

REAL-TIME DYNAMIC HAND SHAPE GESTURE CONTROLLER

by

RAJESH RADHAKRISHNAN

Presented to the Faculty of the Graduate School of  
The University of Texas at Arlington in Partial Fulfillment  
of the Requirements  
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2011

## ACKNOWLEDGEMENTS

I would like to thank my supervising professor Dr. Vassilis Athitsos for his guidance and invaluable advice during the course of my Master's studies. I would also like to thank Dr. Venkat Devarajan, without his help I wouldn't have taken the research option. I wish to thank my other committee member Dr. Alan Davis for his interest in my research and for taking time to serve in my dissertation committee.

I am grateful to all the teachers who taught me and guided me. I would also like to thank all of my friends who spent their time in helping me collect data for my thesis work.

November 14, 2011

## ABSTRACT

### REAL-TIME DYNAMIC HAND SHAPE GESTURE CONTROLLER

RAJESH RADHAKRISHNAN, M.S.

The University of Texas at Arlington, 2011

Supervising Professor: Dr. Vassilis Athitsos

The main objective of this thesis is to build a real time gesture recognition system which can spot and recognize specific gestures from continuous stream of input video. We address the recognition of single handed dynamic gestures. We have considered gestures which are sequences of distinct hand poses. Gestures are classified based on their hand poses and its nature of motion. The recognition strategy uses a combination of spatial hand shape recognition using chamfer distance measure and temporal characteristics through dynamic programming. The system is fairly robust to background clutter and uses skin color for tracking.

Gestures are an important modality for human-machine communication, and robust gesture recognition can be an important component of intelligent homes and assistive environments in general. Challenging task in a robust recognition system is the amount of unique gesture classes that the system can recognize accurately. Our problem domain is two dimensional tracking and recognition with a single static camera. We also address the reliability of the system as we scale the size of gesture vocabulary.

Our system is based on supervised learning, both detection and recognition uses the existing trained models. The hand tracking framework is based on non-parametric histogram bin based approach. A coarser histogram bin containing skin and non-skin models of size  $32 \times 32 \times 32$  was built. The histogram bins were generated by using samples of skin and non-skin images. The tracker framework effectively

finds the moving skin locations as it integrates both the motion and skin detection. Hand shapes are another important modality of our gesture recognition system. Hand shapes can hold important information about the meaning of a gesture, or about the intent of an action. Recognizing hand shapes can be a very challenging task, because the same hand shape may look very different in different images, depending on the view point of the camera. We use chamfer matching of edge extracted hand regions to compute the minimum chamfer matching score. Dynamic Programming technique is used align the temporal sequences of gesture.

In this paper, we propose a novel hand gesture recognition system where in user can specify his/her desired gestures vocabulary. The contributions made to the gesture recognition framework are, user-chosen gesture vocabulary (i.e) user is given an option to specify his/her desired gesture vocabulary, confusability analysis of gesture (i.e) During training, if user provides similar gesture pattern for two different gesture patterns the system automatically alerts the user to provide a different gesture pattern for a specific class, novel methodology to combine both hand shape and motion trajectory for recognition, hand tracker (using motion and skin color detection) aided hand shape recognition.

The system runs in real time with frame rate of 15 frames per second in debug mode and 17 frames per second in release mode. The system was built in a normal hardware configuration with Microsoft Visual Studio, using OpenCV and C++. Experimental results establish the effectiveness of the system.

## TABLE OF CONTENTS

ACKNOWLEDGEMENTS .....	ii
ABSTRACT .....	iii
LIST OF FIGURES.....	viii
LIST OF TABLES.....	x
LIST OF SYMBOLS USED .....	xi
CHAPTER	Page
1. INTRODUCTION.....	1
1.1 Gesture Recognition .....	1
1.2 Goals .....	2
1.3 Contributions.....	2
1.4 Definition of Terms .....	3
1.4.1 Gestures.....	3
1.4.2 Features .....	3
1.4.3 Frame Sequence.....	3
1.5 Organization of Thesis.....	3
2. LITERATURE OVERVIEW .....	5
2.1 Approaches to Gesture Recognition .....	5
2.2 Gesture Representations .....	5
2.2.1 Static Gesture Representation.....	5
2.2.2 Dynamic or Motion-based Gesture Representation.....	6
2.2.2.1 Vision-based extraction of motion information.....	6
2.3 Related Work in Gesture Recognition.....	7
3. SYSTEM OVERVIEW .....	9

3.1 Training Phase .....	9
3.2 Confusability Analysis of Gesture Vocabulary .....	11
3.3 Real-time Gesture Spotting and Recognition .....	11
4. HAND TRACKING .....	14
4.1 Motion Detection .....	14
4.1.1 Background Subtraction.....	14
4.1.2 Frame Differencing .....	15
4.2 Skin Color Detection .....	16
4.2.1 Skin Detection Results .....	17
5. HAND SHAPE DETECTION.....	22
5.1 Chamfer Distance .....	22
5.2 Approach.....	23
5.2.1 Template Vector Space and Distance Transform Approach .....	24
5.2.2 Multi Scale Chamfer Matching .....	26
5.3 Observations .....	27
6. GESTURE SPOTTING .....	32
6.1 Continuous Dynamic Programming .....	33
6.1.1 Algorithm for Continuous Dynamic Programming.....	33
6.2 Memory Efficient CDP Implementation .....	34
6.3 Approach.....	34
6.3.1 Algorithm .....	35
6.3.2 Gesture Recognition .....	36
7. DYNAMIC TIME WARPING.....	38
7.1 DTW Algorithm.....	38
8. EXPERIMENTS .....	41
8.1 Training Vocabulary .....	41
8.2 Measures of Accuracy.....	43

8.3 Reliability of System on Different Scales of Vocabulary .....	46
9. CONCLUSION AND FUTURE DIRECTIONS.....	50
9.1 Future Directions.....	50
REFERENCES.....	51
BIOGRAPHICAL INFORMATION.....	54

## LIST OF FIGURES

Figure	Page
3.1 An example Hand Shape Template Image, (a) Original Image, (b) Edge Template Image (T).....	9
3.2 Vision-based Recognition System .....	10
3.3 Flow Chart of Dynamic Gesture Recognition System.....	12
4.1 An example of Background Subtraction technique Where (a) is the reference image which is captured before the recognition algorithm is invoked, (b) the current input frame sequence and (c) shows the background subtracted results obtained by absolute differencing between the reference and input image .....	15
4.2 An example of Frame Differencing, (a) is the previous frame of the input frame sequence. (b) is the current input frame sequence. (c) is the next input frame sequence. (d) shows the frame difference (d) shows the frame difference image which is the minimum of absolute difference between current, previous input frame and current, next input frame .....	16
4.3 Skin Detector Result .....	18
4.4 An example of false detection, Face region been Detected .....	19
4.5 Skin Detector result with Face regions Nullified .....	19
5.1 An example of Chamfer Matching between two set of Points (circle and square). (a) represents two set of points (circle and square). (b) represents directed chamfer distance from square to circle. (c) represents directed chamfer distance from circle to square .....	23
5.2 Edge Oriented Binary Image from the Input Frame Sequence. (a) shows the input frame sequence image and (b) shows the corresponding edge oriented image .....	24
5.3 Distance Transform of the Edge Oriented Image .....	25
5.4 Hand shape detection pipeline in Cluttered background condition without background subtraction, (a) Represents the input edge oriented image, (b) represents	



	Its distance transform image and (c) shows the detection result obtained from chamfer matching .....	28
5.5	Hand shape detection with background subtraction (cluttered background) (a) represents the input edge oriented image. (b) represents Its distance transform image and (c) shows the detection result obtained from chamfer matching .....	29
6.1	Spotting function and classification.....	32
7.1	An example Warping Path between two sequences .....	39
8.1	Gesture Motion Patterns .....	42
8.2	The nine hand-shapes dataset used in our experiments.....	42
8.3	Nine Different template edge images used in our experiments.....	43
8.4	Recognition rates at different scales of vocabulary .....	47
8.5	Type: 5 Hand Shape Pattern Gesturing in Upward Motion (a), (b), (c) and (d) represents the input video frame sequences .....	48

## LIST OF TABLES

Table		Page
4.1	Hand tracker results of user wearing full sleeves .....	20
4.2	Hand tracker results of user wearing short sleeves .....	21
5.1	Shape detector accuracy in different scenarios .....	30
8.1	Recognition rates of gestures in same direction .....	44
8.2	Confusion matrix of training template types 4, 5, 8 and 9 for same motion pattern .....	44
8.3	Results of Single hand shape, all motion trajectories .....	45
8.4	Recognition rates in different background conditions .....	46
8.5	Comparison with other approach .....	49

## LIST OF SYMBOLS USED

<b>Symbols</b>	<b>Explanation</b>
T	Hand Shape template Image
BG	Background Subtraction
R	Reference Image for Back Ground Subtraction
$V_t$	Input Frame Sequence
$FD_t$	Frame Differencing
$D_{CM}$	Chamfer Distance
E	Input Edge Image
DT	Distance Transform Image
S_W	Sub Window
$D(i, j)$	Cumulative cost function
$d(i, j)$	Local Cost Function
M	Model Gesture feature vectors
Q	Query Gesture feature vectors
W	Warping path in DTW
X, Y	Edge Image

## CHAPTER 1

### INTRODUCTION

#### 1.1 Gesture Recognition

One of the goals in the modern human-computer interaction arena is to develop an end to end system, wherein a user can automatically control an application using gestures. To accomplish this larger goal, there is a need to develop applications that run faster by recognizing input test gestures in real time. Although conventional user interface are based on keyboards and mouse, a central limitation of these interfaces with machines is the requirement that people need to communicate with machines in a manner very different from the way people interact with each other. To enable natural, human like communication, we need to make use of natural communication signals such as speech, facial and body gestures.

Gesture recognition is important for developing an attractive alternative for human-computer interaction. Gestures can present themselves in various forms. Gestures in modern societies are everything from a smile or a hand and arm movements. Most people add meaning to their spoken words by gesturing. In most cases this is done subconsciously and is therefore hard to suppress.

Many situations are imaginable, where it would be of considerable benefit to add the ability to communicating with computers using gestures. Machinery could be controlled this way, it could simply be a more natural way of conveying certain information to the computer or even it can aid disabled people who cannot use other means of interacting with computer.

Several very successful approaches have been made to solve the gesture recognition problem [5], many of these are computationally expensive. The idea of finding easy ways of implementing gesture recognition even on medium sized architectures therefore seems appealing. Such methods are definitely available, but care has to be taken in weighing the advantages of relaxed hardware requirements versus the disadvantages of a decrease in recognition rates.

Thus, developing a gesture recognition system that has the capability to run in real-time even on medium sized architectures, which can be interfaced with a webcam to control active windows application is the basic motivation behind this thesis.

### 1.2 Goals

In this work we have focused on the problem of recognition of dynamic gestures. The goal of this work is to develop methods to recognize useful gestures from a continuous stream of input frame sequence and classify them into one of the specific classes. The dominant gesture which is considered for detection and recognition are sequences of distinct hand shapes. We have considered single handed gestures which are sequences of distinct hand shapes. A given hand shape can undergo motion and discrete changes. These gestures are distinguished on the basis of shapes involved and the nature of motion. We have developed a real time gesture recognition system which can reliably recognize these gestures despite individual variations.

### 1.3 Contributions

In this work, we have developed a Dynamic Programming based gesture recognition system which uses both the temporal and shape characteristics of the gesture for recognition. Our system is robust to background clutter, does not require special gloves to be worn and runs in real time. The contributions of this paper are as follows,

- Use of Dynamic Programming for gesture recognition is not new but the methodology adopted for combining shape and temporal characteristics is new contribution of this work.
- Confusability analysis and evaluation of training gesture similarities is another novel contribution to this work. This analysis is used to alert the user when the gesture vocabulary includes gestures that the computer has trouble telling apart.
- Using memory efficient implementation of Gesture Spotting and recognition algorithm for matching each input frame sequence with all the training models without crashing the heap.

- Real time implementation of the gesture recognition system with the ability to send commands or keystrokes to any active windows application.
- Schemes have been devised for automatic start and end point detection of the gesture sequence for the training phase, with the assumption that no back ground movement is seen during the training/observation phase.

The performance, accuracy and the robustness of the proposed gesture recognition system is discussed in detail in Chapter 8.

#### 1.4 Definition of Terms

Some terminology will be repeatedly used in this thesis and it is important to give clearly defined meaning to the terms used in order to prevent confusion or misinterpretation.

##### 1.4.1 Gestures

The term gesture is defined as “movement of a part or body to convey meaning” [29]. This definition is considered applicable throughout this thesis.

##### 1.4.2 Features

Features are defined as the notable or a specific piece of information which is relevant for solving the computational task related to a certain application. In the context of this thesis features will describe certain quantifiable aspects of gestures that make them differentiable and classifiable, for example, specific structures of the image itself such as edges, points or object regions.

##### 1.4.3 Frame Sequence

In the context of this thesis frame sequence is defined as the single image in a sequence of images that are recorded or obtained continuous from a frame buffer, a highly accessible part of the video RAM.

### 1.5 Organization of thesis

In the rest of this thesis, we present the details of our work. In chapter 2 we give a general review of the relevant literature. In chapter 3 we present the general overview of our gesture recognition system. In chapter 4 we describe the hand tracker framework used. Chapter 5 details the hand shape detection and chamfer matching methodologies used for estimating and extracting the hand shapes. Chapter 6 and 7 explains on the Dynamic Programming based gesture spotting and recognition algorithm used. In Chapter 8 we discuss the experimental results obtained along with discussions and comparison with other state-of-the art gesture recognition systems. We conclude and provide possible future directions in Chapter 9.

## CHAPTER 2

### LITERATURE OVERVIEW

#### 2.1 Approaches to Gesture Recognition

Gesture recognition is an ideal example of multidisciplinary research. There are different tools for gesture recognition, based on approaches ranging from statistical modeling, computer vision and pattern recognition. Gesture recognition is mainly accomplished by combining both Image-processing techniques [12] such as analysis and detection of shape, texture, color motion, optical flow, image enhancement, segmentation and contour modeling [13], followed by machine learning and pattern recognition techniques, involving feature extraction, object detection, clustering, and classification, which have been successfully used for many gesture recognition systems [14]. In what follows will provide a general review of related literature to gesture recognition.

#### 2.2 Gesture Representations

Gestures are expressive, meaningful body motions involving physical movements of the fingers, hands, arms, head, face, eyes or body with the intent of conveying meaningful information. Gestures can be static (the user assumes a certain pose or configuration) or motion based which requires one to consider both temporal as well as spatial segmentation. Typically, the meaning of a gesture can be dependent on the following, (i) spatial information: where it occurs, the orientation of the gesture, (ii) path information: the path it takes, (iii) symbolic information: the sign it makes.

##### 2.2.1 Static Gesture Representation

Static gesture representation is concerned with spatial data, one frame at a time. The information used for static gesture may be templates [15], normalized silhouettes [16], or postures [17]. The goal of



static gesture recognition is mainly to recognize various postures. Human hands are mainly used for static gesture recognition [17]. The first step in using hand posture and gestures in computer applications is gathering raw data. Raw data is collected in two ways. The first is by using instrumented gloves or tracking devices. The second way is to use a computer-vision-based approach by which one or more cameras collect images of the user's hand. Once the raw data has been collected from a glove/tracking device or vision based data collection system, it must be used as a model to recognize any unknown postures or gestures using any of the gesture recognition algorithms.

### 2.2.2 Dynamic or Motion-based Gesture representation

Motion-based recognition deals with the recognition of an object based on motion in a sequence of images. In this approach, a sequence containing a large number of frames is used to extract motion information. There are two main steps in motion-based recognition. The first step consists of finding an appropriate representation for the objects and motions we want to model, from the motion in the image sequence. Once we extract features or models from sequence of images, the second step consists of matching of some unknown input with the model. Many motion-based gesture representations use vision-based motion capture, there are also few tracking device interface that is used for motion-based gesture representation. One example of tracking device based motion gesture representation is Moving Light Display's (MLD). MLD's consist of bright spots attached to the joints of a human moving in front of a dark background. The collection of spots carry only two dimensional two-dimensional information, their relative movement created a vivid gesture. The recognition is based solely on the data collected by the motion of these spots [20].

#### 2.2.2.1 Vision-based extraction of motion information

There are two main steps in motion-based recognition. A motion representation must be first defined, so that the appropriate motion information is extracted from a sequence of images and organized into motion models. Second, representation of an unknown model must be compared with a stored model for recognition.

A vision-based approach is very effective in extraction of motion information from a sequence of images. Motion correspondence is one such effective method to extract motion information from images. Motion correspondence deals with extracting interesting points, characteristics, and features in an image that can be tracked in time [3]. The generation of motion trajectories from a sequence of images typically involves the detection of tokens in each frame. Tokens include edges, corners, interest points, and regions.

### 2.3 Related work in Gesture Recognition

Gesture recognition in its various incarnations is not a new topic. A fair amount of research has been performed on different aspects of gestures recognition. Several incentives can be found in the literature to suggest the use of gesture recognition system as a user-input mechanism. Many of the gesture recognition systems use either static or dynamic gesture representations and there are very few gesture recognition system developed by considering both the spatial (i.e. Shape) as well as temporal changes.

A more relevant work was carried out in [1], where gestures were considered which is sequences of distinct hand shapes and nature of motion. The recognition strategy used by [1] is similar to our work but the methodology used is completely different. In [1] gesture recognition system was based on Kalman filter and HMM based gesture recognition and classification. The detection rates achieved through the Kalman Filter based hand tracking in [1] is lesser than our proposed method. In [2] a similar approach is described, using both dynamic programming based motion tracking as well as chamfer matching for evaluating the spatial gesture pattern. But their method uses only a single shape template using which all the gestures has to be performed and they eradicate gesture patterns performed without that particular shape template. Our method uses a distinct shape for each gesture class which aids in using a larger vocabulary size with higher recognition rates when compared to [2]. [10] Have proposed a similar gesture recognition framework which can be used as a user interface based on an Eigen space model. But they require large set of training datasets for building Eigen space model. Our method can recognize gestures effectively with a minimal training set. [11] Presents a similar work in recognition of restricted hand shapes

and a moving hand based on simple background subtraction and masking color region to obtain the hand shape template. The hand region tracking and extraction process used by [11] is somewhat similar to our method, but we use a more robust hand histogram bins to detect and track moving hand regions. [23] Have used a similar gesture recognition strategy for pedestrian detection, they combine both the chamfer matching as well as Hessian-Laplace detector for detecting pedestrians from a live single view point camera. In this paper, we explore the use of chamfer matching and dynamic programming based gesture spotting and recognition framework. We also compare the performance of our system with [1][2] and prove that the gesture recognition system developed by us has a better recognition rates even under scalable vocabulary size.

## CHAPTER 3

### SYSTEM OVERVIEW

In this work we have developed a dynamic programming based gesture recognition system which uses both the temporal and shape characteristics of the gesture for recognition. The system consists of two main phases, interactive training phase and real-time gesture spotting and recognition phase. There are three steps involved in the training phase, hand shape template extraction, training temporal characteristics of the gesture and confusability analysis of gesture vocabulary. Once a training data object model is built with the hand shape template and the temporal characteristics of the gesture, the system testing phase or the real time gesture spotting and recognition module is invoked.

#### 3.1 Training Phase

In the training phase, the user is initially asked to provide a hand shape template, one for each of the training gesture patterns. The hand shape template is required to be provided in a uniform background. The system stores the co-ordinate edge pixel locations of the binary edge transformed hand shape template image (T), as shown in figure 3.1.

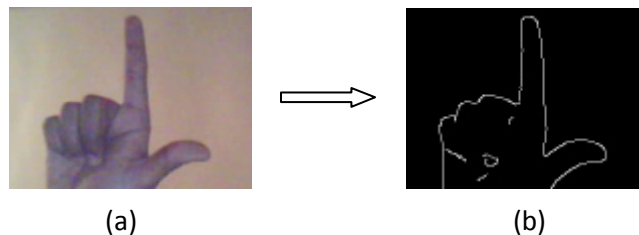


Figure 3.1 An example Hand Shape Template Image, (a) Original Image, (b) Edge Template Image (T)

Hand gestures are considered as dominant feature with the assumption that the training is done in a static background with no skin color object present in the background. Once the hand shape template is obtained, the temporal characteristics of the training gesture are tracked and a motion trajectory vector

containing co-ordinate locations and its corresponding shape similarity measure is obtained for each training gesture pattern. Chamfer matching is used to get the corresponding shape similarity measure. In case of tracking the temporal characteristics of the training gesture pattern, the system needs to know the specific start and end point of the training gesture pattern where it should look for the motion trajectory features of the gesturing hand. For the end point detection, the system uses the motion parameter estimate such as the frame differencing value. If the maximum frame difference value in the specific frame sequence is below a specified threshold and if it remains below threshold continuously for more than ten frame sequence, the system considers it as end of gesture pattern and stores the temporal features to the training data model. Similarly, the system provides an interactive interface to the user which requests the user to be ready to gesture the temporal characteristics of the hand shape gesture pattern. Confusability analysis of gesture then checks for similarities between the trained gesture pattern and the existing trained data object model. The user is asked to retrain the training gesture pattern when a resemblance is encountered. This way we increase the recognition rate by mitigating the recognition mismatches and also improve the accuracy of recognition.

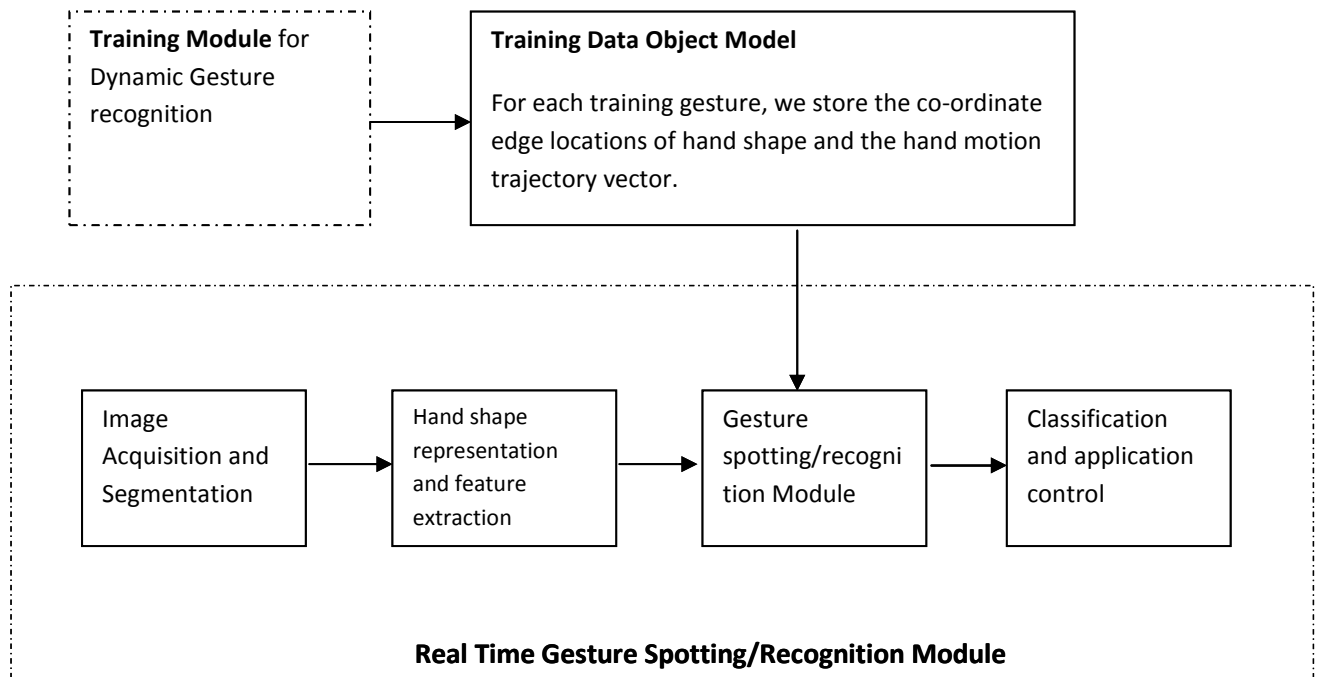


Figure 3.2 Vision-Based Recognition System

### 3.2 Confusability Analysis of Gesture Vocabulary

When the user intends to train gesture pattern for more than one class, there are chances that the user may provide similar set of training gesture pattern for two different classes of gestures. This situation may arise when users are not necessarily aware of the limitations of existing gesture recognition methods. Hence, after the user has chosen a gesture vocabulary, it is important for the system to verify that the training gestures of different classes do not resemble each other. DTW (Dynamic time Warping) method is used to get the training gesture similarity measure. The user is asked to re-train the gesture pattern of the specific class if the DTW distance measure is less than the threshold value. The DTW (Dynamic Time Warping) method is illustrated in Section 6.1 of chapter 6.

### 3.3 Real-time Gesture Spotting and Recognition

In the gesture spotting and recognition phase, the system identifies gestures from a continuous stream of input video based on the temporal sequence of hand shapes and their corresponding motion pattern. We have considered single handed gestures and it is assumed that only single handed dynamic gestures are used. There are three main modules in gesture spotting and recognition framework, hand tracking, chamfer distance based hand shape detection and gesture spotting/recognition. Once the training phase builds training data model, the gesture spotting and recognition phase is called which captures continuous stream of input frame sequence from a fixed point single view web cam input. The hand tracker is based on motion detection and skin color detection, before the system starts spotting and recognizing gestures, face detection module is called which looks for any face locations in the scene. The system then masks the face region which prevents the hand location been lost in the presence of face. This improves the detection accuracy and in turn increases the recognition rate. We have used Haar-cascade classifier for detecting face by using a publicly available face training set [22].

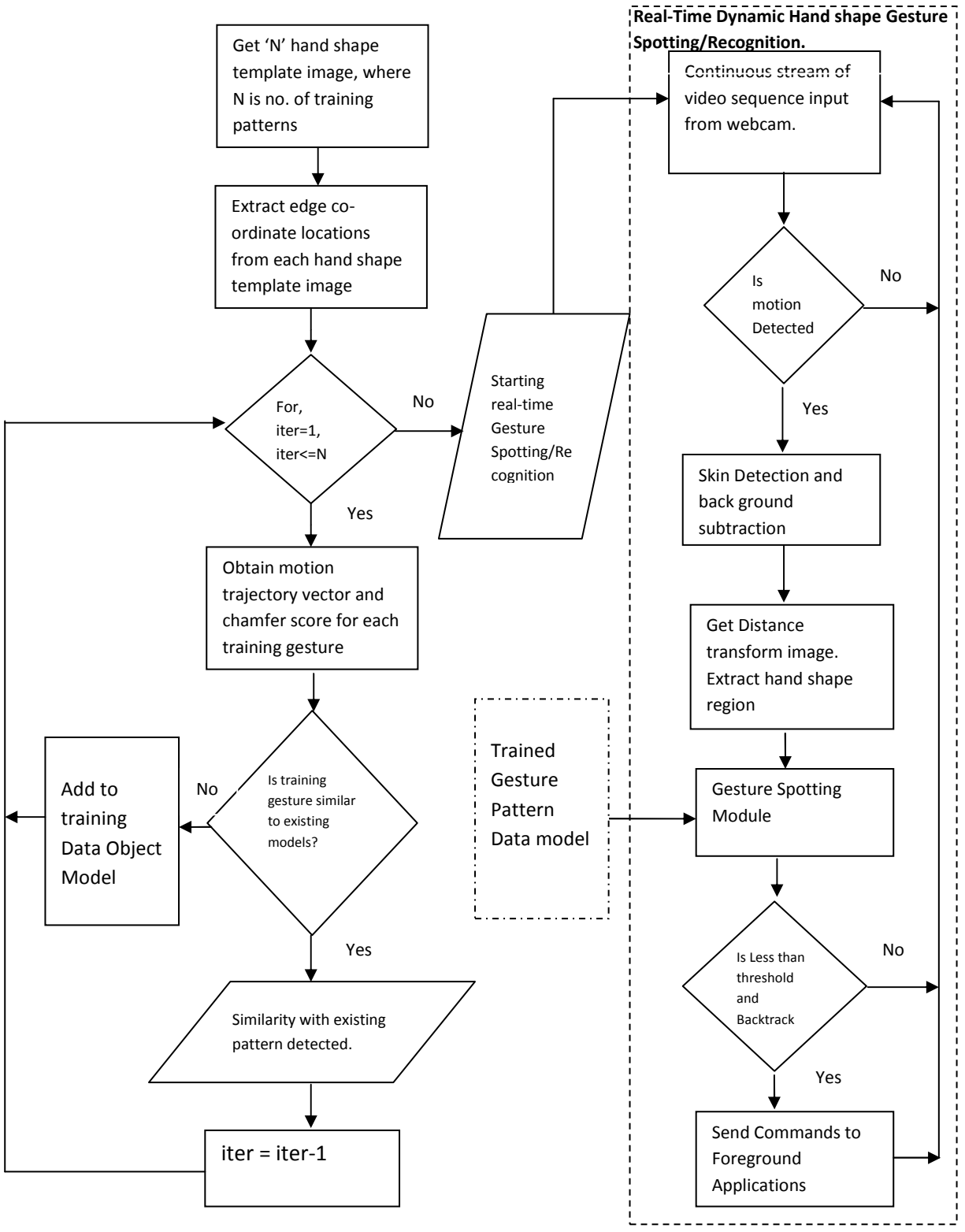


Figure 3.3 Flow Chart of Dynamic Gesture Recognition System

Figure 3.3 shows the flow chart of the overall gesture recognition system. As seen from figure 3.3, the gesture spotting and recognition module starts tracking and recognizing gestures when motion is detected in the specific frame sequence. The hand tracker module extracts hand regions and provides a sub-window containing hand region which is then fed to the hand shape detection and chamfer matching module.

The hand shape detection module looks for specific hand shapes in the test frame sequence by matching the sub-window with the co-ordinate edge pixel locations of the hand shape template. The hand shape detector then returns the best chamfer matching score between the test image and hand shape templates and also the co-ordinate temporal location of the detected hand which is then fed to the gesture spotting algorithm. The shape detection module is explained in detail in Chapter 5. The gesture spotter calculates the cumulative score between the training features and test feature to conclude if the gesture can be classified to a specific class. Continuous Dynamic Programming (CDP) is the method used to check alignment of the test and training gesture pattern. The classification of the spotted gesture into a specific gesture class is evaluated based on two parameters namely the threshold measure and number of frames have elapsed since the candidate was detected. Finally, if the current frame sequence is recognized as one of the specific gesture class, the control or command mapped to specific gesture class is send to active foreground windows application.



## CHAPTER 4

### HAND TRACKING

The first step in recognition of dynamic hand gestures is to extract hand regions from continuous stream of video sequence. The objective of the hand tracker is to segment the moving hand from the background by integrating multiple clues such as motion detection and skin color detection.

#### 4.1 Motion Detection

The main goal of the motion detector is to detect foreground regions from input frame video. The foreground objects must be separated from the static background in order to extract the motion trajectories of the gesture. Background subtraction and frame differencing are two main steps involved in extracting the foreground regions from the background.

##### 4.1.1 Background Subtraction

The basic approach for background subtraction is to store the background image as the reference image, in which there is no movement and then every other frame sequence subtract the reference image to extract alien objects in the scene. The reference image used in our case is the first frame sequence captured by the camera/webcam. Even though the background did not remain constant due to illumination changes, setting proper threshold and integrating the background subtraction with frame differencing and skin detector results, resulted in accurate hand tracking. Figure 4.1 shows an example of background subtraction.

$$BG(q) = \text{AbsDiff}(V_t(q), R(q)) , \text{ where } t \text{ is the input frame at time } t \quad \text{Equation(1)}$$

Where BG is the background subtracted image,  $V_i$  is a single frame sequence, and R is the reference image.

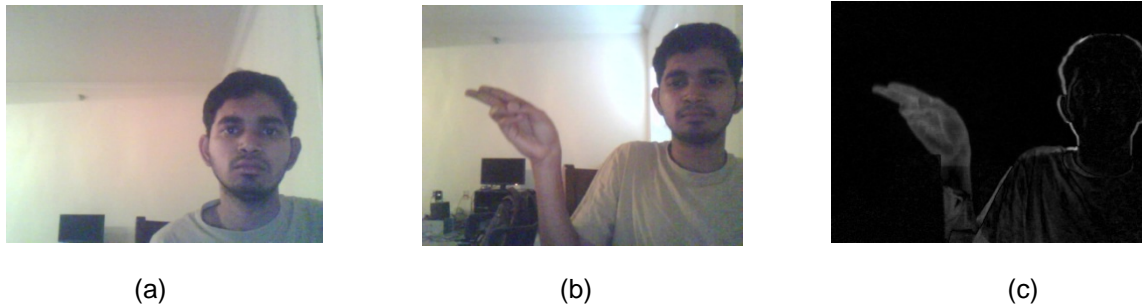


Figure 4.1 An example of background subtraction technique, where (a) is the reference image which is captured before the recognition algorithm is invoked, (b) the current input frame sequence and (c) shows the background subtracted result obtained by absolute differencing between the reference and the input frame sequence image

#### 4.1.2 Frame Differencing

Frame differencing approach is also relevant to background subtraction where instead of finding the difference of the current frame sequence with a reference image, frame differencing value results in difference between two consecutive frame sequences. Frame differencing leaves only a short foreground trail behind the moving object as its memory does not extend beyond the previous frame. The frame differencing case can be expressed in a form of equation as given in equation 2.

$$FD_t(q) = \text{Minimum} \{ \text{AbsDiff}(V_{t-1}(q), V_t(q)), \text{AbsDiff}(V_t(q), V_{t+1}(q)) \}, \text{ Equation}(2)$$

Where  $V_t$  represents input frame sequence at time t for all spatial locations q and  $FD_t(q)$  is the frame difference value for all spatial locations q of the input frame sequence at time t.

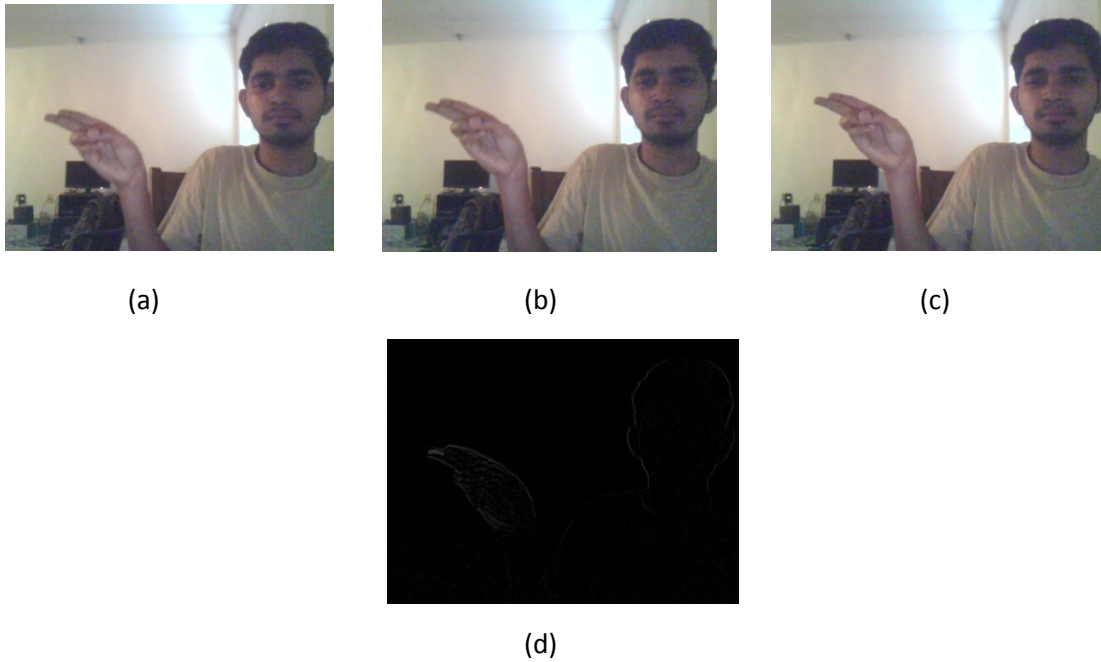


Figure 4.2 Example of frame differencing. (a) Is the previous frame of the input frame sequence. (b) is the current input video frame sequence, (c) is the next input frame sequence. (d) Shows the frame difference image which is the minimum of absolute difference between current, previous input frame and current, next input frame

As seen from figure 4.2, the frame difference value is more or less similar to the background subtraction results but it only has a shorter trail of the moving object.

#### 4.2 Skin Color Detection

Skin detection is based on histogram-based statistical modeling using RGB color space values. In this approach we build skin and non-skin histogram bins using samples of skin and non-skin images. If for example, we consider a color image RGB quantized to 8 bits per color, we'll need an array of  $2^{24}$  elements ( $256 \times 256 \times 256$ ) to store skin probabilities. To reduce the amount of needed memory and to account for possible training data sparsity, coarser color space sampling is used –  $32 \times 32 \times 32$  in our case. The reason for choosing  $32 \times 32 \times 32$  coarse histogram bin is based on the evaluation of different RGB samplings by Jones and Rehg [24], they have illustrated that  $32 \times 32 \times 32$  shows the best performance. The performance of the histogram-based non-parametric skin models directly depends on the representation of the training image sets. In our experiments, we have used 100 samples of skin and non-skin images to

build 32x32x32 coarser skin and non-skin histogram bins. Given the training samples of skin and non-skin, count for how many pixels have color RGB in each bin (R, G, B) of histogram.

Algorithm Steps:

- Create a cell histogram bin array of skin and non-skin each of size 32x32x32 and initialize to zero.
- For each skin and non-skin training color image, get the three (RGB) color space value of each pixel and store it in a temporary variable namely red, green and blue.
  - Red= training\_image( i, j, 1)
  - Green = training\_image( i, j, 2)
  - Blue = training\_image( i, j, 3), where  $1 \leq i \leq \text{width of the Image}$ , and  $1 \leq j \leq \text{Image Height}$
- For a 32x32x32 histogram bin, each intensity value has to be divided by a factor 8 to fit from 256x256x256 to 32x32x32. For example consider we are training skin histogram bin then,
  - Skin [red/8][blue/8][green/8] = Skin [red/8][blue/8][green/8]+1
- The values of histogram bin are normalized to fit into a probabilistic distribution framework by dividing each bin values with the sum of all the histogram bins.
- Once the histogram bins are generated, we need to estimate for each input test image, the P (RGB | skin) and P (RGB | non-skin).
- Maximum likelihood estimate of skin color in the test image is given by,  
$$P(\text{skin\_score}) = P(\text{RGB} | \text{skin}) / \{P(\text{RGB} | \text{skin}) + P(\text{RGB} | \text{non-skin})\}.$$

#### 4.2.1 Skin Detection Results

Skin detector is an important module in the detection and feature extraction phase as our dominant gesture is the single handed gestures. The skin detection module performance was highly dependent on other objects or lighting conditions present in the gesturing environment. The trained histogram bins which is used to estimate the probability of the skin score in each pixel of the input frame sequence has been trained by considering the skin color of Caucasians and Asians. The skin detector

performance may falter if the system is been used by a user whose skin color is close to black pixel intensity. Since we integrate a lot of parametric estimates to detect a gesturing hand in a frame, even if the skin detector module fails to get accurate skin scores the motion detector may help to increase the overall detection accuracy of the system under the consideration that there is no moving skin color object in the background.



Figure 4.3 Skin Detector Result

Figure 4.3 shows the skin values of the input frame sequence obtained from skin detector. Another problem that was encountered in the detection phase is the high probability of face regions been detected. This affected the overall accuracy of the system. An example of false detection is shown in figure 4.4.



Figure 4.4 An example of false detection, Face region been detected

In order to overcome this problem and increase the detection accuracy a cascade classifier based face detection interface was used [19]. The training data set used for this cascade classifier was publicly available [22]. The face detector module is invoked till a face is been spotted from the input video. Once the face region is detected, the face detector module returns the coordinate location of the face region sub-window. The sub-window region returned by the face detector is then nullified from the skin scores obtained from skin detector. Figure 4.5 shows an example skin scores output with face regions nullified.

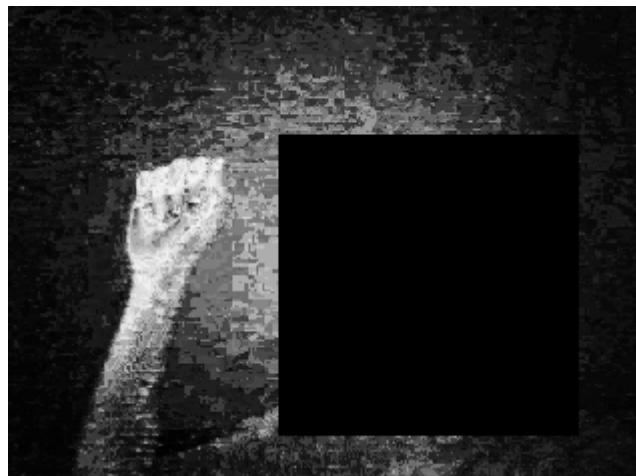


Figure 4.5 Skin Detector result with Face regions Nullified

The detection accuracy and the recognition rates obtained was relatively high when the face regions nullification was done. It should be noted that when there are more than one face regions present in the input frame sequence, only a single face region with higher probability detected by the cascade classifier will be considered. The table given below shows the hand tracking accuracy results with and without face region nullification. The testing of the hand tracker accuracy was made in two different scenarios, user wearing short sleeves and user wearing long sleeves. The results were collected by testing the system with 10 people including Caucasians and Asians.

Table 4.1 Hand Tracker results of user wearing full sleeves

Gesturing Hand	Hand tracking accuracy with face nullification	Hand Tracking accuracy without face nullification
In static non-skin color Background	120/120 = 100%	116/120 = 96.6%
In static skin color Background	118/120 = 98.3%	110/120 = 91.6%
In moving non-skin color background	106/120 = 88.3%	84/120 = 70%
In moving skin color background	74/120 = 61.6%	67/120 = 55.8%

These results show that the face region nullification has significant increase in detection rates. The main effectiveness of this is seen in the case of moving non-skin color background regions which has nearly 18 percent of increased detection rate.

Table 4.2 Hand Tracker results of user wearing short sleeves

Gesturing Hand	Hand tracking accuracy with face nullification	Hand Tracking accuracy without face nullification
In static non-skin color Background	117/120 = 97.5%	114/120 = 96.6%
In static skin color Background	115/120 = 95.8%	105/120 = 91.6%
In moving non-skin color background	99/120 = 82.5%	74/120 = 61.6%
In moving skin color background	66/120 = 55%	57/120 = 47.5%

The results shown here are based on the single candidate detection of the best probable location of the gesturing hand. The results of detection accuracy was improved by considering 'n' candidate detections of probable hand locations, where n is a number chosen between 1 and 15.



## CHAPTER 5

### HAND SHAPE DETECTION

Hand tracker finds the maximum likelihood region of the gestured hand from the frame sequence. The hand shape detection module then finds the chamfer distance measure from the contour of the temporal hand shape. Although many shape matching algorithms have been proposed, chamfer matching remains to be the preferred method when speed and robustness are considered.

#### 5.1 Chamfer Distance

Chamfer matching [4] is the popular technique to find the best alignment between two edge maps. Chamfer matching is specifically incorporated using the edge orientation information because of its tendency to be more stable than intensity images which show variations with respect to different lighting conditions. Chamfer matching is a measure of distance between the edge template image  $T$  and the binary edge transformed test image. Given two edge images,  $X$  and  $Y$ , chamfer distance  $D_{CM}(X, Y)$  is given as,

$$D_{CM}(X, Y) = \frac{1}{|X|} \sum_{x \in X} \min_{y \in Y} \|x - y\| + \frac{1}{|Y|} \sum_{y \in Y} \min_{x \in X} \|y - x\| \quad \text{Equation (3)}$$

Where  $\|a - b\|$  denotes the Euclidean distance between two pixel locations  $a$  and  $b$  [6].  $D_{CM}(X, Y)$  penalizes for points in either edge image that are far away from any point in the other edge image.

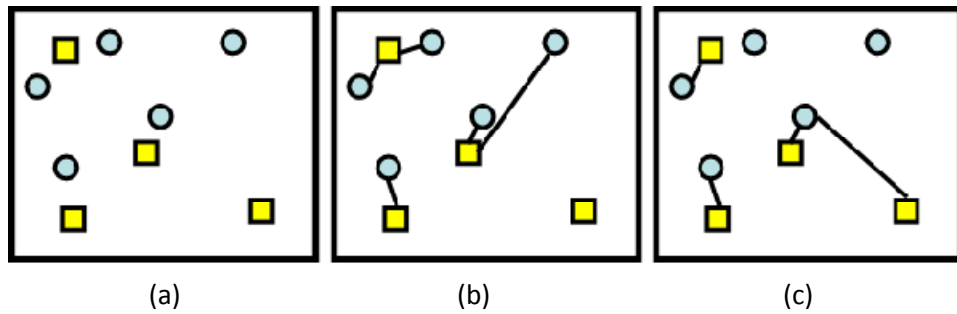


Figure 5.1 An example of the chamfer matching between two sets of points (circle and square). (a) Represents the two sets of points (circle and square). (b) Represents directed chamfer distance from square to circle. (c) Represents directed chamfer distance from circle to square [6]

In figure 5.1, the left image shows two set of points, points in the first set are shown as circles, and point in the second set are shown a square. The middle image shows a link between each circle and its closest square. The circle to square directed chamfer distance is the average length of those links. The right image shows a link between each square and its closest circle. The final chamfer distance measure is the sum of the circle to square and square to circle chamfer distance.

## 5.2 Approach

The first step is to extract image features by applying edge operators for tracing boundaries. Canny edge detection is used in our case to extract the binary edge transformed image from the input frame sequence. To avoid background clutter, the input frame sequence is compared with the reference background image and the difference between current frame sequence and background reference image is given as input of the canny edge detector. A detailed analysis is given in session 5.3 on how the hand-shape detection accuracy improved by considering the background subtracted edge image. Once the edge oriented image is obtained, we need to find the best alignment between the hand shape template and edge oriented frame sequence image. Figure 5.2 shows the sample edge transformed image from the input frame sequence.

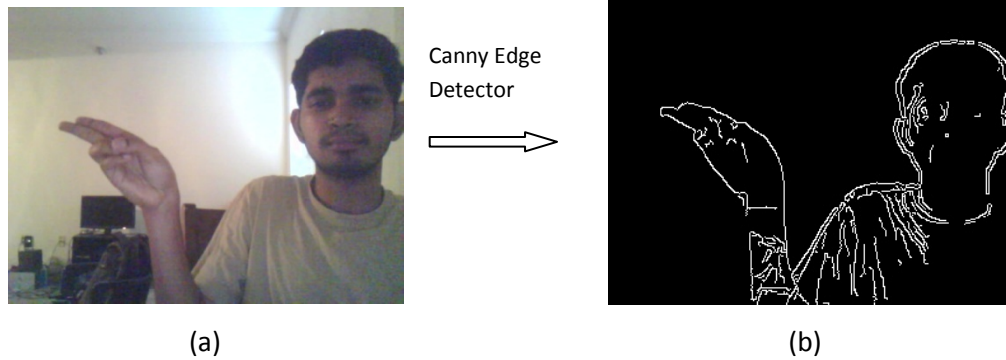


Figure 5.2 Edge oriented Binary image from the input frame sequence. (a) Shows the input frame frame sequence image and (b) shows the corresponding edge oriented image

In figure 5.2, the left image is the input frame sequence image which is compared with the background reference image to generate the binary edge image of the moving objects shown in the right side of figure 5.2. In order to avoid back ground clutter in the edge image, we compare the input frame sequence with the background reference image so as to increase the recognition rates by decreasing the number of false recognition.

### 5.2.1 Template Vector Space and Distance Transform Approach

Computing chamfer distance of the query edge images at all window, all scales with template image can be very time consuming. In order to reduce the search space of chamfer matching we consider a sub-window of the query edge image of width 160 and height 120 drawn around the maximum likelihood hand region obtained from the hand tracking result. Then the similarity between the two sets of edge maps is the Euclidean sum of distances between each template image edge pixel and the query sub-window edge pixel. This similarity measure between template and query sub-window is computed efficiently by transforming the query edge oriented image into a distance transformed array representing for each pixel, the distance to the nearest edge pixel. The distance transform can be computed in two passes over the image [9] and using which the cost function can be evaluated in linear time  $O(n)$  as explained in equation(3).

$$D_{CM}(E_a, E_b) = \frac{1}{n} \sum_{e_i \in E_a} DT_v(e_i) \quad \text{Equation (4)}$$



Figure 5.3 Distance transform of the edge oriented Image

Figure 5.3 shown above is an example distance transform image of the edge oriented image shown in figure 5.2.

The cost function measure is further approximated by embedding the edge template hand shape images into a vector space. The vector space holds the co-ordinate location of the edge pixels in the template image. This type of embedding drastically increased the speed up of matching time as we just sum up the vector space coordinate values in the distance transform image. Embedding of arbitrary spaces into a vector space is a general approach for speeding up nearest neighbor retrieval. Let  $X$  be set of objects, the set of edge images of hands in our case, and  $D$  is the chamfer distance. An embedding function  $F: X \rightarrow R$  is a function that maps objects (co-ordinate locations in our case) of  $X$  into a  $d$ -dimensional real vector space  $R^d$ , For example suppose we have a  $120 \times 90$  edge oriented hand shape template image  $T$ , we then map the co-ordinate locations of the edge pixel present in the template image to 1-D vector space which holds the co-ordinate locations of the edges present in the hand shape template image. Then given an object  $X$ , which is the distance transform image in our case, it is more efficient to map points of  $X$  to vectors and obtain the distance measure. This can be mathematically expressed as given in equation 5,

$$F^R(X) = DT(X,R)$$

Equation (5)

DT(X, R) represents the distance transform image X mapped to coordinate locations present in the vector space R. Once the mapping is done, the distance measurement is then the summation of the mapped points of X. The mapping of distance transform image X to vector space objects is done by scaling and resizing the distance transform image to the hand shape template image size of 120x90.

### 5.2.2 Multi scale Chamfer Matching

Scaling and resizing of the distance transform image is done to best approximate the chamfer distance. Scaling and resizing of the entire frame sequence distance transform image at all scales and at all windows can be very time consuming, hence in order to increase the computation, we consider a small region or a sub-window of size 160x120 centered on the maximum likelihood region of the hand obtained from the hand tracker module. We then evaluate a multi scale, multi linear search to get the best chamfer matching score and the co-ordinate location of the best window that matched the hand shape template T. The sub window size and the template image size are chosen in such a way so as to match the aspect ratio of the input frame sequence image. The algorithm that illustrates the multi scale, multi linear chamfer matching of template with distance transform image is explained below,

#### Algorithm

1. Initiate Scales S,  $i = 1$
2. Loop till  $i \leq \text{size}(\text{scales})$ 
  - a. Initialize Sub Window, S\_W, to size, [width, height]  $\rightarrow [160 * \text{scale}[i], \text{height } 120 * \text{scale}]$ .
  - b. Get the hand region location of the distance transform image DT and resize to fit to S\_W.
  - c. Set S\_W width offset to 40 and height offset to 30, which is the aspect ratio of the input frame sequence (4:3) multiplied by 10.
  - d. Iteratively loop through,
    - i. For  $k=1$  to width offset
    - ii. For  $k_1= 1$  to height offset
      1. Crop S\_W ( $k : k+(\text{width}-\text{width\_offset})$ ,  $k_1 : k_1+(\text{height}-\text{height\_offset})$ )

2. Resize S\_W to size of template
3. Calculate Chamfer score.
  - e. Get the minimum chamfer score measure and its corresponding co-ordinate location.
3. End Loop
4. Store and return the minimum chamfer score and co-ordinate location in a class object.

The multi scale chamfer matching algorithm is invoked for each input frame sequence during the testing as well as training phase.

### 5.3 Observations

We tested the hand shape detection module in outside environment with and without/minimal background clutter and also in indoor environment, considering both background clutter and without/minimal background clutter. User wearing full sleeve shirts was only considered for testing. We made analysis on the performance of the hand shape detection by considering the edge image of the current input frame sequence with background subtraction and without background subtraction. The number of false detections was higher when computing the chamfer score without background subtraction in a window which had lot of background edges. The hand shape detection accuracy was tested with background clutter and without background clutter in two different environments.

Total of four video sets was used for testing the hand shape detection and feature extraction, with two video in outside environment and two video in indoor environment with and without background clutter. The test was made in a static background environment and no moving objects were considered for testing the hand shape detector.

Figure 5.4 shown below is an example frame sequence image used to illustrate the case of false detection when the background has lot of edges and no background subtraction is performed. The top most image in figure 5.4 shows how much of edge information is extracted from the input frame sequence. We then perform distance transform on the edge image. It can be seen from the distance transform image (middle image of figure 5.4), that there is no well-defined region which shows the hand

shape. The hand tracker framework provides a specific region of probable location of gestured hand, which is shown in the middle image of Figure 5.4 as highlighted red color region. Only the highlighted region is searched for hand shape and features extraction which reduced the time of execution of multi-scale chamfer matching by searching from sub-image region to template instead of image to template searches.



(a)



(b)



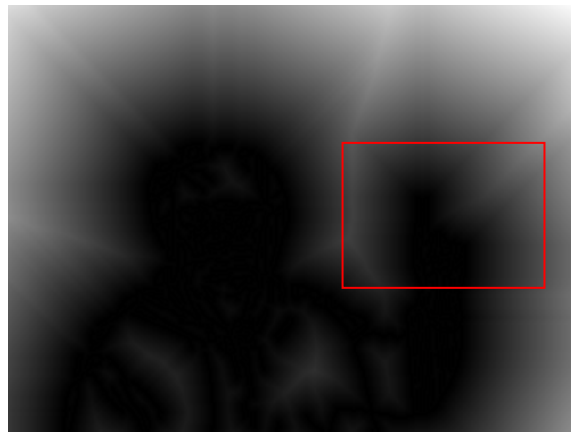
(c)

Figure 5.4 Hand shape detection pipeline in a cluttered background condition without background subtraction. (a) Represents the input edge oriented image, (b) Represents its distance transform image and (c) shows the detection result obtained from chamfer matching

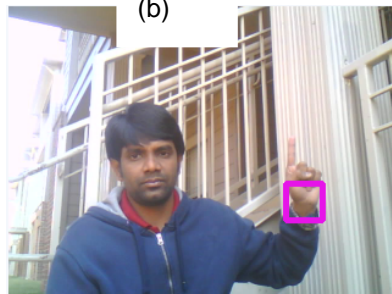
The same test was performed by considering back ground subtraction and the results are shown in figure 5.5.



(a)



(b)



(c)

Figure 5.5 Hand shape detection with background subtraction (cluttered background). (a) Represents the input edge oriented image, (b) represents its distance transform image and (c) shows the detection result obtained from chamfer matching



It can be seen from Figure 5.4 that the background subtracted edge image represents the hand shape information better than compared with the edge oriented image without back ground subtraction in the case of cluttered background. Table 5.1 illustrates the performance of the hand shape detector. The table is populated by considering the test taken from four video sequences, with each video sequence containing approximately [420 to 480] frame sequences. The video was taken in different lighting conditions, outside environment and in presence of background clutter. But no moving background was considered during the test.

Table 5.1 Shape detector accuracy in different scenarios

Background Scenario	Detection Accuracy - Indoor environment (without moving background)	Detection Accuracy - Outdoor Environment (without moving background)
Clutter – Without Background subtraction	82.6%	76.4%
Clutter – With Background Subtraction	95.4%	92.2%
Minimal Clutter – Without Background subtraction	96%	81.5%
Minimal Clutter – With back ground subtraction	99.8%	94%

The detection accuracy is quite high because of the reduced search space of the hand shape detector module (i.e.) the hand tracker module already specifies a sub-window location to search for and to get the contour of the specific hand shape.

The detection accuracy of the outdoor environment was relatively less because of several factors like high luminance condition, effect of wind and slight background changes affected the edge image. It should be noted that the results shown here are not the recognition rates of the system. It is the accuracy of correction detection which is different from the recognition rates. Recognition is a different task when compared to detection which will be discussed in Chapter 8.

## CHAPTER 6

### GESTURE SPOTTING

The objective of the dynamic hand shape recognition system is to create a gesture interface with the capability to extract meaningful gestures from continuous input frame sequence. This is precisely the goal of gesture spotting, to locate the start and end point of the gesture pattern, and to classify the gestures as belonging to one of the predetermined gesture class. A spotting function has the segmentation-free characteristic which ignores gracefully most real world input data which do not belong to the task domain [8]. Gesture spotting can be defined as the task of locating the start and end frame of the video stream that correspond to a gesture of interest, while at the same time rejecting non-gesture motion patterns.

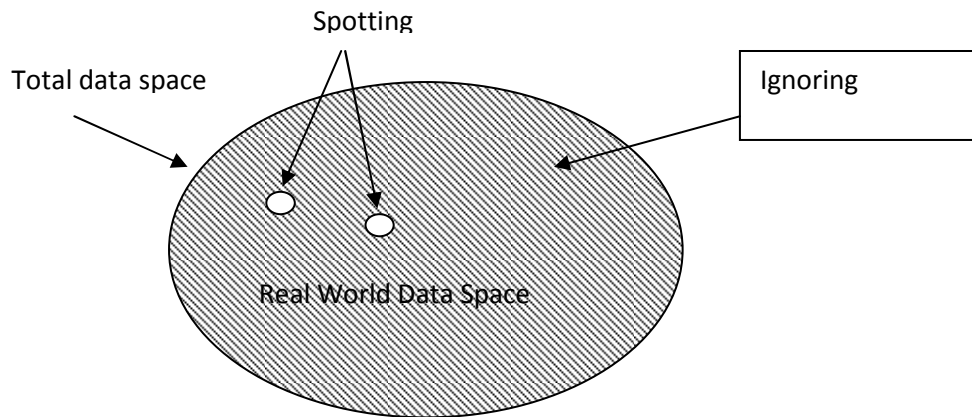


Figure 6.1 Spotting function and classification [8]

As shown in Figure 6.1, spotting method can carry out both recognition and segmentation simultaneously. This is also equivalent to ignoring the part which is outside the task domain determined by reference. There are three basic types of spotting that have been proposed so far and the rest all

proposed methods are more or less variants of these three basic types. Segmentation based spotter [26], Hidden Markov Model-based spotter [25], and Dynamic Programming based spotter. Arguably, the most principled methods for spotting dynamic gestures are based on dynamic programming (DP) [18]. The key advantage of DP is that it can find the best alignment in polynomial time. This is achieved by reducing the problem of finding the best alignment to many sub problem that involve matching parts of model to parts of video sequence.

### 6.1 Continuous Dynamic Programming

Continuous Dynamic Programming (CDP) is a class of dynamic programming based matching algorithm between an input feature sequence and a set of standard sequence patterns. When applying CDP to a spotting problem, the task domain is described by a set of reference sequence pattern. A standard algorithm is described to understand the spotting method.

#### 6.1.1 Algorithm for Continuous Dynamic Programming

Let  $M = (M_1, M_2, \dots, M_m)$  be a model gesture, in which each  $M_i$  is a feature vector extracted from model frame  $i$ . Similarly, Let  $Q = (Q_1, \dots, Q_j, \dots)$  be a continuous stream of feature vectors, in which each  $Q_j$  is a feature vector extracted from input frame  $j$ . We assume that a cost measure  $d(i, j) = d(M_i, Q_j)$ , between two feature vectors  $M_i$  and  $Q_j$  is given. CDP computes the optimal path and the minimum cumulative distance  $D(i, j)$  between the model subsequence  $M_{1:i}$  and the input subsequence  $Q$ . The cost function used to compute the distance measure is given below,

$$D(i, j) = \min\{ D(i-1, j), D(i-1, j-1), D(i, j-1) \} + d(i, j) \quad - \quad \text{Equation (6)}$$

Since our framework is based on temporal motion sequences and also hand shapes, the chamfer matching score is considered along with the cumulative distance measure. The final matching cost becomes,

$$D(i, j) = \min\{ D(i-1, j), D(i-1, j-1), D(i, j-1) \} + d(i, j) + \text{chamfer\_matching\_score} \quad - \quad \text{Equation (7)}$$

The chamfer matching score will be the minimum matching score obtained for a particular hand shape template. For the algorithm to function correctly the cumulative distance has to be initialized properly. This is achieved by introducing a dummy gesture model frame 0 that matches all input frames perfectly for all  $j$ , that is,  $D(0, j) = 0$ . Initializing this way enables the algorithm to trigger new warping path at every input frame.

## 6.2 Memory Efficient CDP Implementation

Continuous Dynamic Programming (CDP) is used to spot gestures by finding the best warping path matching all models with the continuous stream of feature vectors  $j$ . Generally the matching between the input frame feature vectors and all the models needs to be stored as accumulative matrices in the memory between each model and continuous stream of input feature vectors. In the online version of CDP, the local distance  $d(i, j)$  and the cumulative distance  $D(i, j)$  need not be stored as matrices in memory. It suffices to store for each model two column vectors, the current column vector  $col_j$  corresponding to input frame  $j$ , and the previous column  $col_{j-1}$  corresponding to input frame  $j-1$ . Every vector element consists of the cumulative distance  $D$  of the corresponding cell, and possibly other useful data such as frame number, chamfer score and start frame. The previous and the current column vectors for each model are updated for each incoming input frame.

## 6.3 Approach

There are two basic approaches to detection of candidate gesture boundaries, the direct approach and indirect approach. Methods that belong to direct approach compute the motion parameters such as velocity, trajectory curvature [27] or human body activity [28], and then look for abrupt changes (e.g. , zero crossing) in those parameters to find the candidate gesture boundaries. In the indirect approach, the gesture boundaries are detected using the recognition scores. Most indirect methods are based on extensions of Dynamic Programming (DP) algorithms and one such extension is Continuous Dynamic Programming (CDP) which is used in our case to spot and recognize gestures. In CDP, after a provisional set of candidate has been detected, a set of rules is applied to select the best candidate. The

underlying algorithm used for gesture spotting and finding the best candidate gesture is explained next session.

### 6.3.1 Algorithm

The input to the algorithm is the input frame  $j$ , input feature vector  $Q_j$ , chamfer scores of current input frame and set of hand shape templates and the previous column vector  $col_{j-1}$ . The output is the current column vector  $col_j$ . The size of each col is initialized to number of features present in each training model. Each element of the col vector is an object of class which contains members such as frame number, start flag and cumulative distance measure (D). On initialization, the start flag member of the previous column  $col_{j-1}$  is set to zero. For illustration purposes we show the algorithm steps considering a single model, but the same steps have to be repeated for all the training models.

Whenever a motion is detected in the input frame sequence, the chamfer score and co-ordinate location of the detected hand is computed after which the gesture spotting algorithm is invoked.

Input to the Algorithm:

- Current input frame sequence
- Co-ordinate location feature vector  $Q_{jk}$  of the query or the input frame sequence
- Co-ordinate location feature vectors of the model  $M$ .
- Chamfer distance measure  $D_{CM}$  between the input frame sequence and current model hand shape template.

Steps:

1.  $i = 1$
2. If previous  $col_{j-1} \rightarrow start\_flag == 0$ 
  - a. Update the previous  $col_{j-1}$ 
    - i. While  $i \leq m$ , where  $m$  is the size of the model feature vector
      1.  $Col_{j-1}[i] \rightarrow frame\_number = input\ frame\ number$

2. If  $i==1$ 
  - a.  $Col_{j-1}[i] \rightarrow D = D_{CM} + \text{local cost} (d (M_i, Q_{jk}))$
3. else
  - a.  $col_{j-1} [i] \rightarrow D = col_{j-1} [i-1] + D_{CM} + \text{local cost} (d (M_i, Q_{jk}))$
4. end
5.  $col_{j-1} [i] \rightarrow \text{start\_flag} = 1$
- ii. end
3. else Update current  $col_j$ 
  - a. while  $i \leq m$ ,  $m$  is the size of the model feature vector
    - i.  $col_j [i] \rightarrow \text{frame\_number} = \text{input frame number}$
    - ii. if  $i == 1$ 
      1.  $col_j [i] \rightarrow D = D_{CM} + \text{local cost} (d (M_i, Q_{jk}))$
    - iii. else
      1.  $\text{temp} = \text{Min} \{ col_{j-1} [i] , col_{j-1} [i-1], col_j [i-1] \}$
      2.  $col_j [i] \rightarrow D = D_{CM} + \text{local cost} (d (M_i, Q_{jk})) + \text{temp}$
      3. Update previous  $col_{j-1} [i-1]$  with current  $col_j [i-1]$ .
    - iv. End
  - b. End
4. End

Algorithm illustrated above is invoked separately for every gesture model  $M$ . After the cumulative distance measure between the current input frame and for all the models is found, the best candidate gesture or the end point detection algorithm is invoked.

### 6.3.2 Gesture Recognition

The cumulative distances between observation of the current input frame sequence and all the observations in the training model are saved, which is then compared with the specific global threshold value. The threshold value is estimated by aligning the model of a specific class with all the models of the

same class using DTW (Dynamic Time Warping) and the threshold is set to the maximum distance among these distances. The best candidate gesture model which has the cumulative distance measure below the threshold is considered for end point detection. The end point detection classifies the current frame sequence candidate as “spotted” (i.e.) into one of the specific gesture class, if a specified number of frames have elapsed since the candidate was detected. Once the candidate has been detected the start flag of the previous  $col_{j-1}$  is reset to zero for all the models.



## CHAPTER 7

### DYNAMIC TIME WARPING (DTW)

Dynamic Time Warping (DTW) is a method for comparing sequences of observations with the stored observation sequences [18]. DTW is used to align two video sequences to find the best alignment between them. In our experiments, DTW is used for evaluating the global threshold value and also in confusability analysis of gesture vocabulary. We use hand movements and their co-ordinate location as features to measure the similarity between video frames. The features of two video sequences are aligned to find the distances between the two model sequences. DTW finds the appropriate alignment between the two sequences and consequently the minimum DTW score. The lower the score, higher is the similarity between two sequences. The cost function is used by DTW algorithm to find an optimal alignment, referred as optimal warping path [21].

In our case, DTW is used to validate the measure of similarity between gesture sequences (i.e.) if the warping path between two training sequence is below a specific minimum threshold, then the two training gestures are supposed to be aligned or similar to each other. The user is asked to retrain the specific training gesture class in such scenarios.

#### 7.1 DTW Algorithm

Let  $Q = \{Q_1, Q_2, \dots, Q_n\}$  be the feature vectors representation of the query video with  $n$  frames. Let  $M = \{M_1, M_2, \dots, M_m\}$  be the feature vector representation of the model video with  $m$  frames. DTW finds the nearest match for  $Q$  (query) in all possible  $M$ 's training database model. Larger costs will mean larger distances, and will imply less degree of similarity. Warping path specifies the correspondence between  $Q_i$  and  $M_j$  which are vector quantities specifying the co-ordinate locations of the gesturing hand. Let  $W$  be the legal warping path,  $W$  must satisfy the following constraints [18],

- Boundary Conditions: This requires the warping path to start by matching the first element of the query with the first element of M, and end by matching the last element of the query with the last element of M.
- Monotonicity:  $w_{i+1,1} - w_{i,1} \geq 0$ ,  $w_{i+1,2} - w_{i,2} \geq 0$ . This forces the warping path indices  $w_{i,1}$  and  $w_{i,2}$  to increase monotonically with  $i$ .
- Continuity:  $w_{i+1,1} - w_{i,1} \leq 1$ ,  $w_{i+1,2} - w_{i,2} \leq 1$ , This restricts the warping path indices  $w_{i,1}$  and  $w_{i,2}$  to never increase by more than 1, so that the warping path does not skip any elements of Q, and also does not skip any elements of M.

The optimal warping path between Q and M is the legal warping path with smallest cost [18]. Computing the DTW distance takes time  $O(|Q||M|)$ , i.e., time proportional to the product of the lengths of the two sequences.

28.6531	27.4591	31.4006	41.7732	45.3431	49.6488	57.7062	63.5059	64.2573	72.7805
27.4591	25.7099	28.7924	38.0789	40.6079	43.8634	51.0392	56.6039	56.7979	64.56
27.0185	25	27.2947	35.4683	37.2156	39.6989	46.2277	51.614	51.4782	58.8218
28.7924	26.5707	27.4591	33.2415	33.5261	34.4819	39.6232	44.5982	43.8292	50.3289
35.2278	33.0151	32.6497	35.5106	34	32.7567	35.3553	39.4081	37.4833	42.0595
37.4833	35.3553	34.2053	35.1141	32.5576	30.0167	31.0161	34.4819	32.0156	35.8469
44.0454	42.0119	40.3113	39.4462	35.8469	31.7805	30.1496	32.28	28.6356	29.8329
53	51.1077	48.8262	46.0109	41.4005	35.8469	31.0483	31.0644	26.0192	22.4722
60.8358	59.0931	56.356	52.0096	46.7547	40.3113	33.2866	31.3847	25.632	17.1172
68.9638	67.4463	64.195	58.1378	52.3546	45.2769	36.4005	32.5269	26.9072	14.2127

Figure 7.1 An example warping path between two sequences

DTW arranges two sequences of videos on sides of a grid with the unknown sequence on the top and the stored template training video up the left hand side. The row of cell entries shown in figure 7.1 represents the matching score between each known sequence/model feature with all of the unknown sequences. Similarly the column shown in Figure 7.1 represents the matching score between each unknown sequence with all the known sequence. Both the sequences start on the top of the grid. Inside each cell we place a cost of comparing the corresponding features of the two sequences. To find the

best match between two sequences we can find a path through the grid which minimizes the local distance between them. The path shown in yellow in Figure 7.1 gives an example. It should be noted that the minimum path chosen must follow the DTW constraints such as monotonicity, continuity and Boundary conditions. Once an overall best path has been found, the total distance between the two sequences can be calculated. In our experiments, the total warping path distance is compared with a threshold value to test for training gesture similarity.

## CHAPTER 8

### EXPERIMENTS

The experiments were conducted by building a training model and using different test scenarios to measure the performance of the system. The experimental results were conducted under the following hypothesis:

- Gestures are made in a properly illuminated condition.
- The gesturing hand should not move very fast because it might cause motion blur and as a result shape will not be recognized properly.

The foremost importance was to build a vocabulary of training data set which has both the shape and trajectory information. The performance of the algorithm was measured based on the recognition rates, number of false recognitions and number of missed gestures. The system performance was tested by scaling the size of vocabulary. Results are discussed for the reliability of the system on different scales.

#### 8.1 Training Vocabulary

Training vocabulary is built with the hand shape templates and the motion trajectories. A set of nine different hand shape templates and five different trajectory patterns was used. For each trajectory patterns we had two motion trajectory vector models performed by different persons. The orientations of the motion patterns and hand shape templates are shown in figure 8.1 and 8.2 respectively.

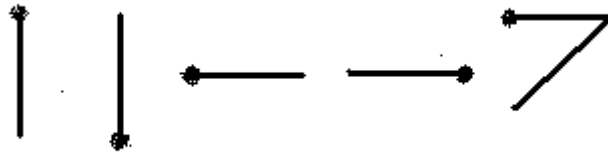


Figure 8.1 Gesture Motion Patterns



Figure 8.2 The nine hand-shapes dataset used in our experiments

The motion patterns shown in figure 8.1 starts at the point where there is a dot and then follows the specified trajectory, we have used five direction (left, right, up, down and number '7') motion patterns. The edge information of each hand shape templates are embedded in a vector space that contains the co-ordinate locations of the binary edge pixel present in a hand shape template. For each hand shape data set, we have a singly linked list vector containing the co-ordinate location of the binary edge pixel.

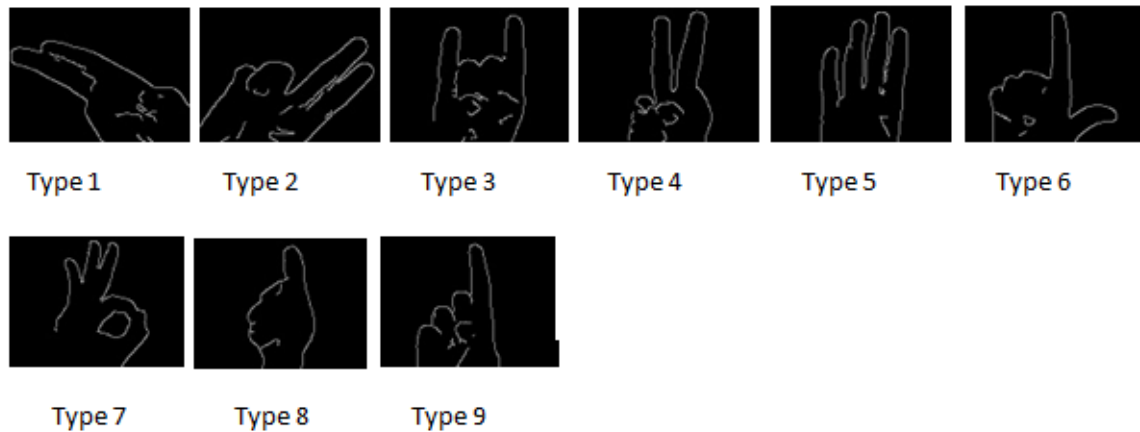


Figure 8.3 Nine different template edge images used in our experiments

## 8.2 Measures of Accuracy

The experiments were made with the combination of different shape and motion patterns. The size of vocabulary was scaled to see the performance of the system. The total number of gesture classes was scaled from 2 to 25. We have nine different shapes and five different classes which gave us a total of 45 different gesture classes that can be used, but there are some shapes which were more or less similar, as a result were not able to use them in the same motion trajectory path.

In order to show the robustness of the system we consider three cases to check the accuracy of the system.

Case 1:

All shapes gestured in same motion pattern.

The system was tested with all nine shapes gestured in the same 'upward moving' motion pattern. The recognition rate was about 65%. The main reason for the reduced recognition rate was the number of false detection between type 4, type 5, type 8 and type 9 hand shapes. The rest of the hand

shapes alone was giving a recognition accuracy of more than 90 %. The tabulation of the result is given below.

Table 8.1 Recognition rates of gestures in same direction

Gestures	Recognition Rate	No of mismatches	No of missed Gestures (No recognition)	No. of over predicted Gestures
All nine shapes	75%	25/120	1/120	4/120
Only five shapes (excluding type 4, 5, 8, 9).	94%	4/120	0	2/120
Shapes (type 4, 5, 8, 9) alone	51%	53/120	5/120	7/120

The confusion matrix of the test made using hand shape type 4, 5, 8, 9 is shown below.

Table 8.2 Confusion matrix of training template types 4, 5, 8 and 9 for same motion pattern

	Predicted/Spotted gestures by Gesture Spotting Algorithm			
	Type 4	Type 5	Type 8	Type 9
Type 4(total 30)	17	1	7	3
Type 5(total 30)	1	15	11	5
Type 8(total 30)	0	0	16	14
Type 9(total 30)	1	0	14	13

The reason for gesture mismatch was mainly because of type 8 gesture which had the shape information more or less similar with the rest of the shape templates. Upon further experiments by not considering the type 8 shape template, we achieved a recognition rate of about 85 %. These results show the effectiveness in our confusability analysis method which checks and specify if the temporal and spatial information of the training gesture pattern are similar.

Case 2:

Single shape gestured in all motion patterns. Only the motion trajectory information is used for classification of gestures. The recognition results and the confusion matrix are shown below.

Table 8.3 Results of Single hand shape, all motion trajectories

Recognition Rate	No of mismatches	No of missed Gestures (No recognition)	No. of over predicted Gestures
91.6%	5/120	1/120	3/120

The mismatches score was high because of the recognition of a gesture class during transition phase from one gesture pattern to another.

Case 3:

The objective of the gesture recognition system is to have a robust recognition framework. Out of the existing hand shape templates we used different combinations of hand shapes and motion patterns to get better recognition accuracy. The vocabulary size was scaled from 2 to 25 gesture classes. An example best vocabulary model and its results are shown below.

The vocabulary consisted of type 1, 4, 5, 6, 7 which was gestured in all specified motion trajectories. The test was performed with 4 users with the total of 200 gestures.

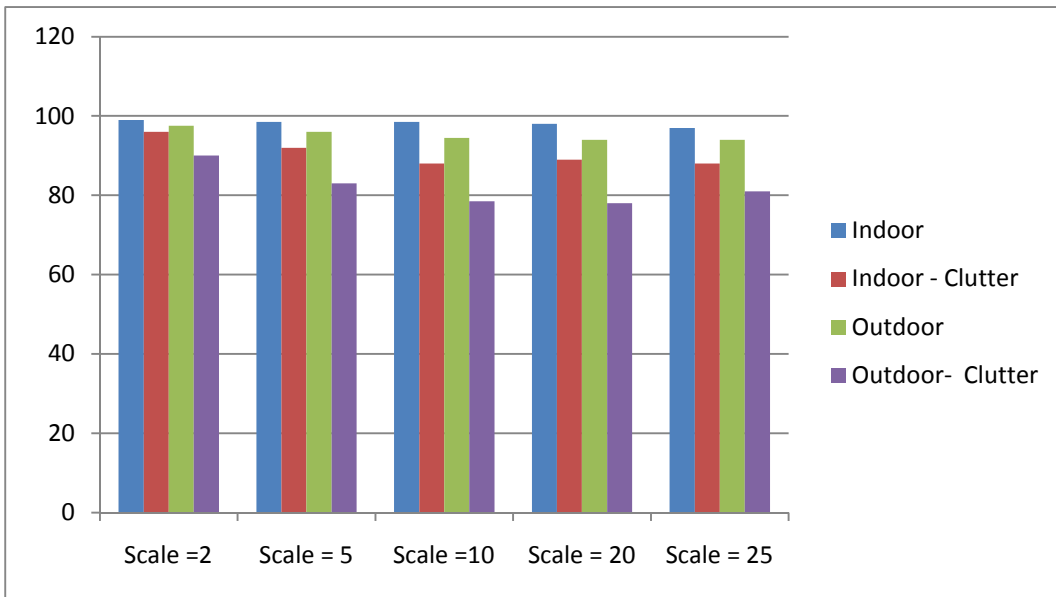


Table 8.4 Recognition rates in different background conditions

Background Setup	Recognition Rate	No of mismatches	No of missed Gestures (no recognition)	No. of over predicted Gestures
Indoor condition uniform background	97%	4	0	2
Outdoor condition uniform background	94%	7	4	1
Highly cluttered Indoor environment	88%	16	10	0
Highly cluttered outdoor environment	81%	21	17	0

### 8.3 Reliability of System the on different Scales of Vocabulary

The reliability of the system at different scales of output is shown in the figure 8.4.



Scale = No. of training gesture classes

Figure 8.4 Recognition rates at different scales of vocabulary

It can be seen that the system has resulted in more or less similar recognition results at different scales of vocabulary. It should be noted that the recognition rates highly depend on the kind of vocabulary preferred and lighting conditions. If the chosen vocabulary has similar motion gesture patterns or hand shapes, then the recognition accuracy may decrease dramatically. The results shown in figure 8.4 are based on the best vocabulary chosen from the set of motion and hand shape patterns gestured under uniform lighting conditions.

Figure 8.5 shows the example input frame sequence where the hand tracking framework captures the gesturing hand locations in consecutive frames.

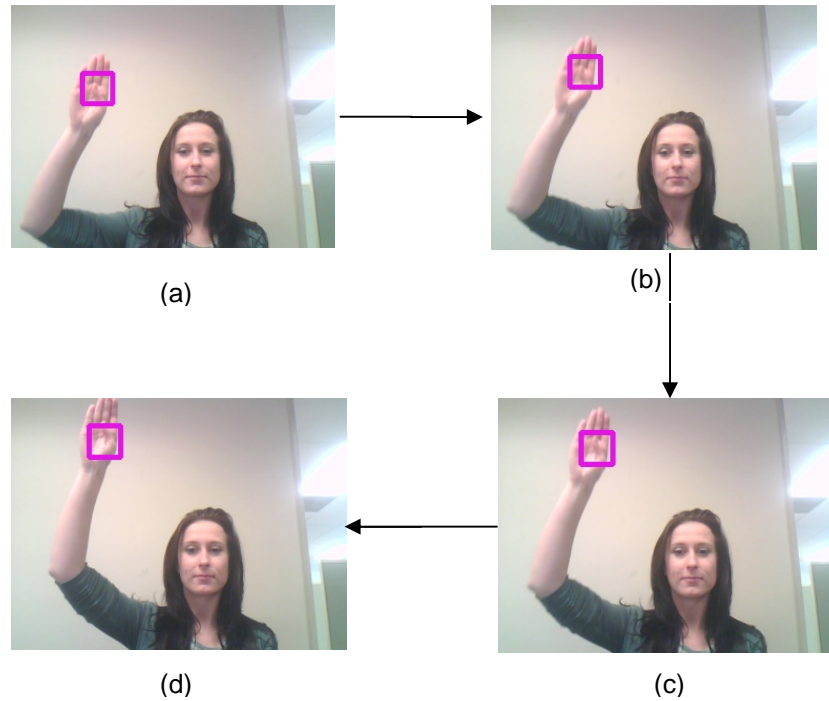


Figure 8.5 Type-5 Hand Shape Pattern Gesturing in Upward Motion. (a), (b), (c) and (d) represents the input video frame sequences

We compared the performance of our proposed system with another system proposed in [1] which uses the same hand shape and motion information for gesture classification. But the recognition strategy used in [1] does not use Dynamic Programming or chamfer distance score. [1] Used the contour discriminant based hand shape detection and Hidden Markov Model framework (HMM) for gesture classification.

Gesture Vocabulary Used:

- In order to test the accuracy of our system compared to [1], we used four gesture vocabulary similar to the one used in [1].
- [1] Used four gesture models with two different hand shapes (open hand and closed hand) and two different motion trajectories (left and right nature of motion).
- We used same motion trajectories with type 4 and type 6 hand shapes, the results of which are shown below.

Table 8.5 Comparison with other approach

Recognition System Used	Recognition Rates	No. of Mismatches
Our Method	96%	4/80
System proposed in [1]	77.5%	18/80

The experiment was conducted in uniform lighting condition as required by [1] with two different users. The experimental results show that our proposed recognition system has a better performance when compared to the existing system which uses the same recognition strategy.

## CHAPTER 9

### CONCLUSION AND FUTURE DIRECTIONS

This thesis is focused on creating a robust gesture controller which can be used as an input interface just as a keyboard, mice or touch pad. This is achieved by spotting and recognizing specific gestures from a continuous stream of input video obtained through any webcam or camera. The proposed gesture spotting and recognition framework runs in real-time, works well even in the presence of background clutter. The advantages of the system are the following,

- User does not need to wear a glove or any tracking device.
- User can set his own desired gestures (i.e.) Hand shapes and nature of motion.

Some of the main contributions of this work are a novel approach for combining the motion and shape parameters for recognizing gestures, confusability analysis of gesture vocabulary, scheme for automatic start and end point detection and real-time gesture recognition.

#### 9.1 Future Directions

There are several research topics that we are interested in pursuing as a future work, with the goal of improving the system performance, user experience and scaling the size of gesture vocabulary. The current system uses a two dimensional web cam interface which results in a mapping of real world data into a two dimensional data. Specifically we neglect the orientation or the angle at which the hand is gestured from the view point. This can be solved by considering hand shapes as 3D models which increases the chances of effectively using an increased gesture vocabulary. We are also interested in building a similar system using a GPU (Graphical Processor Unit) and compare the execution time differences with the proposed system.

## REFERENCES

- [1] Aditya Ramamoorthy, NamrathaVaswani, SantanuChaudhury, Subhashis Banerjee," Recognition of dynamic hand gestures," in the Journal of The Pattern Recognition Society, 2003, pp. 2069-2081.
- [2] M .K. Bhuyan, P. K. Bora and D. Ghosh," Trajectory Guided Recognition of Hand Gestures having only Global Motions," in World Academy of Science, Engineering and Technology, 2008, pp. 832-843.
- [3] Ming-Hsuan Yang, NarendraAhuja, and Mark Tabb," Extraction of 2D Motion Trajectories and its Application to Hand Gesture Recognition," IEEE Transactions of Pattern Analysis and Machine Intelligence, Vol. 24, No. 8, Aug 2002, pp. 1061-1074.
- [4] Ming-Yu Liu, OncelTuzel, Ashok Veeraraghavan, and Rama Chellappa," Fast Directional Chamfer Matching," Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR '10), Jun 2010, pp. 1696 – 1703.
- [5] SushmitaMitra, and TinkuAcharya," Gesture Recognition: A Survey," IEEE Transactions of Systems, Man and Cybernetics, Vol. 37, No. 3, May 2003, pp. 311-324.
- [6] MichalisPotamias and VasillisAthitsos," Nearest Neighbour Search Methods for Handshape Recognition," Conference on Pervasive Technologies Related to Assistive Environments (PETRA), July 2008.
- [7] Jonathan Alon, VassilisAthitsos and Stan Scarloff," Accurate and Efficient Gesture Spotting via Pruning and Subgesture Reasoning," IEEE Workshop on Human Computer Interaction, October 2005, pp.189-198.
- [8] Ryuichi Oka," Spotting Method for Classification of Real World Data," in The Computer Journal, Vol. 41, No.8, 1998, pp.559-565.
- [9] Donald G. Bailey," An Efficient Euclidean Distance Transform," International Workshop on Combinatorial Image Analysis, 2004, pp. 394-408.
- [10] Shuhei Horimoto, Daisaku Arita and Rin-ichiro Taniguchi," Real-time Hand Shape Recognition for Human Interface," in Proceedings of the 12<sup>th</sup> Conference on Image Analysis and Processing (ICIAP'03), 2003.

- [11] Nobuharu Yasukochi, Aya Mitome, and Rokuya Ishii, "A Recognition Method of Restricted Hand Shapes in Still Image and Moving Image as a Man-Machine Interface," *Human System Interactions*, May 2008, pp. 306-310.
- [12] Rafael C. Gonzalez, Richard E. Woods, "Digital Image Processing," 3<sup>rd</sup> Edition, Reading MA: Addison-Wesley, 1992.
- [13] M. Kass, A. Witkin and D. Terzopoulos, "SNAKE: Active Contour Models," *International Journal of Computer Vision*, 1988, pp. 321-331.
- [14] R. O. Duda and P. E. Hart, "Pattern Classification and Scene Analysis," New York: Wiley, 1973.
- [15] C. Sul, K. Lee and K. Wohn, "Virtual Stage: A location-based Karaoke System," *Multimedia, IEEE*, Vol.5, Apr-1998, pp.42-52.
- [16] L. Campbell and A. Bobick, "Using Phase Space Constraints to represent Human Body Motion," in *International Workshop on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June 1995, pp. 338-343.
- [17] Joseph J. La Viola Jr., "A Survey of Hand Posture and Gesture Recognition Techniques and Technology," Technical Report, Brown University, June 1999.
- [18] A. Stefan, H. Wang, and V. Athitsos, "Towards Automated Large Vocabulary Gesture Search," in *IEEE Workshop on Computer Vision and Pattern Recognition for Human Communicative Behavior Analysis (CVPR)*, 2008, pp.1-8.
- [19] Philip Ian Wilson and John Fernandez, "Facial Feature Detection using Haar Classifiers," *Journal of Computing Sciences in Colleges*, Vol. 21, Apr- 2006, pp. 127-133.
- [20] Sturman, David J., and David Zeltzer, "A survey of Glove-based Input," in *IEEE Computer Graphics and Applications*, Vol. 14, 1994, pp. 30-39.
- [21] H. Wang, A. Stefan, S. Moradi, V. Athitsos, C. Neidle, and F. Kamnagar, "A System for Large Vocabulary Sign Search," in *Workshop on Sign, Gesture and Activity (SGA)*, September 2010, pp. 1-12.
- [22] Source Forge, "Haar-Cascade Classifier Trained Face Detector Dataset," Website, 2005.  
[http://opencvlibrary.svn.sourceforge.net/viewvc/opencvlibrary/trunk/opencv/data/haarcascades/haarcascade\\_frontalface\\_alt2.xml](http://opencvlibrary.svn.sourceforge.net/viewvc/opencvlibrary/trunk/opencv/data/haarcascades/haarcascade_frontalface_alt2.xml)

- [23] Edgar Seeman, Bastian Leibe, Krystian Mikolajczyk, and Bernt Schiele, "An Evaluation of Local Shape-based Features for Pedestrian Detection," in Proceedings of British Machine Vision Conference (BMVC), 2005.
- [24] Michael J. Jones and James M. Rehg, "Statistical Color Models with Application to Skin Detection," in Technical Report Series, Cambridge Research Laboratory, Dec 1998.
- [25] Rabiner, L. R., "A Tutorial on Hidden Markov models and selected applications in Speech Recognition," in Proc. IEEE, Vol -77, 1989, pp. 257-286.
- [26] Alexandros Sterios Manos, "A Study of Out-of-Vocabulary Word Modeling for a Segment-Based Keyword Spotting System," Master's Thesis, Massachusetts Institute of technology, April 1996.
- [27] Kang H., Lee C. Jung K., "Recognition-based Gesture Spotting in video games," Pattern Recognition letters, Vol. 25, 2004, 1701-1714.
- [28] Kahol K., Tripathi P., Panchanathan S., "Automated Gesture Segmentation from Dance Sequences," in Automatic Face and Gesture Recognition, 2004, 883-888.
- [29] Collins English Dictionary and Thesaurus, Harper Collins Publishers, 1981.



## BIOGRAPHICAL INFORMATION

Rajesh Radhakrishnan was born in Chennai, India. He received his B.S. degree from SASTRA University, India, in 2009, his M.S. degree from The University of Texas at Arlington in 2011, in Electrical Engineering. His current research interest is in Computer Vision, Machine Learning, Distributed Computing and Embedded Systems.