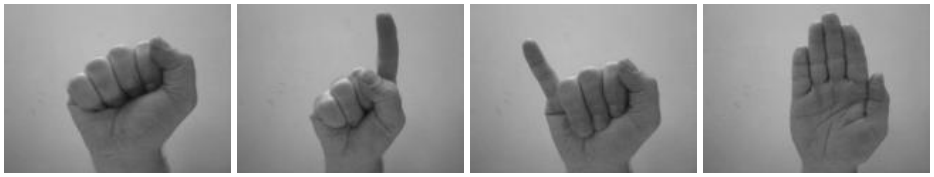


Figure 5.2: Hand postures used in the training stage.

5.3.2 Testing Stage

The testing stage is shown in Figure 5.3. Before capturing frames, we loaded the file, which contains the M highest eigenvectors and weights of training images. After capturing frames from a video camera or video file, we detected the face and subtracted it before using a skin detection and hand postures contours comparison algorithm because the skin detection will detect the face and the face's contours is close to the fist hand posture contours. To eliminate other skin like objects that may be detected in the image, the contours of the skin area had to comply with the contours of any hand posture template contours to detect and save the hand posture only in a small image (160×120 pixels). Then, the small image that contains the hand posture only was classified by comparing its position in hand posture space with the positions of known hand posture training images using xml file built in the training stage.

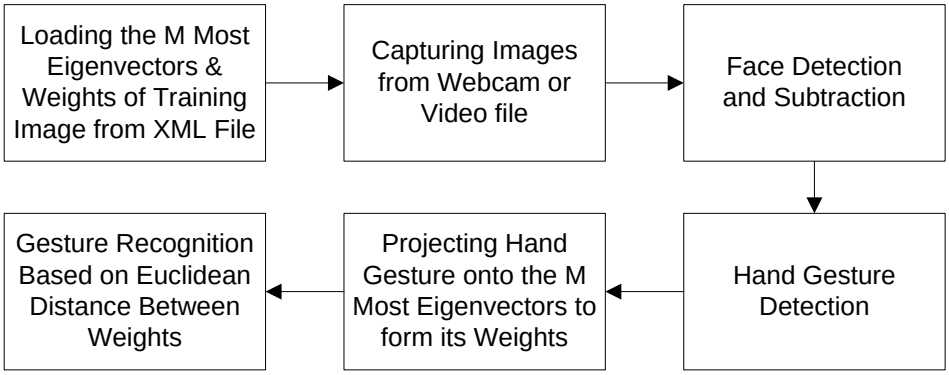


Figure 5.3: Testing stage.

5.3.2.1 Face Detection and Subtraction

We used the skin detection and contour comparison algorithm used in Chapter 3 for detecting the hand posture, and this algorithm can also detect the face since the face is also skin color and its contours are similar to the hand fist posture contours. To eliminate the face area, we

detected the face using the Viola and Jones method [Viola04] and then subtracted the face before applying the skin detection algorithm to detect the hand posture only by replacing face area with a black circle for every frame captured. In this way, we make sure that the skin detection process was applied for hand posture detection only as shown in Figure 5.4.

Figure 5.4: Face detection and subtraction.

5.3.2.2 Hand Posture Detection

Detecting and tracking the human hand in a cluttered background will enhance the performance of hand posture recognition using PCA in terms accuracy and speed because the extracted hand posture space will represent the hand posture only. Furthermore, we will not be confined with the frame resolution size captured from a video camera or video file, because we will always extract the PCA features of the small image (160×120 pixels) that contains the detected hand posture area only and not the complete frame. In this way the speed and accuracy of recognition will be the same for any frame size captured from a video camera such as 640×480, 320×240 or 160×120 and the system will be also robust against a cluttered background because we process the detected hand posture area only. The small image size (160×120 pixels) that contains the detected hand posture area only in the testing stage has to be matched with the training images size of training stage.

Before capturing the frames from a video camera, we loaded the four templates of hand postures used in Chapter 3 as shown in Figure 3.2, namely the fist, index, little and palm to extract their contours and saved them for comparison with the contours of skin detected area of every frame captured as we discussed in Chapter 3. After detecting skin area for every frame captured, we used a contour comparison of that area with the loaded hand posture template contours to eliminate other skin like objects existing in the image. If the contour comparison of detected skin area comply with one of the stored hand posture templates, a small image (160×120 pixels) will enclose the hand posture area only. The small image will be used for extracting the PCA features.

5.3.2.3 Hand Posture Recognition

As it was mentioned before, the XMLfile that contains the weights of hand posture training images in the eigenspace was loaded before capturing frames from a video camera and was used in recognition. The small image (160×120 pixels) that contains the detected hand posture only for every frame captured was converted into a PGM format and projected onto the M most eigenvectors for forming its weights (coefficient vector) using the XML file that contains the M most eigenvectors and weights of every hand posture training image in the eigenspace. Finally, the minimum Euclidean distance between the detected hand posture weights (coefficient vector) and the training weights (coefficient vector) of each training image was determined to recognize the hand posture.

5.4 Experimental Results

We tested the proposed PCA system for the case of four hand postures: the fist posture, the index posture, the little finger posture and the palm posture. The camera used for recording video files in our experiment is a low-cost camera Logitech like before that provides images with a resolution of 320×240, 15 frames-per second. This frame rate matches the real-time speed.

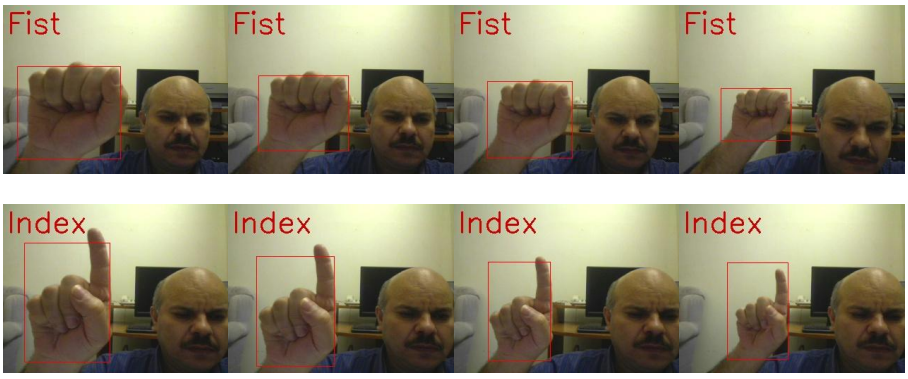
Ten video files with the resolution of 640×480 had been recorded for each hand posture: fist, index, little finger and palm using a commercial grade video camera. The length of each video file was 100 images. The hand postures were recorded with different scales, rotations and illumination conditions and a cluttered background. The test was run for the forty video files to evaluate the performance of the PCA classifier model for each posture. The trained PCA classifier showed an acceptable degree of robustness against scale, rotation, illumination and cluttered background.

Table 5.1 shows the performance of the proposed PCA classifier for each posture with testing against scale, rotation, illumination and a cluttered background. The recognition time did not increase with a cluttered background or with increasing the resolution of video file since the PCA features were extracted from the small image (160×120 pixels) that contains the detected hand posture only. We repeated the experiment with other ten video files for each hand posture with the resolution of 320×240 pixels, the same time was needed to recognize every frame with the same accuracy as 640×480 videos results because the PCA features were extracted in both cases from the small image that contains the hand posture only. Figure 5.5 shows some correct

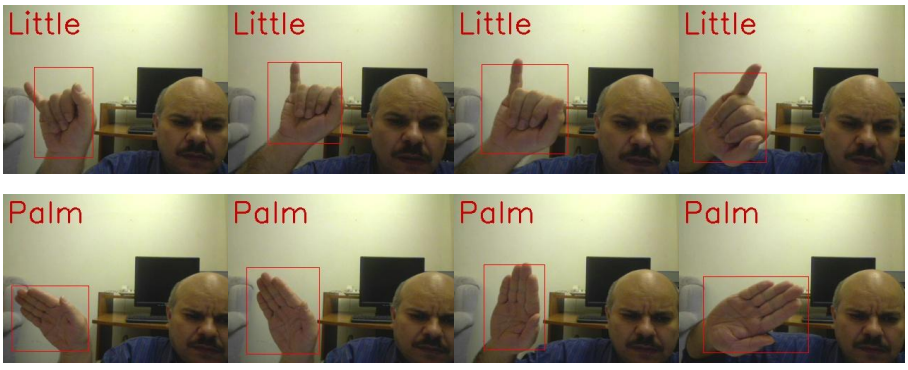
samples for hand posture recognition using PCA classifier for fist, index, little and palm postures with testing against scale, rotation, illumination and cluttered background.

TABLE 5.1: THE PERFORMANCE OF PCA CLASSIFIER WITH CLUTTERED BACKGROUND (640×480 PIXELS)

Posture Name	Number of frames	Correct	Incorrect	Recognition Time (Second/frame)
Fist	1000	933	67	0.055
Index	1000	928	72	0.055
Little	1000	912	88	0.055
Palm	1000	945	55	0.055



(a)



(b)

Figure 5.5: Hand posture detection and recognition with cluttered background against: (a) scale, (b) rotation transformations.

5.5 PCA vs. Bag-of-Features

From the performance results in Table 5.1 and Table 4.3 in Chapter 4, it is obvious that hand posture recognition using a bag-of-features approach has higher recognition accuracy rate and speed than using PCA. This is because Bag-of-Features is based on SIFT, which is invariant against scale, rotation, and partially invariant to illumination conditions while PCA is highly sensitive to them. Besides, increasing number of training images to build the multiclass SVM classifier will not affect the recognition speed of SVM classifier in the testing stage for bag-of-features model. With a PCA model, the more training images used with different scales, rotations and illumination conditions for increasing robustness against them, the more time required for the testing stage for deciding the minimum Euclidean distance between the test image and all of the training images for recognition. Therefore, there is no advantage to use the PCA method instead of Bag-of-Features method. However, our proposed system using a PCA model still achieves satisfactory real-time performance as well as high classification accuracy under variable conditions.

As previously described hand posture recognition systems have two challenges: hand detection and hand posture recognition. The hand detection stage has to be done before the posture recognition stage. According to section 4.4.2, the real-time hand posture recognition systems [Chen07, Marcel99, Chung09, Fang07b, Yun09, Ren10] suffered from inaccurate approaches for bare hand postures detection and tracking before recognition, which degraded the recognition accuracy. While our proposed systems using bag-of-features and PCA models achieved real-time performance and accurate recognition for the detected hand postures because we achieved real-time and accurate approach for bare hand postures detection and tracking before recognition.

5.6 Hand Gesture Interaction with Gaming

To show the effectiveness of our hand gesture recognition system using PCA described in this chapter for human-computer interaction (HCI), a game employing gesture-based interaction with a 3D gaming virtual environment (VE) was implemented in the following sections. We will present the “Fly over the city” game [Dardas11c] that can be controlled with a bare hand using our proposed system described in this chapter to provide natural, user-independent, user-friendly,

and real-time interaction between human and computers. The user in the game controls and directs the movements of a helicopter over a city by a set of hand gesture commands.

5.7 Fly Over The City Game

The game “Fly over the city” [Dardas11c] presented in this paper was developed using Java 3D. Java 3D is a client-side API which was developed by Sun Microsystems. It uses Java programming language to render interactive 3D graphics. Java 3D aims to enable quick development of complex 3D applications. It also enables fast and efficient implementation on a variety of platforms such as PCs, and workstations. Java 3D provides rich libraries for building shapes, describing behaviours, interacting with the user, and controlling rendering details. It allows software developer to build 3D scenes programmatically, or through loading 3D content from VRML, OBJ and other external files.

The game “Fly over the city” [Dardas11c] is composed of different objects that represent different parts of the city such as high buildings, town homes, stadium, markets, city hall, streets, and airport as shown in Figure 5.6. These objects are all static, and have been imported from Alice v2.0 libraries.

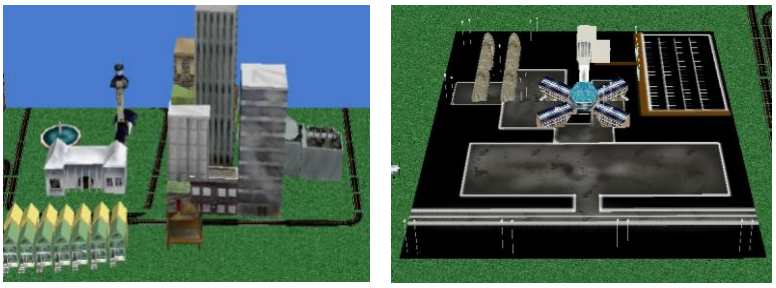


Figure 5.6: Part of the city.

Dynamic objects such as people and birds perform certain behaviors, where people walk in the streets and birds fly in the sky. The main object in this game is the Helicopter as shown in Figure 5.7(a). Creating the helicopter object requires to define its geometry, components, and sound. The game starts when the helicopter departs the airport as shown in Figure 5.7(b). The player uses the 4 arrows keys in keyboard to fly helicopter. To move up/down the helicopter, the user simply presses the up/down arrow key accordingly. To move right/left the helicopter, the user just presses the right/left arrow key accordingly. The user can also speed up or slow down the helicopter simply by pressing the letter key A/Z on the keyboard accordingly. If the user is

willing to hold helicopter in the sky, she/he does not press any keys on the keyboard. To fly it again, she/he presses any of the four arrows keys in keyboard.

(a) **(b)**

Figure 5.7: (a) Helicopter (b) Flying over the city.

While flying, the helicopter may collide with other objects such as birds, buildings, and high-rise buildings. A function called collision is used to address the collision. When the helicopter is about to collide with a building, it will move backward. When it is about to collide with a bird, it will move downwards. Figure 5.8 describes the state machine diagram for the helicopter.

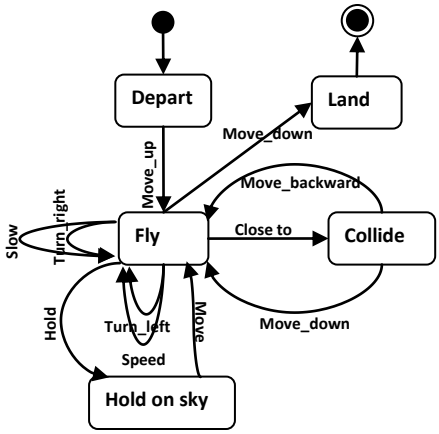


Figure 5.8: State machine diagram.

5.8 Flying Over The City Game Using Hand Gestures

We have two independent applications that can run at the same time: the helicopter game and the hand posture detection and recognition system described in this chapter. We first ran the 3D gaming virtual environment (VE) in which we can control the movements of helicopter using keyboard. Then, we ran our hand gesture recognition system independently to send events or commands to the keyboard to control movements of helicopter.

The user, who is characterized by the avatar helicopter, can explore through the 3D gaming VE to navigate the virtual world and interact with objects. A good exploration and interaction interface to the 3D gaming VE should be natural and intelligent. In the meantime, the hand gesture interface should also be easy to learn and effective to permit the user to execute the actions. The interactions should be direct and accurate in order that the user and the VE are in concurrence on what object is being interacted with.

Since the user is characterized by the avatar helicopter in the VE, we used our vision-based hand gesture recognition system developed in this chapter to control the helicopter by a set of hand gesture commands. In order to use hand gestures as a HCI device for VE applications, the hand gesture recognition system must satisfy the requirements in terms of real-time, accuracy, and robustness. The avatar helicopter can be moved forward/backward and turned left/right. In addition, the helicopter can be held in the sky and resume flying and its speed can be changed. To implement these functions, we mapped these commands by integrating the recognized hand gestures and hand motions directions according to Table 5.2. These gesture commands take into account the intuitiveness for the user to explore the 3D gaming VE. To move the helicopter up/down, the user moves her/his palm up/down accordingly in front of the camera. To move right/left the helicopter, the user moves her/his palm right/left accordingly. The user can also speed up or slow down helicopter simply by moving her/his palm forth/back accordingly in front of the camera. If the user is willing to hold the helicopter in the sky, she/he changes the hand posture from palm to fist. To fly it again in any direction, she/he changes the hand posture from fist to palm and then moves her/his palm to that direction. It can also be zoomed in/out of the screen by moving her/his fist back and forth. The screen will be zoomed in if the scale of the fist becomes bigger and vice versa.

We tested the exploration of the VE with the defined hand gesture commands. Our system shows the gesture-based interface can attain an improved interaction. Compared with controlling the avatar helicopter with the arrow keys on the keyboard, it is more intuitive and comfortable for users to use hand gestures to direct the movement of the avatar helicopter.

Table 5.2: Interaction commands performed by hand gestures for direct control of a helicopter.

Current State	Next State	Direction	Send Key	Event
palm	Palm	Up	{UP}	Move Up
Palm	Palm	Down	{DOWN}	Move Down
Palm	Palm	Right	{RIGHT}	Move Right
Palm	Palm	Left	{LEFT}	Move Left
Palm	Palm	Forth	A	Speed Up
Palm	Palm	Back	Z	Slow Down
Palm	Fist	Hold	H	Hold
Fist	Palm	Hold	Nothing	Hold
Fist	Fist	Forth	^{+}	Zoom In
Fist	Fist	Back	^{-}	Zoom Out

5.9 Summary

This chapter presented a real-time system, which includes detecting and tracking bare hand in cluttered background using skin detection and hand postures contours comparison algorithm after face subtraction, and recognizing hand postures using Principle Components Analysis (PCA). In the training stage, a set of hand postures images with different scales, rotation and lighting conditions were trained. Then, the eigenvectors of training images were determined, and the training weights were calculated by projecting each training image onto the most eigenvectors. In the testing stage, for every frame captured from a video camera, the hand posture was detected using our algorithm described in Chapter 3. Then, the small image that contains the detected hand posture was projected onto the eigenvectors of training images for forming its test weights. Finally, the minimum Euclidean distance was determined between the test weights and the training weights of each training image for recognizing the hand posture. The testing stage proves the effectiveness of the proposed scheme in terms of accuracy and speed as the coefficient vector represents the detected hand posture only. Experiments show that the system can achieve satisfactory real-time performance regardless of the frame resolution size,

and provides high classification accuracy above 90% under variable scale, orientation and illumination conditions and a cluttered background.

We showed how a game running on a computer can be controlled with hand based gestures using our system in this chapter to hand gesture recognition which can be utilized in natural interaction between human and computers. A vision-based hand gesture interface for interacting with objects in a 3D VE was designed and implemented. With this interface, the user in the “Fly over the city” game can control and direct the movements of the helicopter over the city by a set of hand gesture commands. Our system showed the hand gesture recognition interface can attain an enhanced and more intuitive and flexible method of interaction for the user than other HCI devices.

Chapter 6

Conclusions and Future Work

One of the main objectives of the interactive virtual environments is to provide natural, efficient, and flexible communication between the user and the computer. Human gestures including positions and movements of the fingers, hands, and arms represent one of the richest non-verbal communication modality, which together with the face expressions and head movements allow human users to interact naturally with the virtual environment. Gestures can be static, where the human takes on a specific pose, or dynamic, defined by motion.

Real-time vision based hand gesture recognition is one of the most challenging research areas in the human computer interaction field. Vision-based hand gesture recognition can rely on generic video cameras already available on a large variety of computers, tablets, smart phones, etc to recognize hand gestures. Furthermore, the non-contact visual gesture recognition is more natural, efficient, and convenient than other interactive human-computer systems like data gloves, keyboards or mice.

This thesis proposed an original hand gesture recognition system, which works under different lightning conditions with different transformations and cluttered background. When hand gesture recognition is combined with other technologies, such as voice recognition, it is possible to eliminate the traditional mouse and keyboard for user interaction with the computer. The hand gesture recognition systems will provide a more efficient and natural interaction, modality for artistic applications. Medical applications will also significantly benefit from using visual hand gesture interaction for an efficient access to significant patient data during medical procedures in a sterile environment. Another significant application will be in the sign language recognition for the deaf people.

The Human Computer Interface (HCI) proposed in this thesis uses the SIFT feature for hand gesture recognition with a bag-of-features model. After extracting the keypoints for all of the training images for different hand postures using the SIFT algorithm, a vector quantization is performed on the keypoints for all the training images using a k-means clustering model, which finally results in a visual word vocabulary. The quantization process maps the keypoints into a

unified dimensional “bag-of-words” vector for every training image or specifically, as a vector containing the (weighted) count of each visual word in that image, which is used as a feature vector input for building the multi-class SVM classifier model. Finally, a multi-class SVM classifier model was used to classify hand gestures captured from a webcam after constructing visual words vector from the clustering model for every frame captured.

This thesis discusses two-level architecture for the real-time hand gesture recognition system using only one camera as the input device. The low-level of the architecture focuses on, first, the hand posture detection and tracking algorithm proposed in Chapter 3 using face subtraction, skin-colour detection, and contour comparison algorithms and, second, the hand posture recognition system described in Chapter 4 using bag-of-features and multi-class Support Vector Machine (SVM) techniques. The high-level hand gesture recognition and motion analysis is based on the sequences of events recovered in the low level of the architecture. The recognized hand postures from consecutive frames are concatenated in a temporal sequence forming a hand gesture pattern. Sequences of hand postures are then classified using a syntactic grammar-based pattern recognition method. Distinct hand gesture patterns will have different temporal structures of hand postures. These sequences of hand postures are recognized using a prior defined syntax rules such as the regular grammars. The hand gesture recognition grammar was implemented in Chapter 4 using a finite state machine (FSM) parsing the sequence of hand postures. The gesture classifier recognizes the detected hand posture and the result of the recognition are fed into a FSM. The grammar of the FSM decides the real tasks. These tasks can be commands or events to be executed by the keyboard.

A second real-time visual hand gesture recognition system proposed in Chapter 5 consists of two modules: one for hand detection and tracking using face subtraction, skin detection and contours comparison algorithms proposed in Chapter 3, and the second one which performs gesture recognition using Principle Components Analysis (PCA). The developed PCA algorithm consists of two stages: a training stage where hand postures training images are processed to calculate the M highest eigenvectors and weights for every training image, and a testing stage where the weights of the detected hand gestures are matched with the weights of the training images for hand gesture recognition.

In order to validate the efficiency of our proposed hand gesture recognition systems for human-computer interaction, we developed two applications of gesture-based interaction system

for a 3D gaming virtual environment. With vision-based hand gesture interface built in Chapter 4, the user can issue game commands to point and shoot down targets in the exertion game. While with vision-based hand gesture interface built in Chapter 5, the user can control and direct the movements of the helicopter over the city by a set of hand gesture commands.

To conclude, in this thesis a real-time system was proposed that consists of three modules: hand detection and tracking using face subtraction, skin detection and contour comparison algorithm, posture recognition using bag-of-features and multiclass SVM, and a grammar that generates a large number of gesture commands by monitoring the scale of the detected hand posture, its movement direction, and the transitions among postures. In the training stage, after extracting the keypoints for all the training images using SIFT algorithm, VQ was performed on the keypoints for every training image using a k-means clustering to map them into a unified dimensional bag-of-words vector, which is used as input vector for building the multiclass SVM classifier model. Then, the multiclass SVM classifier will be used in the testing stage to classify the detected hand posture captured from a webcam after constructing visual words vector for keypoints of the small image (50 × 50 pixels) that contains the detected hand gesture only. The testing stage proves the effectiveness of the proposed scheme in terms of accuracy and speed as the keypoints extracted represent the detected hand gesture only. Experiments show that the system can achieve satisfactory real-time performance regardless of the frame resolution size as well as high classification accuracy of 96.23% under variable scale, orientation and illumination conditions, and cluttered background. Three important factors affect the accuracy of the system, which are the quality of the webcam in the training and testing stages, the number of the training images, and choosing number of clusters to build the cluster model.

For the future work, there are several potential enhancements that can extend this work. The interactive hand gestures recognition algorithms developed in this thesis were tested in a videogame application, but there are several other applications such medical systems, human-robot interaction, video editing, TV production, and sign language recognition, which can benefit from these new algorithms.

An interesting extension will be to explore the use of combined hand gesture recognition, face expression recognition, eye gaze, and voice recognition as a smart multi-modal interaction

method between people and service robots assisting the sick and disabled, and for environment surveillance applications.

Another interesting extension for our work will be tracking and recognizing two hands postures; this will provide a large set of gestures. Two hand tracking can be exploited by using one hand to make gestures for generating events and the other hand to carry out all the motions required. Two hand tracking can be integrated with virtual environments, to communicate in a more natural way with the computer and objects in the virtual environment. A more advanced extension could be hand tracking in a 3D space.

References

[Agarwal06] A. Agarwal and B. Triggs. “Hyperfeatures: Multilevel local coding for visual recognition”, In European Conference on Computer Vision (ECCV), 2006.

[Al-Amin06] M. Al-Amin Bhuiyan. On gesture recognition for human-robot symbiosis. In Proc. 15th IEEE International Symposium on Robot and Human Interactive Communication ROMAN 2006, pages 541-545, 2006.

[Albiol01] A. Albiol, L. Torres, E. J. Delp. An Unsupervised Color Image Segmentation Algorithm For Face Detection Applications. Proc. of International Conference on Image Processing 2001, 7-10 Oct 2001.

[Anders95] J. Anderson. An Introduction to Neural Networks. The MIT Press, 1995.

[Aran06] O. Aran, C. Keskin, L. Akarun. Computer Applications for Disabled People and Sign Language Tutoring. Proceedings of the Fifth GAP Engineering Congress, 26-28 April 2006, S¸anlıurfa, Turkey.

[Argyros06] A. Argyros, M. Lourakis, “Vision-Based Interpretation of Hand Gestures for Remote Control of a Computer Mouse”, Proc. of the workshop on Computer Human Interaction, 2006, pp. 40-51.

[Askar04] S. Askar, Y. Kondratyuk, K. Elazouzi, P. Kauff, O. Scheer. Vision-Based Skin-Colour Segmentation of Moving Hands for Real-Time Applications. Proc. Of. 1st European Conference on Visual Media Production, CVMP, London, United Kingdom, 2004.

[**Baklou08**] M. Baklouti, E. Monacelli, V. Guitteny, & S. Couvet, “Intelligent assistive exoskeleton with vision based interface”, in ‘Proceedings of the 6th international conference on Smart Homes and Health Telematics’, Vol. 5120. 123–135, 2008.

[Bannach07] D. Bannach, O. Amft, K. Kunze, E. Heinz,G. Tröster, P. Lukowicz. Waving real-hand gestures recorded by wearable motion sensors to a virtual car and driver in a mixed-reality parking game. In Proceedings of the Second IEEE Symposium on Computational Intelligence and Games (Honolulu, Apr. 1–5, 2007), 32–39.

[Barczak05] A. Barczak, F. Dadgostar, “Real-time hand tracking using a set of co-operative classifiers based on Haar-like features”, Res. Lett. Inf. Math. Sci., 2005, Vol. 7, pp 29-42.

[Barczak05b] A. L. C. Barczak, F. Dadgostar, and C. H. Messom, “Real-time hand tracking based on non-invariant features,” in Proc. IEEE Instrum. Meas. Technol. Conf., Ottawa, ON, Canada, 2005, pp. 2192–2197.

[Bay08] H. Bay, A. Ess, T. Tuytelaars, L. Gool, "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008.

[Bergh11] M. Bergh, D. Carton, R. Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. Gool, M. Buss, “Real-time 3D Hand Gesture Interaction with a Robot for Understanding Directions from Humans”, Proceedings of 20th IEEE International Symposium on Robot and Human Interactive Communication , Atlanta Georgia , 2011.

[Bhuiyan07] M. Bhuiyan, M. E. Islam, N. Begum, M. Hasanuzzaman, Chang Hong Liu, and H. Ueno. Vision based gesture recognition for human-robot symbiosis. In Proc. 10th international conference on Computer and information technology iccit 2007, pages 1-6, 2007.

[Binh05] N. Binh, E. Shuichi, T. Ejima. “Real-time hand tracking and gesture recognition system”. In Proceedings Graphics, Vision and Image Processing (GVIP), 2005.

[Blake99] A. Blake, B. North, and M. Isard. Learning multi-class dynamics. In Proc. Advances in Neural Information Processing Systems (NIPS), volume 11, pages 389-395, 1999.

[Bosch07] A. Bosch, X. Munoz, and R. Marti, “Which is the best way to organize/ classify images by content?” Image Vis. Comput., vol. 25, no. 6, pp. 778–791, Jun. 2007.

[Bradski98] G. Bradski. Computer-vision face tracking for use in a perceptual user interface. Intel Technology Journal (Q2 1998).

[Bramwe09] T. Bramwell. "E3: MS execs: Natal not derived from 3DV". Eurogamer. Eurogamer Network. Retrieved June 5, 2009.

[Brashe06] H. Brashear, V. Henderson, K. Park, H. Hamilton, S. Lee, T. Starner. American Sign Language recognition in game development for deaf children. In Proceedings of ACM SIGACCESS Conference on Assistive Technologies (Portland, OR, Oct. 23–25). ACM Press, New York, 2006, 79–86.

[Bretzn02] L. Bretzner, I. Laptev, and T. Lindeberg. Hand gesture recognition using multi scale colour features, hierarchical models and particle filtering. In Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on, pages 405-410, 2002.

[Brooks06] A. Brooks and C. Breazeal. Working with robots and objects: revisiting deictic reference for achieving spatial common ground. In HRI '06: Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction, pages 297-304, New York, NY, USA, 2006.

[Brown04] E. Brown, P. Cairns. "A grounded investigation of game immersion". In: CHI'04 extended abstracts on human factors in computing systems. ACM, New York, pp 1297–1300, 2004.

[Calinon07] S. Calinon. A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (Arlington, VA, 2007), 255–262.

[Chang01] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[Chen03] F. Chen, C.M. Fu, and C.L. Huang. Hand gesture recognition using a real-time tracking method and Hidden Markov Models. Image and Vision Computing, 21(8):745-758, 2003.

[Chen04] Q. Chen, E. M. Petriu, and X. Yang, "A comparative study of fourier descriptors and Hu's seven moments for image recognition," in Proc. IEEE Canadian Conference on Electrical and Computer Engineering, vol. 1, 2004, pp. 103–106.

[Chen07] Q. Chen, N. Georganas, E. Petriu, "Real-time Vision-based Hand Gesture Recognition Using Haar-like Features", IEEE Instrumentation and Measurement Technology Conference Proceedings, IMTC 2007.

[Chen07b] Y. Chen, K. Tseng. Developing a multiple-angle hand-gesture-recognition system for human-machine interactions. In Proceedings of the 33rd Annual Conference of the IEEE Industrial Electronics Society (Taipei, Nov. 5–8, 2007), 489–492.

[Chen08] Q. Chen. Real-Time Vision-Based Hand Tracking and Gesture Recognition. PhD thesis, University of Ottawa, 2008.

[Chen08b] Q. Chen, N. Georganas, and E. Petriu, "Hand gesture recognition using Haar-like features and a stochastic context-free grammar," IEEE Trans. Instrum. Meas., vol. 57, no. 8, pp. 1562–1571, Aug. 2008.

[Chen11] S. Chen, Z. Pan, M. Zhang and H. Shen, "A case study of user immersion-based systematic design for serious heritage games", Multimedia Tools and Applications, 2011.

[Chui92] C. Chui. An Introduction to Wavelets. Academic Press, San Diego, 1992.

[Chung09] W. Chung, X. Wu, and Y. Xu, "A real-time hand gesture recognition based on Haar wavelet representation," in Proc. IEEE Int. Conf. Robot. Biomimetics, 2009, pp. 336–341.

[Conde03] C. Conde, A. Ruiz and E. Cabello, "PCA vs Low Resolution Images in Face Verification", Proceedings of the 12th International Conference on Image Analysis and Processing (ICIAP'03), 2003.

[Cordea08] M. Cordea, E. Petriu, D.C. Petriu, "Three-Dimensional Head Tracking and Facial Expression Recovery Using an Anthropometric Muscle-Based Active Appearance Model", IEEE Trans. Instrum. Meas., vol. 57, no. 8, pp. 1578 – 1588, 2008.

[Corrad01] A. Corradini, "Real-time gesture recognition by means of hybrid recognizers," Lecture Notes in Computer Science, Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, vol. 2298, pp. 34–46, 2001.

[Cote06] M. Cote, P. Payeur, and G. Comeau. Comparative study of adaptive segmentation techniques for gesture analysis in unconstrained environments. In IEEE Int. Workshop on Imagining Systems and Techniques, pages 28-33, 2006.

[Crowley95] J. Crowley, F. Berard, and J. Coutaz. Finger tracking as an input device for augmented reality. In International Workshop on Gesture and Face Recognition, Zurich, June 1995.

[Cutler98] R. Cutler, M. Turk. View based Interpretation of Real-time Optical Flow for Gesture Recognition, 3rd IEEE Conf. on Face and Gesture Recognition, Nara, Japan, April 1998.

[Dardas10] N. Dardas, Q. Chen, N. Georganas, E. Petriu, "Hand Gesture Recognition Using Bag-of-Features and Multi-Class Support Vector Machine", HAVE 2010, 9th

IEEE International Workshop on Haptic Audio Visual Environments and Games, Oct. 16-17, 2010, Phoenix, AZ, USA.

[Dardas11] N. Dardas, N. Georganas, "Real-time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques", IEEE Transactions on Instrumentation and Measurement, VOL. 60, no. 11, pp. 3592 - 3607, Nov 2011.

[Dardas11b] N. Dardas, E. Petriu, "Hand Gesture Detection and Recognition Using Principal Component Analysis", CIMSA 2011, 4th IEEE International Conference on Computational Intelligence for Measurement Systems and applications, Sep. 19-21, 2011, Ottawa, ON, Canada.

[Dardas11c] N. Dardas, M. Alhaj, "Hand Gesture Interaction with a 3D Virtual Environment", The International Journal of ACM JORDAN, Vol.2, No.3, September 2011.

[Davis86] F.D. Davis, 1986. A technology acceptance model for empirically testing new end-user information systems: theory and results. Massachusetts Institute of Technology. Available at: http://dspace.mit.edu/handle/1721.1/15192.

[Davis94] J. Davis and M. Shah. Visual gesture recognition. Vision, Image, and Signal Processing, 141(2):101-106, 1994.

[Dollár05] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. "Behavior recognition via sparse spatio-temporal features", IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005.

[Dudley09] B. Dudley. "E3: New info on Microsoft's Natal -- how it works, multiplayer and PC versions". Brier Dudley's Blog. The Seattle Times. Retrieved June 3, 2009.

[Dumas98] J. Dumas. Usability Testing Methods: Subjective Measures: Part II - Measuring Attitudes and Opinions. October issue of Common Ground, The newsletter of the Usability Professionals' Association, 4-8, 1998.

[Eickel98] S. Eickeler, A. Kosmala, G. Rigoll. Hidden Markov Model based continuous online gesture recognition. In Int. Conference on Pattern Recognition, 1998.

[ElSawah08] A. El-Sawah, N. Georganas, and E. Petriu, "A prototype for 3-D hand tracking and gesture estimation," IEEE Trans. Instrum. Meas., vol. 57, no. 8, pp. 1627–1636, Aug. 2008.

[Estrada04] F. Estrada, A. Jepson, D. Fleet. Local Features Tutorial. Nov. 8, 2004.

[Fang07] G. Fang, W. Gao, & D. Zhao, 'Large-vocabulary continuous sign language recognition based on transition-movement models', Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 37(1), 1–9, 2007.

[Fang07b] Y. Fang, K. Wang, J. Cheng, and H. Lu, "A real-time hand gesture recognition method," in Proc. IEEE Int. Conf. Multimedia Expo, 2007, pp. 995–998.

[Fei04] H. Fei. A hybrid hmm/particle filter framework for non-rigid hand motion recognition. Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP'04). IEEE International Conference on, 5, 2004.

[Ford98] A. Ford and A. Roberts, "Color space conversions," Westminster Univ., London, U.K., Aug. 11, 1998.

[Frati11] V. Frati, D. Prattichizzo, "Using Kinect for hand tracking and rendering in wearable haptics", IEEE World Haptics Conference, 2011.

[Friedm97] J. H. Friedman. Another approach to polychotomous classification. Technical report. Department of Statistics, Stanford University, 1997.

[Garg09] P. Garg, A. Naveen, S. Sangeev. "Vision Based Hand Gesture Recognition." World Academy of Science, Engineering and Technology. 49.173 (2009): 972-977.

[Gavrila96] D. Gavrila and L. Davis. 3-D model-based tracking of humans in action: a multi-view approach. In Proc. IEEE Computer Vision and Pattern Recognition (CVPR), pages 73-80, 1996, 1996.

[Gilbert09] A. Gilbert, J. Illingworth, and R. Bowden. "Fast realistic multi-action recognition using mined dense spatio-temporal features", IEEE International Conference on Computer Vision (ICCV), 2009.

[Gonzal04] R. Gonzalez, R. Woods, and S. Eddins, Digital Image Processing Using MATLAB. Englewood Cliffs, NJ: Prentice-Hall, 2004.

[Goshorn08] R. Goshorn, D. Goshorn, and M. Kolsch. "The Enhancement of Low-Level Classifications for Ambient Assisted Living". Proceedings of the 2nd Workshop on Behaviour Monitoring and Interpretation, BMI'08, 2008.

[Graetz04] C. Graetzel, T. Fong, C. Grange, C. Baur. A non-contact mouse for surgeon-computer interaction. Technology and Health Care 12, 3 (Aug. 24, 2004), 245–257.

[Grange04] S. Grange, T. Fong, C. Baur. M/ORIS: A medical/operating room interaction system. In Proceedings of the ACM International Conference on Multimodal Interfaces (State College, PA, 2004). ACM Press, New York, 2004, 159–166.

[Grauman05] K. Grauman and T. Darrell. "The pyramid match kernel: Discriminative classification with sets of image features", IEEE International Conference on Computer Vision (ICCV), 2005.

[Greens01] H. Greenspan, J. Goldberger, and I. Eshet, "Mixture Model for Face Color Modeling and Segmentation," Pattern Recognition Letters, vol. 22, pp. 1525-1536, Sept. 2001.

[Gutta96] S. Gutta, H. Huang, F. Imam, and H. Wechsler. Face and hand gesture recognition using hybrid classifiers. In Proc. Second International Conference on Automatic Face and Gesture Recognition, pages 164-169, 1996.

[Habib06] H. Habib and M. Mufti. Real-time mono vision gesture based virtual keyboard system. Consumer Electronics, IEEE Transactions on, 52(4):1261-1266, 2006.

[Hang04] Z. Hang and R. Qiuqi. Visual gesture recognition with color segmentation and support vector machines. In Proc. 7th International Conference on Signal Processing ICSP '04, volume 2, pages 1443-1446 vol.2, 2004.

[Heap96] A. J. Heap and D. C. Hogg, "Towards 3-D hand tracking using a deformable model", In 2nd International Face and Gesture Recognition Conference, pages 140–145, Killington, USA,Oct. 1996.

[Hong06] S. Hong, N. A. Setiawan, and C. W. Lee. Human-robot interaction using context awareness and active plane model. In Proc. International Joint Conference SICE-ICASE, pages 4770-4775, 2006.

[Höysni05] J. Höysniemi, P. Hämäläinen, L. Turkki, T. Rouvi. Children's intuitive gestures in vision-based action games. Commun. ACM 48, 1 (Jan. 2005), 44–50.

[Hsu02] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines, IEEE 373 Transactions on Neural Networks, 13(2002), 415-425.

[Hu62] M. K. Hu, "Visual Pattern Recognition by Moment Invariant," IRE Trans. Info. Theory, vol. IT – 8, pp. 179– 187, Feb. 1962.

[Hua04] G. Hua and Y.Yes Wu. Multi-scale visual tracking by sequential belief propagation. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004.

[Hurst12] W. Hurst, C. Wezel, "Gesture-based interaction via finger tracking for mobile augmented reality", Multimedia Tools and Applications, 2012.

[Iglesi10] J. A. Iglesias, P. Angelov, A. Ledezma, A. Sanchis, Evolving classification of agents' behaviors: a general approach. Evolving Systems 2010;1(3):161-171.

[Ikizler07] N. Ikizler and P. Duygulu. Human action recognition using distribution of oriented rectangular patches. In HUMO07, pages 271–284, 2007.

[Imagawa00] K. Imagawa, H. Matsuo, R. Taniguchi, D. Arita, S. Lu, and S. Igi. Recognition of local features for camera-based sign language recognition system. In Proc. International Conference on Pattern Recognition, volume 4, pages 849–853, 2000.

[Ionescu05] B. Ionescu, D. Coquin, P. Lambert, V. Buzuloiu. Dynamic Hand Gesture Recognition Using the Skeleton of the Hand. EURASIP Journal on Applied Signal Processing 2005:13, 2101-2109.

[Ivanov97] Y. Ivanov, A. Bobick. "Probabilistic parsing in action recognition", 1997.

[Ivanov00] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. IEEE Trans. Pattern Anal. Mach. Intell., 22(8):852–872, 2000.

[Iwai99] Y. Iwai, H. Shimizu, and M. Yachida. Real-time context-based gesture recognition using hmm and automaton. In Proc. International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, pages 127-134, 1999.

[Foley96] J. Foley, Computer Graphics Principles and Practice (2nd edition in C), Addison Wesley, 1996.

[Jhuang07] H. Jhuang, T. Serre, L. Wolf, T. Poggio. "A biologically inspired system for action recognition", IEEE International Conference on Computer Vision, 2007.

[Jiang07] Y. Jiang, C. Ngo, and J. Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In ACM Int'l Conf. on Image and Video Retrieval, 2007.

[Jolliffe02] I. Jolliffe, "Principal component analysis", Springer-Verlag, 2002.

[Jun08] H. Jun and Z. Hua, "A real-time face detection method in human-machine interaction," in Proc. International Conference on Bioinformatics and Biomedical Engineering (ICBBE), 2008.

[Juan09] L. Juan, O. Gwun, " A Comparison of SIFT, PCA-SIFT and SURF", International Journal of Image Processing (IJIP), Volume(3): Issue (4), 2009.

[Juang91] B. Juang and L. R. Rabiner. Hidden Markov Models for speech recognition. Technometrics, 33:251-272, 1991.

[Junejo08] I. Junejo, E. Dexter, I. Laptev, and P. Pérez. "Cross-view action recognition from temporal self-similarities", In European Conference on Computer Vision (ECCV), 2008.

[Just06] A. Just, Y. Rodriguez, S. Marcel. "Hand posture classification and recognition using the modified census transform". In: Proceedings of the Seventh IEEE international conference on automatic face and gesture recognition (FG'06), University of Southampton, UK, pp 351–356, 2006.

[Kawara04] N. Kawarazaki. I. Hoya, K. Nishihara, T. Yoshidome. Cooperative welfare robot system using hand gesture instructions. Lecture Notes in Control and Information Sciences 306, Springer, Berlin, 2004, 143–153.

[Ke04] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In Proc. Of the IEEE Conf. on Computer Vision and Pattern Recognition, 2004.

[Kelly08] W. Kelly, A. Donnellan, and D. Molloy, "Screening for objectionable images: A review of skin detection techniques," in Proc. IMVIP, 2008, pp. 151–158.

[Ketteb05] S. Kettebekov, M. Yeasin, and R. Sharma. Prosody based audiovisual coanalysis for coverbal gesture recognition. Multimedia, IEEE Transactions on, 7(2):234-242, 2005.

[Kim96] J. Kim, W. Jang, and Z. Bien. A dynamic gesture recognition system for the korean sign language (ksl). Systems, Man, and Cybernetics, Part B, IEEE Transactions on, 26: 354-359, 1996.

[Kim06] C. Kim, J. H. Yi. An Optimal Chrominance Plane in the RGB Color Space for Skin Color Segmentation. International Journal of Information Technology vol.12 no.7, pp.73-81, 2006.

[Kläser08] A. Kläser, M. Marszalek, and C. Schmid. "A spatio-temporal descriptor based on 3D-gradients", In British Machine Vision Conference (BMVC), 2008.

[Kölsch04] M. Kölsch, M. Turk, T. Höllerer. Vision-based interfaces for mobility. In Proceedings of the International Conference on Mobile and Ubiquitous Systems (Boston, Aug. 22–26, 2004), 86–94.

[Kölsch04b] M. Kölsch and M. Turk, "Robust hand detection," in Proc. 6th IEEE Conference on Automatic Face and Gesture Recognition, 2004.

[Kölsch04c] M. Kölsch and M. Turk, "Analysis of rotational robustness of hand detection with a Viola-Jones detector," in Proc. International Conference on Pattern Recognition, vol. 3, 2004, pp. 107–110.

[Korten96] D. Kortenkamp, E. Huber, R. Bonasso. Recognizing and interpreting gestures on a mobile robot. In Proceedings of the 13th Conference on Artificial Intelligence, Aug. 4–8, 1996, Portland, OR.

[Kostom11] M. Kostomaj, B. Boh. "Design and evaluation of user's physical experience in an Ambient Interactive Storybook and full body interaction games". Multimedia Tools and Applications (August 2011), 54 (2), pg. 499-525, 2011.

[Laptev08] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. "Learning realistic human actions from movies", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008.

[Lazebn06] S. Lazebnik, C. Schmid, J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In Proc. Of IEEE Conference on Computer Vision and Pattern Recognition, 2006.

[Lee06] T. Lee, J. Mody. "Behavioral Classification", In Proceedings of the 15th European Institute for Computer Antivirus Research (EICAR) Annual Conference, 2006.

[Lee09] J. Lee. "Project Natal".Procrastineering. Retrieved June 6, 2009.

[Lement04] J. Lementec and P. Bajcsy. Recognition of arm gestures using multiple orientation sensors: gesture classification. In Proc. 7th International IEEE Conference on Intelligent Transportation Systems, pages 965-970, 2004.

[Li11] K. Li, Y. Du, Z. Fu. "TreeHeaven: a table game using vision-based gesture recognition". In Proceedings of the 2011 ACM symposium on the role of design in UbiComp research & practice (RDURP '11). ACM, New York, NY, USA, 11-14, 2011.

[Liang98] R. Liang and M. Ouhyoung, "A real-time continuous gesture recognition system for sign language," in Proc. 3rd International Conference on Automatic Face and Gesture Recognition, 1998, pp. 558–565.

[Licsar04] A. Licsar and T. Sziranyi. Hand gesture recognition in camera-projector system. In CVHCI04, pages 83-93, 2004.

[Lienhart02] Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In ICIP02, 2002.

[Likert32] R. Likert. "A Technique for the Measurement of Attitudes". Archives of Psychology, Vol. 140, No. 55, 1932.

[Lloyd82] S. Lloyd. "Least squares quantization in PCM".IEEE Transactions on Information Theory 28 (2): 129–137, 1982.

[Lowe04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. Int. J. Computer Vision, 60(2):91-110, 2004.

[Lu03] S. Lu, D. Metaxas, D. Samaras, and J. Oliensis, "Using multiple cues for hand tracking and model refinement," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2003, pp. 443–450.

[MacKay03] D. J. C. MacKay, Information Theory, Inference, and Learning Algorithms. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[Maggio95] C. Maggioni. GestureComputer - new ways of operating a computer. In Int. Workshop on Automatic Face and Gesture Recognition, pages 166-171, Zurich, Switzerland, 1995.

[Malik02] S. Malik, C. McDonald, and G. Roth, "Hand tracking for interactive pattern-based augmented reality," in Proc. International Symposium on Mixed and Augmented Reality, 2002.

[Malric08] F. Malric, A. El Saddik, and N.D. N. D. Georganas. Artificial neural networks for real-time optical hand posture recognition using a color-coded glove. In Computational Intelligence for Measurement Systems and Applications, 2008.

[Marcel99] S. Marcel, "Hand posture recognition in a body-face centered space," in Proc. Conf. Human Factors Comput. Syst. (CHI), 1999, pp. 302–303.

[Marcel00] S. Marcel. Hand Gesture Recognition using Input Output Hidden Markov Models, Proc. 4th IEEE Int. Conf, 2000.

[Marsza06] M. Marszalek and C. Schmid. "Spatial weighting for bag-of-features", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.

[Marsza09] M. Marszalek, I. Laptev, and C. Schmid. "Actions in context", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[Martin98] J. Martin, V. Devin, and J. Crowley. Active hand tracking. In IEEE Conference on Automatic Face and Gesture Recognition, pages 573-578, Nara, Japan, 1998.

[Mckenna04] S. Mckenna and K. Morrison, "A comparison of skin history and trajectory-based representation schemes for the recognition of user- specific gestures," Pattern Recognition, vol. 37, pp. 999–1009, 2004.

[Messom06] H. Messom and A.L.C. Barczak. Fast and efficient rotated haar-like features using rotated integral images. In Australian Conference on Robotics and Automation (ACRA2006), 2006.

[Micilo04] A. Micilotta and R. Bowden, "View-based location and tracking of body parts for visual interaction," in Proc. BMVC, 2004, pp. 849–858.

[Murase95] H. Murase and S. Nayar. Visual learning and recognition of 3-d objects from appearance. International Journal of Computer Vision, 14:5-24, 1995.

[Nair02] V. Nair, J. Clark. Automated Visual Surveillance Using Hidden Markov Models, In VI02, pp 88, 2002.

[Nallap07] K. Nallaperumal, S. Ravi, C. N. K. Babu, R. K. Selvakumar, A. L. Fred, C. Seldev, and S. S. Vinsley, "Skin detection using color pixel classification with application to face detection: A comparative study," in Proc. IEEE Int. Conf. Comput. Intell. Multimedia Appl., 2007, vol. 3, pp. 436–441.

[Nickel07] K. Nickel, & R. Stiefelhagen, 'Visual recognition of pointing gestures for human-robot interaction', Image Vision Computing 25(12), 1875–1884, 2007.

[Niebles08] J. Niebles, H.Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. International Journal of Computer Vision. 79(3): 299-318. 2008.

[Nishik03] A. Nishikawa, T. Hosoi, K. Koara, D. Negoro, A. Hikita, S. Asano, H. Kakutani, F. Miyazaki, M. Sekimoto, M. Yasui, Y. Miyake, S. Takiguchi, M. Monden. FAce MOUSe: A novel human-machine interface for controlling the position of a laparoscope. IEEE Transactions on Robotics and Automation 19, 5 (Oct. 2003), 825–841.

[Ntalia10] K. Ntalianis, A. Doulamis, N. Tsapatsoulis and N. Doulamis, "Human action annotation, modeling and analysis based on implicit user interaction", Multimedia Tools and Applications, Volume 50, Number 1, Pages 199-225, 2010.

[Oka02] K. Oka, Y. Sato, and H. Koike, "Real-time fingertip tracking and gesture recognition," in Proc. IEEE Computer Graphics and Applications, vol. 22, no. 6, 2002, pp. 64–71.

[Ong04] E. Ong and R. Bowden, "Detection and segmentation of hand shapes using boosted classifiers," in Proc. IEEE 6th International Conference on Automatic Face and Gesture Recognition, 2004, pp. 889–894.

[Pan10] Z. Pan, Y. Li, M. Zhang, C. Sun, K. Guo, X. Tang, S. Zhou. "A real-time multi-cue hand tracking algorithm based on computer vision". In: Virtual reality conference (VR). IEEE, Piscataway, pp 219–222, 2010.

[Pavlov97] V. Pavlovic, R. Sharma, and T. Huang, "Visual interpretation of hand gestures for human-computer interaction: a review," IEEE Trans. Pattern Anal. Machine Intell., vol. 19, no. 7, pp. 677–695, 1997.

[Pearson901] K. Pearson. "On Lines and Planes of Closest Fit to Systems of Points in Space" (PDF). Philosophical Magazine 2 (6): 559–572, 1901.

[Pentla97] A. Pentland, D. Becker. Sensei: A Real-Time Recognition, Feedback, and Training System for T'ai Chi Gestures. Masters Thesis. Harvard University, Cambridge, MA, 1997.

[Phung01] S.L. Phung, D. Chai, and A. Bouzerdoum, "A Universal and Robust Human Skin Color Model Using Neural Networks," Proc. INNS-IEEE Int'l Joint Conf. Neural Networks, vol. 4, pp. 2844-2849, July 2001.

[Poskan91] J. Poskanzer, The PBMPLUS portable bitmap toolkit, 1991.

[Rabiner86] L. Rabiner and B. Juang. An introduction to hidden markov models. ASSP Magazine, IEEE, 3:4-16, 1986.

[Raheja11] J. Raheja, A. Chaudhary, K. Singal, "Tracking of Fingertips and Centres of Palm using KINECT", IEEE Third International Conference on Computational Intelligence, Modelling & Simulation, 2011.

[Rajko07] S. Rajko, G. Qian, T. Ingalls, and J. James. Real-time gesture recognition with minimal training requirements and on-line learning. In Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR '07, pages 1-8, 2007.

[Ramamo03] A. Ramamoorthy, N. Vaswani, S. Chaudhury, and S. Banerjee, "Recognition of dynamic hand gestures," Pattern Recognition, vol. 36, pp. 2069–2081, 2003.

[Rao06] V. Rao and C. Mahanta. Gesture based robot control. In Proc. Fourth International Conference on Intelligent Sensing and Information Processing ICISIP 2006, pages 145-148, 2006.

[Rausch02] I. Rauschert, P. Agrawal, R. Sharma, S. Fuhrmann, I. Brewer, A. MacEachren. "Designing a human-centered, multimodal GIS interface to support emergency management". In Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems (McLean, VA, Nov. 8–9). 2002.

[Rehg94] J. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: an application to human hand tracking. Proc. European Conference on Computer Vision, 2:35-46, 1994.

[Rehg94b] J. Rehg and T. Kanade. Digiteyes: Vision-based hand tracking for human-computer interaction. In Workshop on Motion of Non-Rigid and Articulated Bodies, pages 16-24, Austin Texas, November 1994.

[Rehg95] J. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In Proc. International Conference on Computer Vision (ICCV), pages 612-617, 1995.

[Remagn01] P. Remagnino, G. A. Jones. Classifying surveillance events from attributes and behaviour. In Proceedings of the British Machine Vision Conference (BMVC), 2001.

[Ren10] Y. Ren and C. Gu, "Real-time hand gesture recognition based on vision," in Proc. Edutainment, 2010, pp. 468–475.

[Ren11] Z. Ren, J. Yuan, Z. Zhang, "Robust Hand Gesture Recognition Based on Finger-Earth Mover's Distance with a Commodity Depth Camera", Proc. of ACM Intl. Conf. on Multimedia (ACM MM 11), Scottsdale, Arizona, USA, Nov. 28-Dec. 1, 2011.

[Ren11b] Z. Ren, J. Meng, J. Yuan, Z. Zhang, "Robust Hand Gesture Recognition with Kinect Sensor", Proc. of ACM Intl. Conf. on Multimedia (ACM MM 11) (Technical Demo), Scottsdale, Arizona, USA, Nov. 28-Dec. 1, 2011.

[Rogalla02] O. Rogalla, M. Ehrenmann, R. Zöllner, R. Becher, R. Dillmann. Using gesture and speech control for commanding a robot assistant. In Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication (Berlin, Sept. 25–27, 2002), 454–459.

[Sabeti07] L. Sabeti, Q. M. Jonathan Wu. High-Speed Skin Color Segmentation for Real-Time Human Tracking. IEEE Internatinonal Conference on Systems, Man and Cybernetics, ISIC2007, Montreal, Canada, 7-10 Oct. 2007.

[Savare06] S. Savarese, J. Winn, and A. Criminisi. "Discriminative object class models of appearance and shape by correlations", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.

[Schlöm08] Schlömer, T., Poppinga, B., Henze, N., and Boll,S. Gesture recognition with a Wii controller. In Proceedings of the Second International Conference on Tangible and Embedded Interaction (Bonn, Germany, Feb. 18–20). ACM Press, New York, 2008, 11–14.

[Schüldt04] C. Schüldt, I. Laptev, and B. Caputo. "Recognizing human actions: A local SVM approach", International Conference on Pattern Recognition (ICPR), 2004.

[Seatovic11] D. Seatovic, V. Meiser. "Ergonomic Touch-Less Control of SCATh Application", Proceedings of the 2011 SCATh Joint Workshop on New Technologies for Computer/Robot Assisted Surgery, 2011.

[Shimada98] N. Shimada, Y. Shirai, Y. Kuno, and J. Miura. Hand gesture estimation and model refinement using monocular camera - ambiguity limitation by inequality

constraints. In IEEE Int. Conf. on Face and Gesture Recognition, pages 268-273, Nara, Japan, 1998.

[Sigal04] L. Sigal, S. Sclaroff, and V. Athitsos. Skin color-based video segmentation under time-varying illumination. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 26 (7):862-877, July 2004.

[Silva11] J. Silva, A. El Saddik. An adaptive Game-Based exercising framework. Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS), 2011.

[Sipser97] M. Sipser. Thoery of Computation. PWS Publishing Company, Massachusetts, 1997.

[Sirovi87] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. Journal of the Optical Society of America, 4:519-524, March 1987.

[Shi10] L. Shi, Y. Wang, and J. Li, "A real-time vision-based hand gestures recognition system," in Proc. 5th ISICA, 2010, pp. 349–358.

[Song11] Y. Song, S. Tang, Y. Zheng, T. Chua and Y. Zhang, "Exploring probabilistic localized video representation for human action recognition", Multimedia Tools and Applications, 2011.

[Sparac08] F. Sparacino, "Natural interaction in intelligent spaces: Designing for architecture and entertainment", Multimedia Tools and Applications, 2008.

[Starner97] T. Starner , J. Weaver, A. Pentland , "A wearable computer based American sign language recognizer", Proc. IEEE Int. Symp. Wearable Computing, pp. 130–137, Oct. 1997.

[Starner00] T. Starner, B. Leibe, B. Singletary, J. Pair. Mindwarping: Towards creating a compelling collaborative augmented reality game. In Proceedings of the Fifth International Conference on Intelligent User Interfaces (New Orleans, Jan. 9–12). ACM Press, New York, 2000, 256–259.

[Stenger01] B. Stenger, P. Mendonca, & R. Cipolla, "Model-Based 3D Tracking of an Articulated Hand". In IEEE Conference on Computer Vision and Pattern Recognition, 310–315, 2001.

[Stenger01b] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee, and J. Buhmann. Topology free hidden markov models: Application to background modeling. IEEE International Conference on Computer Vision, 1:294-301, 2001.

[Stenger06] B. Stenger, "Template based Hand Pose recognition using multiple cues", In Proc. 7th Asian Conference on Computer Vision: ACCV 2006.

[Su00] M. Su. A fuzzy rule-based approach to spatio-temporal hand gesture recognition. IEEE Transactions on Systems, Man, and Cybernetics Part C, 30(2):276-281, 2000.

[Takaha09] D. Takahashi. "Microsoft games exec details how Project Natal was born".VentureBeat. Retrieved June 6, 2009.

[Takaha11] M. Takahashi, M. Fujii, M. Naemura and S. Satoh, "Human gesture recognition system for TV viewing using time-of-flight camera", Multimedia Tools and Applications, 2011.

[Terril99] J. Terrillon, S. Akamatsu. Comparative Performance of Different Chrominance Spaces for Color Segmentation and Detection of Human Faces in Complex Scene Images. Proc. of Vision Interface, pp.180-187, Trois-Rivieres, Canada, May 1999.

[Tokatl05] A. Tokatlı, U. Halıcı. 3D Hand Tracking in Video Sequences. MSc Thesis, September 2005, Middle East Technical University.

[Travie03] C. Travieso, J. B. Alonso, and M. A. Ferrer. Sign language to text by svm. In Proc. Seventh International Symposium on Signal Processing and Its Applications, volume 2, pages 435-438 vol.2, 2003.

[Triesch98] J. Triesch, C. Malsburg. Robotic gesture recognition by cue combination. Gesture and Sign Language in Human-Computer Interaction. Lecture Notes in Computer Science. Springer, Berlin, 1998, 233–244.

[Turk91] M. Turk and A. Pentland, "Eigenfaces for recognition", Journal of Cognitive Neuroscience, 3(1), 1991.

[Utsumi98] A. Utsumi and J. Ohya. Image segmentation for human tracking using sequential-image-based hierarchical adaptation. In Proc. IEEE Computer Vision and Pattern Recognition (CVPR), pages 911-916, 1998.

[Vapnik98] V. Vapnik. Statistical Learning Theory. Wiley-Interscience, 1998.

[Varona09] J. Varona, A. Jaume-i-Capo, J. Gonzalez, F. Perales. "Toward natural interaction through visual recognition of body gestures in real-time". Interact. Comput. 21, 1-2 (January 2009), 3-10, 2009.

[Viola01] P. Viola and M. Jones. Robust real-time object detection. In IEEE Workshop on Statistical and Computational Theories of Vision, Vancouver, Canada, 2001.

[Viola04] P. Viola, M. Jones. Robust Real-Time Face Detection. International Journal of Computer Vision 57, 2 (May 2004), 137–154.

[Wachs08] J. Wachs, H. Stern, Y. Edan, M. Gillam, C. Feied, M. Smith, J. Handler. A hand-gesture sterile tool for browsing MRI images in the OR. Journal of the American Medical Informatics Association 15, 3 (May– June 2008), 321–323.

[Wachs11] J. Wachs, M. Kölsch, H. Stern, Y. Edan. "Vision-Based Hand-Gesture Applications", Communications of the ACM, Volume 54 Issue 2, February 2011.

[Wang07] Y.Wang, P. Sabzmeydani, and G. Mori. Semi-latent dirichlet allocation: A hierarchical model for human action recognition. In HUMO07, pages 240–254, 2007.

[Wang08] C. Wang, K. Wang, "Hand Gesture recognition using Adaboost with SIFT for human robot interaction," Springer Berlin, ISSN 0170-8643, Volume 370, 2008.

[Weng10] C. Weng, Y. Li, M. Zhang, K. Guo, X. Tang, Z. Pan. "Robust hand posture recognition integrating multi-cue hand tracking". In: Proceedings of the entertainment for education, and 5th international conference on E-learning and games. Springer, New York, pp 497–508, 2010.

[Weston99] J. Weston, C. Watkins. Multi-class support vector machines. Proceedings of ESANN99, M. Verleysen, Ed., Brussels, Belgium, 1999.

[Willems08] G. Willems, T. Tuytelaars, and L. Van Gool. "An efficient dense and scale invariant spatio-temporal interest point detector", In European Conference on Computer Vision (ECCV), 2008.

[Wong07] S.Wong, T. Kim, and R. Cipolla. Learning motion categories using both semantic and structural information. In CVPR07, pages 1–6, 2007.

[Wu99] Y. Wu and T. S. Huang, "Human Hand Modeling, Analysis and Animation in the Context of HCI", in IEEE International Conference Image Processing". Kobe, Japan, 1999.

[Wu99b] Y. Wu and T. T. Huang. Capturing human hand motion: A divide-and-conquer approach. In Proc. International Conference on Computer Vision (ICCV), pages 606-611, Greece, 1999.

[Wu00] P. Wu, B. S. Manjunath, S. Newsam, and HD Shin. A texture descriptor for browsing and similarity retrieval. Signal Processing: Image Communication, 16(1-2):33-43, 2000.

[Wu01] Y. Wu, J. Lin, and T. Huang, "Capturing Natural Hand Articulation". In IEEE International Conference on Computer Vision II, 426–432, 2001.

[Wu01b] Y. Wu and T. S. Huang, "Hand modeling analysis and recognition for vision-based human computer interaction," IEEE Signal Processing Magazine, Special Issue on Immersive Interactive Technology, vol. 18, no. 3, pp. 51–60, 2001.

[Wu02] Y. Wu and T. S. Huang, "Non-stationary color tracking for vision-based human computer interaction," IEEE Trans. on Neural Networks, vol. 13, no. 4, pp. 948–960, 2002.

[xbox12] "Get started with kinect - xbox.com." http://www.xbox.com/en-US/Kinect/GetStarted, 2012.

[Yang98] J. Yang, W. Lu, and A. Waibel, "Skin-color modeling and adaptation," in Proc. ACCV, 1998, pp. 687–694.

[Yao02] J. Yao and J. R. Cooperstock, "Arm gesture detection in a classroom environment," in Proc. IEEE Workshop on Applications of Computer Vision, 2002, pp. 153–157.

[Yao04] Y. Yao, M. Zhu, Y. Jiang, and G. Lu. A bare hand controlled ar map navigation system. In Proc. IEEE International Conference on Systems, Man and Cybernetics, volume 3, pages 2635-2639 vol.3, 2004.

[Yeffet09] L. Yeffet and L. Wolf. "Local trinary patterns for human action recognition", IEEE International Conference on Computer Vision (ICCV), 2009.

[Yin06] **X.** Yin, X. Zhu. Hand-posture recognition in gesture-based human-robot interaction. In Proceedings of the IEEE Conference on Industrial Electronics and Applications (Singapore, May 24–26, 2006), 1–6.

[Yuan05] Y. Yuan, Y. Ying, and K. L. Barner. Tactile gesture recognition for people with disabilities. In Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05), volume 5, pages v/461-v/464 Vol. 5, 2005.

[Yun09] L. Yun and Z. Peng, "An automatic hand gesture recognition system based on Viola-Jones method and SVMs," in Proc. 2nd Int. Workshop Comput. Sci. Eng., 2009, pp. 72–76.

[Zabulis09] X. Zabulis, H. Baltzakis and A. Argyros. "Vision-based hand gesture recognition for human- computer interaction". Chapter 34, in "The Universal Access Handbook", Lawrence Erlbaum Associates, Inc. (LEA), 2009.

[Zarit99] B. Zarit, B. Super, and F. Quek, "Comparison of five color models in skin pixel classification," in Proc. ICCV Int. Workshop Recog., Anal. Tracking Faces Gestures Real-Time Syst., 1999, pp. 58–63.

[Zhang01] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer, "Visual panel: Virtual mouse keyboard and 3D controller with an ordinary piece of paper," in Proc. Workshop on Perceptive User Interfaces, 2001.

[Zhao06] W. Zhao, Y. Jiang, and C. Ngo. Keyframe retrieval by keypoints: Can point-to-point matching help? In Proc. of 5th Int'l Conf. on Image and Video Retrieval, pages 72-81, 2006.

[Zhao08] Z. Zhao , A. Elgammal. "Spatiotemporal Pyramid Representation for Recognition of Facial Expressions and Hand Gestures", IEEE International Conference on Automatic Face & Gesture Recognition, 2008.

[Zhao09] Y. Zhao, X. Shen, N. Georganas, E. Petriu, "Part-based PCA for Facial Feature Extraction and Classification," Proc. HAVE 2009 - IEEE Int. Workshop on Haptic, Audio and Visual Environments and Games, pp. 99-104, Italy, Nov. 2009.

[Zhou03] H. Zhou, T. Huang, "Tracking articulated hand motion with Eigen dynamics analysis", In Proc. Of International Conference on Computer Vision, Vol 2, pp. 1102-1109, 2003.

[Zhu04] Q. Zhu, C.-T. Wu, K.-T. Cheng, and Y.-L. Wu, "An adaptive skin model and its application to objectionable image filtering," in Proc. ACM Multimedia, 2004, pp.

Appendix 1. Questionnaire

For each one of the following statements, please mark the option below that closest reflects your level of agreement with the statement.

1) Moving the crosshairs left and right with hand gestures was easy to accomplish.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

2) I did not perceive any delay when moving the crosshairs left and right using hand gestures

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

3) I had difficulty hitting the targets because the application would not respond all the time to my left/right hand gestures.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

4) Shooting in the game by using hand gestures was easy

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

5) I did not perceived delays when shooting using the hand gesture

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

6) I had difficulty hitting the targets because the application would not respond all the time to my ‘shoot’ hand gesture

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

7) I felt comfortable interacting with the game by means of hand gestures

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

8) Playing the game using hand gestures felt natural and I was able to focus on playing the game

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

9) If I wanted to use an exercise bike on a regular basis I would rather use this exergame with the hand gestures, instead of a traditional bike

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]

10) The whole experience with the exergame and hand gesture interface was awkward and I would generally avoid using it for my exercising

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
[ ]	[ ]	[ ]	[ ]	[ ]