A Framework for Sign Language Recognition using Support Vector Machines and Active Learning for Skin Segmentation and Boosted Temporal Sub-units

by

George Awad

Bsc., Msc.

A thesis submitted in fulfilment of the requirement for award of
Doctor of Philosophy (Ph.D.)

to the

Dublin City University
Faculty of Engineering and Computing
School of Computing

Supervisor: Dr. Alistair Sutherland

2007

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award pf Ph.D. is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: George Awad (George Awad)
ID No.: 5414671
Date: 10/8/2007

Acknowledgement

Beginning a new life in a new country with new people and culture is a hard thing to cope with. Doing a Ph.D. in such a new environment is even more hard without the help and encouragement of many people and the guidance of God. That's why I would like to thank God first and foremost for helping me to finish my work and sending me a lot of people across my journey from the first day till the end.

At first, I want to thank deeply my supervisor Dr. Alistair Sutherland for encouraging me to apply for Ph.D at first place and supporting me even from before I arrive to Ireland. He was there all the time guiding my work, reviewing my publications, reading and correcting my thesis and even supporting me during my searching for job opportunities. I am also very grateful to the school of computing in DCU for funding my Ph.D and supporting me in all my conference travels.

Also, I would like to thank Dr. Junwei Han very much for collaborating with me in my research and sharing with me his experience to help us making our ideas come true. I would like to thank all the postgraduate students here that I have met and dealt with, either personally or in work and especially Tommy Coogan my group colleague who I shared with him our lab place, conference journeys, writing papers, and developing our demo.

I would like to thank Dr. Richard Bowden for sharing with us his sign language dataset. Also, I want to thank all the Egyptian community that I have met in our church here in Dublin, my brother in the states and my special friend in Canada for all their continuous support and praying for me during my down times.

Finally, I am indebted so much to my parents in Egypt for all their love, sup-

port, encouragement and endless care about my studies and life, as without them I wouldn't have been here right now. I owe them my past, present and future.

Thank you all, so much.

George Awad

A Framework for Sign Language Recognition using Support Vector Machines and Active Learning for Skin Segmentation and Boosted Temporal Sub-units

George Awad

Dublin City University

Glasnevin,

Dublin 9

Abstract

This dissertation describes new techniques that can be used in a sign language recognition (SLR) system, and more generally in human gesture systems. Any SLR system consists of three main components: Skin detector, Tracker, and Recognizer. The skin detector is responsible for segmenting skin objects like the face and hands from video frames. The tracker keeps track of the hand location (more specifically the bounding box) and detects any occlusions that might happen between any skin objects. Finally, the recognizer tries to classify the performed sign into one of the sign classes in our vocabulary using the set of features and information provided by the tracker.

In this work, we propose a new technique for skin segmentation using SVM (support vector machine) active learning combined with region segmentation information. Having segmented the face and hands, we need to track them across the frames. So, we have developed a unified framework for segmenting and tracking skin objects and detecting occlusions, where both components of segmentation and tracking help each other. Good tracking helps to reduce the search space for skin objects, and accurate segmentation increases the overall tracker accuracy.

Instead of dealing with the whole sign for recognition, the sign can be broken down into elementary subunits, which are far less in number than the total number of signs in the vocabulary. This motivated us to propose a novel algorithm to

model and segment these subunits, then try to learn the informative combinations of subunits/features using a boosting framework. Our results reached above 90% recognition rate using very few training samples.

Chapter 1

Introduction

1.1 Introduction

In daily life, human beings communicate with each other and interact with computers using gestures. As a kind of gesture, sign language (SL) is the primary communication media for deaf people. Everyday, millions of deaf people all over the world are using SL to get useful information and exchange ideas. Therefore, in recent years, SL recognition has gained a lot of attention and a variety of solutions have been proposed. Sign gestures might be treated as a composition of hand shape, motion, position, and facial expression. Thus, SL recognition requires knowledge of all of these. Generally, a SL recognition system should contain three major modules: skin segmentation and tracking (SST), feature extraction, and recognition. The first module is to acquire and locate hands and face across the video frames. The purpose of the second module is to prepare useful features for classification.

Fig. 1.1 demonstrates a general system architecture overview for a SLR system. Based on segmented hands and face, we can extract the hand shape, orientation, and facial expression features. Through analyzing the tracked skin objects, we obtain the hand motion trajectories, hand position, and lip movement. Finally, classifiers are trained to recognize the signs.

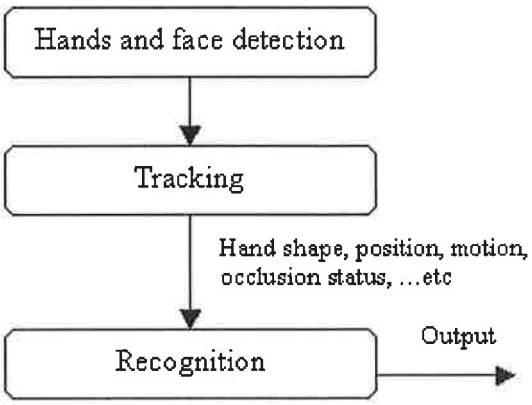


Figure 1.1: System architecture

1.2 Device vs Vision approach to SLR

According to the means of capturing features, SL recognition techniques can be classified into two groups: glove-based and vision-based. The former group of approaches requires users to wear data or colour gloves. The glove enables the system to avoid or simplify the segmentation and tracking task. However, its disadvantages are apparent. On the one hand, users have to carry a hardware device, which makes them feel uncomfortable. Sometimes, they cannot perform accurate gestures with the gloves. On the other hand, the glove-based methods might lose the facial expression information, which is very important for the SL recognition as well.

In comparison, the vision-based methods rely on computer vision techniques without needing any gloves, which is more natural for users. However, one difficulty is how to accurately segment and track hands and face. SST plays an important role in vision-based SL recognition. Only after skin objects have been acquired, useful descriptions such as hand shape, motion, and facial expression, and further recognition are possible. In other words, SST is the cornerstone of SL recognition. In order to produce high quality SST, two techniques must be developed: a powerful skin colour model and a robust tracker. The skin colour model offers an effective way to detect and segment skin pixels. It should be able to handle illumination and human skin variations. The tracker is responsible for locating skin objects. For SL recognition, it should be capable of predicting occlusions that frequently happen in real world SL

conversations. The purpose of occlusion detection is to keep track of the status of the occluded parts, which helps to reduce the search space in the recognition phase.

1.3 Overview of the proposed SLR system

This work aims to provide an SST framework for SL recognition, then given that we can acquire the required useful features we propose a novel solution for SLR based on boosting SL subunits. To achieve precise skin segmentation, we introduce a novel skin colour model integrating SVM active learning and region segmentation. This model consists of two stages: a training stage and segmentation stage. In the training stage, first, for the given gesture video, a generic skin colour model is applied to the first few frames, which obtains the initial skin areas. Afterwards, a binary classifier based on SVM active learning is trained using obtained initial skin areas as the training set. In the segmentation stage, the SVM classifier is incorporated with the region information to yield the final skin colour pixels. The contribution that distinguishes the proposed model from other existing skin colour algorithms is twofold. First, the SVM classifier is trained using the training data automatically collected from the first several video frames, which does not need human labour to construct the training set. More importantly, the training is performed for every video sequence. It is adaptive to different human skin colours and lighting conditions. The skin colour model can also be updated with the help of tracking to deal with illumination variation. Second, region information is adopted to reduce noise and illumination variation. Moreover, active learning is employed to select the most informative training subset for the SVM, which leads to fast convergence and better performance.

As for the tracker, we extend the previous work of our group in three ways. First, in the previous work they used the colour glove to avoid the segmentation issue. In this work, we are more interested in improving SL recognition in natural conversation. Three features, skin color, motion, and position, are fused to perform accurate skin object segmentation. Additionally, the previous work tracked two hands wearing

color glove only. However, the proposed work can segment and track two hands and face. The obtained face information definitely could facilitate the recognition. Second, we apply a Kalman filter (KF) to predict occlusions in the same way as the previous work. Nevertheless, our KF is based on the skin colour instead of colour glove. Third, in the proposed work, tracking and segmentation tasks are approached as one unified problem where tracking helps to reduce the search space used in segmentation, and good segmentation helps to accurately enhance the tracking performance.

Despite the great deal of effort in SLR so far, most existing systems can achieve good performance only with small vocabularies or gesture datasets. Increasing vocabulary inevitably incurs many difficulties for training and recognition, such as the large size of required training set, signer variation and so on. Therefore, to reduce these problems, some researchers have proposed to decompose the sign into subunits. In contrast with traditional systems, this idea has the following advantages. First, the number of subunits is much smaller than the number of signs, which leads to a small sample size for training and small search space for recognition. Second, subunits build a bridge between low-level hand motion and high-level semantic SL understanding. In general, a subunit is considered to be the smallest contrastive unit in a language in the field of linguistics. A number of researchers have provided evidence that signs can be broken down into elementary units. However, there is no generally accepted conclusion yet about how to model and segment subunits in the computer vision field.

This work investigates the detection of subunits from the viewpoint of human motion characteristics. We model the subunit as a continuous hand action in time and space. It is a motion pattern that covers a sequence of consecutive frames with interrelated spatio-temporal features. In terms of the modelling, we then integrate hand speed and trajectory to locate subunit boundaries. The contribution of our work lies in three points. First, our algorithm is effective without needing any prior knowledge like the number of subunits within one sign and the type of sign. Second, the trajectory of hand motion is combined so that the algorithm does not rely on clear

pauses as in some previous related work. Finally, because of the use of an adaptive threshold in motion discontinuity detection and refinement by temporal clustering, our method is more robust to noise and signer variation.

After segmenting the SL subunits, we attempt to develop an effective SLR system using the AdaBoost algorithm which tries to learn informative subunit and feature combinations needed to achieve good classification performance. To our best knowledge, very little work has been done using Adaboost in SLR. We present two variations for learning boosted subunits. In the first case, we train the sign classes independently, and in the second case, we train the classes jointly, which permits the various classes to share the weak classifiers to increase the overall performance.

The presented work enables us to efficiently recognize SL with a large vocabulary using a small training dataset. One important advantage of our algorithm is that it is inspired by human signing behaviour and recognition ability so it can work in a manner analogous to humans. Experiments on real-world signing videos and the comparison with classical HMM-based weak classifiers demonstrate the superiority of the proposed work.

In this thesis, we aimed to provide new different techniques that can be applied in SLR applications. Our goal was to contribute to research in skin segmentation, hand and face tracking, modelling and recognizing signs efficiently based on human behaviour in performing and recognizing signs using informative subunits of the signs.

1.4 Overview of the Thesis

The next chapter introduces the reader to the literature review of the different SLR systems proposed by different research groups to solve the problem of SLR.

Chapter 3 gives a review of current skin segmentation techniques and discusses our proposed skin segmentation algorithm with various evaluation results.

Chapter 4 introduces our proposed SST system and provides some experimental results for skin segmentation and tracking.

Chapter 5 introduces the subunit modelling and segmentation algorithm and ends

with some evaluation experiments.

Chapter 6 introduces our SLR system based on learning boosted subunits and presents the experimental results of the classification.

Chapter 7 concludes with a summary, and gives some future work directions.

Chapter 2

Sign Language Recognition Literature review

2.1 Introduction

In taxonomies of communicative hand/arm gestures, Sign Language (SL) is often considered as the most structured form of gesture, while gestures that accompany verbal discourse are described as the least standardized. SL communication also involves non-manual signals (NMS) through facial expressions, head movements, body postures and torso movements [Ong and Ranganath 05].

SLR therefore requires observing these features simultaneously together with their synchronization, and information integration. As a result, SLR is a complex task and understanding it involves great efforts in collaborative research in machine analysis and understanding of human action and behaviour; for example, face and facial expression recognition [Kong et al. 04, Pantic and Rothkrantz 00], tracking and human motion analysis [Gavrila 99, Wang et al. 03], and gesture recognition [Pavlovic et al. 97].

As non-SL gestures often consist of small limited vocabularies, they are not a useful benchmark to evaluate gesture recognition systems. However, SL on the other hand can offer a good benchmark to evaluate different gesture recognition systems because it consists of large and well-defined vocabularies, which can be hard to disambiguate

by different systems.

In real life, we can imagine many different useful applications for SLR such as:

- sign-to-text/speech translation system or dialog systems for use in specific public domains such as airports, post offices, or hospitals
  [McGuire et al. 04, Akyol and Canzler 02].

- In video communication between deaf people, instead of sending live videos, SLR can help to translate the video to notations which are transmitted and then animated at the other end to save bandwidth [Kennaway 03].

- SLR can help in annotating sign videos [Koizumi et al. 02] for linguistic analysis to save a lot of human labour manually in ground truthing the videos.

SL gesture data is mainly acquired using cameras (vision-based) or sensor devices (glove-based) [Sturman and Zeltzer 94]. We are interested here in the vision-based approach as the glove-based approach has the limitation of being an unnatural way of performing signs. However, it can simplify a lot the tasks of segmentation (especially in the presence of occlusions) and tracking. But it ignores the fact that we need the facial expression as an important feature. In the next sections, we will try to summarize the related work done by different research groups in SLR. We will cover the three main tasks of hand detection and tracking, feature extraction and classification.

2.2 Hand detection and tracking

In almost all SLR systems, the hand(s) must be detected in the image sequence, and this is usually based on features like colour, motion, and/or edge. The colour cue is used by skin colour detection or using colour gloves such as in [Sweeney and Downton 96, Sutherland 96, Bauer and Kraiss 02, Assan and Grobel 97, Bauer and Kraiss 01]. When skin colour is used, the user is usually required to wear long sleeves to avoid the skin colour of the arm area. Skin colour was combined with a motion cue in

[Akyol and Alvarado 01, Imagawa and Igi 98, Yang et al. 02] and with edge information in [Terrillon et al. 02]. Different assumptions were used to distinguish the hands from the face such as that the head is relatively static compared to the hands [Akyol and Alvarado 01, Imagawa and Igi 98] or that the head is bigger than the hands [Yang et al. 02].

A common requirement for the motion cue, is that the hand must be continuously moving as in [Huang and Jeng 01] where the hand was detected by logically ANDing difference images with edge maps and skin-color regions. In [Cui and Weng 00, Cui and Weng 99] a hierarchical nearest neighbour decision rule was used to map partial views of the hand to previously learned hand contours to obtain an outline of the hand.

In [Huang and Huang 98] the hands were detected assuming that it is the only object moving against a stationary background and that the head is relatively stationary. In [Ong and Bowden 04] they used a boosted cascade of classifiers to detect hand shapes, where dark backgrounds were used and signers were asked to wear long-sleeved dark clothing. Other related work also tried to localize body parts such as the body torso [Bauer and Kraiss 02, Assan and Grobel 97], or elbow and shoulders [Hienz et al. 96] along with the hands and face, based on the body geometry and colour cues. This helps to reference the position and movement of the hands to the signer's body.

Hand tracking can be done either in 2D or 3D. In 2D tracking, approaches can be classified to boundary-based [Huang and Huang 98, Cui and Weng 00], view-based [Huang and Jeng 01], blob-based [Tanibata et al. 02, Imagawa and Igi 98], and matching motion regions [Yang et al. 02]. One of the hard problems in tracking is occlusion. Generally speaking, in most systems that are based on skin colour, occlusion handling is poor and not satisfactory. Some systems try to predict the hand location based on the model dynamics and previous frame positions with the assumption of small constant hand motion [Starner et al. 98, Imagawa and Igi 98].

In [Starner et al. 98] they subtracted the face region from the merged face/hand blob, but unfortunately this method only can handle small overlaps. In [Imagawa 00]

they applied a sliding observation window over the merged blob of face/hand and the likelihood of the window subimage was calculated to classify it to one of the possible hand shape classes. The overlapping hands and face were distinguished in [Tanibata et al. 02] by using hand and face texture templates. This method is not robust to change in hand shape, face orientation, or large change in facial expression.

Another interesting approach that does not track the hands and face separately [Zieren et al. 02, Sherrah 00], but rather applies probabilistic reasoning (such as heuristic rules [Zieren et al. 02] or Bayesian networks [Sherrah 00]) for simultaneous assignment of labels to the possible hand/face regions, assuming that the skin blobs can only be assigned to the hands, thus not allowing for other skin regions in the background. This allows more robust tracking that can deal with high overlapping and fast hand movement, along with complex hand interactions. Multiple features were used such as motion, colour, orientation, size and shape of blobs, distance relative to other body parts, and Kalman filter prediction.

In [Assan and Grobel 97, Bauer and Kraiss 02, Huang and Huang 98] uniform backgrounds were used to simplify the constraints. However, a few systems such as in [Chen et al. 03], allow complex cluttered background that include moving objects and apply background subtraction to extract the foreground under the assumption that the hand is constantly moving. In contrast to the above approaches, there are some systems that use 3D models [Vogler and Metaxas 97, Downton and Drouet 92] by using multiple cameras to estimate the body parts and/or avoid occlusions but of course with a great computational cost.

As skin segmentation is one of the main research areas that we will address in this work, the next section reviews the major techniques used to detect skin pixels in images or videos.

2.2.1 Skin segmentation review

In general, skin detection methods [Vezhnevets et al. 03] can be classified into two groups. Pixel-based methods that classify each pixel as skin or non-skin indepen-

dently from its neighbours, and Region-based methods that take into consideration the spatial arrangement of skin pixels [Kruppa et al. 02, Yang and Ahuja 98, Jedynak et al. 02]. This survey concentrates on Pixel-based methods, as it is the most commonly used among existing skin segmentation techniques. In the past few years, a number of comparative studies of skin colour pixel classification have been reported. The first large skin database, the compaq database, was created by [Jones and Rehg 02] and used to compare three different techniques namely, thresholding the red/green ratio, colour space mapping with 1D indicator, and RGB skin probability map [Brand and Mason 00]. Gaussian and Gaussian Mixture Models (GMMs) were compared across nine chrominance spaces in [Terrillon 00]. A comparison of five colour spaces and two non-parametric skin-modelling methods (lookup table and Bayes skin probability map) has been provided in [Zarit et al. 99]. In [Lee and Yoo 02] the single Gaussian and GMM have been compared in different chrominance spaces and a new model is proposed.

2.2.2 Colour representations

Many colour spaces exist with different properties and many of them have been used in skin segmentation. There has been no agreement on whether to use only the pixel chrominance ignoring the luminance or not. We will briefly review the most popular colour spaces used in the image-processing field [Foley et al. 90]:

- RGB: The three primary colours red (R), green (G), and blue (B).
- HSV: The hue (H), saturation (S), and intensity value (V), which are the three attributes in human perception of colour.
- YCbCr: The luminance (Y) and chrominance (Cb and Cr channels).

In our skin segmentation approach, we adopt the RGB colour space. We think that by using only the chrominan information we might be losing information, so we try to cope with the original colour 3D features.

2.2.3 Skin segmentation classification algorithms

The objective of the classification algorithms is to build a decision rule that will discriminate between skin and non-skin pixels. This is usually done by building a metric that measures the distance of the pixel to the skin colour. The type of this metric is defined by the skin colour modelling method [Vezhnevets et al. 03]. We can categorize skin modelling methods into three main categories: explicit decision rules, non-parametric modelling and parametric modelling.

2.2.3.1 Explicitly defined skin region

An obvious method to build a skin classifier is to define empirically some decision rules for skin clustering in some colour space. *Algorithm 1* shows a particularly widely used rule [Peer 03] that is developed for RGB colour space.

Algorithm 1 (R,G,B) is classified as skin if: R>95 and G>40 and B>20 and max{R,G,B} - min{R,G,B}>15 and |R-G|>15 and R>G and R>B

The advantage of this method is the simplicity of the decision rules, which makes it very fast and easy to implement. The disadvantage of this method is the need for a suitable colour space and adequate decision rules (tested on a representative dataset) that result in reasonable classification accuracy for a particular application.

2.2.3.2 Nonparametric skin distribution modelling

The key idea here is to use the training data to estimate the skin colour distribution without defining an explicit model of the skin colour [Brand and Mason 00]. Many variations exist in the literature from histogram based modelling to the Self Organizing Map (SOM) based on neural networks.

- **Normalized lookup table (LUT)**

The histogram-based approach to skin segmentation has been used in several face detection and tracking algorithms [Zarit et al. 99, Vezhnevets et al. 03]. This approach relies on the assumption that skin colours form a cluster in some colour space

[Jones and Rehg 02]. A 2D or 3D histogram is constructed by quantizing the colour space into a number of bins, each corresponding to particular range of colour components (pairs or triads). These bins form the lookup table (LUT) and each of them stores the frequency of this particular colour in the training skin images. Normalizing the histogram bin values converts the histogram values to a discrete probability distribution:

$$P_{skin}(c) = \frac{skin[c]}{N} \tag{2.1}$$

Where *skin[c]* gives the count in the histogram bin associated with the colour vector c, and N is the normalization coefficient, (sum of all bin values [Jones and Rehg 02], or maximum bin value [Zarit et al. 99]). The normalized values of the histogram bins form the likelihood that the corresponding colours are from skin pixels.

- **Bayes Classifier**

Using two histograms; one for skin and one for non-skin, we can use Bayes Theorem to choose the most likely hypothesis given the value of a colour vector c. Therefore the probability that a given colour c belongs to skin class or non-skin class is given by the equations 2.2 and 2.3 respectively:

$$P(c|skin) = \frac{skin[c]}{T_s} \tag{2.2}$$

$$P(c|\sim skin) = \frac{\sim skin[c]}{T_n} \tag{2.3}$$

Where *skin[c]* and *˜skin[c]* is the pixel count in bin c of the skin and non-skin histograms respectively, and T_s and T_n are the total counts in skin and non-skin histograms respectively. Then we have from Bayes Theorem:

$$\frac{P(skin|c)}{P(\sim skin|c)} = \frac{P(c|skin)P(skin)}{P(c|\sim skin)P(\sim skin)} \tag{2.4}$$

This corresponds to Maximum a posteriori (MAP) criterion, where *P(skin)* and *P(~skin)* can be estimated from the overall number of skin and non-skin samples in the training set [Jones and Rehg 02, Zarit et al. 99]. A decision rule can then be obtained by comparing equation 2.4 to a threshold. After some manipulations, eq. 2.4 can be written as [Vezhnevets et al. 03]:

$$\frac{P(c|skin)}{P(c|\sim skin)} > \Theta \tag{2.5}$$

$$\Theta = K \star \frac{1 - P(skin)}{P(skin)}, \; where \, K \, is \, constant$$

Which is seen to be a Maximum likelihood (ML) detector. It turns out [Zarit et al. 99] that MAP and ML Bayes classification rules are equivalent with different threshold values, this is because the priori probability *P(skin)* affects only the choice of the threshold Θ. There exist an optimal threshold that minimizes the overall classification cost. These histogram-based methods are also termed Skin Probability Maps (SPM) [Brand and Mason 00].

- **Self Organizing Map (SOM)**

SOM is one of the most famous and widely used types of unsupervised artificial neural network. It has mainly been used to find patterns in and classify high dimensional data, although it works equally as well with low dimensional data. In [Brown et al. 01], they trained two SOMs (skin and skin/non-skin). Their results have shown that SOM skin detectors performance doesn't change significantly with different colour spaces. It has also shown better results than Gaussian Mixture Models. According to the authors, the SOM method needs less resources than GMMs and histogram-based models using a specialized hardware.

- **Nonparametric methods summary**

Two obvious advantages of non-parametric methods are that they are fast in training and usage and also they are independent to the shape of the distribution and therefore

to the colour space. On the other hand, they require much storage space, are unable to generalize or interpolate the training data, are not adaptive from one dataset to another and also they need a representative dataset to capture accurately the distribution.

2.2.3.3 Parametric skin distribution modelling

The need for more compact models together with the ability of interpolating the training data motivated using parametric techniques such as Single Gaussian and Mixture Gaussian Models.

- **Single Gaussian**

An elliptical Gaussian pdf for skin colour distribution can be modelled as:

$$P(c|skin) = (2\Pi)^{-d/2}|Cs|^{-1/2}exp\left\{\frac{-1}{2}(c-Ms)^T Cs^{-1}(c-Ms)\right\} \tag{2.6}$$

Where Ms and Cs are the mean vector and covariance matrix of the skin class and d is the dimension of the feature vector. We can use the $P(c|skin)$ probability to measure how "skin-like" the c colour is [Vezhnevets et al. 03], or alternatively, the Mahalanobis distance can be used as well assuming a uniform distribution of non-skin class. So the ML rule in equation 2.5 can be reduced to the following equation:

$$(c-Ms)^T Cs^{-1}(c-Ms) <= \Theta \tag{2.7}$$

Where Θ is a threshold and the left hand side is the squared Mahalanobis distance.

- **Mixture of Gaussians**

Generalizing the single Gaussian, a more advanced model is a mixture of Gaussians. The pdf in this case is given by:

$$P(c|skin) = \sum_{i=1}^{k} \pi_i P_i(c|skin) \tag{2.8}$$

Where K is the number of mixture components, π_i is the contribution of the i^{th} component, obeying the normalization constraint $\sum_{i=1}^{k} \pi_i = 1$, and $P_i(c/skin)$ are the Gaussians pdfs, each with its own corresponding mean and covariance matrix. The Expectation Maximization (EM) algorithm is well-known technique to train the model given the number of components k [Jones and Rehg 02, Terrillon 00]. By comparing the probability $P(c/skin)$ to a threshold, we can decide the class of the pixel. Note that the choice of k is very important so that the model explains the training data without over-fitting.

- **Multiple Gaussian Clusters**

The skin colour cluster can be approximated with k 3D Gaussians using a variant of the k-means clustering algorithm [Phung et al. 02]. Then the Mahalanobis distance is measured from the colour vector c to the clusters' centres and classified as skin if the distance to the closest model cluster is below a threshold.

- **Parametric methods summary**

Three important advantages of parametric methods are (1) their speed is acceptably fast (except for GMMs with many components), (2) they need little storage space, and (3) they are able to interpolate and generalize incomplete training data. On the other hand their performance depends on the skin distribution shape and therefore the colour space. The parameters can change from one image to another if environmental conditions change, and it is hard to determine whether to use a single or mixture Gaussian Model and how many components to use in the mixture Gaussian. In addition, to capture accurately the distribution, we need a large training dataset which in some cases might be very expensive to collect.

2.3 Feature extraction

The type of features extracted for classification of signs largely depends on exactly what type of signs the system aims to classify. For example, for understanding hand signing, usually researchers are interested in hand shape, motion,

position, orientation, facial expressions. While for finger spelling systems such as in [Wu and Huang 00, Birk et al 97, Handouyahia et al 99], and [Gupta and Ma 01, Deng and Tsui 02] the features mainly describe the hand configuration and orientation and the field of view (FOV) for the camera is mainly on the hand alone with uniform background. However in more general signing, the FOV has to be on the whole upper body to capture the movement of the hands and occlusions.

A common feature for the hand position is the "centre-of-gravity" of the hand blob which can be measured in raw image coordinates [Starner et al. 98], relative to the first frame [Cui and Weng 00], or relative to other body parts [Assan and Grobel 97, Bauer and Kraiss 02, Tanibata et al. 02].

Approaches that use multiple cameras to obtain the 3D position of hands and other body parts can provide better accuracy but with high computational cost [Vogler and Metaxas 97, Matsuo et al. 97]. A stereo camera was used to find the 3D position of both hands in a body-centered coordinate frame in [Matsuo et al. 97], while in [Vogler and Metaxas 97] the orientation parameters and the 3D wrist position coordinates were extracted relative to the signer's spine. In general, these approaches might achieve high accuracy in terms of tracking and resolving occlusions but they still suffer from problems such as scalability to large vocabulary, signer independence, segmenting sign segments from continous signs and not including facial features

With regard to motion features, Yang et al [Yang et al. 02] have used hand trajectories, while in [Min et al. 97] they used the chain code of hand positions to train an HMM. Optical flow was used in [Chen et al. 03].

Hand appearance features can be obtained from segmented hand images, hand blobs (silhouettes), or hand contours. Usually, the segmented images are normalized for illumination change, size, scale, rotation and translation and techniques like Principal Component Analysis (PCA) are used to reduce the dimensionality before further processing [Birk et al 97, Imagawa 00].

In [Kennaway 03, Cui and Weng 99], they used geometric moments of hand blobs. In [Bauer and Kraiss 02, Assan and Grobel 97], sizes, distances and angles between

coloured fingers were calculated, while Fourier descriptors (FD) were calculated on hand contours in [Huang and Huang 98, Chen et al. 03].

In [Al-Jarrah and Halawani 01], they used the distance between the finger tips and hand centroid. In [Huang and Jeng 01] hand contours were represented by Active Shape Models. PCA was used in training hand contours in [Bowden and Sarhadi 02]. Generally speaking, contour-based representations have the advantage of using invariant features to translation, scaling, and 2D rotation. However, different hand shapes with similar contours can be very hard to distinguish as their contour-based representations (features) become very similar and thus increasing the ambiguity in classification.

2.4 Classification methods

Two main approaches are used to classify signs. One approach uses a single classification stage using the different features collected, while the other approach tries to classify sign components (hand shape, motion, position, ...etc) and then integrate their classification results into a final sign-level classification stage. SLR systems are divided into two groups, one group tries to recognize isolated signs, while the other recognizes continuous signs. Neural networks are one of the famous methods for classification, especially for hand shapes [Waldron and Kim 95, Vamplew and Adams 98, Handouyahia et al 99]. Hand position, movement type and orientation have been classified by NNs in [Waldron and Kim 95, Vamplew and Adams 98], while whole signs were classified in [Huang and Huang 98].

Time-Delay NNs that can handle temporal processing but that require standard temporal length were used by [Yang et al. 02] to classify signs from hand trajectory. A fixed temporal length is not required by Recurrent NNs and were used by to classify sign words [Murakami and Taguchi 91].

Due to their popularity in speech recognition to process time series data, HMMs have been used in gesture recognition as well. HMMs have the ability to segment continuous speech into individual trained words or phonemes which are chained to-

gether in a tree-like structure. This idea was also used by researchers for recognition of continuous signs [Wang et al. 02, Bauer and Kraiss 02, Liang and Ouhyoung 98, Gao et al 00]. Some researchers tried to segment the signs into sequential subunits, where the HMM states model the subunits for each sign [Bauer and Kraiss 01]. This idea is very similar to the speech phonetic acoustic models

In [Bauer and Kraiss 02], the recognition performance of 150 HMM subunits trained on 100 isolated signs reached 92.5 percent. In [Vogler 03, Wang et al. 02, Yuan et al. 02], they defined subunits from the view-point of linguistics instead of using unsupervised learning. PCA and Multiple Discriminant Analysis (MDA) were used by [Cui and Weng 00]. They classified 28 ASL signs by constructing a recursive partition tree and applied PCA and MDA at the node. In [Deng and Tsui 02], they first applied PCA, and then using Gaussian distributions they clustered the data by coarse classification before applying MDA locally on each cluster. This gave them better results for a larger vocabulary instead of using MDA on the whole dataset. Signs were classified based on only the beginning and ending hand shape in [Imagawa 00, Deng and Tsui 02]. In [Wu and Huang 00], they used MDA to classify 14 hand shapes by training a classifier on a large unlabeled dataset and small labeled dataset modelled by the same mixture density.

Other methods have been used for classification of signs and hand shapes such as Template matching [Gupta and Ma 01], nearest-neighbour matching [Kramer and Leifer 87], and decision trees [Hernandez et al. 04]. In terms of classifying sign components and then integrating their classification results, a common approach [Hernandez et al. 04, Imagawa 00, Vamplew and Adams 98] is to hand-code lexicons, that define a sign from these components, and then classification is performed by comparing the recognized components with the ideal lexicon classes.

In [Vamplew and Adams 98], they employed a nearest-neighbour algorithm to do the matching with a heuristic distance measure. HMMs were used to recognize the sign components in [Liang and Ouhyoung 98], then Dynamic Programming was used to classify the sign. NNs were used in [Gao et al 00] to classify sign components then these labels were used as observations for HMM states that were then used to classify

the sign word.

In [Tanibata et al. 02], they modelled each hand with one HMM channel, then combined the results by multiplying the probabilities for isolated word recognition. NNs in [Waldron and Kim 95] were used to combine component-level results. The advantage of the component-level approach is that fewer classes have to be classified at the component level (which means simpler classifiers with fewer parameters). Then combining the results of these components can be used to classify a large vocabulary. This approach scales well [Hernandez et al. 04].

2.5 Vision-based imaging restrictions

In a survey [Ong and Ranganath 05] of the commonly used imaging restrictions and constraints in vision-based SLR systems, it has been shown that most systems use at least two of the following restrictions:

- Long-sleeved clothing [Assan and Grobel 97, Bauer and Kraiss 02],
  [Huang and Huang 98, Starner et al. 98, Tanibata et al. 02],
  [Imagawa 00, Yang et al. 02].
- Coloured gloves [Assan and Grobel 97, Bauer and Kraiss 02],
  [Matsuo et al. 97, Holden and Owens 00].
- Uniform background [Assan and Grobel 97, Bauer and Kraiss 02],
  [Holden and Owens 00, Yang et al. 02],
  [Huang and Huang 98, Matsuo et al. 97].
- Complex but stationary background
  [Cui and Weng 00, Starner et al. 98, Tanibata et al. 02, Imagawa 00].
- Head is required to have less motion than the hands [Cui and Weng 00],
  [Huang and Huang 98, Starner et al. 98, Tanibata et al. 02, Imagawa 00].
- Hands are moving continuously [Cui and Weng 00].

- Specific initial hand location or fixed body location or pose
  [Starner et al. 98, Tanibata et al. 02].
- Face and/or left hand is excluded from FOV
  [Cui and Weng 00, Huang and Huang 98, Holden and Owens 00].
- Unnatural signs to avoid overlapping with face or occlusion with other hand
  [Cui and Weng 00, Huang and Huang 98].

The message that we get from these restrictions is that SLR (especially vision-based) is still in it's early age of investigation and needs a lot of effort and time to reach a mature level of research. Systems need to overcome these restrictions and scale well to handle large vocabularies with single and two-hand signs, be independent of the signers, and work in a natural environment not just inside academic laboratories. In our proposed work, our system can be characterized by the following:

- No restrictions on long-sleeved clothing for tracking, but for recognition we do require it to extract the hand shape features from the palm area.
- No colour gloves.
- Background can be non-uniform as far as initially there is no big skin-like areas.
- Initially we require that no other skin-like colours exist other than the signer's face and hands.
- Background can be non-stationary during tracking (as we operate inside search windows instead of whole frame).
- No restrictions on head and hands movements.
- No specific hand or head positions.
- We detect occlusions but without segmenting occluded objects.

2.6 Summary

In this chapter we presented a review of SLR systems from various research groups. We reviewed the common methods used in the three main stages for sign classification: Locating and tracking the hands, feature extraction and classification methods. Hands detection and tracking is a very important task to enable us to extract the relevant features like hand shape, position, orientation, motion type. Also, non-manual features are considered a necessary channel for useful features although many systems ignore them and focus only on the hands. Two major divisions exist in terms of hand segmentation: colour-gloves and skin segmentation. Also, two major schemes for sign classification exist: either classifying sign components first then integrating their results, or classifying directly the sign using all the existing features. In general, there is a trade-off between the restrictions and constraints on the imaging environment and the vocabulary size. As we restrict more the environmental conditions, we can handle more signs and vice-versa. In general, the main technical challenges can be summarized as follows: handling large vocabulary (scalability), signer independence, incorporating all relevant features such as facial expressions and spatial relationships between them, using the context and grammar in recognition, segmenting sign words from continuous signs, flexible imaging restrictions without decreasing the total accuracy.

Chapter 3

Skin Segmentation

3.1 Introduction

Skin segmentation has attracted a lot of research interest in recent years. It aims to detect human skin regions in an image. For example skin detection has been used in applications such as speaker recognition and speaker location [Wang and Brandstein 99], the detection of pornographic material [Forsyth and Fleck 96] and video data indexing [Hsu et al. 98]. It is also commonly used in applications for hand gesture analysis [Zhu et al. 00], face detection [Hsu et al. 02, Yang and Waibel 96], and objectionable image filtering [Jones and Rehg 02]. In these applications, the skin regions detection helps to reduce the search space of skin objects such as faces or hands. Also in face and hands tracking algorithms it can give an initial estimate or be used as an important clue to the tracking algorithm [Zarit et al. 99]. The skin colour as a feature is very attractive as it allows fast processing, and it is highly robust to the pose of the object. In addition, the characteristic colour of the skin (a particular tone that does not vary much with different races of the same colour) makes it easy to be recognized by human eyes [Vezhnevets et al. 03]. Also, the concentration of the skin colour in a small region in the RGB colour space can help us to develop a corresponding colour model. On the other hand, the main challenge that faces skin detection algorithms is to make them robust to the large variations in its appearance. For example, the skin appearance can change in shape and colour and is often af-

fected by occlusions and changes in position, intensity, colour of light sources. Other objects may cast shadows on skin regions and other image noise regions can appear as a skin-like colour. Finally other objects may have skin-like colours e.g. clothes,...etc, which makes it difficult task to discriminate skin from non-skin regions.

Recently, a new trend in research into skin colour models, called adaptive models, has attracted increasing attention. Technically, it emphasizes how to model the skin colour under the condition of varying illumination. In [Yang et al. 98, Mckenna et al. 99] and [Wu et al 00], they firstly formulated the skin colour distribution as a GMM. Then, parameters, such as mean and covariance, of the GMM were dynamically updated over time by collaborating prediction from the previous tracked skin objects with the Maximum Likelihood (ML) algorithm [Yang et al. 98, Mckenna et al. 99] or the Discriminant Expectation- Maximization [Wu et al 00].

In [Soriano et al. 03], Soriano et al. applied adaptive histogram back-projection to update the skin colour model. Another interesting study was published by [Sigal et al. 04]. It made use of a dynamic Markov model whose parameters could be estimated by ML over time, to predict the evolution of the skin colour histogram under the assumption that objects are performing affine motion. The histogram adaptation also required current segmentation results.

3.2 Proposed skin colour model

3.2.1 Introduction

We attempt to develop a skin colour model for SL recognition. If we look at signing videos, it is quite easy to observe that one signing video generally only contain very few signers and the signers' skin colours keep consistent across the frames. This motivates us to train a skin model for every signing video. Thus, it is unnecessary to collect a lot of training samples. That is to say, the skin colour model for SL recognition might be transferred to a small sample size classification problem. We propose to use a Support Vector Machine (SVM) to learn the skin colour because of its outstanding performance in dealing with classification with small sample size

[Vapnik 98]. To overcome the problem of imbalance in training data, Active Learning is adopted to select the most informative training subset for the SVM, which leads to fast convergence and better performance. Finally, region information is incorporated to further reduce the noise and illumination variation. Our skin model is learned for each new signing video so that it is adaptive to the different human skin types and lighting conditions. In chapter 4 after we explain our proposed tracking algorithm, we will show that within one video, the proposed skin model can also be updated based on the newly predicted training data from the tracking, which works well under time-varying illumination. Compared with existing approaches, our model automatically constructs the training set from the first several frames which saves human labour.

Fig.3.1 shows the basic architecture of the proposed skin colour model. The framework consists of two stages: training stage and segmentation stage. In the former stage, first, for a signing video, a generic skin colour model is applied to the first several frames, which obtains the training samples. Then, a binary classifier based on SVM Active Learning [Wang 03] is trained using the obtained training set. Note that the generic skin model has to classify correctly a reasonable amount of skin pixels in order for the SVM to work well. This in general is not hard as the generic skin model is designed using a huge number of skin training samples, thus in most cases it outputs the desired number of skin samples. In the segmentation stage, the SVM classifier is incorporated with the region information to produce the final skin colour pixels.

3.2.2 The generic skin model

In our work, the target of adopting the generic skin model is to collect some training data for SVM classifier. The generic skin model is implemented by defining a fixed skin colour range in one colour-space. Technically, any generic skin model could be used in our work. Because of simplicity, here, we adopted the generic skin model presented in [Peer 03], which defined skin pixels in RGB space as follows:

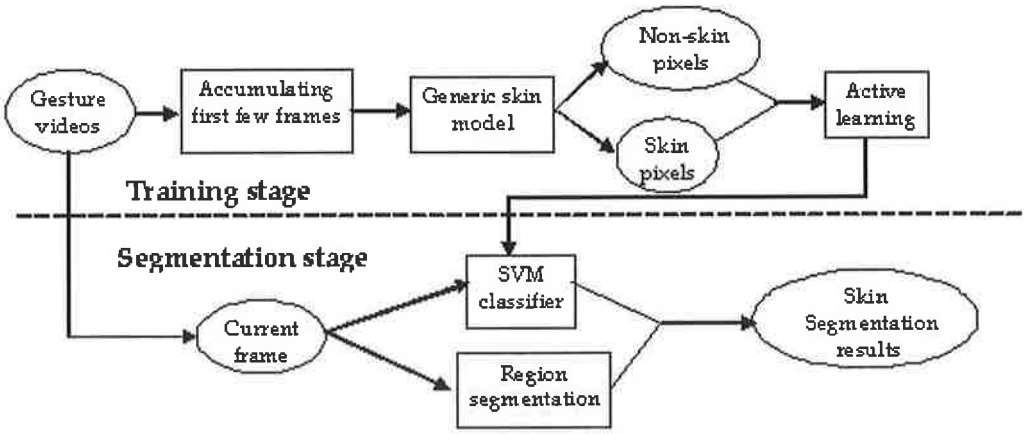


Figure 3.1: The basic architecture of the proposed skin colour model

Algorithm 2 (i) under uniform daylight:
(R>95) & (G>40) & (B>20) & (max{R,G,B} - min{R,G,B}>15) & (|R-G|>15) & (R>G) & (R>B) & (G>B)
(ii) under flashlight:
(R>220) & (G>210) & (B>170) & (|R-G|≤15) & (R>B) & (G>B)

3.2.3 SVM active learning

SVMs are learning machines that can perform binary classification and regression estimation tasks. Invented by Vapnik [Vapnik 95], they have strong theoretical foundations and excellent empirical successes. They have been applied to tasks such as handwritten digit recognition [Vapnik 98], object recognition [Papageorgiou et al. 98], and text classification [Joachims 98]. The high performance of SVMs is due to their ability to make a good generalization from a limited training dataset. In a binary classification setting, given linear separable training set $\{x_1, x_2, ..., x_n\}$with their labels $\{y_1, y_2, ..., y_n\}, y_i \in \{-1, 1\}$, the SVM is trained and the optimal hyperplane is yielded, which separates the training data by a maximal margin. Specifically, the optimal hyperplane might be found by solving an optimization problem:

$$minimize : \emptyset(\mathrm{w}) = \frac{1}{2} \parallel w \parallel^2 \tag{3.1}$$

$$subject\ to: y_i(w \bullet x_i + b) \geq 1$$

Here:

$$w = \sum \alpha_i y_i x_i \tag{3.2}$$

The optimal hyperplane divides the data points into two groups. Points lying on one side are labelled -1, and the other points are marked 1. When a new example is inputted for classification, a label (1 or -1) is issued by its position with respect to the hyperplane that is:

$$f(x) = sign(\sum_i \alpha_i y_i (x_i \bullet x) + b) \tag{3.3}$$

For the case of data that is not linearly separable, the SVM first projects the original data to a higher dimensional space by a Mercer kernel function K such as Gaussian RBF kernels and polynomial kernels, and then linearly separates them. The corresponding nonlinear decision boundary is:

$$f(x) = sign(\sum_i \alpha_i y_i K(x_i \bullet x) + b) \tag{3.4}$$

The SVM can be easily applied as a skin colour model. Given a gesture video, the generic skin model is performed on the first several frames so that the training set containing skin and non-skin data could be obtained. Afterwards, the SVM classifier is constructed using the training set from the previous frames to segment the future frames one by one. In practice, one problem is the imbalance in the training data: i.e. the number of the negative examples (non-skin pixels) is far larger than the number of the positive examples (skin pixels). Fig.3.2 displays the fact. The left picture is the original image, and the right one is the segmented results. In the right image, the points labelled green are the skin pixels and other points are considered as the non-skin pixels. The imbalance of training examples may make the learning less reliable. Moreover, it results in a long learning time.

A feasible way to reduce this limitation is to use Active Learning. Active Learning is named in contrast to the traditional passive learning. Most machine learning approaches belong to passive learning because they are usually based on the entire training set or randomly selected data [Tong and Chang 01]. In contrast, Active Learning tries to find the most informative data to train the classifier. Its goal is to achieve better performance and faster convergence with less training examples. Lately, Active Learning has been successfully introduced to document classification [Schohn and Cohn 00], image retrieval [Tong and Chang 01], and text classification [Tong and Koller 01]. Whereas, to our best knowledge, very little work has been done in the skin segmentation field.

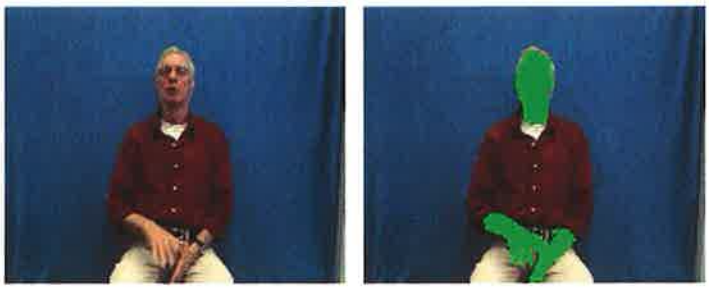


Figure 3.2: The imbalance of the training examples

The key idea of Active Learning is to extract the most informative samples from all available training data. Tong et al. explain Active Learning from the viewpoint of version space theory in [Tong and Chang 01, Tong and Koller 01]. The version space was defined as a set of all hyperplanes that could classify the given training data correctly. To a new-labelled sample, all hyperplanes in the version space were used to classify the new data again. Those hyperplanes that made the wrong classification were removed from the version space so that the size of version space was reduced. The informative samples are able to reduce the size of version space as much as possible. Based on this idea, Tong et al. found informative samples are always near the hyperplane. In other words, the importance of one instance point depends on its distance from the hyperplane. As for our application, we attempt to find the small but informative subset of negative examples with a similar size to the training set of positive examples. The instances closer to the SVM hyperplane generally

give a larger influence to the learning so that they are more informative than other instances. This motivates us to design a similarity-based sampling strategy to select more informative negative examples for our specific application.

Let F be the training set, F^+ and F^- are the positive (skin pixel) and the negative (non-skin pixel) training set, respectively. $F = F^+ \cup F^- \& F^+ \cap F^- = \emptyset$.

We hope to get the small subset of negative examples F^-_{active} by Active Learning. Here, $F^-_{active} \subset F^-$. First, a region segmentation scheme JSEG [Deng and Manjunath 01] is employed to segment F^- into different regions, $R^{F-}_1, R^{F-}_2, ..., R^{F-}_N, \sum_{i=1}^{N} R^{F-}_i = F^-$ based on the colour-texture feature. Second, the similarities between each $R^{F-}_i$and $F^+$are described by the colour histogram based distances. More specifically,

$$D(R^{F-}_i, F^+) = \| H(R^{F-}_i) - H(F^+) \| \tag{3.5}$$

Where $H(\bullet)$ is the colour histogram vector. Note that this distance measures how close is every region colour to the positive samples, and this is measured across all the segmented regions individually. A smaller distance between R^{F-}_i and F^+ indicates that the region colour is similar to skin colour. In the feature space, the skin pixels generally form a cluster. If one negative instance is closer to F^+ , it is closer to the SVM hyperplane as well. Therefore, it is more likely to be an informative example. Finally, we sample the negative examples according to a principle called "most similar-highest priority". To be specific, more negative instances are extracted from the R^{F-}_i with smaller distance to F^+, but less negative instances are selected from the R^{F-}_j with the larger distance to F^+. The sampled examples construct the F^-_{active} , and its size is approximately equal to the size of F^+. The advantage of our similarity-based sampling strategy is that not only can it get more informative examples, but also the obtained set F^-_{active} covers all kinds of negative examples from different regions that were considered by the generic skin model to be negative examples. In some cases, the generic skin model detects some false positives, but this should not have a major effect overall as the SVM can handle some noise in the

data. In summary, the SVM Active Learning for skin segmentation is fulfilled by the following steps:

1. Apply the generic skin model to the first several frames to obtain F^+ and F^-;
2. Segment F^- into different regions $R_1^{F-}, R_2^{F-}, ..., R_N^{F-}$, and compute the distances between each R_i^{F-} and F^+ in the colour feature space;
3. Construct the F^-_{active} from F^- in accordance with similarity based sampling scheme;
4. Train the binary SVM classifier using F^+ and F^-_{active} ;
5. Classify every current frame into skin and non-skin pixels by the trained SVM

In general, any similarity based sampling scheme can be employed. In our case we chosed a simple yet effective way to do the sampling. Let m be the minimum colour histogram distance such that:

$$m = arg\ min\ d_i,\ where\ d_i = D(R_i^{F-}, F^+) \tag{3.6}$$

we pick from every region R_i^{F-} a ratio of pixels proportional to its distance to F^+, we define this ratio r as:

$$r_i = \frac{m}{d_i} \tag{3.7}$$

In a comparison of our sampling scheme to Adaboost sampling principle, we can find some differences such as: Adaboost selects training samples using the weights updated every iteration based on the classification error, while in our case, we are working on the region level where regions can be weighted by their distance to the positive samples then samples from regions are selected based on this distance. Adaboost iterates on this process for a fixed number of iterations, while we do this one time only. Adaboost linearly combines weak classifiers each cycle, while we train SVM once directly.

3.2.4 Combining SVM active learning and region information

Although the performance of SVM Active Learning is outstanding, it cannot produce perfect skin segmentation results due to noise and illumination variation. However, region information is considerably more robust to noise and illumination variation (Fig. 3.3 shows a sample of region segmentation result on one frame). Hence, in order to solve this problem, we incorporate region information to further refine the segmentation result. First, the JSEG algorithm [Deng and Manjunath 01] is adopted to parse the frame into regions. Then, if the majority of pixels of one region R_i belong to skin pixels, the whole region is declared as the skin area. To be exact, one region satisfying:

$$\frac{NS(R_i)}{NT(R_i)} > \eta \tag{3.8}$$

is decided as skin area. Here, $NS(R_i)$ denotes the number of the skin pixels in the region R_i, $NT(R_i)$ refers to the number of pixels in R_i, and η is an empirically defined constant. We should note here that η is critical for the decision whether this region is a skin or not. So, to avoid this limitation we don't combine the region information in the tracking part (chapter 4) and substitute it's effect by fusing more cues for segmentation.

Figure 3.3: Region segmentation sample. the left image is the original frame. The right image is the corresponding frame after being region segmented

3.3 Experimental results

We tested the proposed skin colour model with 8 video sequences from the European Cultural Heritage Online (ECHO) database [ECHO] and other signing videos captured by ourselves. They were captured with different signers and under different lighting conditions. Almost every video sequence from ECHO is over 15 minutes long. To quantitatively evaluate our work, we randomly picked 240 frames from those test sequences and manually segmented the skin pixels to construct the ground truth (Appendix A). An SVM classifier was trained using the first accumulated three frames. As in [Phung et al. 05], three metrics, correct detection rate (CDR), false detection rate (FDR), and overall classification rate (CR) were employed to measure the performance of the techniques. They are described as follows:

CDR: percent of correctly classified skin pixels;

FDR: percent of wrongly classified non-skin pixels;

CR: $\frac{N_s}{max(N_s^A, N_s^G)} \star 100\%$, where N_s is the number of skin pixels detected both by the algorithm and the ground truth, N_s^A is the number of skin pixels detected by the algorithm, and N_s^G is the number of skin pixels detected by the ground truth.

The CDR metric doesn't capture over-segmentation, as it only measures the true positives. So we can get high CDR with the cost of high FDR, while CR is sensitive to over-segmentation and under-segmentation and can give a good true measure for the overall accuracy. Three experiments were constructed to evaluate our skin colour model. First we test the performance of Active Learning. We then examine the effect of combining region information. Next, comparisons with some traditional skin segmentation techniques are reported.

3.3.1 Testing the performance of active learning

To demonstrate the performance of SVM Active Learning, we compared the SVM classifiers with and without Active Learning using our test data. Fig.3.4 shows one set of sample results. The first, second and third image display the original frame, the SVM without active learning, and SVM with active learning, respectively. It can be seen in this example that when more training samples were taken from the trousers region (as its colour is near skin-colour), the results become much more better. Table 3.1 lists the statistical results including the precision and training time. As can be seen from the experimental results, the SVM Active Learning is superior in both accuracy and computational complexity. It can enhance the overall accuracy almost by 6%, and decrease average training time by 114 seconds. The reduction of the training time is due to the selection of a small informative subset from the negative samples instead of using all the possible samples, thus the total number of training samples decreases which reduces the training time.

original without AL with AL

Figure 3.4: Experimental samples with and without Active Learning (AL)

	CDR (%)	FDR (%)	CR (%)	Training time (s)
No active learning	85.12	2.43	61.97	121.14
With active learning	82.83	1.39	67.60	7.33

Table 3.1: The Statistical Precision and Training Time Comparisons with and without Active Learning

3.3.2 Evaluation of combining region information

This experiment is to evaluate the segmentation results with and without region information. Fig.3.5 displays some sample results, and Table 3.2 lists the statistical

precision comparisons. In Fig.3.5, the first column shows the original frames, the second column shows the segmentation results without region information, and the third column shows the results with region information. Clearly, the algorithm with region information is better, which can reduce the noise and refine the segmentation results. Incorporation of region information enhanced the overall accuracy by 9%.

original without region with region

Figure 3.5: Comparison results with and without region information

	CDR (%)	FDR (%)	CR (%)
The proposed method without region	82.83	1.39	67.60
The proposed method with region	86.34	0.96	76.77

Table 3.2: The Statistical Precision Comparisons with and without Region Information

3.3.3 Comparisons with traditional skin segmentation techniques

To demonstrate the effectiveness of the proposed work, we compared the proposed model with two existing skin segmentation algorithms, the generic skin model [Peer 03, Chai and Ngan 99] and a Gaussian model [Phung et al. 05, Zhu et al. 04]. The Gaussian models [Phung et al. 05] can be described as follows. They employed the Bayesian decision rule:

$$\frac{P(c|skin)}{P(c|nonskin)} \geq \xi \tag{3.9}$$

to classify the skin and non-skin pixels. Here, $P(c|skin)$ and $P(c|nonskin)$ refer to the probability density function (pdf) of skin and non-skin colour, respectively. ξ is

a threshold. The colour pdf could be modelled as a single Gaussian (Eq. 2.6), or Gaussian mixture (Eq. 2.8).

In [Phung et al. 05], Phung proposed two strategies: modelling only skin pixels as Gaussian (called one-Gaussian in our experiments) and modelling both skin and non-skin pixels as Gaussian (called two-Gaussian in our experiments). In this experiment, we implemented these two strategies. Notice we used Gaussian mixture to model pdfs. To estimate the gaussians, we used the output of the generic skin model (first 3 frames). Fig.3.6 shows some results. The segmentation results by generic model, one-Gaussian, two-Gaussian, and the proposed approach are displayed in the first, second, third, and fourth column, respectively. Table 3.3 lists the statistical accuracy comparisons. As we can see from the comparison results, the proposed model has the highest overall accuracy with the second lowest false detection rate. Although the two-Gaussian model has the best correct detection rate, its false detection rate is the worst. This result is not surprising because as the Gaussian try to achieve high correct detection rate, it's parameters become more flexible to catch more false positives thus increasing the false detection rate. The real challenge is to achieve high CDR with low FDR.

Figure 3.6: Sample results of generic skin model, one-Gaussian model, two-Gaussian model, and the proposed model

	CDR (%)	FDR (%)	CR (%)
The generic skin model	71.51	0.79	65.10
One-Gaussian model	72.74	1.04	66.85
Two-Gaussian model	90.88	4.41	57.06
The proposed model	86.34	0.96	76.77

Table 3.3: Statistical Accuracy Comparisons of Existing Models and the Proposed Model

3.4 Summary

In this chapter, a completely adaptive skin segmentation algorithm for gesture recognition system has been proposed. A binary SVM classifier was trained using the training data automatically collected from the first several video frames. More importantly, Active Learning and region segmentation were combined to further improve the performance. One important advantage of the proposed work is that it is easy to implement and does not need human labour to construct the training set. In addition, it may be efficiently incorporated in a gesture recognition system or other human body related applications with the minor revision. The evaluating experiments on real-world SL videos demonstrated that the proposed work is promising.

Chapter 4

Hand and Face Tracking

4.1 Introduction and literature review

Skin object tracking is an essential component of an SL recognition system. It may provide SL recognition with useful spatio-temporal features such as hand location and trajectory. Many efforts have been devoted into research of skin and hand tracking. Generally speaking, there are two streams of scheme proposed to solve the problem: device-based and vision-based. The principal idea of a device-based tracker is to capture hand motion by asking users to wear gloves or markers [Shamaie and Sutherland 05, Gao et al 00], by using an infrared camera [Sato et al. 00], or by using a laser rangefinder as mentioned in [Strickon and Paradiso 98]. Obviously, some specific devices make the task of hand tracking simple, however, it is expensive, even impossible in some applications to use the device examples. Definitely, vision-based hand tracking draws more attention.

In [Yang et al. 02], Yang et al. implemented hand tracking based on region matching using affine transformations. The regions were yielded by a multi-scale image segmentation algorithm. Their hand tracking results finally were incorporated into an American SL recognition system. In [Chen et al. 03] and [Huang and Jeng 01], the authors combined multiple features like motion, edge, and skin colour to detect and further locate the hand. However, one shortcoming is that their approach worked well only for a single hand. It is not straightforward to extend their algorithm to

SL recognition because most signs are two-handed and occlusions among hands and face happen very often.

In [Imagawa and Igi 98, Martin et al. 98, McAllister et al. 02], a related work has tried to use a Kalman Filter (KF) to track the hands. Firstly, according to information from the previous frames, a linear KF was built to estimate the motion velocity of the hand. Then, the location of hands in the next frame could be predicted using the estimated velocity and position in last frame. These trackers worked very fast. Unfortunately, they cannot perform accurate hand segmentation. In order to reduce the limitation of KF, the CONDENSATION algorithm was proposed by Isard et al. [Isard and Blake 98], which used conditional density propagation to track curves in clutter. The propagation was based on the fusion of learned dynamical models and visual observations. This algorithm was successfully applied to track hand contours. In [Black and Jepson 98] Black et al. revised the CONDENSATION algorithm to recognize gesture and expressions. In [Mammen et al 01], they extended the CONDENSATION to track both hands simultaneously that can deal with occlusions. In [Rehg and Kanade 94, Stenger et al. 01, Lu et al. 03] another interesting research stream called model-based hand tracking was reported. These algorithms required some prior knowledge like 2D or 3D hand shape. In DigitEyes system [Rehg and Kanade 94], researchers tracked hands by a 3D hand model with 27 degrees of freedom. They assumed the hand as a collection of 16 rigid bodies and modelled them by kinematic chains. In [Stenger et al. 01], Stenger et al. built a 3D hand model by truncated quadrics. The Unscented KF (non-linear filter) was applied to estimate the hand pose and then track hands. Lately, Lu et al. [Lu et al. 03] introduced a deformable model for hand tracking. It defined a geometric model to represent the shape and structure of hand, which is based on the measurement of an average male. The model-based approaches are effective under the assumption of known shape. However, in SL, hand shape varies quickly, which might result in poor performance in tracking.

Most of the above works can achieve hand tracking well. However, very few of them can perform hand segmentation with the accuracy required to provide SL recognition

with shape features. In addition, the presence of occlusions makes it challenging to track the face and hands. For this reason, some systems avoid signs that include occlusions, perform unnatural signs, or choose camera angles that doesn't capture occlusions [Ong and Ranganath 05]. However, occlusions between face and hands or between the two hands occur frequently in many signs in the real world. Occlusion detection is necessary because it might help to reduce the search space in the recognition phase.

In our work, we aim to deal with the segmentation and tracking problems as one unit which simplifies the process of locating the skin objects, unlike other works that separate the two tasks of segmentation and tracking. We introduce a method for combining colour, motion and position information to segment skin objects. The tracking is based on the fusion of KF and blob matching from segmentation results. In the previous work of our group [Shamaie and Sutherland 05], they proposed a colour glove-based method to detect occlusion between the two hands using KF prediction. Here, we extend this by using skin detection techniques and handling occlusion between skin objects (face and two hands) in a robust way to keep track of the status of the occluded parts. In [Sherrah and Gong 00], the authors propose a very related solution for tracking the face and two hands. Their approach is based on Bayesian Belief Networks (BBNs) to fuse high-level contextual knowledge about the human body with sensor-level observations such as colour, motion and hand orientation. In the next sections we will explain in detail our proposed system for skin segmentation and tracking (SST).

4.2 Skin segmentation and tracking system overview

A block diagram for the system architecture is shown in Fig. 4.1. In general we track three objects: the face and two hands. Two main components form the proposed algorithm. The first component, skin segmentation is responsible for segmentation of skin objects by combining different useful information. The second component, object tracking, is responsible for matching the resulted skin blobs of the segmenta-

tion component to the previous frame blobs. Keeping track of the occlusion status of the three objects is done by the knowledge of the occlusion alarms between any pair of the objects in addition to the number of the new detected objects.

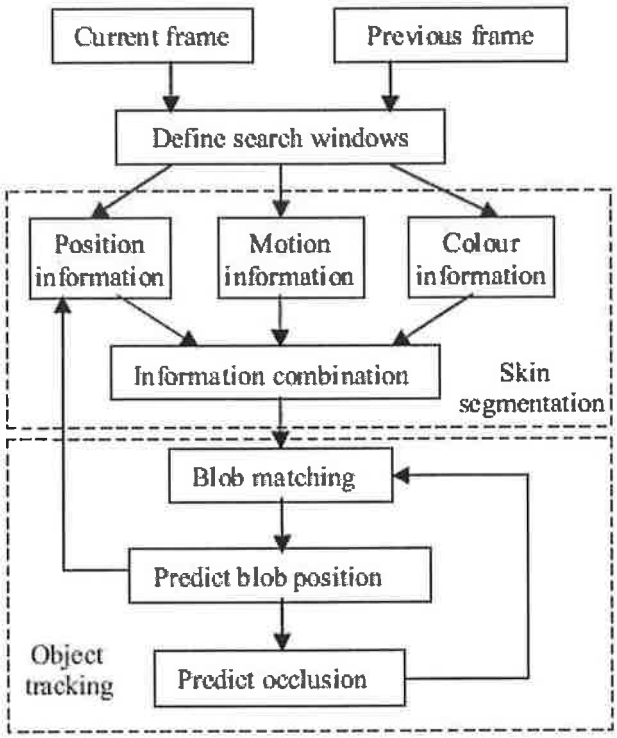


Figure 4.1: SST system architecture

4.2.1 Skin Segmentation

4.2.1.1 Colour information

We apply the proposed skin colour model as discussed in chapter 3 in small search windows around the predicted positions of the face and hand objects and return decision values from the SVM representing how likely the pixels are to be skin. As the training of the SVM classifier is based on the first few frames, it can miss some skin pixels. Therefore, we propose another colour distance metric to take advantage of the prior knowledge of the last segmented skin object. This prior knowledge colour metric is denoted as $dist(C_{skin}, X_{ij})$, where C_{skin} is the median RGB colour vector of the previously segmented skin object, X_{ij} is the current pixel RGB colour vector in

the search window in row i and column j, and *dist* is defined as the Euclidean distance between the two vectors. Finally, we normalize the values of the SVM classifier P_{svm}, and the prior knowledge colour metric P_{col}. Fig. 4.2 shows the search window in a sample frame surrounding the right hand (a), and after we apply the prior knowledge colour metric (b) we can see that high values (more bright pixels) represent the hand region while low values (more dark pixels) represent non-skin regions.

We would like to note here that we are using our proposed skin colour model without combining the region segmentation information as were discussed in chapter 3. This is because here we are using other useful features like the motion and position which helps a lot without the need of the region information. Using the region information can also take some processing time and can be considered as a single point of failure for the skin segmentation system because as discussed in chapter 3, the skin areas are detected finally by finding the regions with high overlapping skin pixels (classified by SVM), so if regions are not segmented accurately, the skin segmentation will not precisely represent the true skin objects. Thus depending on more than one feature can be more useful in this case.

(a) (b)

Figure 4.2: Demonstration of the colour feature. (a) search window surrounding the right hand, (b) the normalized values of the Euclidean distance after being subtracted from 1.

4.2.1.2 Motion information

Finding the movement information takes two steps. Firstly, motion detection, then in the next step, finding candidate foreground pixels. The first step examines the local grey-level changes between successive frames by frame differencing:

$$D_i(x, y) = |W_i(x, y) - W_{i-1}(x, y)| \tag{4.1}$$

where W_i is the i^{th} search window, x, y are pixel locations relative to the search window, and D_i is the absolute difference image. We then normalize D_i to convert it to probability values $D_i^{'} = \frac{D_i}{max(D_i)}$. The second step assigns a probability value $P_m(x, y)$ for each pixel in the search window to represent how likely this pixel belongs to a skin object. This is done by looking backward to the last segmented skin object binary image in the previous frame search window OBJ_{i-1} and applying the following model to the pixels in $D_i^{'}$:

$$P_m(x,y) = \begin{cases} 1 - D_i^{'}(x,y) & if\, OBJ_{i-1}(x,y) = 1 \\ D_i^{'}(x,y) & otherwise \end{cases} \quad (4.2)$$

In this way, small values (stationary pixels) in $D_i^{'}$ that were previously segmented as object pixels will be assigned high probability values when subtracted from 1 as they represent skin pixels that were not moved, and new background pixels (that were previously skin pixels) with high $D_i^{'}$ will be assigned small probability values. So simply, this model gives high probability values to candidate skin pixels and low values to candidate background values. Fig. 4.3 demonstrates the process of calculating the motion feature between frame i and frame i+1.

$frame_i$ $frame_{i+1}$ $D_i^{'}(x,y)$ $OBJ_i(x,y)$ $P_m(x,y)$

Figure 4.3: Demonstration of the motion feature for frame i+1

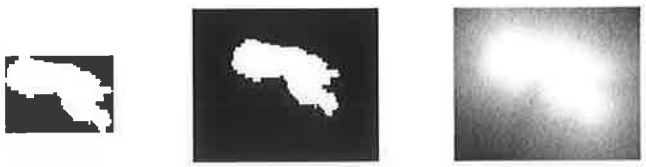
4.2.1.3 Position information

To capture the dynamics of the skin objects, we assume that the movement is sufficiently small between successive frames. Accordingly, a KF model can be used to describe the x and y coordinate of the centre of the skin objects with a state vector S_k that indicates the position and velocity. The model can be described as

[Chui and Chen 99]:

$$\begin{aligned} S_{k+1} &= A_k S_k + G_k \\ Z_k &= H S_k + V_k \end{aligned} \tag{4.3}$$

where A_k is a constant velocity model, G_k, V_k represents the state and measurement noise respectively, Z_k is the observation, and H is the noiseless connection between the observation and the state vector S. This model is used to keep track of the position of the skin objects and predict the new position in the next frame. Given that the search window surrounds the predicted centre, we translate a binary mask of the object from the previous frame to be centred on the new predicted centre. Then the distance transform (spatial distance between every non-object pixel and the nearest object pixel) is computed between all pixels in the search window and pixels of the mask. The inverse of these distance values assigns high values to pixels that are belonging to or near the mask and low values to far away pixels. The distance values are then converted to probabilities P_p by normalization. Fig. 4.4 demonstrates calculating the position feature inside the search window, we can see that high pixel values surrounds the predicted position of the skin object, while low values are assigned to far positions where the skin object is less likely to be located.



(a) (b) (c)

Figure 4.4: Demonstration of the position feature. (a) binary mask in previous frame, (b) binary mask centered on predicted position in search window, (c) normalized distance transform.

4.2.1.4 Information fusion

After collecting the colour, motion and position features, we combine them logically using an abstract fusion formula to obtain a binary decision image $F_i(x, y)$:

$$F_i(x,y) = \begin{cases} 1 & if\ (P_{col}(x,y) > \tau)\, OR\, ((P_{svm}(x,y) > \gamma)\, AND\, (P_m(x,y) > \nu) \\ & \quad AND\, (P_p(x,y) > \sigma)) \\ 0 & otherwise \end{cases} \tag{4.4}$$

where P_{col}, P_{svm} , P_m, and P_p is the decision probability values of the prior knowledge colour metric, skin colour model, motion, and position respectively, and τ, γ , ν and σ are thresholds where σ is determined adaptively by the following formula:

$$\sigma = \frac{size((P_m(x,y) > \nu)\, AND\, (P_p(x,y) \equiv 1))}{size(P_m(x,y) > \nu)} \tag{4.5}$$

The threshold σ determines the margin that we are searching into around the predicted object position. In Eq. 4.5 this is formulated by finding the overlapping between the predicted object position and the foreground pixels above certain threshold value. The other thresholds values are determined empirically.

4.2.2 Skin object tracking

The tracking component is responsible for matching the segmented skin blobs of the new frame to the previous frame skin blobs while keeping track of the occlusion status of the three objects. In general, tracking three objects (face and two hands) results in ten different case scenarios as follows:

1. Face and two hands all exist (3 skin objects).
2. Non-occluded Face while the two hands are occluded (2 skin objects).
3. One hand occluded with the face while the other hand is separate (2 cases, 2 skin objects).

4. Face and one hand non-occluded, while the other hand is hiding (2 cases, 2 skin objects).

5. Face and the two hands are all occluded together (1 skin object).

6. Face alone, and the two hands are hiding (1 skin object).

7. Face occluded with one hand, while the other hand is hiding (2 cases, 1 skin object).

In order to match skin objects between successive frames, we have to keep track of objects that might occlude in the next frame because this affects our final conclusion about what objects exist in the current frame and their occlusion status. In the next section we will explain the basic idea of how we detect occlusion between any of the skin objects. This step is critical and necessary to achieve correct blob matching between previous and current frame skin objects.

4.2.2.1 Occlusion detection

A rectangle is first formed around each of the face and two hands. Then, each rectangle is modelled by a KF filter. To be specific, we model each side of the rectangles by its position, velocity and acceleration as follows:

$$\begin{aligned} S_{j,k+1} &= S_{j,k} + hs'_{j,k} + \frac{1}{2}h^2 s''_{j,k} \\ s'_{j,k+1} &= s'_{j,k} + hs''_{j,k} \end{aligned} \tag{4.6}$$

Where S is the position, S' the velocity, S'' the acceleration, $h > 0$ is the sampling time, j is the rectangle side index, and k is the time. Combining Eq. 4.3 and Eq. 4.6, we can get:

$$\begin{bmatrix} S_{k+1} \\ S'_{k+1} \\ S''_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & h & \frac{1}{2}h^2 \\ 0 & 1 & h \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_k \\ S'_k \\ S''_k \end{bmatrix} + G_k \tag{4.7}$$

$$Z_k = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} S_k \\ S'_k \\ S''_k \end{bmatrix} + V_k$$

Note that we use $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ for matrix H as the position is the only observable feature for the rectangle sides. Applying Eq. 4.7 for every rectangle side can predict the position of the rectangles in the next frame. Accordingly, we check to see if there is any overlap between any of the bounding rectangles in the next frame. If there is an overlap, we raise an occlusion alarm corresponding to the fact that two bounding rectangles are about to overlap. If in the next frame, the number of detected skin objects is less than the number in current frame and an occlusion alarm was raised previously, we conclude that occlusion happened.

On the other hand, if the number of detected skin objects decreases and no occlusion alarms were raised, then this means that one or more skin objects are hiding. The same idea can also be applied between already occluded objects and a new non-occluded object as in the case where face and left hand are already occluded and then the right hand approaches them so that the all-3 objects become occluded. Fig. 4.5 demonstrates two samples of an occlusion between the two hands (of a cartoon character) and hand and face (of real video). At frame i the predicted positions of the bounding boxes in frame i+1 are not overlapping, so no alarms are raised. On the other hand, at frame i+1, the predicted positions are overlapping in frame i+2, so an alarm is raised indicating the possibility of occlusion to happen in the next frame. In frame i+2, we actually detect only 2 skin objects instead of 3 such as in frame i+1 and also an alarm is already raised, so we conclude that occlusion happened between the two skin objects.

(a) frame i (b) frame i+1 (c) frame i+2

Figure 4.5: Demonstration of detecting occlusions.

4.2.2.2 Tracking

As shown in Fig. 4.1, the tracking process takes place by first constructing search windows around each of the objects we are tracking. When two or more objects are occluded, they are treated as one object and one search window is constructed around their position.

Given that the search windows are constructed, we segment the skin objects as described in section 4.2.1. Next, connected regions are labelled after removing noisy small regions. Using the number of detected skin objects and the occlusion alarms as discussed in the previous section, we maintain a high-level understanding of the status of the current frame with respect to the occlusion status using a set of heuristic rules. For example, if we detected one object and an occlusion alarm between the face and left hand is raised, then we conclude that the face and left hand are occluded and the right hand is hiding. This technique can be extended to handle all 7 case scenarios. This technique proved to work well with the following assumptions:

1. The face cannot be hiding.

2. Minimum number of skin objects in any frame is 1 (face) and maximum 3 objects (face and two hands).

3. Initially, the system must begin by detecting 3 skin objects.

So, taking assumption number 2 into consideration, we handle 9 different cases of transitions between sequential frames as shown in Fig. 4.6. This approach is very similar to a Finite State Machine (FSM) except that we don't explicitly execute entry and exit actions in each state. We use this model because of its simplicity in representing the possible states of skin objects that might occur in any frame, and thus the occlusion status.

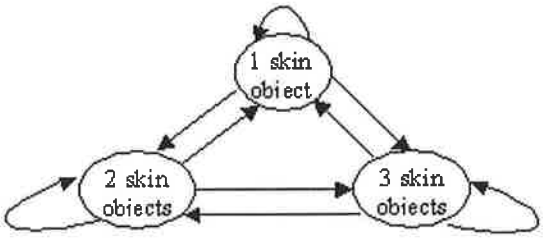


Figure 4.6: Skin objects state transitions between sequential frames

In order to decide the status of the current frame, i.e to know the identity of the current skin objects, we designed a simple algorithm to conclude what objects are present and what is the occlusion status between them using heuristic rules. Algorithm 3 shows the outline of this occlusion status detection process. Note that the following terms apply:

- hand-hand occlusion alarm: occlusion alarm between the two hands.
- L-hand-face occlusion alarm: occlusion alarm between the left hand and the face.
- R-hand-face occlusion alarm: occlusion alarm between the right hand and the face.

- L-hand search window: the segmentation output of the left hand search window.
- R-hand search window: the segmentation output of the right hand search window.

Algorithm 3 Occlusion status detection

```
if number of skin objects == 3 then
    objects are: face, left hand, right hand.
elseif number of skin objects == 2 then
    if hand-hand occlusion alarm is set on AND L-hand-face occlusion alarm
        is set off AND R-hand-face occlusion alarm is set off then
            objects are: face, 2 hands occluded.
    elseif L-hand-face occlusion alarm is set on AND R-hand-face occlusion
        alarm is set off AND hand-hand occlusion alarm is set off then
            objects are: right hand, left hand and face occluded.
    elseif R-hand-face occlusion alarm is set on AND L-hand-face occlusion
        alarm is set off AND hand-hand occlusion alarm is set off then
            objects are: left hand, right hand and face occluded.
    elseif L-hand search window is empty AND R-hand search window is not
        empty then
            objects are: face, right hand, left hand is hiding.
    elseif L-hand search window is not empty AND R-hand search window is
        empty then
            objects are: face, left hand, right hand is hiding.
elseif number of skin objects == 1
    if face-L-hand-R-hand occlusion alarm is set on then
        object is: face and left hand and right hand are all occluded.
    elseif L-hand-face occlusion alarm is set on then
        object is: face and left hand occluded, right hand is hiding.
    elseif R-hand-face occlusion alarm is set on then
        object is: face and right hand occluded, left hand is hiding.
    else
        object is: face, both hands are hiding.
```

We would like to note here that when one of the hands hides, we fix the location of the search window to the last position where the hand was visible. Our assumption is that the hand will reappear again probably in a location near to the place where it disappeared. The advantage of such a technique is its simplicity and speed as it consists of just some conditional statements. It performs very well in terms of accuracy as it covers all the possible cases that can appear in any given frame.

The final step in the tracking part is the blob matching. Given that we concluded

what objects are there in the segmented frame and their occlusion status, we perform the matching between the previous frame skin objects and the new frame skin blobs. The matching is done using the distance between the objects in sequential frames. Here we used the Euclidean distance between the centres of the objects to match the corresponding objects.

4.2.3 Skin colour model adaptive tracking

One of the challenges of our SST system is that lighting conditions might change over time within a video sequence so that the skin colour distribution is not constant. Apparently, the static skin colour model is incapable to handle the illumination change problem. To handle this task, we apply the useful information from tracking to update the skin colour model to solve the problem. The basic idea for adapting the skin colour model is to collect new training data for re-training the SVM classifier every frame.

Specifically, given two consecutive frames f^{t-1} and f^t , we assume their skin colour distributions are different due to a lighting change. For the current frame f^t, we collect the new skin samples from inside search windows that were constructed already around the predicted skin object locations by the tracker using KF. Firstly, we use the generic skin colour model as a filter, which offers only the feasible skin colored pixels to the new skin training set. Then, the filtered skin pixels (x, y) are decided as the new skin samples provided that both the decision probability of motion (in Eq. 4.2) and decision probability of position (in Section 4.2.1.3) are large enough, say over the empirical threshold.

Finally, the rest of the search window pixels are considered as non-skin pixels. Having new skin and non-skin samples, we train the SVM classifier for the f^t and then apply it to classify the pixels of the search window again. The classifier returns a skin colour probability $P_{svm}(x, y)$, which could be combined with the $P_m(x, y)$ and $P_p(x, y)$ to continuously perform the tracking.

We tested skin colour model adaptation with a number of gesture videos under the time-varying illumination. We used a light to simulate the illumination change while

capturing videos. We controlled the light intensity by moving the light closer to or further from the human body, and turning on or off the light. Fig. 4.7 displays some skin segmentation results by the updated skin color model. The visually acceptable results demonstrate the effectiveness of the proposed method.

Figure 4.7: Some segmentation results under the condition of time-varying illumination

4.3 Experimental results

We tested the proposed tracking system on a number of ECHO [ECHO] and self-captured signing videos for different SL speakers under different lighting conditions and with different occlusion conditions. Fig. 4.8 illustrates several examples of the tracked images. We used the rectangles with different colors to represent tracking of the different objects. If some objects were occluded, there rectangles were merged to one rectangle. To quantitatively evaluate the performance, we manually labelled 600 frames to construct the ground truth of the bounding boxes of the skin objects (see Fig. 4.9). Out of 600 frames, 237 frames included occlusions. As in

[Martin et al. 98], we measure the error in the position (x, y) of the centre of the bounding box. Table 4.1 shows the average error in x and y directions respectively and the average error of the tracking process, i.e. when skin object is incorrectly identified (ex. left hand identified as right hand). As shown in Table 4.1, the algorithm accuracy is very high as the maximum error is about 6 pixels, and in terms of tracking errors, only 39 frames had objects identified incorrectly, where 37 frames of them, the error where due to occlusions, and only 2 frames had errors in the absence of occlusion. From the results, we can conclude that the tracking is very robust to occlusions, as out of about 40% occluded frames, the error percentage was about 6.5%. In addition, we pluged the proposed segmentation and tracking system into a PCA-based gesture recognition system developed by a colleage in our research group and replaced the SVM colour feature by the generic skin model. The system worked on a standard PC under Matlab environment using non-optimized code and run at 10 frames/sec.

	Face	Right hand	Left hand
Error in X direction (pixel)	1.722	1.516	4.781
Error in Y direction (pixel)	2.796	2.268	6.236
Tracking error %	6.1%	6.5%	6.5%

Table 4.1: The Statistical Accuracy of the Proposed Tracking System

Figure 4.8: Some samples of the proposed tracking system

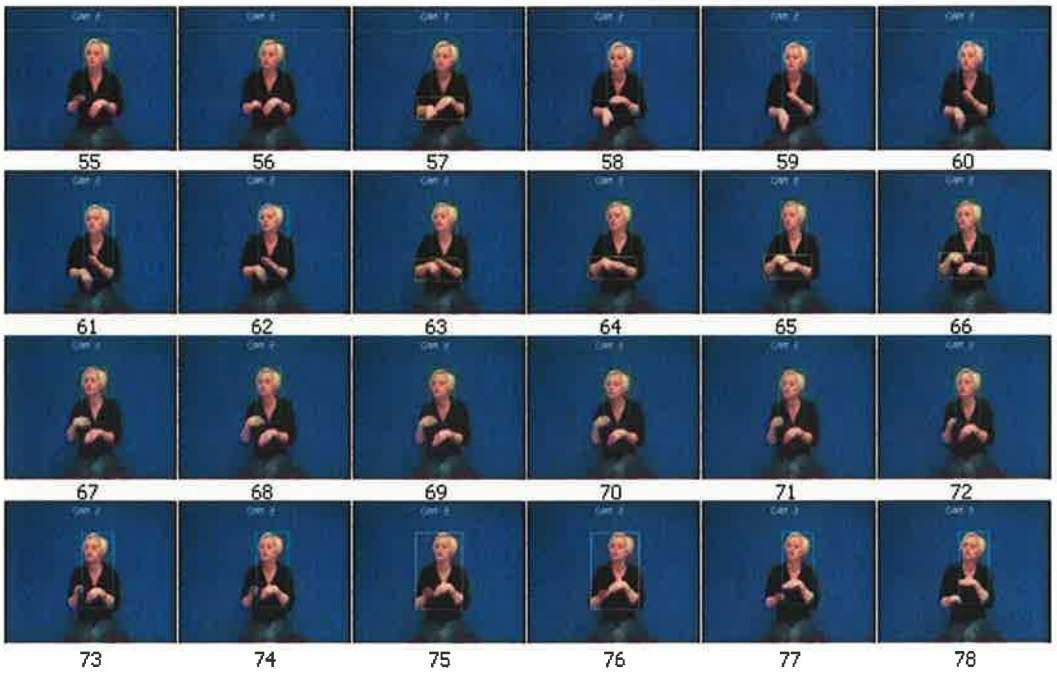


Figure 4.9: Sample ground truth frames, different rectangle colours represents occluded skin objects

4.4 Summary

In this chapter, we presented a complete unified system for segmentation and tracking of skin objects for gesture recognition. The algorithm works on skin detection instead of using colour gloves. Occlusion detection is handled between any of the face and the two hands accurately, which is very important as most of the real-world SL video sequences include many occlusions (about 40% as demonstrated in our testing data). The tracking process uses the occlusion information to maintain a high-level understanding of the occlusion status of all the skin objects and can identify with high accuracy and speed the skin objects in the scene based on simple heurestic rules. More importantly, tracking and segmentation tasks have been approached as one unified problem where tracking helps to reduce the search space used in segmentation, and good segmentation helps to accurately enhance the tracking performance. The system demonstrates a good compromise or trade-off between the computational cost and the overall accuracy. Currently it can run at 10 frames/sec using the generic

skin model instead of the SVM classifier. The system is modular and most of the used components are very simple in computations (except the SVM colour feature) making it easy to replace more faster/accurate components in the future. However, we don't handle the occlusion segmentation problem to separate different occluded skin objects, which might be a useful feature in the recognition phase.

Chapter 5

Modelling and Segmenting Sign Language Subunits

5.1 Introduction and literature review

Despite the great efforts in SLR so far, most existing systems can achieve a good performance with only small vocabularies or gesture datasets. Increasing vocabulary inevitably incurs many difficulties for training and recognition, such as the large size of the required training set, signer variation and so on. Therefore, to reduce these problems, some researchers have proposed a subunit instead of whole-sign based strategy for SLR [Liddell and Johnson 89, Vogler and Metaxas 99, Yeasin and Chaudhuri 00, Bauer and Kraiss 01, Fang et al. 04]. In contrast with traditional systems, this idea has the following advantages. First, the number of subunits is much smaller than the number of signs, which leads to a small sample size for training and small search space for recognition. Second, subunits build a bridge between low-level hand motion and high-level semantic SL understanding. Only after subunits become available are structural and linguistic analysis possible, and the capability of SLR can be greatly improved. In general, in the field of linguistics, a subunit is defined to be the smallest contrastive unit in a language. In [Stokoe 78], Stokoe has provided the evidence that the signs can be broken down into elementary units through the study of American SL. However, there is no generally accepted conclusion yet

about how to model and segment subunits in the computer vision field. Therefore, a number of researchers have put forward a variety of definitions and segmentation solutions. In [Liddell and Johnson 89], Liddell et al. introduced a Movement-Hold model. In this model, signs are sequentially parsed into subunits, called movements and holds. "Movements" are temporal segments during which the signer's configuration changes. In contrast, "holds" mean the hands remain stationary for a short term. Following this model, Vogler [Vogler and Metaxas 99] manually detected the boundaries between movements and holds. The model is effective only under the assumption that there are clear pauses between subunits. Moreover, for a task of large vocabulary SLR, manual segmentation is impossible. Alternatively, Yeasin et al. [Yeasin and Chaudhuri 00] define a subunit as a temporal segment with uniform dynamics. The motion breakpoints are considered as the subunit boundaries, which are located by a change detection algorithm. This scheme is easy to implement, but requires salient movement pauses as well. In addition, due to behaviour variations between different signers, simple change detection using a unified threshold may fail to achieve good performance. Another interesting work was published in [Bauer and Kraiss 01], which proposed to employ a K-means clustering approach to self-organize subunits. Nevertheless, their clustering is based only on the spatial features from each frame. It ignores the temporal information, which might be more important in SL analysis. Recently, Fang et al. [Fang et al. 04] extracted subunits for SLR using Hidden Markov models (HMM). One HMM is trained for each sign first. Then, each state in the HMM is associated with one subunit. This work suffers from the shortcoming that they have to predefine the number of states for the HMM. It implies each sign has the same number of subunits. Unfortunately, this hypothesis is not true most of the time. Another related work in the field of facial expression recognition [Xiang and Gong 04] uses HMM for recognising Action Units (AU) of expressions. Based on the Facial Action Coding System (FACS) which divides the face into upper and lower face actions, motion are divided into action units. AUs are defined as muscle movement that combine to produce expressions. HMMs are trained for each expression where the hidden states model the AUs. To reduce the

limitations of the previous work, we propose to detect subunits from the viewpoint of human motion characteristics. We model the subunit as a continuous hand action in time and space. It is a motion pattern that covers a sequence of consecutive frames with interrelated spatio-temporal features. In terms of the modelling, we then integrate hand speed and trajectory to locate subunit boundaries. The contribution of our work lies in three points. First, our algorithm is effective without needing any prior knowledge such as the number of subunits within one sign or the types of signs. Second, the trajectory of the hand is utilized so that the algorithm does not rely on clear pauses any more. Finally, because of the use of an adaptive threshold in motion discontinuity detection, the approach is adaptive to signer variation and the refinement by temporal clustering (after the temporal segmentation is done) makes it more robust to noise. In general, our main claim is that our algorithm can discover subunits within a segment of raw video without any human supervision. In other words, it performs unsupervised learning on a set of unlabeled data. In the following sections, we will explain our system for subunit detection and the evaluation results of the proposed approach.

5.2 System overview

We define a subunit as a motion pattern with interrelated spatio-temporal features. We attempt to study human motion habits and then address the subunit boundary detection issue in light of the learned useful information. After we watched a large number of SL videos, two observations were noticed. First of all, while shifting from one subunit to the next subunit, the hand movement of signers always goes through three phases: deceleration, acceleration, and uniform motion. This motivates us to locate the subunit boundary by discovering the speed change of hand motion. Secondly, the motion trajectory during a subunit often forms a continuous and smooth curve in 2-D or 3-D space such as in Fig. 5.1.

The trajectory generally displays considerable discontinuities surrounding the subunit boundary. The detection process is thus the recognition of perceptual dis-

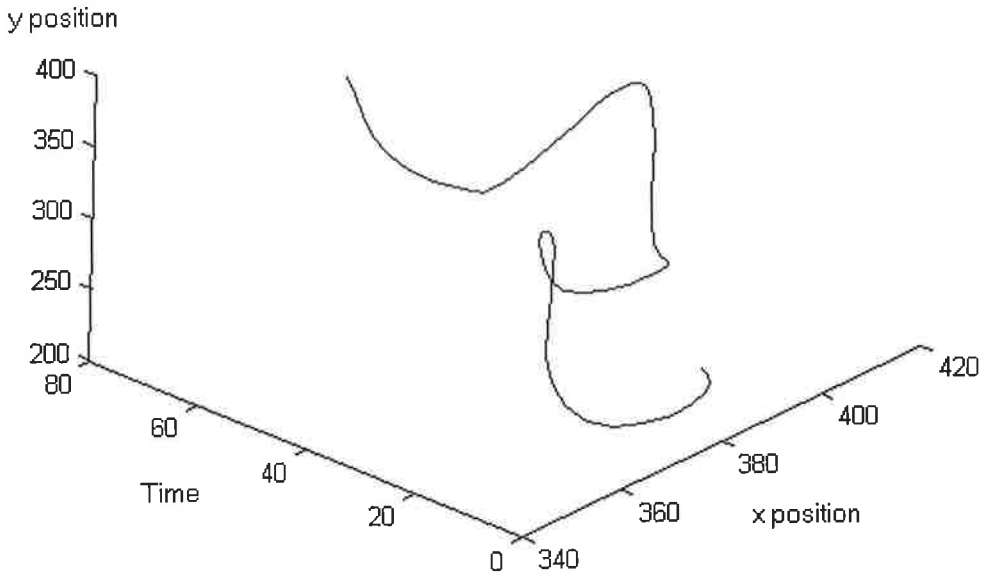


Figure 5.1: Sample trajectory of British sign "Banana"

continuities. As a result, the trajectory information can remove any restrictions on the speed such as time warping as it can verify the turning points where the motion pattern is going to change. Fig. 5.2, 5.3, and 5.4 explain these two observations using examples of real signs. We can see in each figure the motion speed and trajectory curves of the signs and the detected boundary points between subunits. As can be seen from the examples, the discontinuities take place around the subunit boundaries in both the motion speed and the trajectory domain. As a result, we try here to combine motion speed and trajectory in order to segment subunits. We believe that the two features (speed and trajectory) when used together can help to make the detection better and more accurate, as the trajectory information can verify if there is a real visual discontinuity.

A block diagram of the system architecture is shown in Fig. 5.5. The system consists of four major components. The objective of the first component is to return speed

and trajectory information. They are easy to obtain once the hands have been segmented and tracked across frames as discussed in chapter 4. The second component, the speed discontinuity detector, works as follows. The speed difference is calculated to quantify the motion variation from frame k to frame $k+1$. Compared against a threshold T, if the speed difference is $> T$, a motion discontinuity between frames k to frame $k+1$ is recorded. T is automatically decided by an adaptive thresholding technique. The third component, the trajectory discontinuity detector, is responsible for finding "corner" points with significant changes in trajectory by measuring the "sharpness" of the bend in the curve. Afterwards, boundary candidates detected by both detectors are combined and serve as the input into the fourth component of the system, temporal clustering.

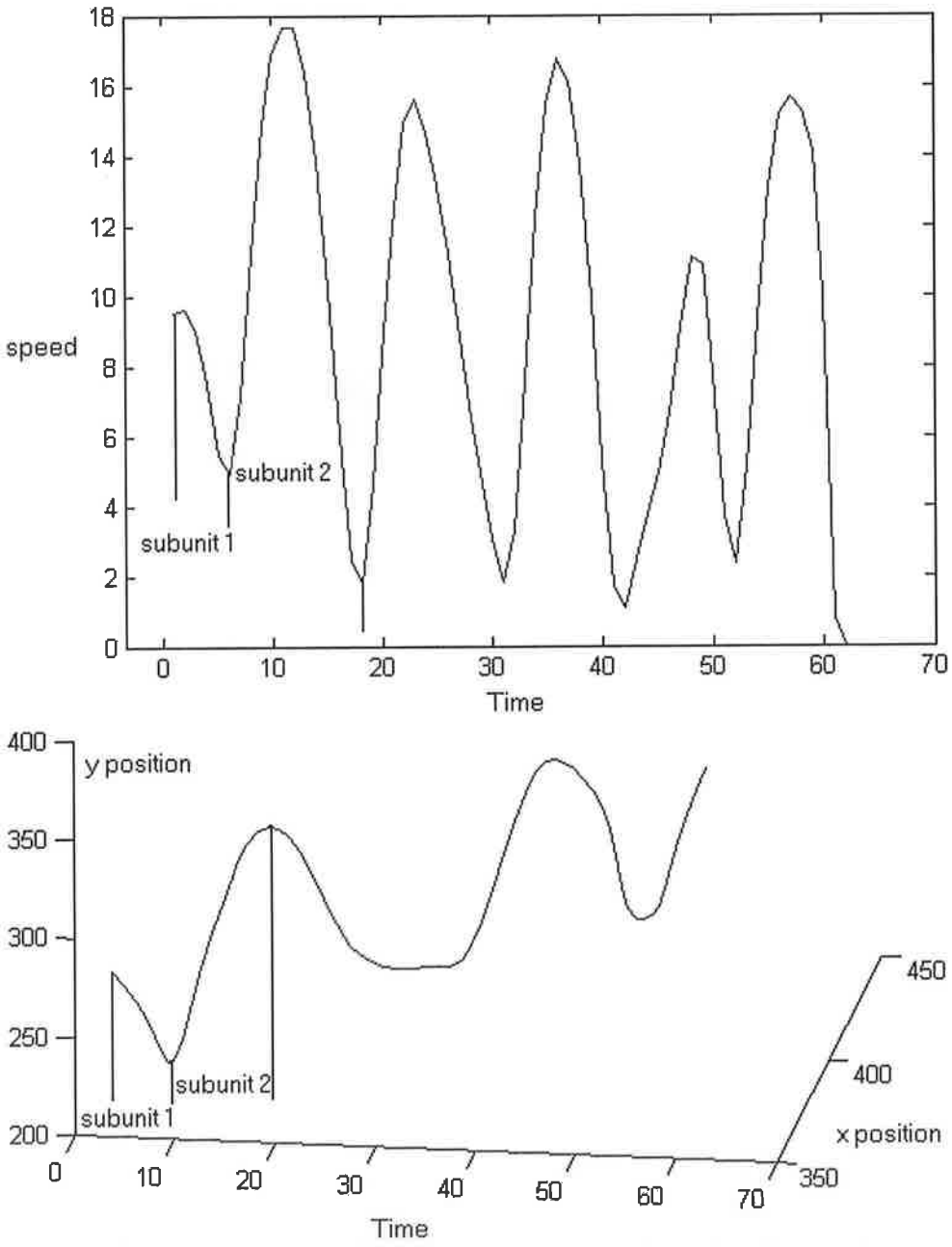


Figure 5.2: An example of the speed and trajectory curves of a real sign (sample 1)

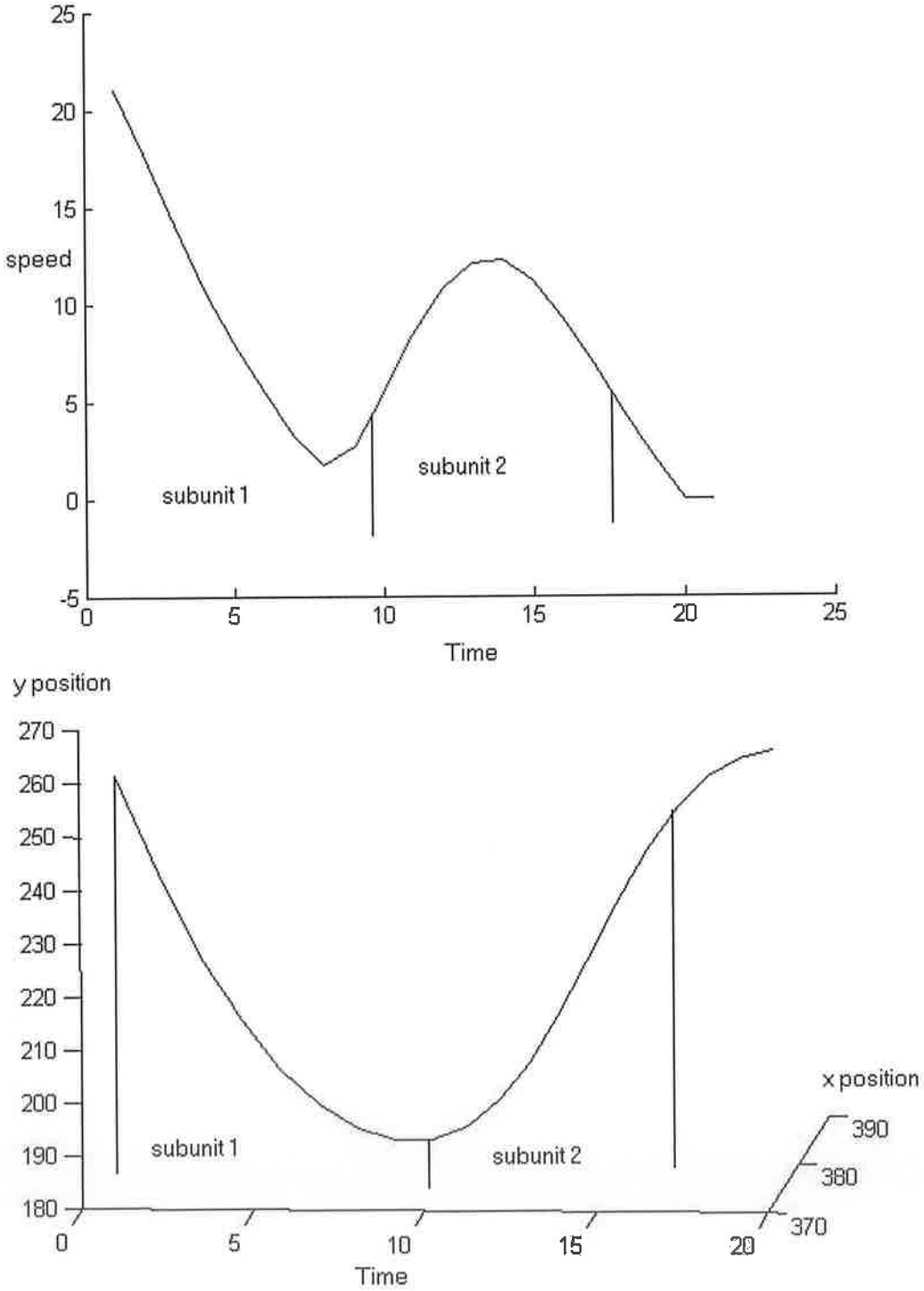


Figure 5.3: An example of the speed and trajectory curves of a real sign (sample 2)

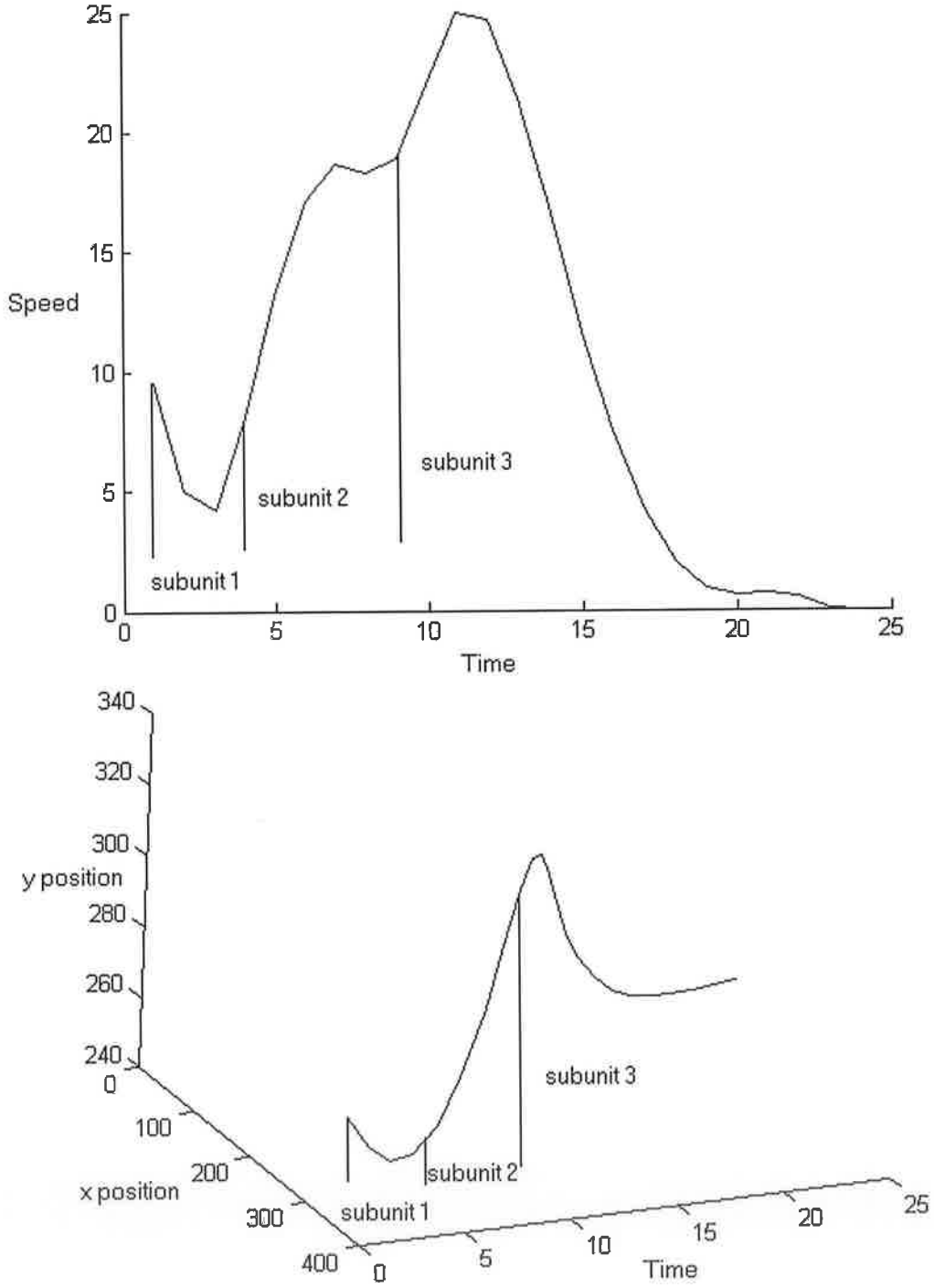


Figure 5.4: An example of the speed and trajectory curves of a real sign (sample 3)

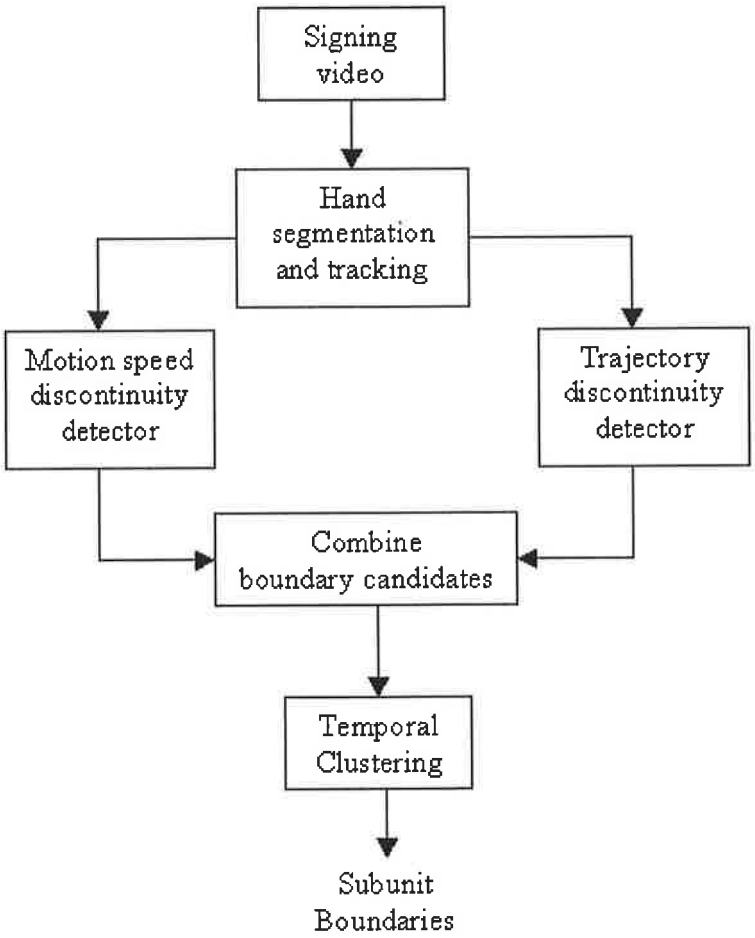


Figure 5.5: Basic architecture of subunit detection system

5.3 System components

5.3.1 Hand segmentation and tracking

The task of this component is to generate motion speed and trajectory information, which is implemented by the following three steps. Firstly, given a frame k of a sign f the hand segmentation and tracking (see chapter 4) algorithm is applied to get the position of the hand in this frame. Secondly, the trajectory of sign f, $Tr_f = [x_k, y_k]'$ can be obtained from the hand position in every frame. The motion speed of the hand S_k is calculated based on frame k and $k+1$:

$$S_k = \| (x_{k+1}, y_{k+1}) - (x_k, y_k) \| \tag{5.1}$$

Finally, both motion speed and the trajectory are smoothed using splines [Lee and Xu 00]. It is worth noting that there are two types of hand movement in SL: dominant hand where one hand performs the sign and bimanual movements where the two hands together perform the sign. The two types of movements can be distinguished by their trajectory information. In our work, for the former case, only spatio-temporal features from the dominant hand movement are used. Otherwise, features from both hands are employed. For simplicity, we illustrate the algorithm utilizing the dominant hand movement.

5.3.2 Motion speed discontinuity detector

This detector works by examining local speed changes of hand movements. The speed difference by subtraction of successive frames is utilized as the discontinuity metric. Given the hand motion speed S_k of the k^{th} frame, its speed difference is defined as:

$$D_k = |S_{k+1} - S_k| \tag{5.2}$$

Then, the obtained discontinuity values are compared with a threshold T_s:

$$M_k = \begin{cases} 1 & \text{boundary candidate} \quad \text{if } D_k > T_s \\ 0 & \text{non-boundary candidate} \quad \text{else} \end{cases} \tag{5.3}$$

Deciding the optimal threshold T_s is a nontrivial problem and there are many famous techniques in the literature proposed for calculating threshold values from histograms usually designed to convert grayscale images to binary ones. For simplicity, we choosed to employ a simple adaptive thresholding method calculated based on the weighted sum of the total histogram average bin values:

$$T_s = \frac{\sum_{i=1}^{N} freq_i * v_i}{\sum_{i=1}^{N} freq_i} \tag{5.4}$$

where N is the total number of bins in the speed difference histogram (histogram of $D_k$values), $freq_i$ is the count of values in bin i, and v_i is the bin value.

5.3.3 Trajectory discontinuity detector

Trajectory segmentation has been previously studied in areas such as video segmentation [Xiang and Gong 04] where techniques such as Discrete Curve Evolution (DCE) uses a distance or similarity measure such as Euclidean distance to measure distance using 3 neighbour points and if it exceeded a threshold, the vertex point is declared as a breaking point. Also, another similar technique, Forward-Backward Relevance (FBR) has been proposed by [Xiang and Gong 04] following DCE method but using non-neighbour points to become more robust to noise. The hand motion trajectory offers rich spatio-temporal information. The purpose of this component is to discover points of perceptual discontinuity along the trajectory curve. A corner can be defined as a point on a curve where the curvature is locally maximal. It is well known [Rosten and Drummong 06] that corners generally correspond to such places where perceptual changes are happening. Hence, the trajectory discontinuity detector is effectively a corner point detector. Here, we apply two metrics to specify the corner points. One is the angle calculated on the local neighbourhood, and the other is the angle difference. If a point's angle is very sharp (acute) or its angle is very different from the angles of its neighbouring points, this point is determined as a corner. Fig. 5.6 shows an example of a trajectory where the corner points can be either acute or obtuse between motion patterns. This is our motivation behind using the two metrics for detecting corner points.

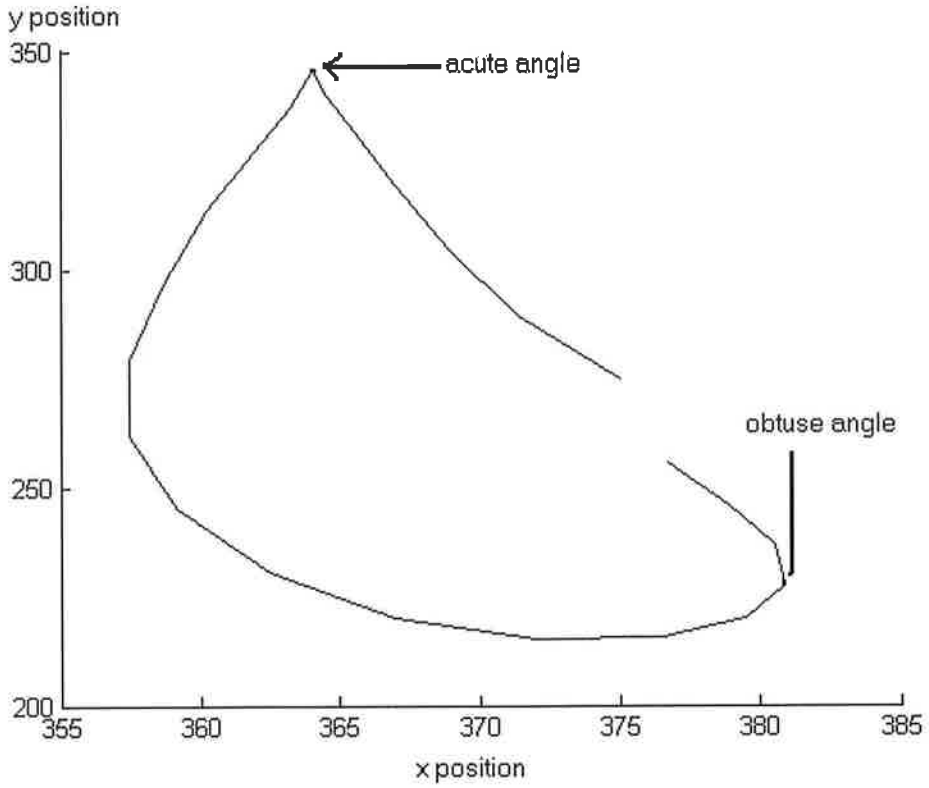


Figure 5.6: Trajectory curve of a sign showing motion discontinuity at different types of corner points

Let $Tr = [x_k, y_k]'$ be a trajectory curve, where x_k and y_k denote the hand's 2-D location in the k^{th} frame. The angle φ_k associated with point (x_k, y_k) is calculated by:

$$\varphi_k = \arccos(\frac{a^2 + b^2 - c^2}{2ab}) \tag{5.5}$$

Here a,b,c are distances among three consecutive points. To be specific:

$$\begin{aligned} a &= \| (x_k, y_k) - (x_{k-1}, y_{k-1}) \| \\ b &= \| (x_{k+1}, y_{k+1}) - (x_k, y_k) \| \\ c &= \| (x_{k+1}, y_{k+1}) - (x_{k-1}, y_{k-1}) \| \end{aligned} \tag{5.6}$$

Then, the angle difference is defined as:

$$D\varphi_k = |\varphi_{k+1} - \varphi_k| \tag{5.7}$$

The trajectory discontinuity detector is thus implemented by:

$$C_k = \begin{cases} 1 & \text{boundary candidate} \quad \text{if } \varphi_k < T_\varphi \text{ or } D\varphi_k > T_{D\varphi} \\ 0 & \text{non-boundary candidate} \quad \text{else} \end{cases} \tag{5.8}$$

Where the two thresholds T_φ and $T_{D\varphi}$ are adaptively calculated using Eq. 5.4. The proposed technique can work online given that we have pre-calculated the speed and angle thresholds of the signer and accumulated few points such as the first 3 trajectory points

5.3.4 Combining boundary candidates

Combining boundary candidates from the speed and trajectory discontinuity detectors M_k and C_k can be done simply by selecting the common boundaries, but as the data can be sometimes noisy, it is quite hard to depend only on the exact matched boundaries. As a result, we decided to use a small window of length 3. If there is no an exact matching boundary at C_i and M_i, we search for the first matching boundary in $M_{i-3}, M_{i-2}, M_{i-1}, M_{i+1}, M_{i+2}, M_{i+3}$. We call the detected boundaries in this stage "preliminary boundaries".

5.3.5 Temporal clustering

In practice, our approach is not able to achieve an outstanding performance because of noise from irregular motion patterns, motion variations from different signers, errors in matching trajectory and speed boundary candidates, and finally from the errors that might occur during the tracking of the hands. These noises and variations normally result in some false subunit boundaries and very small subunit segments. Fig. 5.7 shows an example of a false boundary detected due to mismatching between

the speed and trajectory detectors. In this case, it may be necessary to introduce a temporal clustering process to remove the false boundaries and further improve the results.

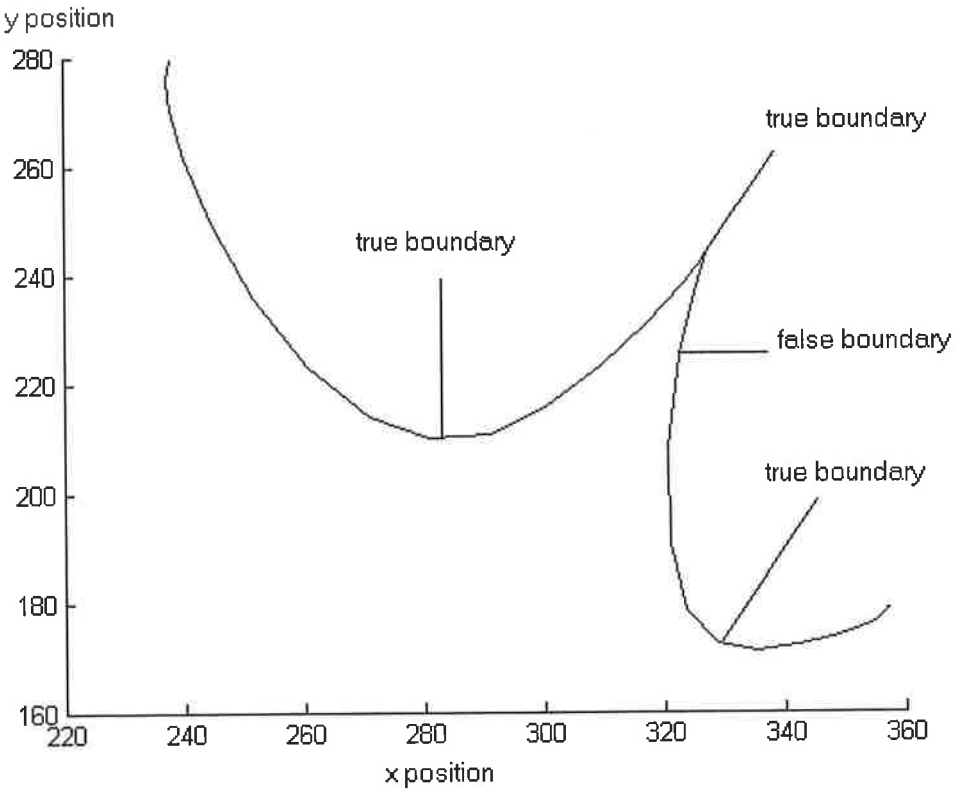


Figure 5.7: An example of real sign trajectory with a false boundary detected

The principal idea of our temporal clustering is to merge the consecutive similar preliminary subunit segments using more spatio-temporal visual features. The key problem is how to measure the similarity between preliminary subunit segments. Hidden Markov Models aim to automatically recognize time series data using the forward-backward or veterbi algorithm. However, the training is computationally expensive and needs many training samples. Other techniques used to learn intrinsic classes such as Entropy and Minimum Description Length treat continuous temporal data as a fixed length data vectors which can affect the results due to the non-linear warping of the time scale. In our approach, we apply DTW (dynamic time warping) to address this problem since it has been acknowledged to be a very good and pop-

ular tool for comparing temporal signals of different length [Fang et al. 04], thus we use it as a similarity metric only without any need for recognition such as in HMMs. A related work in [Ng and Gong 02] was proposed to learn trajectory models based on Levenshtein-distance based DTW. The authors used the inverse of the pairwise DTW distance between trajectories to build an affinity matrix. Then, the clustering of the trajectories was treated as a graph partitioning problem. One difference between their approach and ours is that they didn't assume that the trajectories could be merged or more specifically they assumed that the trajectories were segmented without errors or false boundaries, while our objective here is to use the clustering to detect preliminary sequential subunits that should be merged. DTW uses dynamic programming to find the best warping path that leads to the minimal warping cost between two preliminary subunits. More specifically, suppose we have two preliminary subunits $U = \{u_1, u_2, ..., u_m\}$ of length m and $Q = \{q_1, q_2, ..., q_n\}$ of length n. Here, u_i and q_j represent feature vectors extracted from every frame. The warping path between U and Q is denoted by:

$$W = \{w_{1,}w_{2,}w_{3,}, ...w_K\} \tag{5.9}$$

where $max(m,n) \leq K \leq m+n-1$, with $w_k = (i_k, j_k)$. Each element $w_k = (i_k, j_k)$ is associated with a distance between the two vectors u_{i_k} and q_{j_k}, which is:

$$d(w_k) = d(u_{i_k}, q_{j_k}) = \| u_{i_k} - q_{j_k} \| \tag{5.10}$$

The warping cost of W is given by:

$$Z(W;U,Q) = \sum_{k=1}^{K} d(w_k) = \sum_{k=1}^{K} \| u_{i_k} - q_{j_k} \| \tag{5.11}$$

The warping path is subject to some constraints such as endpoint, continuity, and monotony criterions. From many satisfiable warping paths, we pick the best one with the minimal warping cost, and then define the distance between two preliminary subunits U and Q as:

$$DTW(U,Q) = min\{Z(W;U,Q)\} = min\{\sum_{k=1}^{K} \parallel u_{i_k} - q_{j_k} \parallel\} \qquad (5.12)$$

The search for the best warping path can be implemented by dynamic programming. In order to make DTW work efficiently, the construction of feature vector p_k for the k^{th} frame plays an important role.

In our discontinuity detectors, we only consider the *"local spatio-temporal"* features computed from a pair of consecutive frames. Here, we design our feature vector p_k by taking into account some *"global spatio-temporal factors"* to represent the motion pattern of the whole subunit. These global features are based on subunit trajectory information and are invariant to trajectory translation and scaling so that they are capable of dealing with the motion noises and variations.

If we assume the hand segmentation and tracking system can provide us with the following information: (1) hand location in the k^{th} frame, (x_k, y_k) ; (2) the corresponding preliminary subunit trajectory, Tr ; (3) the centroid of Tr, (x_c, y_c) ; (4) the head position (x_h, y_h) . The feature vector p_k contains 6 factors, which are formulated as:

- *Hand motion speed.* calculated as: $S_k = \parallel (x_{k+1}, y_{k+1}) - (x_k, y_k) \parallel$.
- *Hand motion direction code.* First, the hand motion direction is described by: $\theta = \arctan(\frac{y_{k+1} - y_k}{x_{k+1} - x_k})$. Then, θ is quantized into 18 direction codes of range 20 degree each. The yielded direction code is denoted by α_k.
- *Distance between hand position and trajectory centroid.* calculated as: $\beta_k = \parallel (x_k, y_k) - (x_c, y_c) \parallel$.
- *Orientation angle of vector from hand location to trajectory centroid.* calculated as: $\gamma_k = \arctan(\frac{y_c - y_k}{x_c - x_k})$.
- *Distance between hand and head.* calculated as: $\delta_k = \parallel (x_h, y_h) - (x_k, y_k) \parallel$.
- *Orientation angle of vector from hand to head.* calculated as: $\epsilon_k = \arctan(\frac{y_h - y_k}{x_h - x_k})$.

In these descriptors above, the former 2 descriptors indicate the hand motion velocity information, the middle 2 descriptors measure the hand position relative to the whole trajectory, and the latter 2 descriptors depict the hand position relative to the head. This set of features can provide us with good information about the global motion pattern of the corresponding subunit. We tried to concentrate more on the motion because this is related to our original definition of the subunits as a continuous motion pattern. As a result, we didn't want to use the shape features of the hand as the signer might change his hand configuration while moving. To compute easily, these 6 spatio-temporal features are normalized into the range between 0 and 1. Finally, the feature vector is derived as:

$$ps_k = (N(S_k),\ N(\alpha_k),\ N(\beta_k),\ N(\gamma_k),\ N(\delta_k),\ N(\epsilon_k)) \tag{5.13}$$

where $N(\bullet)$ is a normalization operator. Once the similarity between preliminary subunits can be measured, the last step is to cluster these temporal segments. If consecutive preliminary subunits belong to the same cluster, we merge them into one subunit and then refine the boundary points. DTW distance does not obey the metric axioms and thus can not be directly used in traditional clustering methods which rely on the computation of cluster centroids such as k-means. This motivated us to adopt the agglomerative clustering algorithm [Data clustering] due to its outstanding performance which is a hierarchical algorithm and does not depend on centroid calculations.

However, the quality of the clustering is dependent on the trajectory information from the segmentation and tracker component. Given that the subunit segmentation algorithm has supplied us with the preliminary subunits which represents the different motion patterns, we are sure that trajectory continuity exists within each subunit. Meanwhile, the DTW distance metric combined with the hierarchical clustering technique can cope with some noise that can exist in the trajectory information. In the next chapter, we will explain in detail how we use the clustering algorithm which provides us with codebook entries in the recognition task.

5.4 Experimental results

We tested the proposed work with a number of real-world signing videos. They were collected from three different sources: Echo database [ECHO], self-captured sequences, and data shared from other research group [SLR group]. To evaluate our proposed approach, we will first demonstrate some visual results for subunits segmentation to be able to subjectively judge how promising is the approach. Later, we will present an experiment to quantitatively evaluate the algorithm.

5.4.1 Subjective experiment

We demonstrate here four samples of videos, three of which are from other research group [SLR group], and one self captured by ourselves. Fig. 5.8, 5.9, 5.10, and 5.11 show sample frames from the videos. We denote the detected boundary frames by a rectangle drawn around the frame number. In the first three samples the signer is wearing colour gloves to simplify the segmentation and tracking task, while in the last sample we tested our algorithm using our hand segmentation and tracking system as discussed in chapter 4. In the four samples, we can notice that the algorithm performance is promising for segmenting different motion patterns which we will use as our basic blocks in the recognition phase of the sign in the next chapter.

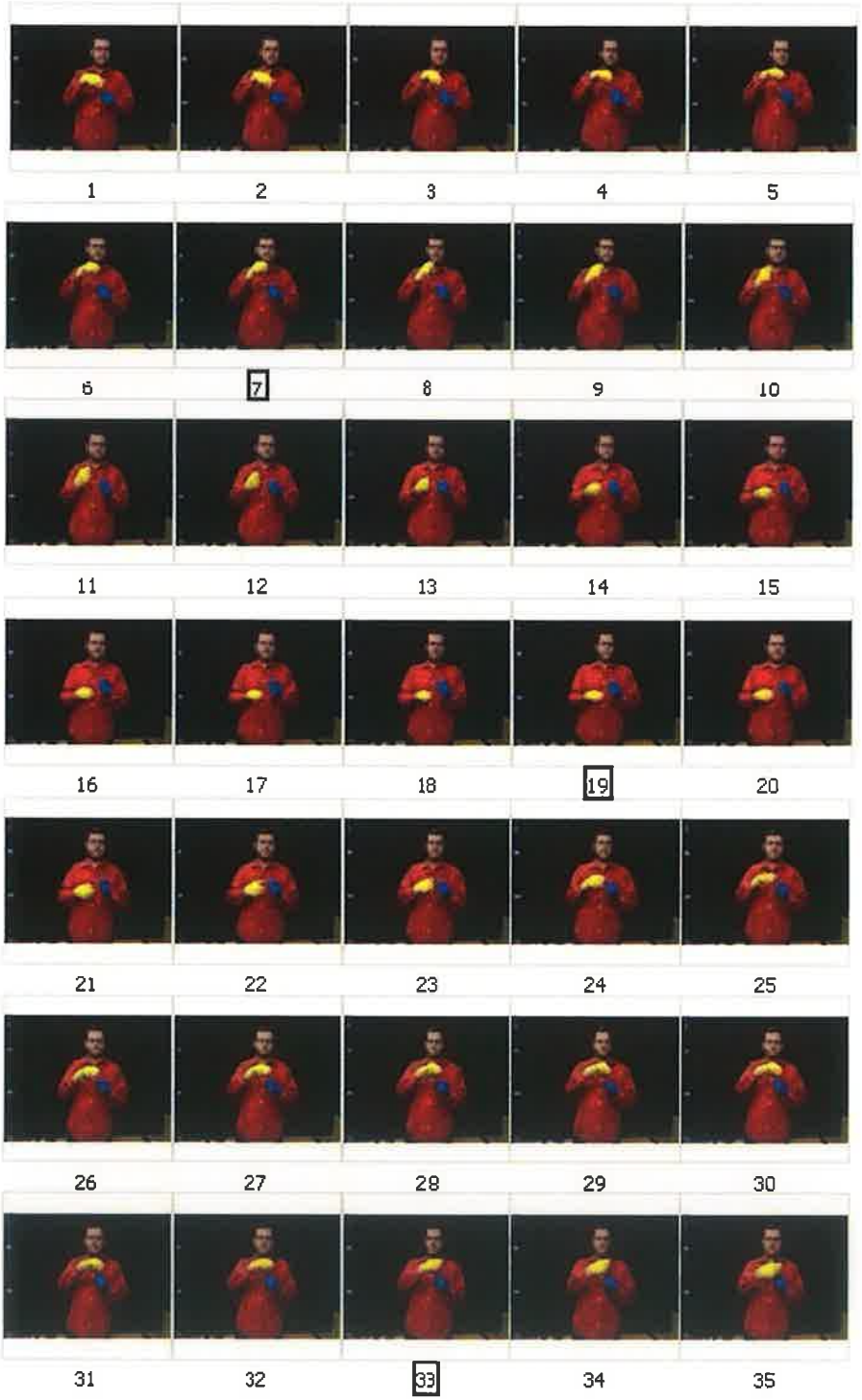


Figure 5.8: Sample 1 of subunits segmentation (sign "banana")

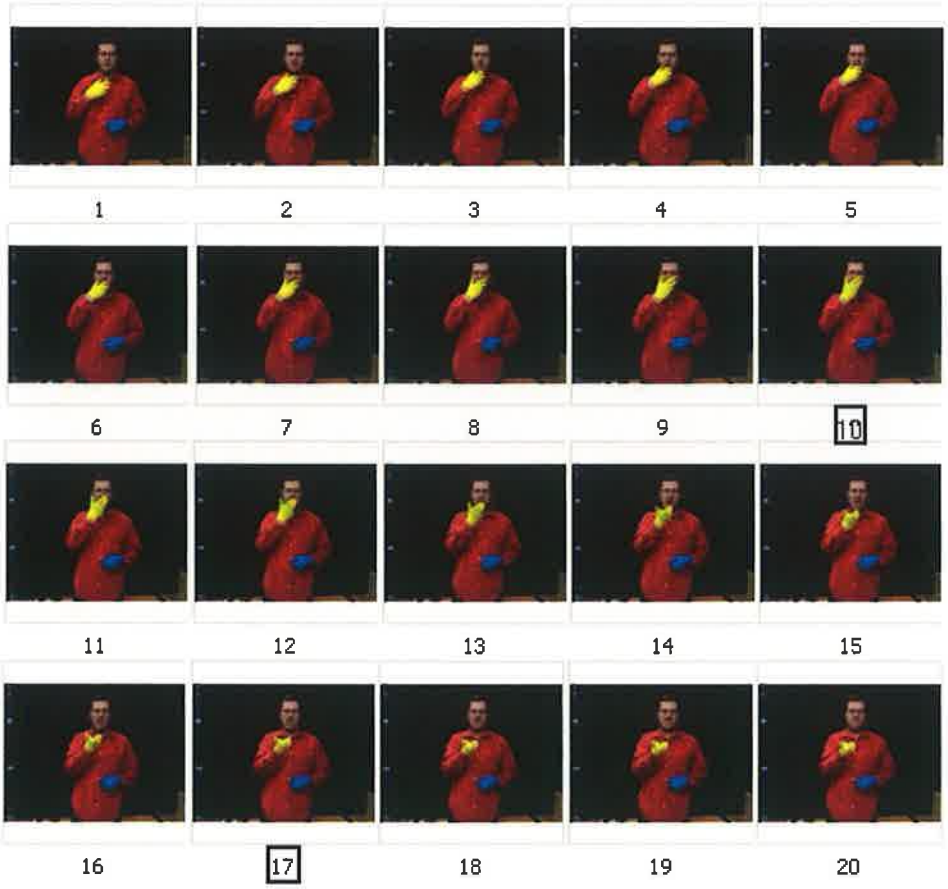


Figure 5.9: Sample 2 of subunits segmentation (sign "apple")

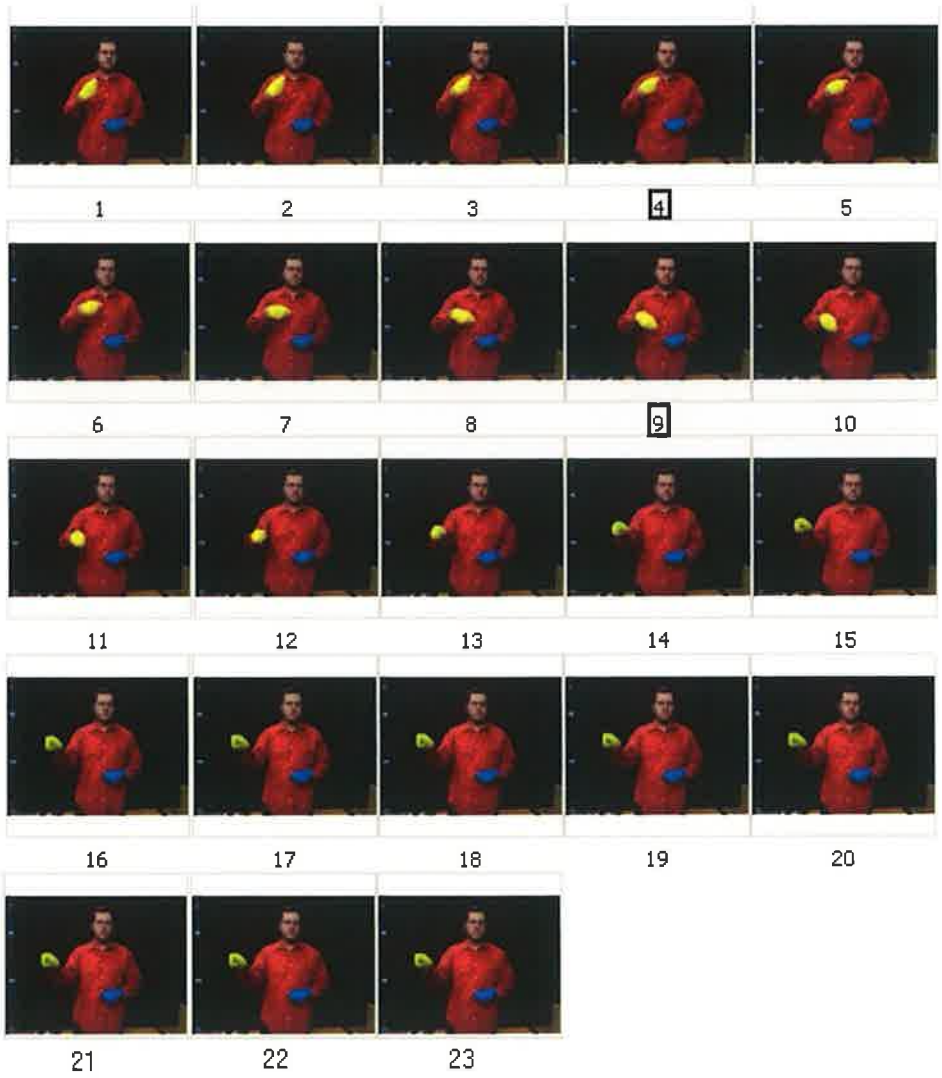


Figure 5.10: Sample 3 of subunits segmentation (sign "bat")

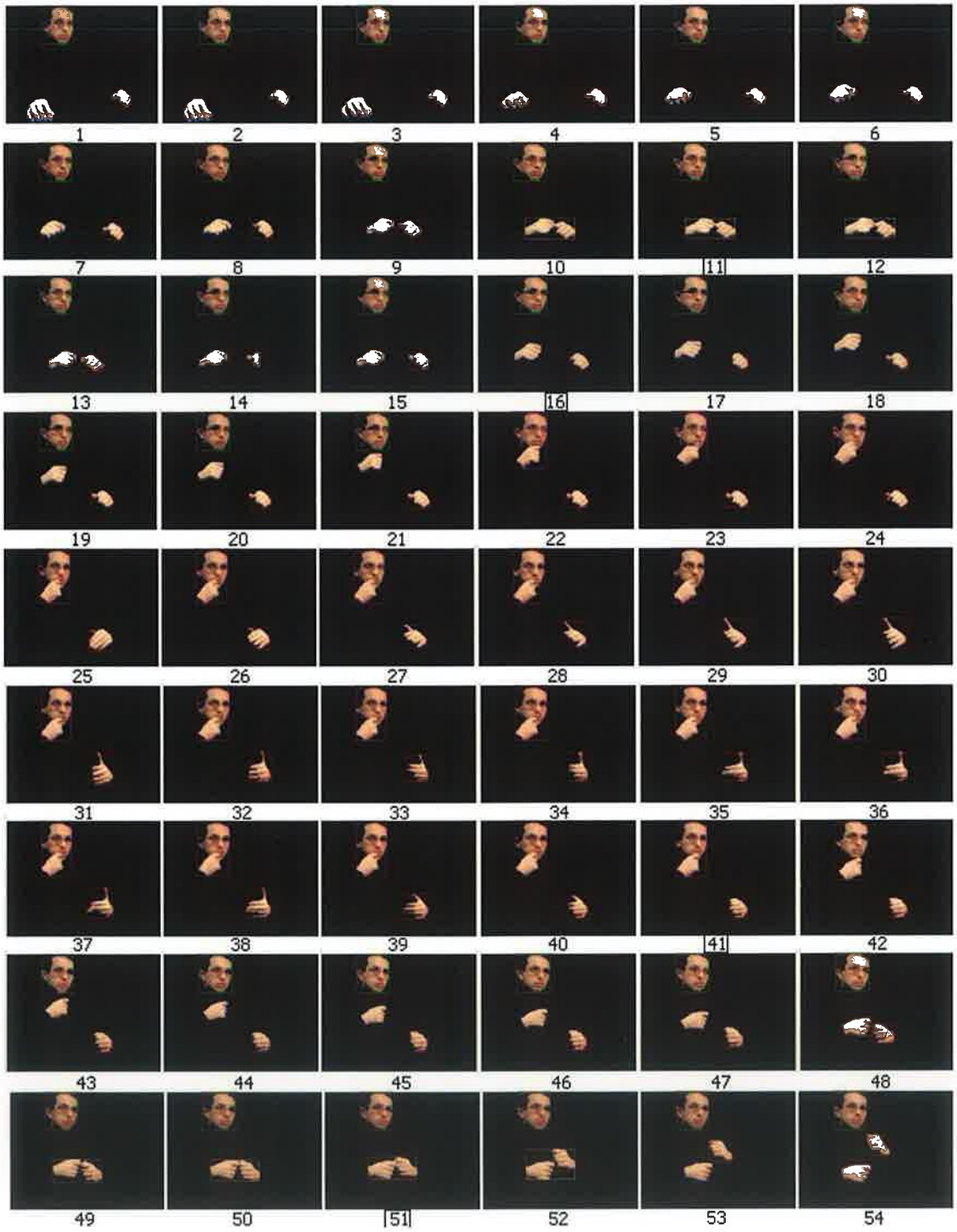


Figure 5.11: Sample 4: subunits segmentation for the signer's right hand. Continued next page

Fig. 5.11 continued.

5.4.2 Quantitative experiment

This experiment was constructed to quantitatively evaluate our work. We randomly selected 10 signs (colour-glove videos, and isolated signs) from our collected dataset [SLR group]. To test the capability of our algorithm in handling noise and motion variations, every sign was performed with 10 repetitions. 5 examples of each sign were utilized to construct the ground truth, and the other 5 examples were used for testing. The ground truth was built through human manual segmentation.

For each training sign, we manually segment the subunits of the 5 samples and cluster them using the DTW distance metric. For each cluster, we calculate the *medoid subunit* (the subunit which has the minimum average distance to all the other subunits in the same cluster). After clustering, a codebook can be constructed for every sign using the medoid subunits, which act as representatives for the subunits that exist in these sign. The experiment measures how accurate the codebook that is generated automatically by our algorithm compared to the ground truth codebook. An important issue in any clustering problem is how to decide the number of clusters. In our case, we use the average number of segmented subunits for the 5 testing samples as a threshold for the number of clusters. The two following metrics, recall and precision, were adopted to measure the performance:

$$Recall = \frac{N_c}{N_g}$$
$$Precision = \frac{N_c}{N_d} \tag{5.14}$$

where :

N_g : the number of the correct subunits in ground truth codebook

N_d : the total number of subunits detected in the algorithm codebook

N_c : the number of correct subunits detected in the algorithm codebook

Table 5.1 lists the statistical detection performance. As can be seen, our algorithm reaches an average recall of around 0.82 and average precision of around 0.76. Through carefully studying experimental results, especially failed cases, we found three factors mainly influence the detection accuracy.

The first one is the noise and varying motions which negatively affect the matching of boundary frames detected by both the speed and trajectory information. The second factor is the information quality provided by the hand segmentation and tracking system. In some cases, the segmentation and tracking system cannot guarantee to return accurate hand positions and motion trajectories due to motion blur, illumination change, complicated background, and occlusion. The third factor is the hand motion complexity. In some cases, the hands are involved in somewhat complex movements such as in the movement of the fingers while the palm is stationary or when the hand is occluded with other skin object. Finally, the previous factors may affect the real number of clusters to generate the final codebook. In general, given that our experiment did not try to avoid the above factors, we may claim that the proposed approach is promising.

Sign number	N_g	N_d	N_c	Recall	Precision
1	2	3	2	1	0.66
2	2	2	2	1	1
3	3	4	3	1	0.75
4	2	2	2	1	1
5	2	2	2	1	1
6	3	2	1	0.33	0.5
7	3	2	1	0.33	0.5
8	1	1	1	1	1
9	2	3	2	1	0.66
10	2	2	1	0.5	0.5

Table 5.1: Statistical detection performance of the proposed subunit segmentation system

To demonstrate the performance of clustering and reasons of possible errors, fig. 5.12 shows the dendrogram of subunits clustered in four clusters. The total subunits were segmented from 10 sign samples. From observing the sign samples, it can be shown that there are four main subunits. Two types of errors occured in this clustering. First, subunit no.38 were originally segmented into 2 subunits instead of 4 due to motion variation. Then as the 2 subunits were clustered into the same cluster, they were merged together, so the whole sign ended up to be in one subunit. And as the average number of subunits for the 10 samples (in this case was 4) is used to determine the number of clusters, this sign is considered as one cluster.
Second, as a direct result to the previous error, two true clusters were merged into one cluster to make the final number of clusters 4. However, assuming that the first error didn't happen, we should have ended up with the 4 true clusters.

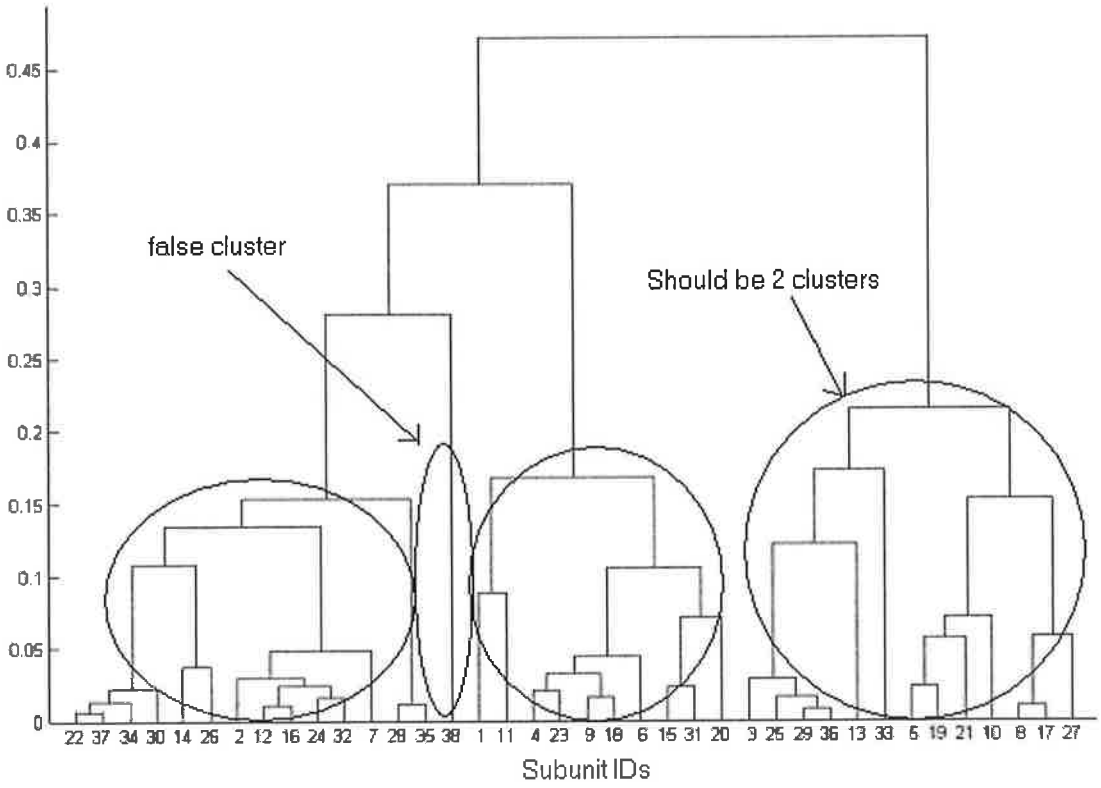


Figure 5.12: An example of a dendrogram for clustering real sign subunits using 10 sign samples

5.5 Summary

In this chapter, we have studied human action characteristics and taken advantage of them to develop a subunit boundary detection model. Dealing with a small number of subunits instead of the whole sign has many advantages especially in the task of SLR as the number of subunits is much more smaller than the total SL vocabulary size. Motion trajectory and speed information derived from hand motion are integrated to generate potential subunit boundaries. A temporal clustering utilizing more spatio-temporal features is then applied to refine the performance. The presented model is robust to various signers and doesn't require any previous knowledge about the signs or the number of subunits, thus it can operate in a completely unsupervised way to discover the subunits in the sign vocabulary. It is very easy to implement, can

operate in real time and may be efficiently incorporated in a gesture/SL recognition system. Subjective and quantitative evaluations based on a real-world data have demonstrated the effectiveness and robustness of the proposed work.

Chapter 6

Subunit-based Sign Language Recognition

6.1 Introduction

A large amount of effort has been devoted to research in SLR. Encouraged by the success of HMMs in speech recognition, most existing approaches apply the same idea to SLR and focus on training classifiers on isolated signs. Some representative work can be found in [Starner et al. 98, Liang and Ouhyoung 98, Vogler and Metaxas 98]. HMMs are capable of modelling temporal signals due to its state-based statistical model. However, one major shortcoming lies in its requirement for extensive training data to handle variations and represent temporal transitions. Normally, HMM-based algorithms need 40-100 training examples for each sign to achieve good performance, which was pointed out by [Kadir et al. 02]. Hence, this group of schemes is not suitable for SLR with a large vocabulary.

In the previous chapter we discussed the subunit-based approach for decomposing the whole sign into small elementary subunits which has the advantage that the number of subunits is much smaller than the number of signs, which leads to a smaller sample size for training and a smaller search space for recognition. Second, subunits build a bridge between low-level hand motion and high-level semantic SL understanding. In this chapter, we attempt to develop an effective SLR system using AdaBoost learning

on subunits.

AdaBoost was originally invented by Freund and Schapire [Freund and Schapire 95]. AdaBoost has been successfully used in a wide variety of learning applications such as image retrieval [Tieu and Viola 04], object detection [Opelt et al. 06], action recognition [Lv and Nevatia 06], and gesture recognition [Lockton and Fitzgibbon 02] within the last decade. Nevertheless, to our best knowledge, very little work has been done in SLR. In our work, AdaBoost is adopted to select discriminative combinations of subunits and features, which are considered as weak classifiers. A strong classifier is finally constructed based on a set of learned weak classifiers.

The proposed system consists of two major stages. In the first stage, we model spatio-temporal features of the hand movement and apply them to break down signs into subunits. Next, in the second stage, we present two variations for learning boosted subunits where in the first case we train the sign classes independently, and in the second case, we train the classes jointly, which permits the various classes to share the weak classifiers to increase the overall performance and reduce the number of weak classifiers due to sharing. The presented work opens the possibility of efficiently recognizing sign language with large vocabulary using small training data. One important advantage of our algorithm is that it is inspired by human recognition abilities so it can work in a manner analogous to humans. Experiments on real-world signing videos and the comparison with classical HMM-based weak classifiers demonstrate the superiority of the proposed work.

6.2 The Adaboost algorithm

The original AdaBoost algorithm [Freund and Schapire 95], is a supervised learning algorithm designed to find a binary classifier that discriminates between positive and negative examples. The input to the learning algorithm is a set of training examples (x_n, y_n), $n = 1, ..., N$, where each x_n is an example and y_n is a boolean value indicating whether x_n is a positive or negative example. AdaBoost boosts the classification performance of a simple learning algorithm by combining a collection

of weak classifiers into a stronger classifier. Each weak classifier is given as a function $h_j(x)$ which returns a boolean value. The output is 1, if x is classified as a positive example and 0 otherwise.

Whereas the weak classifiers only need to be slightly better than a random guessing, the combined strong classifier typically produces good results. To boost a weak classifier, it is required to solve a sequence of learning problems. After each round of learning, the examples are reweighted in order to increase the importance of those which were incorrectly classified by the previous weak classifier. The final strong classifier takes the form of a perceptron, a weighted combination of weak classifiers followed by a threshold. Large weights are assigned to good classification functions whereas poor functions have small weights. A variant of the AdaBoost algorithm has been presented in [Viola and Jones 01]. This variant restricts the weak classifiers to depend on single-valued features f_j only. This allows the algorithm to apply the feature selection process by finding each round the best feature that discriminates between the positive and negative examples. Each weak classifier has the form:

$$h_j(x) = \begin{cases} 1 & if\, p_j f_j(x) < p_j \Theta_j \\ 0 & otherwise \end{cases} \tag{6.1}$$

where Θ_j is a threshold and p_j is either -1 or 1 and thus representing the direction of the inequality. The algorithm determines for each weak classifier $h_j(x)$ the optimal values for Θ_j and p_j, such that the number of misclassified training examples is minimized:

$$(p_j, \Theta_j) = \underset{(p_i,\Theta_i)}{argmin} \sum_{n=1}^{N} |h_i(x_n) - y_n| \tag{6.2}$$

To achieve this, it considers all possible combinations of both p_j and Θ_j, whose number is limited since only a finite number of training examples is given. To be specific, for each feature, the examples are sorted based on feature value. The Adaboost optimal threshold for that feature can then be computed in a single pass

over this sorted list. Note that the weak classifier is not necessarily a simple decision rule like the one above, but can rather be any type of classifier in the machine learning literature. Algorithm 4 outlines the Adaboost technique.

Algorithm 4 The Adaboost algorithm according to [Viola and Jones 01]

- Input: set of examples $(x_1, y_1), ..., (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}\, for\, y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For t = 1,...,T:
  1. Normalize the weights, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
  2. Select the best weak classifier with respect to the weighted error $\epsilon_j = \sum_n^N w_{t,n} \left| h_j(x_n) - y_n \right|$
  3. Choose the classifier h_j with the lowest error ϵ_j and set $(h_t, \epsilon_t) = (h_j, \epsilon_j)$.
  4. Update the weights: $w_{t+1,n} = w_{t,n}\beta_t^{1-e_n}$, $where\, \beta_t = \frac{\epsilon_t}{1-\epsilon_t}$ and $e_n = 0$, if example x_n is classified correctly by h_t and 1, otherwise.
- The final strong classifier is given by: $C(x) = \begin{cases} 1 & if\ \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases}$, where $\alpha_t = log \frac{1}{\beta_t}$

6.3 Subunits as weak learners

6.3.1 Subunits extraction

In the last chapter we discussed the segmentation of sign video x into a set of frame sequences that we called 'subunits', where every subunit represents a motion pattern of the hand that covers a sequence of consecutive frames with interrelated spatio-temporal features. Given a training set of N sample videos for sign x, the subunit segmentation algorithm is applied on all sample videos and we get a set of subunits:

$$S_x = \{su_{1,1}, su_{1,2}, ..., su_{1,i}, su_{2,1}, su_{2,2}, ..., su_{2,j}, ..., su_{N,1}, su_{N,2}, ...su_{N,k}\}$$

where $su_{n,l}$ is the l^{th} subunit in sample n. Every $su_{n,l}$ is then modelled using the 6 global spatio-temporal features we introduced in the last chapter (*Hand motion speed, Hand motion direction code, Distance between hand position and trajectory centroid, Orientation angle of vector from hand location to trajectory centroid, Distance between hand and head, Orientation angle of vector from hand to head*):

$$su_{n,l} = \{f_1^{n,l}, f_2^{n,l}, ..., f_m^{n,l}\}$$

where $f_i^{n,l}$ represents feature vector of frame i in subunit $su_{n,l}$.

6.3.2 Subunits clustering

Hierarchical clustering is a way to investigate grouping in our data, simultaneously over a variety of scales, by creating a cluster tree. The tree is not a single set of clusters, but rather a multilevel hierarchy, where clusters at one level are joined as clusters at the next higher level. This allows us to decide what level or scale of clustering is most appropriate for our application. To perform hierarchical cluster analysis on our data set, we have to follow this procedure:

1. **Find the similarity or dissimilarity between every pair of objects in the data set.** In this step, we calculate the distance between objects using a distance metric. In our case we adopted the DTW metric as discussed in the last chapter.

2. **Group the objects into a binary, hierarchical cluster tree.** In this step, we link pairs of objects that are in close proximity using a linkage algorithm. There are different linkage algorithms. These linkage algorithms are based on different ways of measuring the distance between two clusters of objects. If n_r is the number of objects in cluster r and n_s is the number of objects in cluster s, and x_{ri} is the i^{th} object in cluster r, we adopt the average linkage algorithm which uses the average distance between all pairs of objects in cluster r and cluster s:

$$d(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} dist(x_{ri}, x_{sj})$$

where $dist(x_{ri}, x_{sj}) = DTW(x_{ri}, x_{sj})$. The linkage algorithm uses the distance information generated in step 1 to determine the proximity of objects to each other. As objects are paired into binary clusters, the newly formed clusters are grouped into larger clusters until a hierarchical tree is formed.

3. **Determine where to cut the hierarchical tree into clusters.** In this step, we prune branches off the bottom of the hierarchical tree, and assign all the objects below each cut to a single cluster. This creates a partition of the data. In general, if we know the number of clusters we need, we can easily know where we have to prune the tree from. In our case, we know from the subunit segmentation algorithm the number of subunits that were generated from every signing video sample. We calculate the average number λ of subunits segmented from all the samples of sign x and use this λ as a threshold to prune the hierarchical tree and get out the subunit clusters for sign x.

4. The final step in the clustering task is to construct a codebook for the different subunit clusters. For every cluster, we find the medoid subunit (the subunit which have the minimum average distance to all the other subunits in the same cluster).

$$medoid_j = min(\sum_{i=1}^{n_s} DTW(x_i, x_j)), j \in (1, ..., n_s)$$

The set of cluster medoids form the codebook entries of sign x. Fig. 6.1 and 6.2 demonstrate two examples of codebook construction. Subunits from sign samples where extracted and clustered as shown in the dendrograms. Then the medoid subunits where identified. Fig. 6.3 shows an example of a failed example for a complex motion pattern (sign of "hungry") where the hand moves slowly in small region towards the mouth back and forth. The subunit segmentation algorithm

detected 2 medoids, the first one is a long subunit where the small subunits were merged together, and another short true subunit. In reality, for this sign, we should get 3 medoid subunits: one from the beginning of the hand motion till the mouth, one from the mouth outwards, and one towards the mouth.

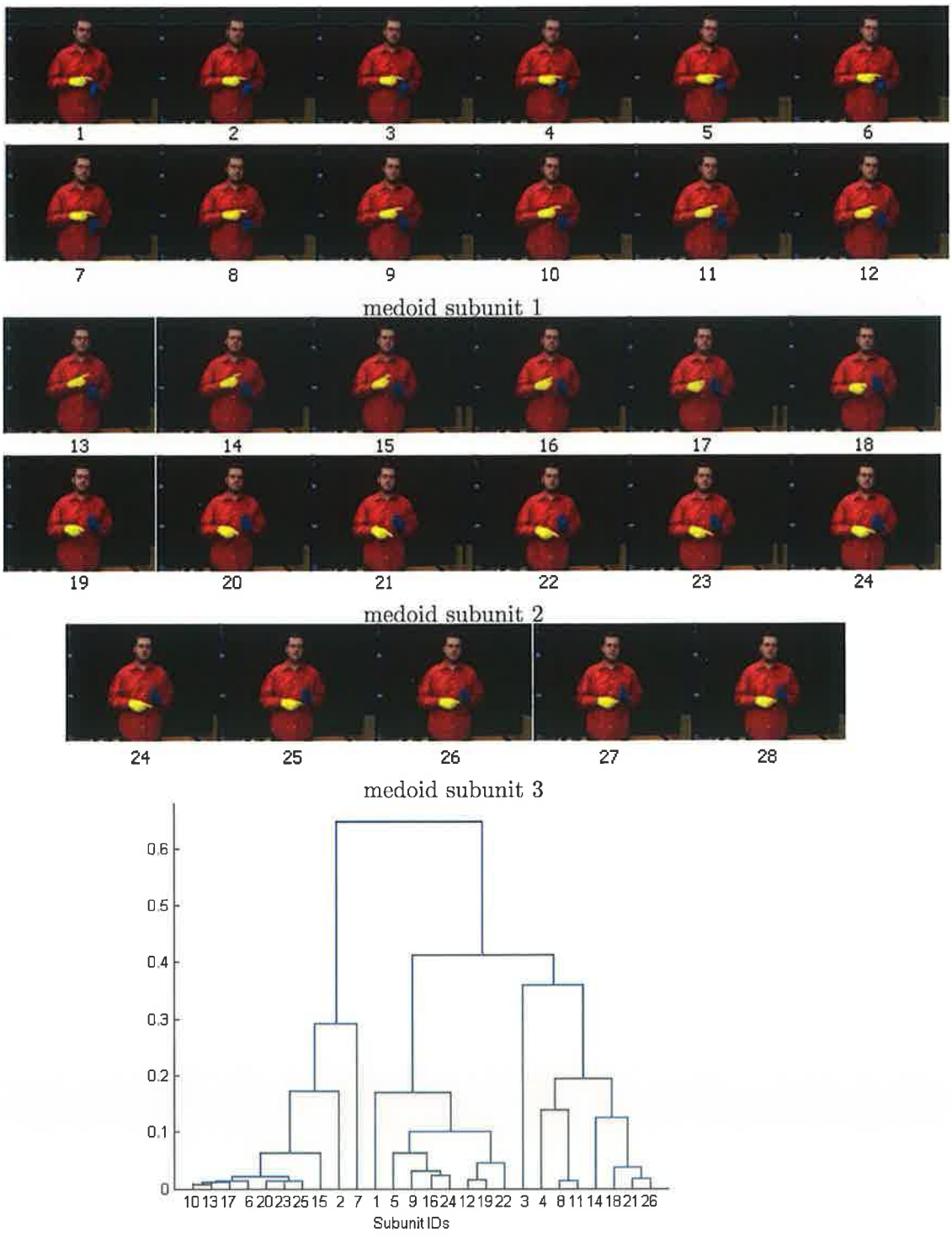


Figure 6.1: An example of a codebook for a real sign with 3 entries, with the dendrogram of the subunits

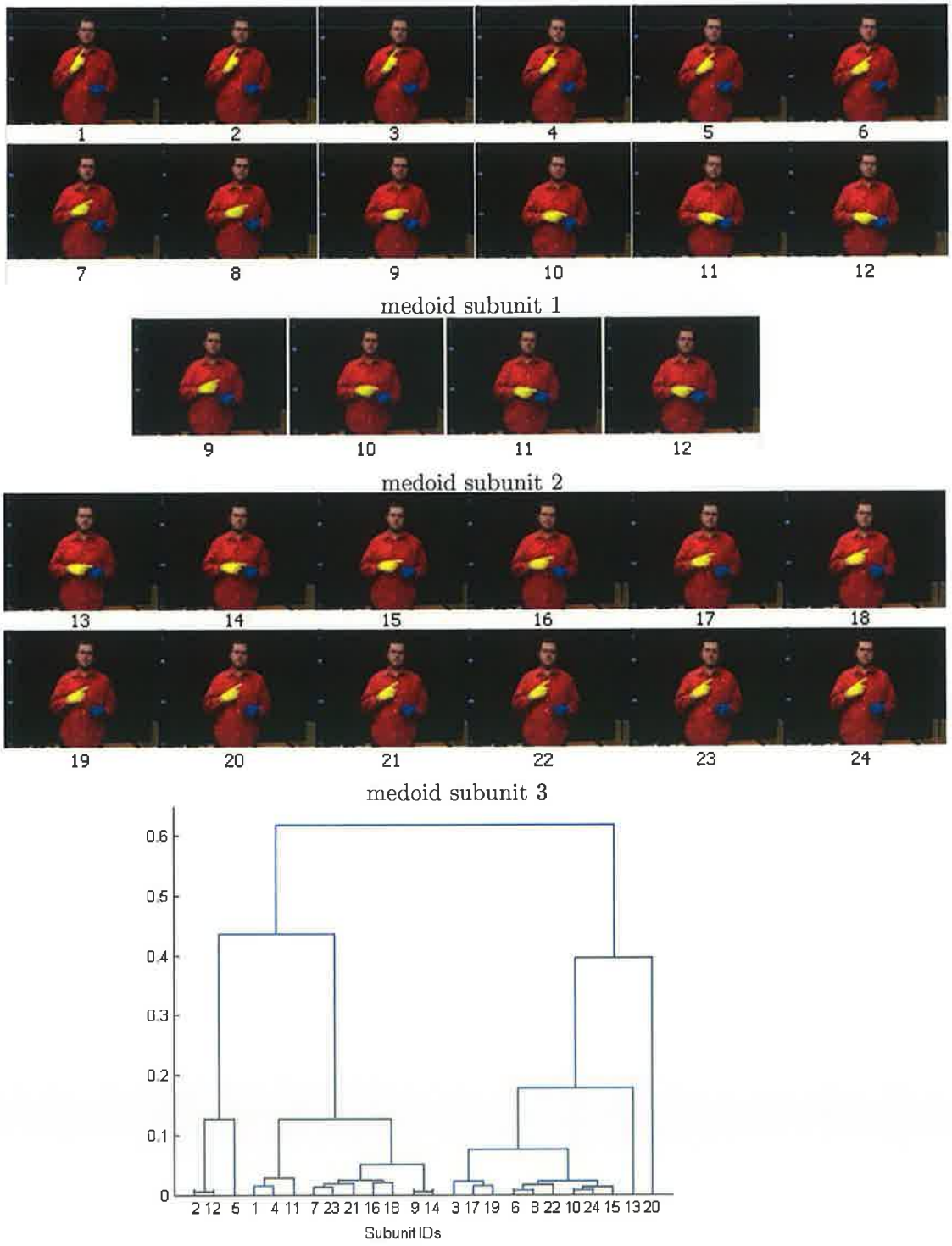


Figure 6.2: An example of a codebook for a real sign with 3 entries, with the dendrogram of the subunits

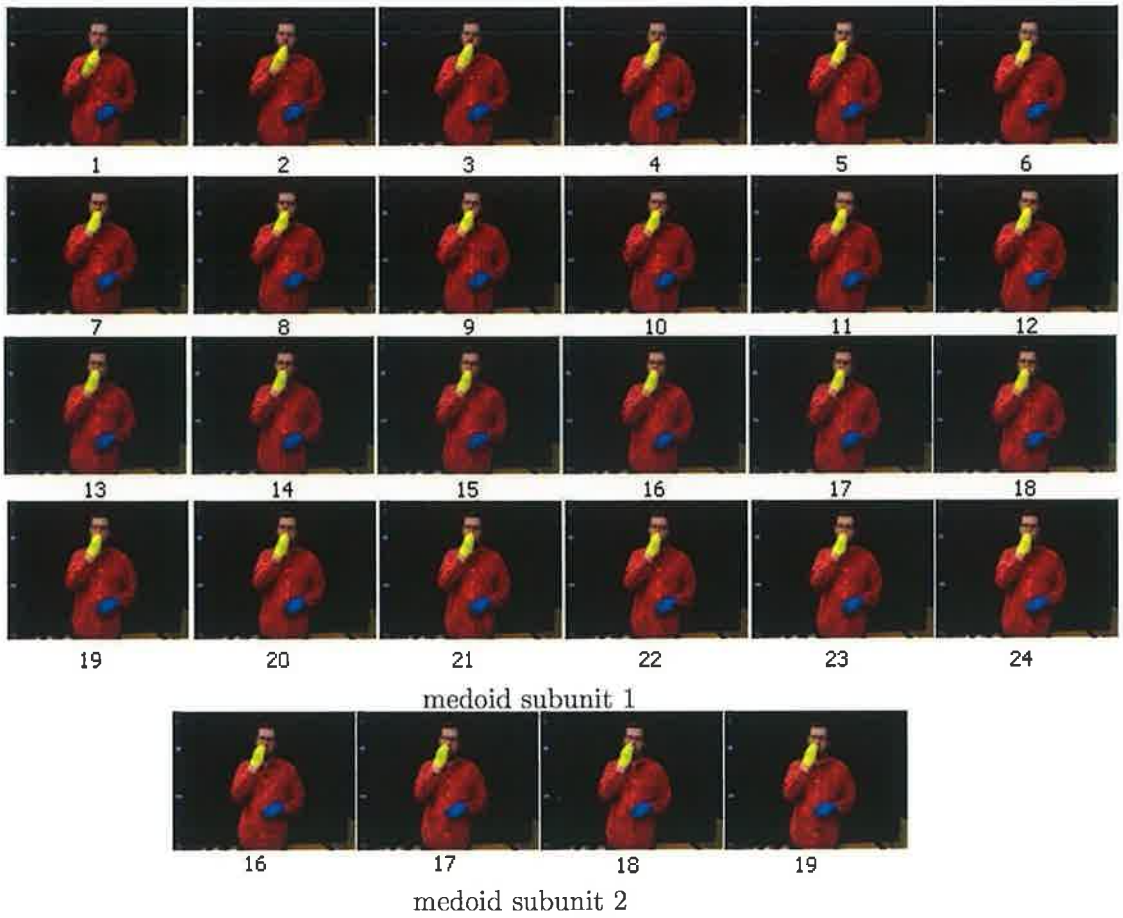


Figure 6.3: An example of failed subunit segmentation in a complex motion pattern

6.3.3 Constructing weak classifiers

In this section we will discuss the construction of weak classifiers using the combinations of subunits and features. Here we introduce a new feature to represent the shape of the hand based on the hand boundary (Fourier Descriptors) and hand region (Moments).

6.3.3.1 Fourier descriptors

Basically, Fourier Descriptors (FD) is obtained by applying Fourier transform (FT) on a shape signature function derived from shape boundary coordinates $\{(x(t), y(t)),\ t = 0, 1, ...N-1\}$. The centroid distance function is a popular shape signature function, which is given by the distance of the contour points from the centroid (x_c, y_c) of the

shape:

$$s(t) = ([x(t) - x_c]^2 + [y(t) - y_c]^2)^{1/2},\ t = 0, 1, 2, ...N - 1 \tag{6.3}$$

where $x_c = \frac{1}{N}\sum_{t=0}^{N-1} x(t)$, $y_c = \frac{1}{N}\sum_{t=0}^{N-1} y(t)$. s(t) is invariant to translation. One dimensional FT is then applied on *s(t)* to obtain the Fourier transformed coefficients:

$$a_n = \frac{1}{N}\sum_{t=0}^{N-1} r(t)\exp(\frac{-j2\Pi nt}{N}),\ n = 0, 1, 2, ...N - 1 \tag{6.4}$$

Ignoring the phase information of a_n and using only the magnitudes $|a_n|$ achieves rotation invariance, while scale invariance can be achieved by dividing the magnitudes by the DC component, i.e. $|a_0|$. FDs are basically the normalized Fourier coefficients [Zhang and Lu 01]. Global shape features are captured with the first few low frequency terms, while higher frequency terms capture finer details of the shape.

6.3.3.2 Moment Invariants

One of the most popular region-based image invariants [Pakchalakis and Lee 99, Reeves et al. 88] is the Moment invariants. Based on regular moments, a set of invariants using nonlinear combinations were first introduced by Hu back in 1961 [Hu 62]. Regular moments are defined as:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x, y)dxdy \quad p, q = 0, 1, 2, ... \tag{6.5}$$

where m_{pq} is the $(p+q)^{th}$ order moment of the continuous image function $f(x, y)$. The central moments of $f(x, y)$ are defined as:

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y)dxdy \tag{6.6}$$

Where $\bar{x} = \frac{m_{10}}{m_{00}}$ and $\bar{y} = \frac{m_{01}}{m_{00}}$, which are the centroid of the image. The central moments obviously are invariant to image translations. To obtain scale invariance, we let $\acute{f}(\acute{x}, \acute{y})$ represent the image $f(x, y)$ after scaling the image by $s_x = s_y = \alpha$, so

$\acute{f}(\acute{x},\acute{y}) = f(\alpha x, \alpha y) = f(x,y)$, and $\acute{x} = \alpha x$, $\acute{y} = \alpha y$, then we can easily prove:

$$\acute{m_{pq}} = \alpha^{p+q+2} m_{pq}$$

Similarly,

$$\acute{\mu_{pq}} = \alpha^{p+q+2}\mu_{pq}\ ,\ \acute{\mu_{00}} = \alpha^2\mu_{00}$$

We can define normalized central moments as:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}},\ \gamma = \frac{p+q+2}{2},\ p+q = 2,3,... \tag{6.7}$$

η_{pq} is invariant to changes of scale because:

$$\acute{\eta_{pq}} = \frac{\acute{\mu_{pq}}}{\acute{\mu_{00}^{\gamma}}} = \frac{\alpha^{p+q+2}\mu_{pq}}{\alpha^{2\gamma}\mu_{00}^{\gamma}} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} = \eta_{pq}$$

Based on normalized central moments, Hu introduced seven moment invariants:

$$\begin{aligned}
\emptyset_1 &= \eta_{20} + \eta_{02} \\
\emptyset_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\emptyset_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\emptyset_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\emptyset_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\emptyset_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\emptyset_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad (3\eta_{21} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned}$$

Hu's seven moment invariants have the nice properties of being invariant under image scaling, translation and rotation. However, their disadvantage is that more

higher order moments is quite hard to compute and reconstructing the shape from the moments are also hard. The first lower moments can capture only global shape properties.

6.3.3.3 Dynamic Time Warping (DTW)

At this stage we have a codebook of subunits. As we don't yet know what subunits are more important (informative) for a sign to be recognized, nor what features are more important to discriminate this sign from other signs, we try here to construct a set of weak classifiers from the codebook entries, each with a different set of feature combinations.

Using a standard Boosting framework, we can learn the informative subunits/features combinations and construct a strong sign classifier for every sign in our vocabulary. Given a codebook $B_x = \{S_1, S_2, ..., S_\lambda\}$ for sign x consisting of λ subunit entries, and feature set $F = \{F_1, F_2, ..., F_7\}$, where the first 6 features ($F_1 ... F_6$) correspond to the 6 global spatio-temporal features mentioned in subsection 6.3.1 and F_7 corresponds to the hand shape feature. F_7 is calculated using the Fourier descriptors (FD) and Hu moments with fixed feature vector size (we used 32 total coefficients, 25 Fourier coefficients and 7 Hu moments).

We can construct a set of weak classifiers using different combinations of these 7 features calculated for every S_i in B_x. Let $W_x = \{w_1, w_2, ...w_7\}$, where W_x is the set of weak classifiers constructed for sign x, and w_i is the set of weak classifiers constructed using i features. So w_4 is the set of weak classifiers constructed using all possible combinations of 4 features from the set F. Also note that these combinations of features are calculated for all the subunits S_i in B_x. In general, we store the information of every weak classifier in a structure such that:

$$w_i = \{(fv, FID, SUID)_1, (fv, FID, SUID)_2, ...(fv, FID, SUID)_y\}$$

where, fv is the feature vector, FID is the ID of the features calculated which can be a string of digits in the range of 1 to 7 and of length i, and $SUID$ is the ID of

the subunit S_i which can be any number between 1 to λ. We want a classifier to fire ($h_i(X) = 1$) if the distance of h_i to a sign video is below a certain threshold:

$$h_i(X) = \begin{cases} 1 & if\, D(h_i, X) < \Theta_{h_i} \\ 0 & otherwise \end{cases} \tag{6.8}$$

$$D(h_i, X) = min(DTW(h_i, x_j)),\ j \in (1, 2, ..., M) \tag{6.9}$$

where the sign video X consists of subunits $x_1, x_2, ..., x_M$, h_i is the weak classifier, and DTW is the dynamic time warping distance metric. In the Adaboost framework, every iteration we select the best weak classifier h_i that minimizes the overall error over the training samples using their current weights. To determine the best threshold Θ_{h_i} for every weak classifier, we sort the distances D between the classifier and the training samples and in a single loop over them we can use each of them as a threshold and calculate the total error of the training samples. Finally, we pick the weak classifier and its corresponding threshold that resulted in the minimum total error.

6.3.3.4 Hidden Markov Model (HMM)

HMMs are famous for their applications in temporal pattern recognition such as speech [Huang et al 90], handwritting [Veltman and Prasad 94], and gesture recognition [Yoon et al. 99]. A HMM has the ability to find the most likely sequence of states that may have produced a given sequence of observations. Formally, the elements of a Hidden Markov Model are defined using the following declarations [Jose and Luis 04]:

- set of observation strings $O = O_1, ...O_t, ...O_T$, $where\, t = 1, ..., T$
- set of N states $S_1, ..., S_N$
- set of k discrete symbols from a finite alphabet $V_1, ..., V_k$
- a state transition matrix $A = \{a_{ij}\}$, where a_{ij} is the transition probability from

state S_i to S_j

- an observation probability matrix $B = \{b_{jk}\}$, where b_{jk} is the probability of generating symbol V_k from state S_j
- the initial probability distribution for the states $\prod = \pi_j, j = 1, 2, ...N$, $\pi_j = Pr(S_j \, at \, t = 1)$

The complete parameter set of an HMM can be expressed compactly as $\lambda = (A, B, \pi)$. For every class c where $c \in \{1, 2, ...N\}$, and N is the maximum number of classes in our problem, given a set of training sequences, one HMM model λ can be trained, and then given a testing input x to be classified, we select the class c with highest probability $Pr(x|\lambda_c)$. The three basic problems must be solved for the application of HMM: classification, decoding, and training. These problems are in general solved using the forward algorithm, Viterbi algorithm, and the Baum-Welch algorithm. We used the classical left-right (basic) states structure, which is typical for motion ordered paths.

We tried here to use HMMs as weak classifiers in the same manner of using the DTW as in the last section. In section 6.3.2 we discussed the construction of a hierarchical tree of subunits which we prune its branches to get a set of subunit clusters. Given the feature set $F = \{F_1, F_2, ..., F_7\}$, and the subunits in every cluster, we train one HMM model for each possible feature combination. Let $C = \{C_1, C_2, ..., C_r\}$ be the set of clusters for sign x, and $F_{comb} = \{F_{comb1}, F_{comb2}, ..., F_{comb127}\}$ be the set of all possible feature combinations, then we train a set of HMM models:

$$\begin{aligned} HMM_{models} = \{ & HMM_1^{c_1}, ..., HMM_{127}^{c_1}, \\ & HMM_1^{c_2}, ..., HMM_{127}^{c_2}, \\ & ..., HMM_1^{c_r}, ..., HMM_{127}^{c_r}\} \end{aligned}$$

where HMM_{models} is the set of all trained HMM models, and $HMM_n^{c_r}$ is the HMM model trained on sample subunits in cluster r using feature combination n, where

$n \in \{F_{comb1}, ...F_{comb127}\}$. In comparison to the DTW weak classifiers discussed above, we want a classifier to fire ($h_i(X, HMM_n^{c_r}) = 1$) if the probability of X given the model $HMM_n^{c_r}$ is above a certain threshold:

$$h_i(X, HMM_n^{c_r}) = \begin{cases} 1 & if\ P(X, HMM_n^{c_r}) > \Theta_{h_i} \\ 0 & otherwise \end{cases} \tag{6.10}$$

$$P(X, HMM_n^{c_r}) = max(Pr(x_j|HMM_n^{c_r}))\,,\ j \in (1, 2, ..., M) \tag{6.11}$$

where the sign video X consists of subunits $x_1, x_2, ..., x_M$, h_i is the weak classifier, and P is the maximum probability between the HMM model $HMM_n^{c_r}$ and the subunits x_j.

6.4 Joint-Adaboost learning

Much recent research on object category recognition has proposed models and learning methods where a new model is learnt individually and independently for each object category [Opelt-PAMI 06]. However, such approaches seem unlikely to scale up to the detection of a large number of different object classes because each classifier is trained and run independently. Another promising approach by [Torralba et al. 04] has been proposed to explicitly learn to share features across multiple object classes (classifiers) [Opelt-CVPR 06]. The basic idea is an extension of the Adaboost algorithm. Rather than training C binary classifiers independently, they train them jointly. The result is that many fewer features are needed to achieve a desired level of performance than if the classifiers were trained independently. This results in a faster classifier (since there are fewer features to compute) and one which works better (since the features are fit to larger shared data sets).

It has been shown in [Torralba et al. 04] that although class-specific features achieve a more compact representation for a single category, the whole set of shared features is able to provide more efficient and robust representations when the system is trained to detect many object classes than the set of class-specific features. One

drawback of class-specific features is that they might be too finely tuned, preventing them from being useful for other object classes.

The learning algorithm is an iterative procedure that adds one feature at each step. Each feature is found by selecting, from all possible class groupings and features, the combination that provides the largest reduction of the multiclass error rate. The feature added in the first iteration will have to be as informative as possible for as many objects as possible, since only the object classes for which the feature is used will have their error rate reduced. In the second iteration the same selection process is repeated but with a larger weight given to the training examples that were incorrectly classified by the previous feature. This process is iterated until a desired level of performance is reached or until a fixed number of iterations T. The algorithm has the flexibility to select class-specific features if it finds that the different object classes do not share any property.

6.4.1 Sharing weak classifiers

Motivated from the related work of joint learning in object recognition and from our observations that different subunits can be shared between signs, we are proposing here to apply joint-Adaboost learning to share weak classifiers across different sign classes. Our aim is two-fold. Firstly to increase the overall performance, as now the weak classifiers are optimized to reduce the total error over all the classes at every iteration and so focus on more general features instead of class-specific features. Secondly, to reduce the total number of weak classifiers required compared to independently learning each class, which helps in constructing a faster, stronger classifier. The joint boosting algorithm is summarized in algorithm 5. We adopted the joint boosting algorithm proposed by [Torralba et al. 04]. The main difference between the two algorithms is the weak classifiers, as here we use the DTW-based metric to measure the distance between the classifier (modelled by the corresponding subunit/feature combination) and the input sign.

The basic idea of the algorithm is that at each boosting round, we examine various subsets $S_n \subseteq C$, and try to fit a weak classifier to discriminate that subset

Algorithm 5 Joint Boosting with DTW-based weak classifiers.

- Input: set of examples $(x_1^c, y_1^c), ..., (x_i^c, y_i^c)$ where $y_i^c \in \{-1, 1\}$ for negative and positive examples respectively, $i = 1...N$,
  $c = 1...C$.
- Initialize weights $w_i^c = 1$, $H(x, c) = 0$.
- For t = 1,...,T:
  - **(a)** Repeat for $n = 1, 2, ..., 2^c - 1$
    1. Find the best shared weak classifier h_t w.r.t. the weights $w_i^{s_n}$:

$$h_t(x, c) = \begin{cases} D(h_t, x) < th_{h_t} & if\ c \in S_n \\ k^c & if\ c \notin S_n \end{cases}$$

    1. Evaluate error:

$$E_n = \sum_c^C \sum_{i=1}^N w_i^c (y_i^c - h_t(x_i, c))^2$$

  - **(b)** Find the best sharing by selecting $n = arg\,min_n\, E_n$, and pick the corresponding shared h_t and S_n
  - **(c)** Update:
    $H(x, c) = H(x, c) + h_t(x, c)$
    $w_i^c = w_i^c e^{-y_i^c h_t(x,c)}$

from the other classes $c \notin S_n$, we do this by considering all the classes in the subset as "positive" examples, and examples from other classes as "negative". This gives us a binary classification problem which can be solved in a manner similar to binary Adaboost outlined above. We then pick the subset that maximally reduces the error on the weighted training set for all the classes. The corresponding best shared weak classifier $h_t(x, c)$ is then added to the strong classifiers $H(x, c)$ for all the classes $c \in S$, and the weights of all the training set examples are updated. For classes that do not share this weak classifier, the function $h_t(x, c)$ is constant k^c different for each class. This constant prevents sharing classifiers due to asymmetry between the number of positive and negative examples for each class and is defined as:

$$k^c = \frac{\sum_i w_i^c y_i^c}{\sum_i w_i^c},\ c \notin S_n$$

Thus each weak classifier holds 2 parameters (feature vector values fv , threshold th_{h_t}) for the positive class, and $C - |S_n|$ parameters for the negative class, and one parameter specifying which subset S_n was chosen. As we don't know which subset is the best for sharing, we have to search over all $2^c - 1$ subsets each round which can be very slow. So we followed [Torralba et al. 04] in their greedy strategy. This starts with selecting the class that achieves alone the lowest overall error, then incrementally adds the next class with the lowest training error. After we finish adding all the classes, we select the subset that gives the best error reduction.

6.5 Experimental results

We tested our proposed subunit-based SL recognition algorithm on 20 different signs (see table 6.1) that were collected from a database from other research groups. Every sign is performed 10 times (total 200 video samples) with variations in speed and the way of performing. Some of the sign samples are presented in figures 6.4,6.5, 6.6, and 6.7. In theory, we have 4 different case scenarios for modelling the weak classifiers in the Adaboost algorithm:

- single subunit / single feature.
- single subunit / all possible feature combinations.
- all possible subunits combinations (or fixed number of subunits) / single feature.
- all possible subunits combinations (or fixed number of subunits) / all possible feature combinations.

In these experiments, we chose to test with only the first 2 scenarios as the last two scenarios are very time-consuming. The computation cost is very heavy to select at every iteration the best weak classifier from such huge pool of weak classifier candidates. For example, if we imagined that a sign consists of 3 subunits, and for the 7 features there is 127 different combinations, then this means that for the 3

subunits, we need $127 * 3 + 127^2 * 3 + 127^3 = 2097151$ weak classifiers for this sign alone using the 4^{th} scenario. However, the third scenario is better as it gives a total of $7*3+7^2*3+7^3 = 511$ weak classifiers and the second scenario gives $127*3 = 381$ weak classifiers. Thus our experiments are based on using the second scenario. Note that the first scenario is a special case of the second option as well.

We experimented our proposed algorithm on different numbers of positive and negative training samples. We begin training with 1 positive sample and 19 negative samples (one from each class) selected randomly. This experiment is repeated 10 times for every sign class. Then sequentially, we repeat the process by using 2 positive samples and 38 negative samples (two from each class) till we reached 9 positive samples and 171 negative samples. The rest of the 10 samples in each class is used for testing. We used T=20 cycles in the Adaboost algorithm to combine 20 weak classifiers and construct a strong classifier for every sign class in independent training, while in joint training we used T=60 to train the whole set of the 20 classes, thus the number of weak classifiers needed is reduced from 400 to 60 which is part from our motivation in using joint training.

Two types of experiments were done to test the performance of the proposed approach for both types of training: independently where we evaluate both types of weak classifiers: DTW-based and HMM-based, and jointly. In general, first, we test the performance of each classifier independently by recording its performance using the recall, precision, specificity, and F-score measures. Then to we test the overall performance in a multiclass test by selecting the best classification across all the classes for each testing video sample and recording the minimum, maximum and average error and accuracy.

We used the following metrics in our evaluations:

$$recall = \frac{N_c}{N_{g+}} \tag{6.12}$$

$$precision = \frac{N_c}{N_d} \tag{6.13}$$

$$F - score = \frac{2 * recall * precision}{recall + precision} \tag{6.14}$$

$$specificity = \frac{T_n}{N_{g-}} \tag{6.15}$$

Where N_c is the number of true positives that were detected by the classifier, N_d is the total number of detected signs by the classifier, N_{g+}, $N_{g-}$are the total number of true positives and true negatives in the ground truth respectively, and T_n is the number of true negatives that were rejected by the classifier. F-score is used as a single performance measure to combine recall and precision, while specificity is used to measure how good the classifier in rejecting the negative samples.

In the case of multiclass test, for each testing sample we apply all the trained classifiers. If only one classifier fired then we select this class to be the result, otherwise when more than one classifier fires, we select the class with maximum distance from its threshold which can represent the confidence of this class.

Sign	1	2	3	4	5
Name	about	apple	ask	bat	before
Sign	6	7	8	9	10
Name	believe	best	black	bottle	boy
Sign	11	12	13	14	15
Name	brown	busy	but	can	can't
Sign	16	17	18	19	20
Name	clever	coffee	confidence	deaf	don't_know

Table 6.1: BSL Signs used in our evaluation tests

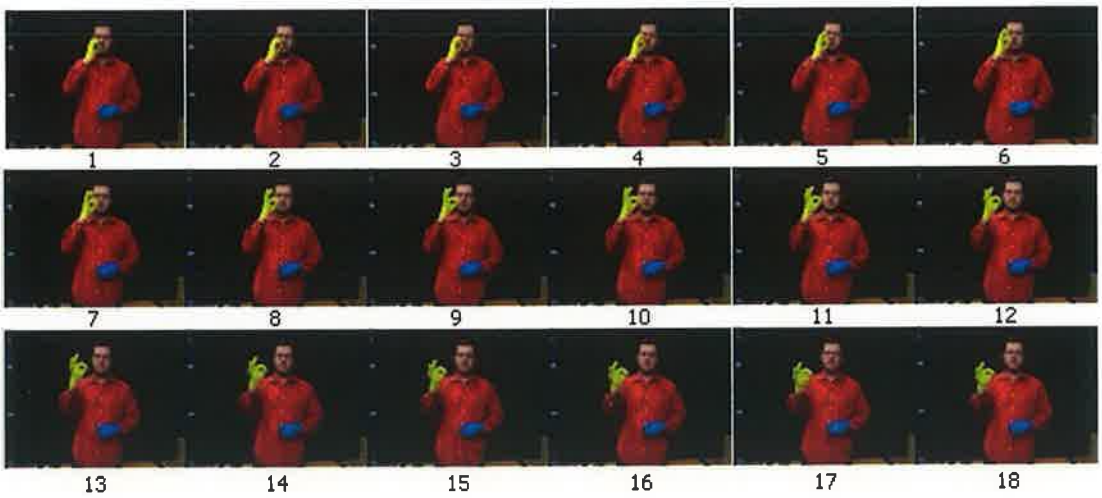


Figure 6.4: A sample of the sign "ask"

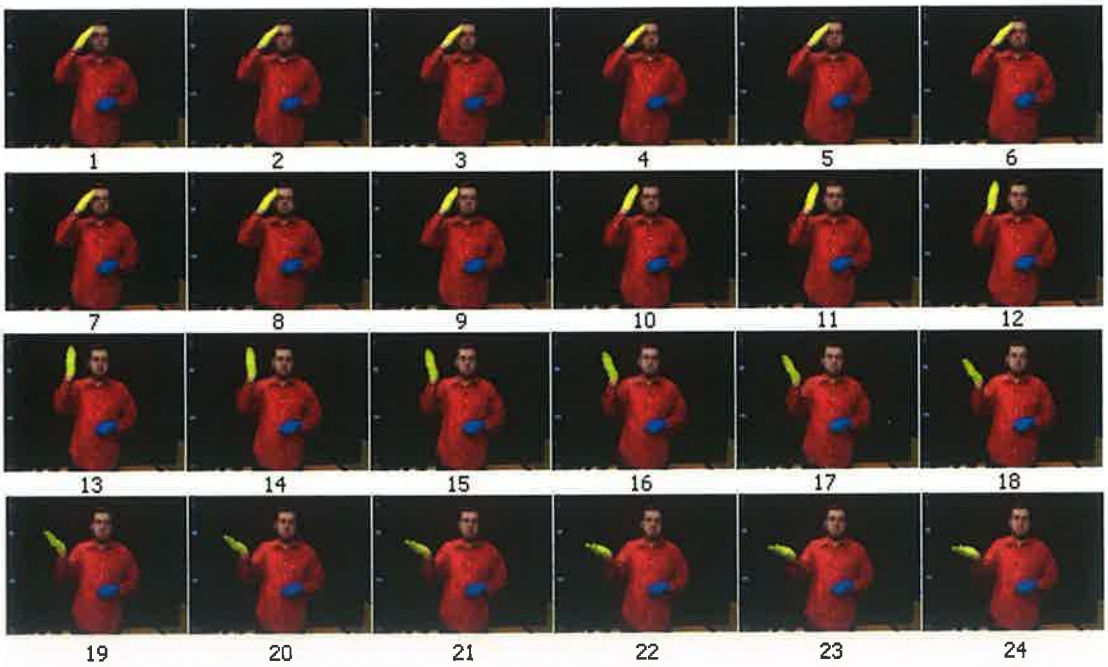


Figure 6.5: A sample of the sign "don't know"

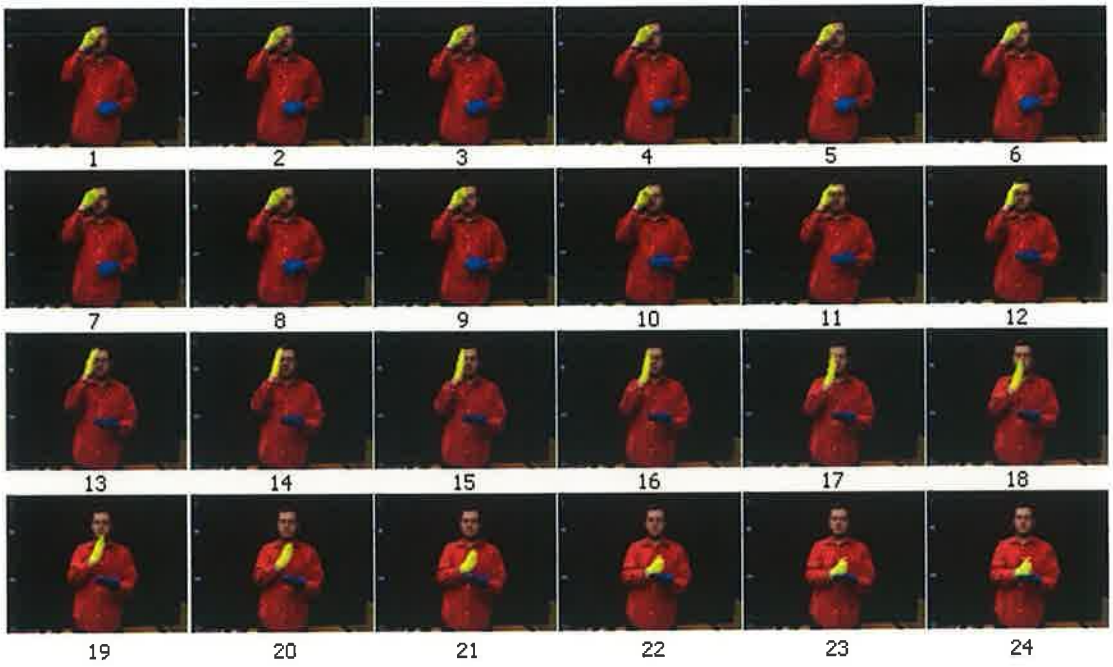


Figure 6.6: A sample of the sign "believe"

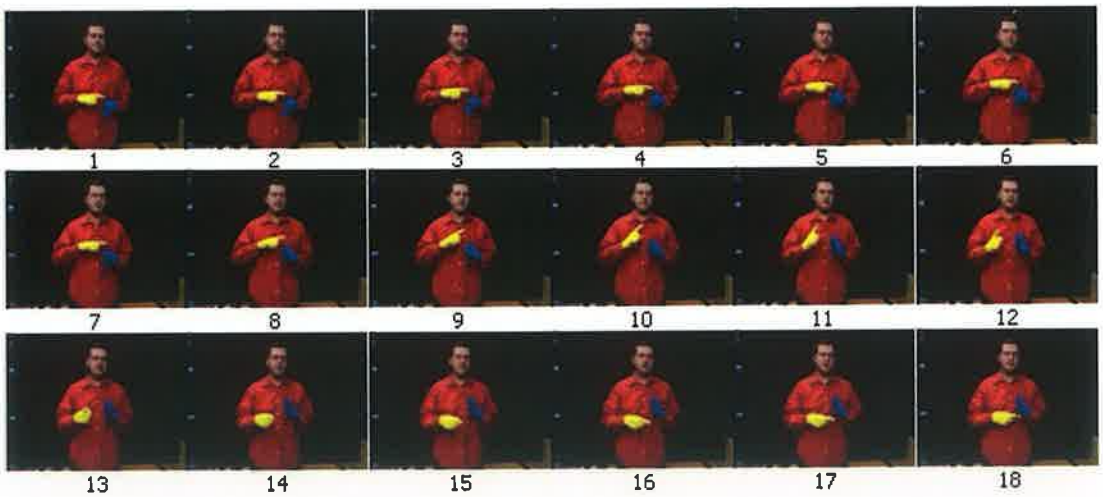


Figure 6.7: A sample of the sign "about"

6.5.1 Independent training

6.5.1.1 DTW-based weak classifiers

Table 6.2 shows the average performance results when each classifier is applied independently on the testing dataset. The values in the table are the average performance

of all the 20 classes with different training samples. Maximum F-score was achieved using 7 training samples with score 0.857 and specificity 0.9997. Table 6.3 shows the results of testing the classifiers on multiclass test. Using only 1 training sample gives about 60% average accuracy with the maximum reaches about 72%. Using 50% of the data set for training (5 samples) gives around 87% average accuracy with maximum 92%, while 9 training samples give 90% accuracy on average with maximum 95%.

Training samples	Recall	Precision	Specificity	F-score
1	0.2922	0.7537	0.9980	0.4109
2	0.5275	0.9489	0.9988	0.6745
3	0.5950	0.9558	0.9992	0.7287
4	0.6758	**0.9715**	0.9991	0.7922
5	0.735	0.9679	0.9985	0.8318
6	0.7225	0.96933	0.9992	0.8238
7	0.7750	0.9633	**0.9997**	**0.8570**
8	**0.7925**	0.9275	0.9992	0.8526
9	0.78	0.775	0.9994	0.7774

Table 6.2: Average statistical results of individual classifiers using independent training (DTW base classifiers)

Training samples	Min. error	Max. error	Mean error	Mean accuracy
1	0.277	0.472	0.400	0.600
2	0.156	0.268	0.213	0.786
3	0.121	0.242	0.197	0.802
4	0.075	0.183	0.133	0.866
5	0.08	0.16	0.126	0.874
6	0.05	0.175	0.112	0.887
7	0.05	0.166	0.098	0.901
8	0.025	0.15	0.087	**0.912**
9	0.05	0.2	0.1	0.9

Table 6.3: Statistical results of multiclass classification using DTW-based weak classifiers.

6.5.1.2 HMM-based weak classifiers

Table 6.4 shows the results of using HMM as weak classifiers. The results are very close to DTW-based classifiers as the maximum F-score 0.85 was achieved using 8 training samples together with maximum recall and precision 0.837 and 0.90 respec-

tively. However, the results of table 6.5 show the performance in multiclass test to be lower than the corresponding DTW results as the average accuracy reached 53% using 1 training sample with a maximum of about 58%, and using 50% of the data set for training (5 samples) gives around 81% average accuracy with maximum 87%, while 9 training samples give 88% accuracy on average with maximum 95%.

Training samples	Recall	Precision	Specificity	F-score
1	0.3766	0.5339	0.9796	0.3866
2	0.5556	0.6689	0.9825	0.5630
3	0.6435	0.7610	0.9866	0.6635
4	0.6474	0.7728	0.9890	0.6759
5	0.739	0.8413	0.9915	0.7614
6	0.7612	0.8458	0.9917	0.7808
7	0.7750	0.8830	0.9943	0.8028
8	**0.8375**	**0.9091**	0.9951	**0.8505**
9	0.83	0.8016	**0.9963**	0.8108

Table 6.4: Average statistical results of individual classifiers using independent training (HMM base classifiers)

Training samples	Min. error	Max. error	Mean error	Mean accuracy
1	0.416	0.516	0.468	0.531
2	0.243	0.375	0.309	0.690
3	0.128	0.285	0.228	0.771
4	0.175	0.308	0.24	0.76
5	0.13	0.26	0.191	0.809
6	0.1	0.237	0.167	0.832
7	0.066	0.233	0.156	0.843
8	0.05	0.2	0.117	**0.882**
9	0.05	0.2	0.12	0.88

Table 6.5: Statistical results of multiclass classification using HMM-based weak classifiers

6.5.1.3 Weak classifiers comparison

In a comparison between HMM and DTW weak classifiers for our problem, we plotted the performance of recall, precision, F-score and multiclass classification and tried to decide which is better. It can be shown in Fig. 6.8 that the recall performance for both of them is nearly the same or very close with a higher average of 8.6% for HMM than DTW. However, in the other three measures (Fig. 6.9, 6.10, 6.11), the DTW is

better than the HMMs by an average 17%, 8% and 8% for the precision, F-score, and classification respectively. Specificity comparisons revealed that both are almost the same with about 1% higher average for DTW than HMMs. We conclude from these results that DTW weak classifiers could be more suitable for training the Adaboost in our problem, thus we use it in our joint training experiments in the following section.

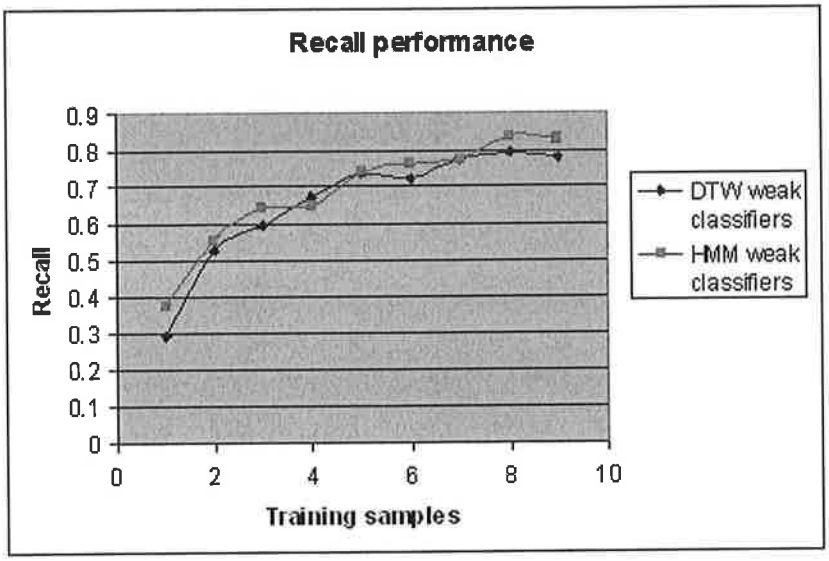


Figure 6.8: Average recall performance for both types of weak classifiers

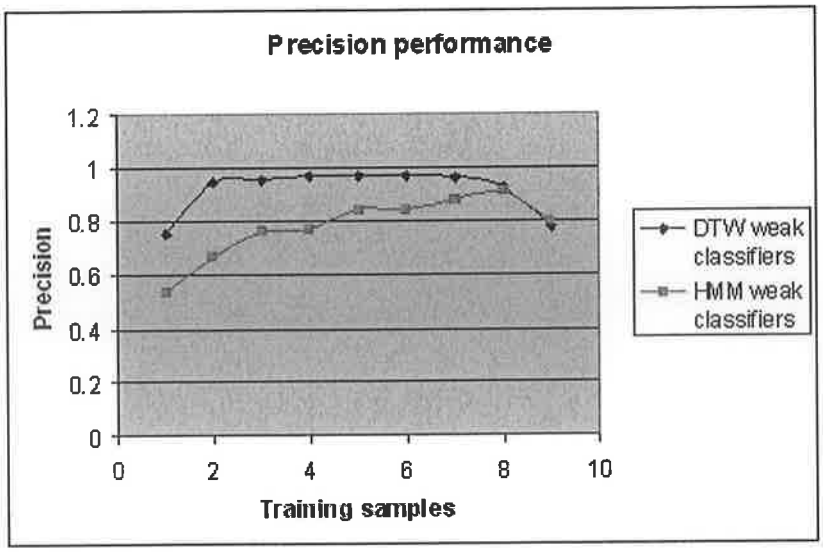


Figure 6.9: Average precision performance for both types of weak classifiers

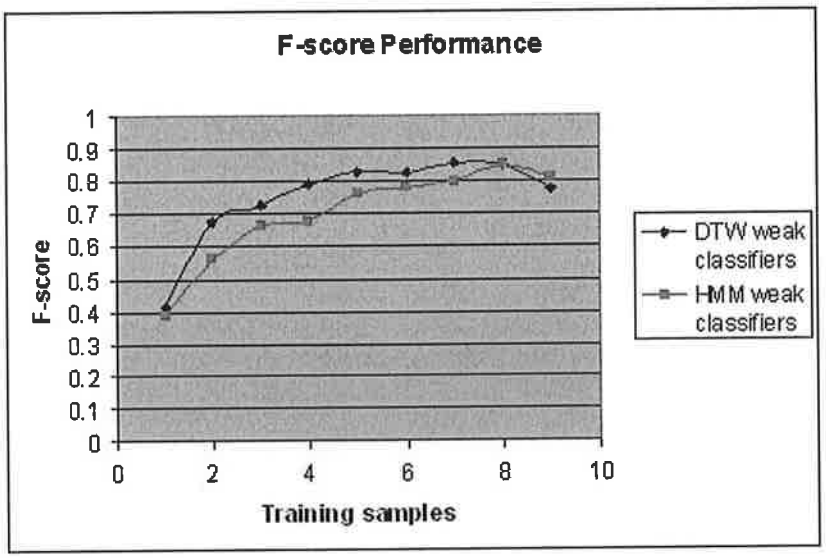


Figure 6.10: Average F-score performance for both types of weak classifiers

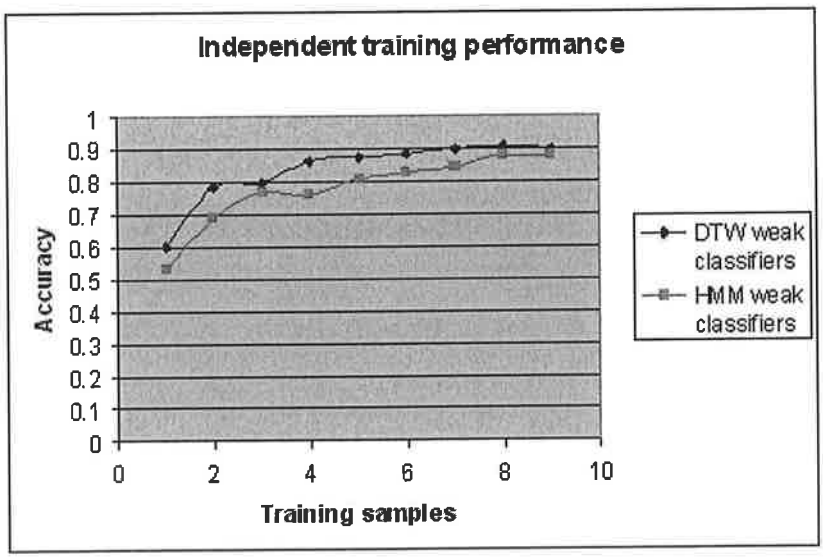


Figure 6.11: Average classification accuracy in multiclass test for both types of weak classifiers

6.5.2 Joint training

Training Adaboost jointly helped us in two main ways: firstly, to reduce the number of weak classifiers, and secondly to increase the overall performance. The gain that we achieve from sharing weak learners can be calculated by $\sum_{i=1}^{c} T_i - T_j$ where T_i is the number of weak learners used to train class i, and T_j is the number of shared

weak learners in joint training. In our case we used 20 weak classifiers for each class trained independently, and we used 60 shared weak classifiers in joint training. So this gives 400 - 60 = 340 or we reduced the number of used weak classifiers by a factor of 6.6 times.

Fig. 6.12 shows an example of the number of classes that share a weak classifier in each round. In this example, the number of classes sharing a weak classifier range from 3 to 8 classes. Using the current weights of the training samples at every round, the subset of classes that reduces the overall error is selected with its corresponding weak classifier.

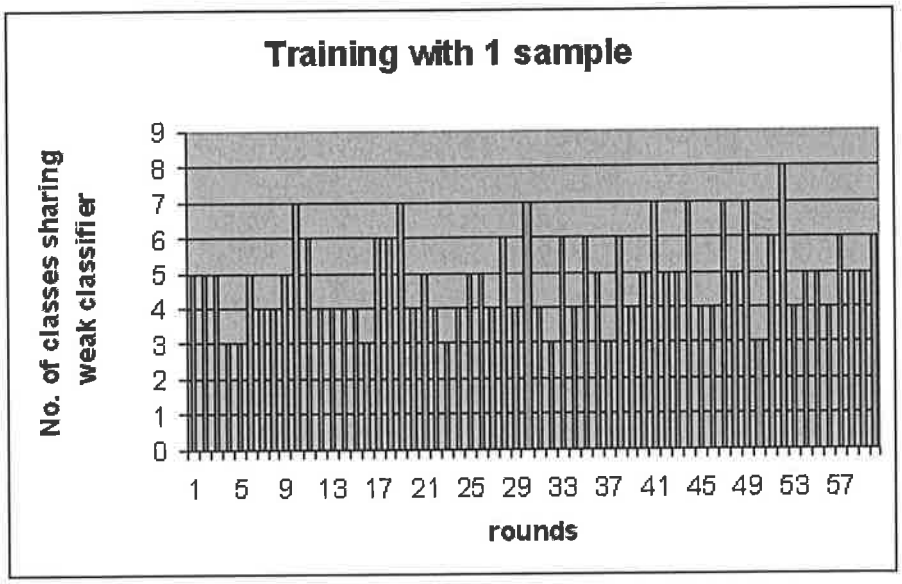


Figure 6.12: An example of sharing 60 weak classifiers between classes in training with 1 sample

Table 6.6 shows the statistical results of individual classifiers. In general, joint training has increased the performance of the recall and F-score by an average 34% and 8% respectively over independent training, while the precision of independent training is better than joint training by about 26%. Regarding the specificity, independent training is better than joint training by about 1.6%, and both can be considered to have high specificity.

In comparison of the performance of multiclass testing for both types of training,

from table 6.7 we can see that joint training has very high accuracy reaching about 85% using 2 training samples and above 90% for 3 to 9 training samples. In addition, we can see that the minimum error reached 0%. Overall, joint training increased the accuracy by about 7% on average over independent training. Fig. 6.13, 6.11, 6.15, and 6.16 show a plot of the F-score, recall, precision, and multiclass classification accuracy for both types of training with various number of training samples. Joint training has the advantage of achieving better performance with less training samples.

Training samples	**Recall**	**Precision**	**Specificity**	**F-score**
1	0.4872	0.4138	0.9596	0.4408
2	0.7937	0.6985	0.9782	0.7404
3	0.8557	0.8030	0.9856	0.8257
4	0.8924	0.8272	0.9877	0.8554
5	0.911	0.8112	0.9850	0.8552
6	0.91	0.8373	0.9876	0.8696
7	0.9416	**0.8559**	**0.9881**	0.8957
8	0.96	0.8501	0.9875	0.9002
9	**0.965**	0.8483	0.9855	**0.9004**

Table 6.6: Average statistical results of individual classifiers trained jointly

Training samples	**Min. error**	**Max. error**	**Mean error**	**Mean accuracy**
1	0.31111	0.4888	0.3972	0.6027
2	0.0562	0.2187	0.1437	0.8562
3	0.0357	0.1714	0.0999	0.9
4	0.0416	0.1	0.07166	0.9283
5	0.02	0.1	0.058	0.942
6	0.0125	0.1375	0.0725	0.9275
7	0	0.1	0.04	0.96
8	0	0.075	0.0325	0.9675
9	0	0.05	0.01	**0.99**

Table 6.7: Average statistical results of multiclass classification using joint training

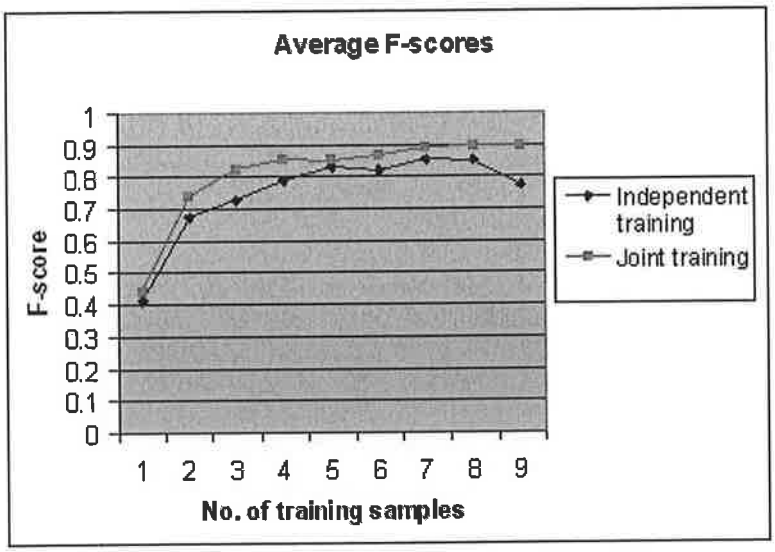


Figure 6.13: Average F-score for both types of training

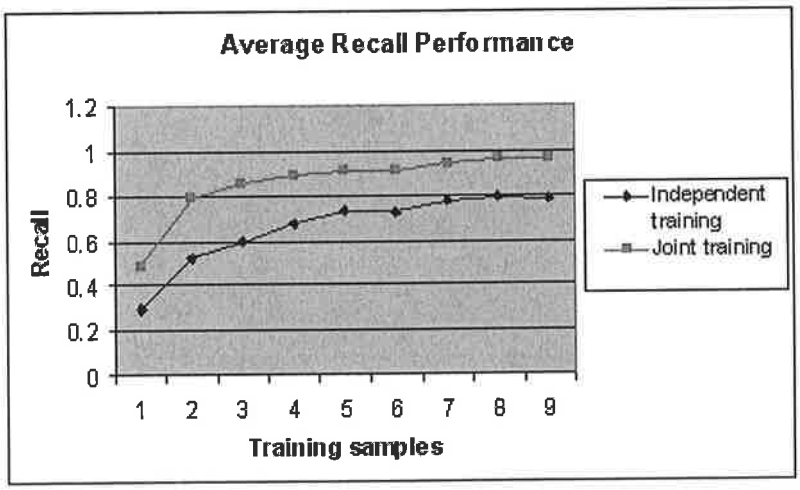


Figure 6.14: Average recall for both types of training

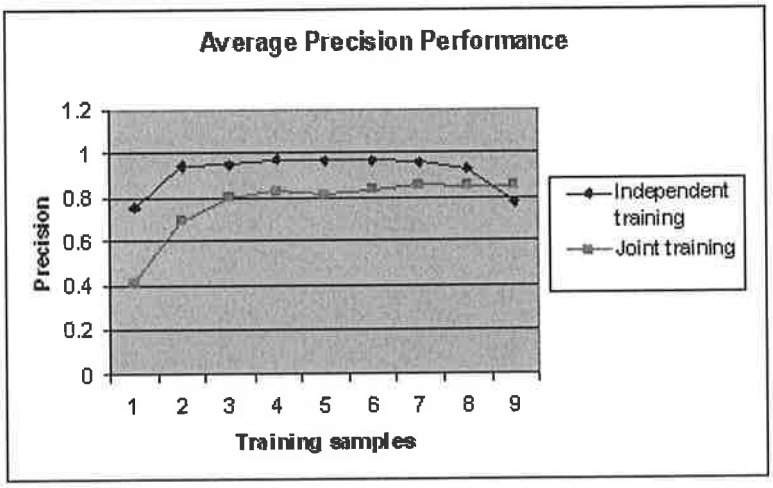


Figure 6.15: Average precision for both types of training

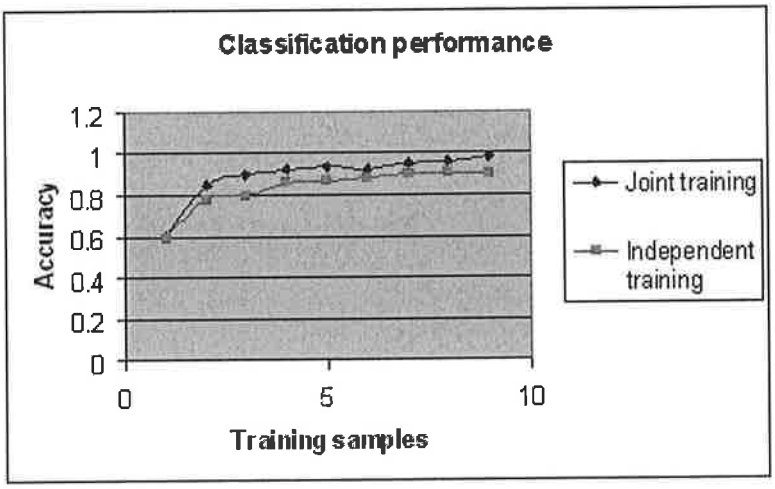


Figure 6.16: Average classification accuracy in multiclass test for both types of training

6.6 Summary

In this chapter we proposed a novel approach for sign language recognition by learning boosted subunits. Based on our subunits segmentation algorithm in the previous chapter, we model the subunits using different combinations of spatio-temporal features which act as the weak classifiers in the Adaboost framework. Then, Adaboost is used to select at every iteration the best informative combination. Two types of

training were adopted: independent training where each class is trained independently and joint training where classes can share the weak classifiers thus reducing the number of required weak classifiers and increasing the performance by allowing the classes to learn weak classifiers that can reduce the error across the whole dataset. We evaluated the performance of each classifier independently using the recall, precision, specificity and F-score metrics and the performance of the whole set of classifiers in a multiclass test using the overall classification accuracy. Joint training using DTW-based weak classifiers proved to be a good combination in terms of classification accuracy and number of required weak classifiers. Training time using Adaboost is very high (usually takes from few weeks to months) especially when the number of rounds are high. However, the classification can be done in real-time especially when joint training is used.

Chapter 7

Summary, Conclusions, and Future Work

7.1 Introduction

Vision-based Sign Language Recognition (SLR) is a complicated task that involves collaborative research efforts in multiple areas such as computer vision, image processing, pattern analysis, machine learning, and understanding of human action and behaviour. Three main tasks are required in any SLR system. First, the signer's hands have to be detected along with his face (some systems ignore the face features as we did in this work) and tracked across the video frames. Second, given that we can track the face and hands, we can extract the relevant features from every frame. Third, given that the useful features have been extracted, a machine learning technique must be used to recognize the performed sign.

In real life, we can imagine many different useful applications for SLR such as sign-to-text/speech translation systems or dialog systems for use in specific public domains [McGuire et al. 04, Akyol and Canzler 02]. In video communication between deaf people, SLR can save bandwidth by translating the video to symbols which are animated at the other end [Kennaway 03]. SLR can be used for annotating sign videos [Koizumi et al. 02] for linguistic analysis.

In this thesis, we aimed to provide new techniques that can be applied in SLR ap-

plications. Our goal was to contribute to research in skin segmentation, hand and face tracking, modelling and recognizing signs efficiently based on human behaviour in performing and recognizing signs using informative subunits of the signs.

7.2 Summary

We started by giving a brief overview of the literature to cover the three main tasks. Different systems handle the three tasks in different ways. Mainly, SLR systems deal with either one-handed signs or two-handed signs, isolated or continuous words. There is a general agreement that SL relevant features includes hand shape, position, orientation, motion, and facial expressions.

Regarding hand detection, approaches vary between colour glove-based methods vs skin detection methods. Although colour glove is simple and easier to deal with when we are segmenting the hands, it's not a natural way for signers or deaf people. On the other hand, skin detection techniques have the problems of dealing with illumination change, skin colour variations, and occlusions with other human skin parts. Different systems try to avoid the occlusion problem by: performing unnatural signs, a choice of vocabulary that doesn't include occlusions, or using stereo cameras but with high computational cost.

Tracking is one of the tasks that get highly affected by the occlusions of both hands or occlusions with the face, thus different assumptions have been made to resolve this problem. We reviewed the imaging restrictions that are usually adopted to simplify hand detection and tracking such as wearing long sleeves, uniform/stationary background, the signer's hand is the only moving object, head has less motion than the hands, and exclusion of face and/or left hand from the Field Of View (FOV).

The hand centre is a common feature for absolute hand position or relative to other body parts, while regarding hand motion, hand trajectories, chain code and optical flow is very popular features. Hand appearance features are mainly extracted from the whole hand blob or hand contour using Fourier descriptors, geometric moments, and PCA. With respect to classification schemes, two main approaches exist. One

approach uses all the features collected to classify the sign directly in one stage, while the other first classifies the different sign components and then integrates their results in one final stage. The most famous methods used in classification are Neural Networks, HMMs, PCA, decision trees, and nearest-neighbour matching.

In an attempt to design a SLR system with limited imaging constraints, we began investigating different existing methods for skin segmentation. Motivated by our observations that sign videos usually contain few signers with small lighting changes, we proposed a new algorithm for skin segmentation where an SVM is trained for every signing video using Active Learning. First, a generic skin model is applied to select initial skin pixels. Then using region segmentation information we select from non-skin regions a ratio of pixels proportional to how close the colour of these non-skin pixels is to skin pixels using a principle we called "most similar-highest priority". Active Learning has the advantage of balancing the number of training samples between skin and non-skin pixels, and increasing the overall accuracy by selecting more informative training samples and less non-informative ones. We define informative pixels as those non-skin pixels that are more similar in their colour to the true skin pixels. Finally, as region segmentation is more robust to noise and some lighting variations, we combine it with the SVM results to refine the final skin segmentation results.

Our experimental results demonstrated that Active Learning increased the overall accuracy by about 6% and reduced the average training time by 114 seconds. While, combining region information enhanced the overall accuracy by 9%. Compared to Gaussian skin models (skin only, and skin/non-skin), our model achieved the highest overall accuracy with the second lowest false detection rate. The skin/non-skin Gaussian model has the best correct detection rate. However, its false detection rate is the worst.

As SLR requires the collection of hand features from the sequence of frames during which the sign was performed, it was necessary to develop a hand tracker component. We decided to approach this problem from a more general viewpoint, so we developed a skin segmentation and tracking system (SST) to segment and track 3

skin objects: face and two hands. Regarding segmenting skin objects, we chose to combine three useful features: "*motion*" as it can distinguish foreground objects from background, "*colour*" as it can help to distinguish skin pixels from non-skin pixels, and "*position*" to reduce the search space by predicting the location of skin objects in the next frame. A Kalman filter was used to predict the position of objects in the next frame, thus helping us to predict occlusions. In every frame, we aimed to identify the existing skin objects and update the occlusion status of all the objects by employing a set of heuristic rules that take advantage of the occlusion alarms and the detected objects in the current frame, then blob matching was done between the previous and current skin objects.

We demonstrated that the tracker can help the skin colour model to be more adaptive to lighting changes by providing the colour model with new training samples and re-training the colour model on the new training samples. Experimenting with our SST system on real signing videos that include large numbers of occlusions, we recorded the average error of the bounding box positions of the face and two hands on 600 frames with 40% occluded frames. The system accuracy is very high as the maximum error is about 6 pixels and the tracking error (by incorrectly identifying skin objects) reached about 6.5%.

The skin segmentation and tracking component was a necessary and a prerequisite step for us to move forward and begin analyzing the characteristics of the human hand motion during signing. From observing the trajectory and speed curves of the hands, it was shown that they are related at some points, where discontinuities happen. In the speed curve, during a continuous motion pattern the hand motion goes through three phases: acceleration, uniform motion and deceleration. While in the trajectory curve, for a certain motion pattern, the trajectory forms a smooth curve without sharp or noticeable corners.

These observations motivated us to model signs using a subunit-based approach, where we define a subunit as a continuous motion pattern in space that covers a certain duration in time. Our objective is to discover these subunits from the sign vocabulary by detecting the subunit boundary points where discontinuities happen

in speed and trajectory. Finally, temporal clustering was employed to refine the results by merging subunits that have been incorrectly segmented due to noise.

To quantitatively evaluate the performance of our subunit boundary detection algorithm, we constructed manual ground truth of the subunits (codebook entries) for 10 signs, and compared it to the output of the automatically constructed codebook by our algorithm by measuring the recall and precision between the two codebooks for every sign. The proposed approach reached an average recall of around 0.82 and average precision of about 0.76.

Towards this end, and assuming that the subunits are the building blocks of a certain sign vocabulary, we need to develop a mechanism to recognize and understand signs based on subunits. Furthermore, as we don't know yet what are the informative subunits or features in the sign vocabulary, this motivated us to adobt the Adaboost algorithm to select discriminative combinations of subunits and features and construct a strong classifier for each sign based on combining weak classifiers. Each weak classifier is composed of a subunit modelled with one feature or more. We considered using all the combinations of features for every subunit. Subunits are extracted from the temporal clustering of each sign class and considered as the medoid subunit of each cluster. Two types of training were applied and compared: independent training where every sign is trained independently of all other classes, and joint training where weak classifiers can be shared across a subset of classes to reduce the total error in the dataset. Two types of weak classifiers were compared in independent training: DTW-based using the medoids of the clusters in each codebook, and HMM-based trained on the subunits in each cluster for every codebook. Using a dataset of 20 sign classes, training was performed using different numbers of training samples from 1 to 9 and we recorded the measures of recall, precision, F-score and specificity for each classifier when tested individually. Also, in a multiclass test, we measured the accuracy of the overall classification performance. Evaluating the performance of DTW and HMM-based weak classifiers using independent training, we found that the recall performance of HMMs on average are higher than DTW-based classifiers by about 8.6%, while DTW is better than the HMMs by an average

17%, 8%, 8%, and 1% for the precision, F-score, classification, and specificity respectively. On average, independent training using DTW-based weak classifiers reached F-score of 0.75 and classification accuracy of 83.6%.

Regarding joint training, the number of weak classifiers were reduced due to sharing by a factor of 6.6 and on average the F-score and classification accuracy reached 0.8 and 89.7% respectively. In general joint training increased the performance of the recall, F-score, and classification accuracy by an average 34%, 8%, and 7% respectively over independent training, while the precision and specificity of independent training is better by an average 26% and 1.6% respectively.

7.3 Conclusions

In this section, we will try to summarize the advantages and disadvantages of the proposed methods that were presented in this thesis work. Regarding the proposed SVM skin segmentation algorithm presented in chapter 3, the proposed approach has the advantage of being adaptive to different users. Training samples are collected automatically. Active Learning and region information are used to refine the results, and it can be used while tracking skin objects in offline mode. On the other hand, some disadvantages are: the general skin model must give initially reasonable skin samples for the training of the SVM. Also, if region information was not good enough, skin regions might not be detected correctly, and finally the SVM is not suitable for real time processing and in this case it could be better to investigate other types of machine learning techniques that can operate faster or use SVM in offline processing of videos.

In chapter 4 we presented a skin segmentation and tracking system (SST) for SLR. In this system the segmentation and tracking parts were integrated together so that good segmentation helps to increase the accuracy of the tracker and good tracking reduces the search space for segmentation. Segmentation doesn't need region segmentation any more by using three perceptually meaningful fast features (except for SVM which is slow, but faster than when applied on the whole frame). The occlusion

status of three skin objects were tracked and identified using simple heuristic rules. An adaptive skin colour model can be used using the feedback from the tracker. While not requiring a lot of imaging restrictions, the system provides a reasonable compromise in terms of computational cost relative to the overall accuracy. The system components and features can be replaced and used easily without much effort as the architecture is very modular and in general can be used in any human related skin object segmentation and tracking application.

On the other hand, the disadvantages can be summarized as follows: the SVM even when used within a search window, cannot operate in a real time. However, testing the system using only the generic skin model (chapter 3) and the simple prior knowledge colour metric (chapter 4) proved to work in real time. In the case that the tracker has lost tracking of an object, it is quite difficult to locate it again unless the object appears again inside the search window where the features are calculated, and currently it is hard to distinguish between when the object is hiding (outside the view of the camera) or if the tracker has missed it. Although we don't require the user to wear a long sleeves, we don't distinguish between the palm area and the arm which is necessary in the recognition phase. Although the system doesn't require uniform background or non-moving background objects, it still can be affected by skin-like colours especially if they are within the search window of the tracker. A last and important point, is that we don't have a module to segment the occluded objects as since to date we treat the occluded parts as one big skin object. However, we do know the identity of the objects that are occluded (i.e, face and hand, or hand and hand,..etc).

Moving forward after developing our SST system, in chapter 5 we proposed a novel algorithm for defining and segmenting SL subunits. The subunit-based approach (inspired from phonemes in speech recognition) has the advantage that the signs can be decomposed into a set of subunits which can be used in the recognition phase instead of depending on extracting the features from the whole sign frames and using them directly in the recognition. The number of subunits should be much smaller than the number of the signs in the vocabulary, thus reducing the recognition space and

supporting small training samples. The proposed approach does not require previous knowledge about the number of subunits or the type of signs. Using adaptive thresholds for detecting discontinuity in trajectory and speed curves makes the algorithm more robust to different users and their way of signing. Temporal clustering refines the results by merging similar sequential subunits that were segmented due to noise. On the other hand, we acknowledge a number of weak points that can affect the performance such as: in trajectory curves with low curvature points (where in reality there was a smooth change in the motion pattern), it is challenging for the trajectory discontinuity detector to detect a corner point. Errors in subunit segmentation affect the average number of subunits segmented for every sign, and thus the merging in the temporal clustering is affected as it uses this average number as a threshold in the dendrogram to construct the clusters of subunits. Some complex movements can be very challenging such as finger movements with fixed palm position, and movement of the hand while occluded with other skin objects, which can affect the quality of the hand position provided by the hand tracker. Finally, due to irregularities in the hand motion (mostly with non native SL users) the boundary points detected by the trajectory and speed discontinuity detectors may not match in the small window, where we search for the subunit preliminary boundary candidates.

The subunit segmentation algorithm was a basic block for our motivation to propose a new SLR algorithm based on boosting subunits in chapter 6. The proposed SLR system based on boosting subunits has the advantage of being inspired from the way humans recognize signs by selecting the informative (discriminative) combinations of subunits and features. It opens the possibility of efficiently recognizing signs of large vocabulary with small training data. Our method is independent of the number of subunits in each sign, so we don't need to predefine any previous knowledge about subunits unlike in HMM-based approaches where the number of states are predefined in most cases. Subunit extraction and construction of weak classifiers are totally automatic. Joint training increases the total performance while reducing the number of required weak classifiers by sharing them across the classes. The classification is simple and can operate in real-time.

On the other hand, the training time of Adaboost is very time-consuming as each iteration has to select one weak classifier from a large pool, and in joint training it considers all the possible subsets of classes. In addition, it depends largely on the number of iterations, so there is a trade off between the performance and time. The weak classifiers and training samples in our experiments are constructed totally automatically which can affect the training performance if they include some errors or noise but can be avoided if subunits are extracted manually. Tracking can still affect the results as the tracker data are used to segment the subunits which are the basis of our recognition system. We experimented with the sharing of weak classifiers but didn't experiment sharing of subunits by constructing one unified codebook for the whole vocabulary which might increase the accuracy.

7.4 Future work

In order to overcome the shortcomings of our proposed techniques in this work, some future investigations have to be done and can be summarized in the following points:

- Regarding the skin segmentation problem, a faster and more reliable machine learning classifier has to be adopted to be able to work in real-time as SVM suffers from this disadvantage.
- In many cases the users are not wearing long sleeves, so research must be done to investigate how to segment the palm area from the rest of the arm.
- Occlusions are very common in SL videos. In this work we proposed a solution for occlusion detection but still future work must be done concerning how to segment skin-occluded areas (face/hand or hand/hand).
- In this work we concentrated on hand features only. However, non-manual features like facial expressions also have big weight in understanding the performed sign and can help in discrimination especially in large vocabularies.
- Our SST system proved to be a good compromise between segmentation and tracking accuracy and real-time performance, but still it can be affected by

skin-like colours inside the search windows, so it might be helpful to look into more useful features to include, such as texture or even investigating texture analysis operators such as Local Binary Patterns (LBP), which are grey-scale invariant.

- More work has to be done in investigating the cases where the palm area is stationary while only the fingers are changing their configuration (thus the hand centre), and how the subunits segmentation algorithm can detect these cases and whether to consider the whole duration of stationary palm as a subunit or not.

- More work has to be done in finding new ways to detect discontinuity in the trajectory curve especially in difficult regions where the curvature is small.

- Regarding the recognition phase, future work must investigate promoting the existing Adaboost learning algorithm to an incremental algorithm where new signs can be learned jointly with the already learned vocabulary, and experimenting with more larger SL vocabulary.

Appendix A

Skin segmentation dataset

As shown below in figures A.1 to A.8 we can see some sample frames from the 8 video sequence used to evaluate the proposed skin segmentation algorithm in chapter 3. Each frame has been ground truthed by editing the skin areas with a predefined colour (RGB: 0,255,0).

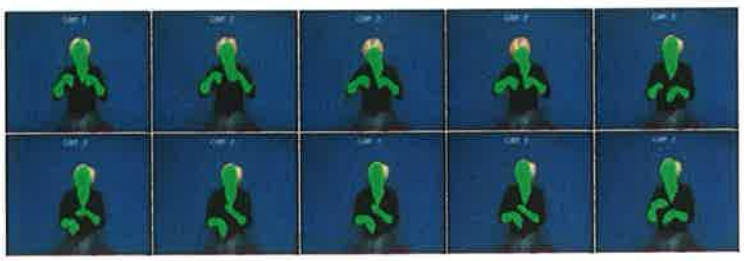


Figure A.1: Sample ground truth frames from video sequence 1

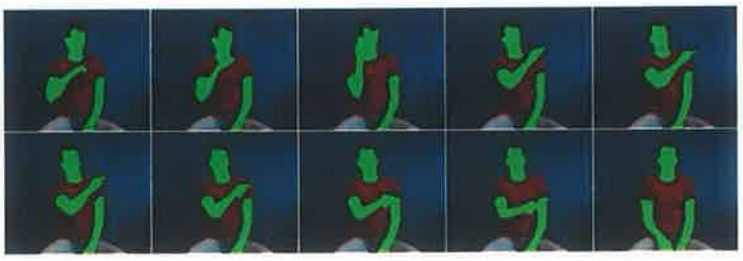


Figure A.2: Sample ground truth frames from video sequence 2

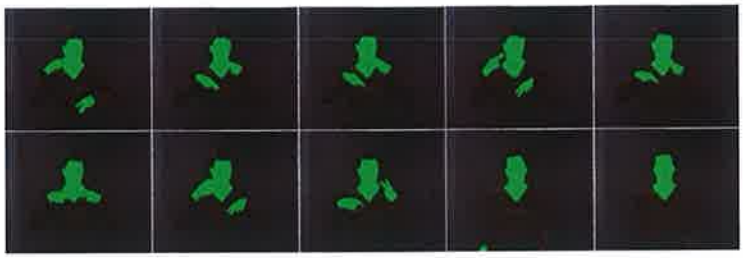


Figure A.3: Sample ground truth frames from video sequence 3

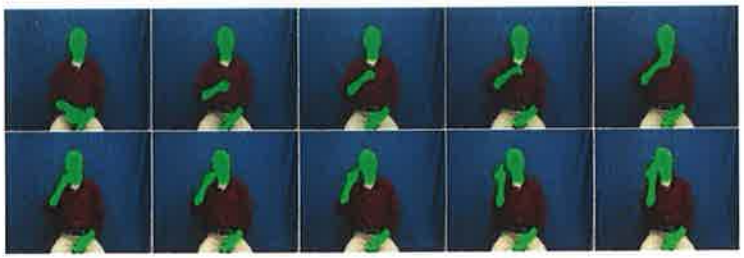


Figure A.4: Sample ground truth frames from video sequence 4

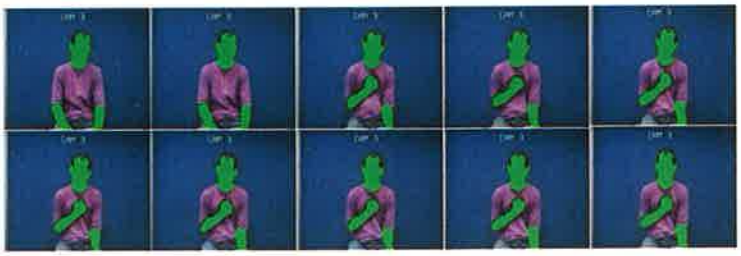


Figure A.5: Sample ground truth frames from video sequence 5

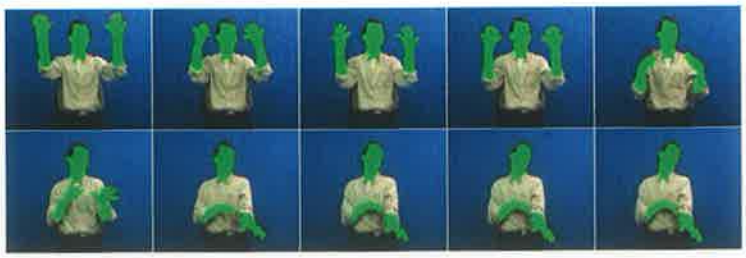


Figure A.6: Sample ground truth frames from video sequence 6

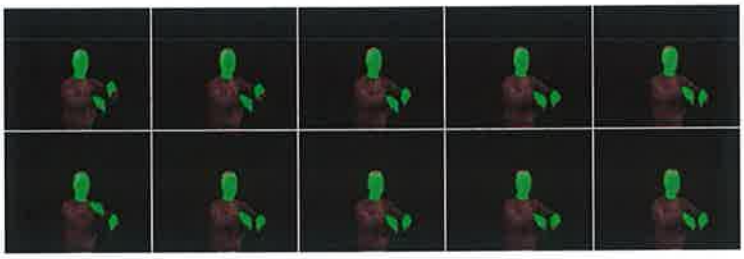


Figure A.7: Sample ground truth frames from video sequence 7

Figure A.8: Sample ground truth frames from video sequence 8

Appendix B

Adaboost sign classifiers results

In the following tables we present the performance of each sign classifier when trained independently and jointly using training samples from 1 to 9. The statistical results are the average of 10 runs

Sign name	Recall	Precision	F-score	Specificity
about	0.333334	0.489899	0.350159	0.984794
apple	0.522222	0.2731325	0.342589	0.937427
ask	0.155556	0.412073	0.17539	0.987136
bat	0.333332	0.5849209	0.3761697	0.98187
before	0.355555	0.493269	0.372809	0.991813
believe	0.555557	0.768651	0.62212	0.99298
best	0.31111	0.404126	0.282006	0.971343
black	0.311112	0.414649	0.316277	0.987719
bottle	0.255555	0.3299249	0.27681	0.974855
boy	0.366666	0.8875	0.469438	0.99883
brown	0.388888	0.39246	0.364452	0.985381
busy	0.255556	0.288889	0.270065	0.97193
but	0.399999	0.763333	0.456539	0.991228
can	0.466665	0.335811	0.364358	0.944444
can't	0.255556	0.401825	0.260011	0.978948
clever	0.488889	1	0.632407	1
coffee	0.466666	0.356822	0.39011	0.949122
confidence	0.5	0.714035	0.547948	0.990643
deaf	0.488889	0.861111	0.530887	0.991812
don't_know	0.322222	0.50641	0.331818	0.980117

Table B.1: Statistical results of independent training (HMM-based) using 1 training sample

Sign name	Recall	Precision	F-score	Specificity
about	0.25	0.416667	0.276046	0.98421
apple	0.6375	0.567885	0.574118	0.969079
ask	0.3	0.487618	0.319618	0.978948
bat	0.475	0.595556	0.457121	0.975656
before	0.6125	0.835834	0.668599	0.988157
believe	0.4625	0.701905	0.546001	0.990788
best	0.6375	0.514551	0.530437	0.959868
black	0.6625	0.834401	0.718404	0.992105
bottle	0.525	0.542619	0.5121	0.976314
boy	0.8625	0.958928	0.906546	0.998026
brown	0.55	0.526389	0.491448	0.982894
busy	0.6	0.645167	0.549169	0.973026
but	0.4	0.6075	0.441404	0.99013
can	0.6125	0.567745	0.553334	0.969736
can't	0.325	0.493453	0.361628	0.978947
clever	0.6125	0.9	0.703634	1
coffee	0.6875	0.704167	0.684928	0.986183
confidence	0.55	0.934265	0.631015	0.994737
deaf	0.7875	0.84264	0.80508	0.989474
don't_know	0.5625	0.701671	0.529602	0.973682

Table B.2: Results of independent training (HMM-based) using 2 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.57143	0.616429	0.572451	0.978946
apple	0.64286	0.713511	0.657741	0.982706
ask	0.271428	0.386588	0.299472	0.978196
bat	0.571428	0.672857	0.604968	0.985713
before	0.571429	0.868333	0.676549	0.995488
believe	0.571429	0.789762	0.624366	0.990224
best	0.785714	0.839683	0.798188	0.990225
black	0.714287	0.846547	0.740971	0.99248
bottle	0.571429	0.704999	0.588941	0.986465
boy	0.914285	0.97	0.932004	0.997744
brown	0.728572	0.807699	0.732778	0.990224
busy	0.714286	0.669048	0.668993	0.977443
but	0.6	0.840357	0.637624	0.988721
can	0.471429	0.407781	0.408002	0.957141
can't	0.457143	0.771429	0.518656	0.984961
clever	0.714285	0.954761	0.798506	0.997744
coffee	0.7	0.829383	0.709176	0.98797
confidence	0.614285	0.845635	0.636139	0.98797
deaf	0.9	0.945833	0.918205	0.996992
don't_know	0.785714	0.739888	0.746404	0.984961

Table B.3: Results of independent training (HMM-based) using 3 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.466667	0.628333	0.492661	0.982457
apple	0.533334	0.836667	0.624682	0.994737
ask	0.366666	0.55389	0.409142	0.983333
bat	0.683333	0.896667	0.766766	0.996492
before	0.65	0.866667	0.721819	0.997368
believe	0.683332	0.77855	0.703321	0.986844
best	0.800001	0.756668	0.757728	0.984211
black	0.799999	0.848095	0.802501	0.991229
bottle	0.583334	0.558355	0.549095	0.977195
boy	0.7	0.98	0.779783	0.999123
brown	0.716667	0.793808	0.738439	0.989474
busy	0.766665	0.836984	0.767614	0.990352
but	0.633334	0.705	0.64189	0.981579
can	0.666668	0.586869	0.610487	0.975439
can't	0.449999	0.598077	0.478202	0.982456
clever	0.633335	0.98	0.742207	0.999123
coffee	0.766665	0.777977	0.756953	0.986843
confidence	0.433334	0.733333	0.535001	0.998246
deaf	0.783334	0.838928	0.799161	0.989475
don't_know	0.833332	0.902381	0.840994	0.994737

Table B.4: Results of independent training (HMM-based) using 4 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.6	0.726667	0.644921	0.995789
apple	0.62	0.890477	0.693759	0.993684
ask	0.4	0.55881	0.434782	0.981052
bat	0.78	0.885	0.814113	0.993683
before	0.74	0.917858	0.793254	0.994737
believe	0.7	0.85	0.74762	0.99263
best	0.82	0.836667	0.818154	0.989473
black	0.86	0.924999	0.883839	0.995788
bottle	0.6	0.804763	0.640295	0.988422
boy	0.86	0.98	0.907778	0.998947
brown	0.88	0.793176	0.80948	0.985263
busy	0.76	0.931429	0.800476	0.995789
but	0.7	0.68631	0.674145	0.985262
can	0.6	0.641667	0.594871	0.978947
can't	0.6	0.905	0.712143	0.995789
clever	0.76	0.96	0.821111	0.997894
coffee	0.9	0.828056	0.846988	0.987367
confidence	0.7	0.921429	0.769841	0.99579
deaf	0.94	0.938095	0.92904	0.995789
don't_know	0.96	0.847261	0.892781	0.989472

Table B.5: Results of independent training (HMM-based) using 5 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.575	0.866667	0.663811	0.998684
apple	0.75	0.896667	0.805555	0.994736
ask	0.4	0.725	0.500715	0.992104
bat	0.8	0.86	0.814762	0.990789
before	0.75	0.98	0.831746	0.998684
believe	0.725	0.775001	0.746429	0.989472
best	0.875	0.905	0.885317	0.994736
black	0.825	0.841667	0.814683	0.990788
bottle	0.625	0.638334	0.625	0.980262
boy	0.975	1	0.985714	1
brown	0.775	0.806667	0.76889	0.988156
busy	0.875	0.811191	0.833355	0.986841
but	0.775	0.805	0.768651	0.992105
can	0.675	0.547778	0.585825	0.971051
can't	0.525	0.875	0.626429	0.994736
clever	0.85	1	0.904762	1
coffee	0.8	0.813334	0.793174	0.988157
confidence	0.875	0.896667	0.866984	0.992105
deaf	0.9	1	0.938095	1
don't_know	0.875	0.872143	0.85725	0.990789

Table B.6: Results of independent training (HMM-based) using 6 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.766668	0.975	0.835714	0.998246
apple	0.766668	0.941667	0.822381	0.996492
ask	0.5	0.741667	0.572381	0.992984
bat	0.700001	0.808334	0.726191	0.989475
before	0.766667	1	0.84	1
believe	0.766668	0.820001	0.760001	0.98772
best	0.966667	0.916667	0.938095	0.994738
black	0.8	0.875	0.783809	0.989475
bottle	0.866668	0.861667	0.842381	0.989474
boy	0.933334	1	0.96	1
brown	0.733333	0.71	0.715	0.992983
busy	0.733335	0.866667	0.786667	0.998246
but	0.733335	0.841667	0.772381	0.996492
can	0.566667	0.676667	0.58881	0.985966
can't	0.600001	1	0.73	1
clever	0.933334	1	0.96	1
coffee	0.800002	0.843334	0.805477	0.989474
confidence	0.833333	0.908333	0.849047	0.994737
deaf	0.933334	0.925	0.917142	0.994738
don't_know	0.800002	0.95	0.851428	0.996492

Table B.7: Results of independent training (HMM-based) using 7 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.75	0.9	0.800001	0.994736
apple	0.85	0.85	0.820001	0.986842
ask	0.5	0.683334	0.543335	0.989472
bat	0.75	0.8	0.766667	0.997368
before	0.8	0.933333	0.840001	0.994737
believe	0.9	0.966667	0.913334	0.997368
best	0.9	0.933334	0.893334	0.994736
black	0.9	0.866668	0.853334	0.989472
bottle	0.75	0.9	0.800001	1
boy	0.85	0.9	0.866667	1
brown	0.95	0.95	0.95	0.997368
busy	1	0.95	0.966667	0.994737
but	0.95	1	0.966667	1
can	0.8	0.750001	0.723335	0.978946
can't	0.9	1	0.933334	1
clever	0.9	1	0.933334	1
coffee	0.85	0.883334	0.843334	0.992104
confidence	0.8	1	0.866668	1
deaf	0.85	1	0.900001	1
don't_know	0.8	0.916667	0.830001	0.994736

Table B.8: Results of independent training (HMM-based) using 8 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.3	0.3	0.3	0.994737
apple	0.8	0.8	0.8	1
ask	0.6	0.5	0.533334	0.984211
bat	1	1	1	1
before	1	1	1	1
believe	0.9	0.9	0.9	1
best	0.8	0.8	0.8	1
black	0.9	0.75	0.800001	0.984211
bottle	1	0.95	0.966667	0.994737
boy	0.9	0.9	0.9	1
brown	1	1	1	1
busy	0.8	0.8	0.8	1
but	0.8	0.8	0.8	1
can	0.9	0.7	0.766668	0.978948
can't	0.9	0.9	0.9	1
clever	0.7	0.7	0.7	1
coffee	0.8	0.733333	0.75	0.989474
confidence	1	1	1	1
deaf	0.6	0.6	0.6	1
don't_know	0.9	0.9	0.9	1

Table B.9: Results of independent training (HMM-based) using 9 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.244443	0.8	0.37446639	1
apple	0.655556	0.943636	0.773648495	0.996101365
ask	0.255555	0.446316	0.325012104	0.985055231
bat	0.233333	0.766667	0.357777422	0.999350227
before	0.233333	0.9	0.370587815	1
believe	0.244443	0.763333	0.370303338	0.999350227
best	0.411111	0.85	0.554184921	0.999350227
black	0.266665	0.763333	0.39525163	0.999350227
bottle	0.11111	0.518182	0.182984058	0.993502274
boy	0.311109	0.8	0.447997811	1
brown	0.144443	0.45	0.218689933	0.999350227
busy	0.41111	0.75	0.531099551	0.999350227
but	0.3	0.703846	0.420689628	0.998050682
can	0.466667	0.72	0.56629238	0.992852502
can't	0.244442	1	0.392853986	1
clever	0.21111	0.8	0.334064543	1
coffee	0.2	0.6	0.3	1
confidence	0.21111	0.9	0.341998542	1
deaf	0.355554	0.9	0.509732915	1
don't_know	0.333332	0.7	0.451611679	1

Table B.10: Results of independent training (DTW-based) using 1 training sample

Sign name	Recall	Precision	F-score	Specificity
about	0.4625	0.8675	0.603336466	0.997807018
apple	0.7125	0.977778	0.824322183	0.998538012
ask	0.4625	0.928571	0.617458185	0.992690058
bat	0.6375	0.852727	0.729571351	0.993421053
before	0.4875	1	0.655462185	1
believe	0.625	1	0.769230769	1
best	0.6125	1	0.759689922	1
black	0.4125	0.9	0.565714286	1
bottle	0.4375	1	0.608695652	1
boy	0.5875	1	0.74015748	1
brown	0.475	0.85	0.609433962	0.999269006
busy	0.5625	0.9	0.692307692	1
but	0.45	0.9	0.6	1
can	0.4375	1	0.608695652	1
can't	0.5	0.931428	0.650698463	0.997807018
clever	0.4875	0.9	0.632432432	1
coffee	0.6375	0.97	0.769362364	0.997807018
confidence	0.5	1	0.666666667	1
deaf	0.55	1	0.709677419	1
don't_know	0.5125	1	0.67768595	1

Table B.11: Results of independent training (DTW-based) using 2 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.600003	0.942858	0.733335833	0.996658312
apple	0.7	0.963492	0.81087784	0.997493734
ask	0.642858	1	0.782609331	1
bat	0.62857	0.966667	0.761790099	0.997493734
before	0.557143	1	0.715596448	1
believe	0.557143	0.946667	0.701456823	0.999164578
best	0.585715	1	0.738739307	1
black	0.514286	1	0.679245532	1
bottle	0.557142	0.91	0.691138581	0.998329156
boy	0.842856	1	0.914728009	1
brown	0.585714	0.926667	0.717758072	0.997493734
busy	0.62857	0.9	0.740185925	1
but	0.528572	1	0.691589274	1
can	0.471427	0.9	0.618748646	1
can't	0.385714	0.885714	0.537399349	0.999164578
clever	0.428573	0.9	0.580646604	1
coffee	0.685714	0.875	0.768878539	0.998329156
confidence	0.714287	1	0.833334208	1
deaf	0.542859	1	0.703705264	1
don't_know	0.742857	1	0.852458922	1

Table B.12: Results of independent training (DTW-based) using 3 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.533334	1	0.695652741	1
apple	0.699998	0.971429	0.813674013	0.998050682
ask	0.666666	0.985714	0.795388481	0.999025341
bat	0.716666	0.833929	0.770863521	0.998050682
before	0.616667	0.966667	0.752982806	0.997076023
believe	0.816665	1	0.899081559	1
best	0.716664	1	0.834949647	1
black	0.716666	0.96	0.820675507	0.998050682
bottle	0.683333	0.95	0.794897734	0.998050682
boy	0.649999	1	0.787878053	1
brown	0.816666	1	0.899082165	1
busy	0.716665	1	0.834950325	1
but	0.5	0.9	0.642857143	1
can	0.483333	1	0.65168509	1
can't	0.6	0.863333	0.707972553	0.995126706
clever	0.716668	1	0.834952361	1
coffee	0.733334	1	0.84615429	1
confidence	0.666667	1	0.80000024	1
deaf	0.533335	1	0.695653592	1
don't_know	0.933333	1	0.965517063	1

Table B.13: Results of independent training (DTW-based) using 4 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.74	0.833572	0.784003884	0.987134503
apple	0.74	0.966667	0.838281375	0.997660819
ask	0.72	1	0.837209302	1
bat	0.88	0.926191	0.902504862	0.996491228
before	0.66	1	0.795180723	1
believe	0.86	1	0.924731183	1
best	0.82	1	0.901098901	1
black	0.76	1	0.863636364	1
bottle	0.76	1	0.863636364	1
boy	0.7	1	0.823529412	1
brown	0.78	1	0.876404494	1
busy	0.66	1	0.795180723	1
but	0.58	1	0.734177215	1
can	0.68	1	0.80952381	1
can't	0.6	0.81	0.689361702	0.995321637
clever	0.58	0.9	0.705405405	1
coffee	0.84	0.922858	0.879481751	0.995321637
confidence	0.76	1	0.863636364	1
deaf	0.64	1	0.780487805	1
don't_know	0.94	1	0.969072165	1

Table B.14: Results of independent training (DTW-based) using 5 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.775	0.926667	0.844074575	0.995614035
apple	0.625	1	0.769230769	1
ask	0.7	0.98	0.816666667	0.998538012
bat	0.8	0.95	0.868571429	0.995614035
before	0.525	0.9	0.663157895	1
believe	0.675	0.9	0.771428571	1
best	0.8	1	0.888888889	1
black	0.75	1	0.857142857	1
bottle	0.65	1	0.787878788	1
boy	0.925	1	0.961038961	1
brown	0.875	1	0.933333333	1
busy	0.75	0.9	0.818181818	1
but	0.65	0.98	0.781595092	0.998538012
can	0.8	1	0.888888889	1
can't	0.475	0.875	0.615740741	0.998538012
clever	0.7	1	0.823529412	1
coffee	0.825	0.975	0.89375	0.998538012
confidence	0.675	1	0.805970149	1
deaf	0.625	1	0.769230769	1
don't_know	0.85	1	0.918918919	1

Table B.15: Results of independent training (DTW-based) using 6 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.800001	0.95	0.868572018	0.998050682
apple	0.733336	1	0.846155621	1
ask	0.866668	1	0.928572194	1
bat	0.866668	1	0.928572194	1
before	0.566666	0.8	0.663414177	1
believe	0.866668	1	0.928572194	1
best	0.766669	1	0.867926023	1
black	0.833335	0.95	0.887851413	0.998050682
bottle	0.833334	1	0.909091306	1
boy	0.933334	1	0.965517598	1
brown	0.766667	0.966667	0.855128543	0.998050682
busy	0.700001	1	0.823530104	1
but	0.733334	1	0.84615429	1
can	0.599999	0.9	0.71999928	1
can't	0.666667	0.9	0.765957667	1
clever	0.766667	0.9	0.828000194	1
coffee	0.833335	1	0.909091901	1
confidence	0.766668	0.9	0.828000778	1
deaf	0.700001	1	0.823530104	1
don't_know	0.900001	1	0.947368975	1

Table B.16: Results of independent training (DTW-based) using 7 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.8	0.9	0.847058824	0.988304094
apple	0.75	0.9	0.818181818	1
ask	0.95	1	0.974358974	1
bat	0.9	0.9	0.9	1
before	0.65	0.9	0.75483871	1
believe	0.9	1	0.947368421	1
best	0.65	0.9	0.75483871	1
black	0.9	1	0.947368421	1
bottle	0.85	1	0.918918919	1
boy	0.85	0.9	0.874285714	1
brown	0.8	1	0.888888889	1
busy	0.85	1	0.918918919	1
but	0.7	0.8	0.746666667	1
can	0.7	0.8	0.746666667	1
can't	0.65	0.85	0.736666667	0.997076023
clever	0.7	0.8	0.746666667	1
coffee	0.85	1	0.918918919	1
confidence	0.85	1	0.918918919	1
deaf	0.6	0.9	0.72	1
don't_know	0.95	1	0.974358974	1

Table B.17: Results of independent training (DTW-based) using 8 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.8	0.8	0.8	1
apple	0.5	0.5	0.5	1
ask	0.8	0.8	0.8	1
bat	1	1	1	1
before	1	1	1	1
believe	0.9	0.9	0.9	1
best	0.8	0.8	0.8	1
black	0.9	0.85	0.874285714	0.994152047
bottle	0.8	0.8	0.8	1
boy	0.9	0.9	0.9	1
brown	0.8	0.8	0.8	1
busy	0.6	0.6	0.6	1
but	0.8	0.75	0.774193548	0.994152047
can	0.6	0.6	0.6	1
can't	0.6	0.6	0.6	1
clever	0.7	0.7	0.7	1
coffee	1	1	1	1
confidence	0.7	0.7	0.7	1
deaf	0.5	0.5	0.5	1
don't_know	0.9	0.9	0.9	1

Table B.18: Results of independent training (DTW-based) using 9 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.377776	0.360239	0.368799139	0.959649123
apple	0.433333	0.34806	0.386043602	0.959064327
ask	0.5	0.400286	0.444620932	0.954385965
bat	0.277777	0.177584	0.216657776	0.934502924
before	0.655556	0.530098	0.586189436	0.959064327
believe	0.355556	0.4095237	0.380636445	0.971345029
best	0.333333	0.257476	0.290534665	0.957309942
black	0.388889	0.540931	0.452479223	0.974853801
bottle	0.366667	0.2624639	0.305935858	0.943859649
boy	0.622223	0.491153	0.548973021	0.968421053
brown	0.533334	0.407972	0.46230522	0.953216374
busy	0.622223	0.341447	0.440931391	0.939181287
but	0.433332	0.425459	0.429359412	0.961988304
can	0.355555	0.23188	0.280698608	0.947953216
can't	0.622223	0.670504	0.645461896	0.98128655
clever	0.266666	0.4100943	0.323181506	0.96374269
coffee	0.7	0.573925	0.630723944	0.965497076
confidence	0.755556	0.505216	0.605532134	0.958479532
deaf	0.78889	0.556842	0.652859686	0.959064327
don't_know	0.355555	0.375531	0.365270091	0.980701754

Table B.19: Results of joint training (DTW-based) using 1 training sample

Sign name	Recall	Precision	F-score	Specificity
about	0.7125	0.670267	0.690738552	0.976973684
apple	0.6375	0.569394	0.601525362	0.984868421
ask	0.85	0.738369	0.790261772	0.978947368
bat	0.5125	0.636158	0.567672841	0.972368421
before	0.8375	0.695374	0.759848135	0.973684211
believe	0.8125	0.752138	0.78115465	0.981578947
best	0.8	0.731305	0.764111656	0.979605263
black	0.85	0.78627	0.816893911	0.985526316
bottle	0.775	0.675164	0.721645414	0.975657895
boy	0.9625	0.71003	0.817209706	0.976315789
brown	0.675	0.660792222	0.667820553	0.980263158
busy	0.85	0.729653	0.785242139	0.974342105
but	0.65	0.73183	0.688492072	0.983552632
can	0.8375	0.612956	0.707847256	0.965789474
can't	0.7125	0.682992	0.697434023	0.982894737
clever	0.75	0.57544	0.651225254	0.969736842
coffee	0.9125	0.780404	0.841298325	0.986184211
confidence	0.8875	0.694624	0.779305288	0.979605263
deaf	0.95	0.808207	0.873385955	0.980921053
don't_know	0.9	0.728747	0.805370386	0.976315789

Table B.20: Results of joint training (DTW-based) using 2 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.657143	0.735714	0.694212407	0.984962406
apple	0.914285	0.727692	0.810386358	0.978947368
ask	0.800001	0.723431	0.759791738	0.981203008
bat	0.628572	0.59058	0.608984034	0.97518797
before	0.871429	0.842152	0.856540397	0.987969925
believe	0.9	0.847894	0.873170341	0.988721805
best	0.971429	0.88308	0.925150022	0.990977444
black	0.9	0.892406	0.896186913	0.992481203
bottle	0.8	0.850183	0.824328453	0.990977444
boy	0.985714	0.744729	0.84844147	0.978947368
brown	0.885714	0.762499	0.819500925	0.981203008
busy	0.9	0.924167	0.911923415	0.994736842
but	0.828571	0.859325	0.843667826	0.993233083
can	0.785715	0.657658	0.716005849	0.968421053
can't	0.799999	0.985714	0.883199276	0.99924812
clever	0.771428	0.728571	0.749387259	0.981954887
coffee	0.871429	0.773331	0.819454583	0.980451128
confidence	0.885715	0.830104	0.857008303	0.987969925
deaf	0.957142	0.888636	0.921617701	0.990977444
don't_know	1	0.81255	0.896582163	0.984210526

Table B.21: Results of joint training (DTW-based) using 3 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.699999	0.810724	0.751303832	0.985964912
apple	0.9	0.754047	0.820584058	0.976315789
ask	0.966667	0.915714	0.940500892	0.994736842
bat	0.949999	0.680555	0.793014607	0.972807018
before	0.883332	0.779763	0.828322628	0.985087719
believe	0.949999	0.95	0.9499995	0.996491228
best	0.95	0.944047	0.947014145	0.996491228
black	1	0.891868	0.942843792	0.989473684
bottle	0.766666	0.848096	0.805327804	0.987719298
boy	0.983333	0.775973	0.867432792	0.981578947
brown	0.766667	0.821428	0.79310336	0.996491228
busy	0.899999	0.795677	0.844628932	0.984210526
but	0.866666	0.858333	0.862479373	0.99122807
can	0.866665	0.836904	0.851524541	0.985964912
can't	0.633333	0.692381	0.661541985	0.992105263
clever	0.866666	0.790735	0.826961175	0.984210526
coffee	1	0.815202	0.898194251	0.984210526
confidence	1	0.813096	0.896914449	0.985964912
deaf	0.916666	0.944047	0.93015504	0.996491228
don't_know	0.983333	0.825475	0.897515721	0.987719298

Table B.22: Results of joint training (DTW-based) using 4 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.86	0.670813	0.753716071	0.974736842
apple	0.96	0.768055	0.853367283	0.982105263
ask	0.92	0.784444	0.846831553	0.984210526
bat	0.94	0.74748	0.832757959	0.98
before	0.88	0.833929	0.856345298	0.986315789
believe	0.92	0.862501	0.890323113	0.989473684
best	0.98	0.921428	0.949811868	0.994736842
black	0.98	0.890318	0.933008868	0.991578947
bottle	0.82	0.72365	0.768818061	0.976842105
boy	0.98	0.811905	0.888068173	0.98
brown	0.9	0.890476	0.89521267	0.987368421
busy	0.94	0.776627	0.850539319	0.983157895
but	0.82	0.93	0.871542857	0.995789474
can	0.84	0.665953	0.742918962	0.974736842
can't	0.76	0.861429	0.807542039	0.989473684
clever	0.78	0.793334	0.786610497	0.986315789
coffee	1	0.70258	0.825312173	0.974736842
confidence	0.98	0.811549	0.887855169	0.986315789
deaf	0.96	0.901389	0.929771735	0.991578947
don't_know	1	0.87619	0.934009882	0.991578947

Table B.23: Results of joint training (DTW-based) using 5 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.875	0.846667	0.860600366	0.988157895
apple	0.85	0.780477	0.813756281	0.984210526
ask	0.975	0.860001	0.913897022	0.989473684
bat	0.9	0.753334	0.820161685	0.978947368
before	0.825	0.770278	0.79670045	0.981578947
believe	0.975	0.937143	0.95569675	0.994736842
best	0.95	0.886667	0.917241558	0.992105263
black	0.95	0.91	0.929569892	0.992105263
bottle	0.875	0.838334	0.856274667	0.989473684
boy	1	0.92	0.958333333	0.994736842
brown	0.875	0.806667	0.839445176	0.993421053
busy	0.95	0.844444	0.894117398	0.986842105
but	0.875	0.98	0.924528302	0.998684211
can	0.775	0.746667	0.760569724	0.975
can't	0.6	0.734444	0.660449446	0.988157895
clever	0.975	0.800001	0.878873843	0.985526316
coffee	1	0.740477	0.850889727	0.975
confidence	1	0.80381	0.891235773	0.981578947
deaf	0.975	0.98	0.977493606	0.998684211
don't_know	1	0.80762	0.893572764	0.984210526

Table B.24: Results of joint training (DTW-based) using 6 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.833335	0.811667	0.822358294	0.985964912
apple	0.900001	0.836667	0.867179146	0.987719298
ask	1	0.875	0.933333333	0.99122807
bat	1	0.855	0.921832884	0.987719298
before	0.933334	0.841667	0.885133504	0.987719298
believe	0.966667	0.809524	0.88114413	0.98245614
best	1	0.925	0.961038961	0.994736842
black	0.966667	0.885	0.924032556	0.99122807
bottle	0.833335	0.85	0.841585008	0.987719298
boy	0.966667	0.9	0.932143012	0.992982456
brown	0.9	0.891667	0.895814122	0.992982456
busy	0.966667	0.9	0.932143012	0.989473684
but	0.966667	0.762857	0.852753344	0.978947368
can	0.900001	0.91	0.904972881	0.992982456
can't	0.866667	0.86	0.863320629	0.987719298
clever	0.866668	0.715834	0.784062732	0.973684211
coffee	0.966667	0.811667	0.882412082	0.985964912
confidence	1	0.885	0.938992042	0.99122807
deaf	1	0.925	0.961038961	0.994736842
don't_know	1	0.867857	0.92925422	0.985964912

Table B.25: Results of joint training (DTW-based) using 7 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.9	0.766668	0.828000778	0.981578947
apple	0.85	0.766667	0.806185751	0.986842105
ask	1	0.900001	0.947368975	0.992105263
bat	1	0.816668	0.899083377	0.984210526
before	0.9	0.816667	0.856310863	0.986842105
believe	0.95	0.823334	0.88214324	0.981578947
best	0.95	0.916667	0.933035887	0.994736842
black	1	0.933334	0.965517598	0.994736842
bottle	1	0.916667	0.956521921	0.992105263
boy	1	0.833334	0.909091306	0.984210526
brown	0.95	0.916667	0.933035887	0.992105263
busy	0.9	0.800001	0.847059384	0.984210526
but	1	0.766668	0.867925383	0.978947368
can	1	0.740001	0.850575373	0.973684211
can't	0.9	0.833334	0.865384975	0.994736842
clever	0.95	0.906667	0.927827823	0.989473684
coffee	1	0.800001	0.888889506	0.981578947
confidence	1	0.883334	0.938053473	0.989473684
deaf	0.95	1	0.974358974	1
don't_know	1	0.866667	0.92857162	0.986842105

Table B.26: Results of joint training (DTW-based) using 8 training samples

Sign name	Recall	Precision	F-score	Specificity
about	0.8	0.683333	0.737078458	0.984210526
apple	0.8	0.75	0.774193548	0.994736842
ask	1	1	1	1
bat	1	0.883333	0.938052909	0.984210526
before	1	0.883333	0.938052909	0.984210526
believe	1	0.95	0.974358974	0.994736842
best	1	0.9	0.947368421	0.989473684
black	1	0.95	0.974358974	0.994736842
bottle	1	0.9	0.947368421	0.989473684
boy	1	0.733333	0.846153624	0.968421053
brown	1	0.95	0.974358974	0.994736842
busy	0.9	0.833333	0.865384436	0.989473684
but	1	0.85	0.918918919	0.984210526
can	0.9	0.75	0.818181818	0.978947368
can't	0.9	0.566666	0.695454043	0.957894737
clever	1	0.9	0.947368421	0.989473684
coffee	1	0.766666	0.867924101	0.968421053
confidence	1	1	1	1
deaf	1	0.933333	0.965517063	0.989473684
don't_know	1	0.783333	0.878504463	0.973684211

Table B.27: Results of joint training (DTW-based) using 9 training samples

Bibliography

[Akyol and Alvarado 01] Akyol, S., and Alvarado, P., Finding Relevant Image Content for mobile Sign Language Recognition, Proc. IASTED Intl Conf. Signal Processing, Pattern Recognition and Application, pp. 48-52, 2001.

[Akyol and Canzler 02] Akyol, S. and Canzler, U., An Information Terminal Using Vision Based Sign Language Recognition, Proc. ITEA Workshop Virtual Home Environments, pp. 61-68, 2002.

[Al-Jarrah and Halawani 01] Al-Jarrah, O., and Halawani, A., Recognition of Gestures in Arabic Sign Language Using Neuro-Fuzzy Systems, Artificial Intelligence, vol. 133, pp. 117-138, Dec. 2001.

[Assan and Grobel 97] Assan, M., and Grobel, K., Video-Based Sign Language Recognition Using Hidden Markov Models, Proc. Gesture Workshop, pp. 97-109, 1997.

[Bauer and Kraiss 01] Bauer, B., and Kraiss, K., Towards an automatic Sign Language recognition system using subunits, Proc. of Intl Gesture Workshop, pp. 64-75, 2001.

[Bauer and Kraiss 02] Bauer, B., and Kraiss, K.F., Video-Based Sign Recognition Using Self-Organizing Subunits, Proc. Intl Conf. Pattern Recognition, vol. 2, pp. 434-437, 2002.

[Birk et al 97] Birk, H., Moeslund, T. B., and Madsen, C. B., Real-Time Recognition of Hand Alphabet Gestures Using Principal Component Analysis, In Proc. of The 10th Scandinavian Conf. on Image Analysis, 1997.

[Black and Jepson 98] Black, M.J., and Jepson, A.D., A Probabilistic Framework for Matching Temporal Trajectories: CONDENSATION-Based Recognition of Gesture and Expressions, Proc. of Fifth European Conf. On Computer Vision, pp. 909-924, 1998.

[Bowden and Sarhadi 02] Bowden, R., and Sarhadi, M., A Nonlinear Model of Shape and Motion for Tracking Fingerspelt American Sign Language, Image and Vision Computing, vol. 20, pp. 597-607, 2002.

[Brand and Mason 00] Brand, J. and Mason, J., A Comparative Assessment of Three Approaches to Pixel-Level Human Skin Detection, Proc. IEEE Intl Conf. Pattern Recognition, vol. 1, pp. 1056-1059, Sept. 2000.

[Brown et al. 01] Brown, D., Craw, I., and Lewthwaite, J., A som based approach to skin detection with application in real time systems, In Proc. of the British Machine Vision Conf., 2001.

[Chai and Ngan 99] Chai, D. and Ngan, K. N., Face Segmentation Using Skin-color Map in Videophone Applications, IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 551-564, Jun. 1999.

[Chen et al. 03] Chen, F. -S., Fu, C. -M., and Huang, C. -L., Hand Gesture Recognition using a Real-Time Tracking Method and Hidden Markov Models, Image and Vision Computing, vol. 21, pp. 745-758, 2003.

[Chui and Chen 99] Chui, C.K., and Chen, G., Kalman Filtering with Real-Time Applications, Springer, Berlin Heidelberg, 1999.

[Cui and Weng 99] Cui, Y., and Weng, J., A Learning-Based Prediction-and-Verification Segmentation Scheme for Hand Sign Image Sequence, IEEE Trans. Pattern Analysis Machine Intelligence, vol. 21, no. 8, pp. 798- 804, Aug. 1999.

[Cui and Weng 00] Cui, Y., and Weng, J., Appearance-Based Hand Sign Recognition from Intensity Image Sequences, Computer Vision Image Understanding, vol. 78, no. 2, pp. 157-176, 2000.

[Data clustering] Data clustering, from Wikipedia, the free encyclopedia, available online at URL: http://en.wikipedia.org/wiki/Data_clustering.

[Deng and Manjunath 01] Deng, Y. and Manjunath, B. S., Unsupervised Segmentation of Color-texture Regions in Images and Video, IEEE Trans. Pattern Anal. Machine Intell., vol. 23, pp. 800-810, 2001.

[Deng and Tsui 02] Deng, J.-W., and Tsui, H.T., A Novel Two-Layer PCA/MDA Scheme for Hand Posture Recognition, Proc. Intl Conf. Pattern Recognition, vol. 1, pp. 283-286, 2002.

[Downton and Drouet 92] Downton, A.C., and Drouet, H., Model-Based Image Analysis for Unconstrained Human Upper-Body Motion, Proc. Intl. Conf. Image Processing and Its Applications, pp. 274-277, Apr. 1992.

[ECHO] ECHO sign language database, available online at URL: http://www.let.ru.nl/sign-lang/echo/

[Fang et al. 04] Fang, G., Gao, X., Gao, W., Chen, Y., A novel approach to automatically extracting basic units from chinese Sign Language, Proc. of Intl. Conf. on Pattern Recognition, pp. 454-457, 2004.

[Foley et al. 90] Foley, J.D., Dam, A.V., Feiner, S.K., and Hughes, J.F., Computer Graphics: Principles and Practice. New York: Addison Wesley, 1990.

[Forsyth and Fleck 96] Forsyth, D. and Fleck, M., Identifying Nude Pictures. In IEEE Workshop on applications of computer vision, 1996.

[Freund and Schapire 95] Freund, Y., and Schapire, R.E, A decision-theoretic generalization of online learning and an application to boosting. In Computational Learning Theory (Eurocolt), 1995.

[Gao et al 00] Gao, W., Ma, J., Wu, J., and Wang, C., Sign Language Recognition Based on HMM/ANN/DP, Intl. J. Pattern Recognition Artificial Intelligence, vol. 14, no. 5, pp. 587-602, 2000.

[Gavrila 99] Gavrila, D., The Visual Analysis of Human Movement: A Survey, Computer Vision Image Understanding, vol. 73, no. 1, pp. 82-98, Jan. 1999.

[Gupta and Ma 01] Gupta, L., and Ma, S., Gesture-Based Interaction and Communication: Automated Classification of Hand Gesture Contours, IEEE Trans. Systems, Man, and Cybernetics, Part C: Application Rev., vol. 31, no. 1, pp. 114-120, Feb. 2001.

[Handouyahia et al 99] Handouyahia, M., Ziou, D., and Wang, S., Sign Language Recognition Using Moment-Based Size Functions, Proc. Intl Conf. Vision Interface, pp. 210-216, 1999.

[Hernandez et al. 02] Hernandez-Rebollar, J.L., Lindeman, R.W., and Kyriakopoulos, N., A Multi-Class Pattern Recognition System for Practical Finger Spelling Translation, Proc. Intl. Conf. Multimodal Interfaces, pp. 185-190, 2002.

[Hernandez et al. 04] Hernandez-Rebollar, J.L., Kyriakopoulos, N., and Lindeman, R.W., A New Instrumented Approach for Translating American Sign Language into Sound and Text, Proc. Intl Conf. Automatic Face and Gesture Recognition, pp. 547-552, 2004.

[Hienz et al. 96] Hienz, H., Grobel, K., and Offner, G., Real-Time Hand-Arm Motion Analysis Using a Single Video Camera, Proc. Intl. Conf. Automatic Face and Gesture Recognition, pp. 323-327, 1996.

[Holden and Owens 00] Holden, E.-J., and Owens, R., Visual Sign Language Recognition, Proc. Intl Workshop Theoretical Foundations of Computer Vision, pp. 270-287, 2000.

[Hsu et al. 98] Hsu, F., Lee, S., and Lin, B., Video Data Indexing by 2D C-Trees. Journal of Visual Languages and Computing, vol. 9, pp. 375-397, 1998.

[Hsu et al. 02] Hsu, R.-L., Abdel-Mottaleb, M., and Jain, A.K., Face Detection in Colour Images, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp. 696- 707, May 2002.

[Hu 62] Hu, M. K., Visual pattern recognition by moment inariants, IRE Trans. Inf. Theory, vol.IT-8, pp.179-187, Feb. 1962.

[Huang and Huang 98] Huang, C.-L., and Huang, W.-Y., Sign Language Recognition Using Model-Based Tracking and a 3D Hopfield Neural Network, Machine Vision and Application, vol. 10, pp. 292-307, 1998.

[Huang and Jeng 01] Huang, C.-L., and Jeng, S.-H., A Model-Based Hand Gesture Recognition System, Machine Vision and Application, vol. 12, pp. 243-258, 2001.

[Huang et al 90] Huang, X. D., Ariki, Y., and Jack, M. A., Hidden Markov Models for Speech Recognition.Edinburgh University Press, Edinburgh, 1990.

[Imagawa 00] Imagawa, K., Matsuo, H., Taniguchi, R.-i, Arita, D., Lu, S., and Igi, S., Recognition of Local Features for Camera-Based Sign Language Recognition System, Proc. Intl. Conf. Pattern Recognition, vol 4, pp. 849-853, 2000.

[Imagawa and Igi 98] Imagawa, K., Lu, S., and Igi, S., Color-Based Hand Tracking System for Sign Language Recognition, Proc. of IEEE Intl. Conf. Automatic Face and Gesture Recognition, pp. 462-467, 1998.

[Isard and Blake 98] Isard, M., and Blake, A., CONDENSATION-Conditional Density Propagation for Visual Tracking, International Journal of Computer Vision, vol. 29, pp. 5-28, 1998.

[Jedynak et al. 02] Jedynak, B., Zheng, H., Daoudi, M., and Barret, D., Maximum entropy models for skin detection. Tech. Rep. XIII, Universite des Sciences et Technologies de Lille, France, 2002.

[Joachims 98] Joachims, T., Text categorization with support vector machines, In Proc. of the European Conference on Machine Learning, Springer-Verlag, 1998.

[Jones and Rehg 02] Jones, M.J. and Rehg, J.M., Statistical Colour Models with Application to Skin Detection, Intl J. Computer Vision, vol. 46, no. 1, pp. 81-96, Jan. 2002.

[Jose and Luis 04] Jose, V., and Luis, S., Feature selection for visual gesture recognition using hidden markov models, In Proc. of Fifth Mexican Int. Conf. in Computer Science (ENC'04), 2004.

[Kadir et al. 02] Kadir, T., Bowden, R., Ong, E., and Zisserman, A., Minimal Training, Large Lexicon, Unconstrained Sign Language Recognition, in Proc. of British Machine Vision Conference, pp. 849-858, 2002.

[Kennaway 03] Kennaway, R., Experience with and Requirements for a Gesture Description Language for Synthetic Animation, Proc. Gesture Workshop, pp. 300-311, 2003.

[Koizumi et al. 02] Koizumi, A., Sagawa, H., and Takeuchi, M., An Annotated Japanese Sign Language Corpus, Proc. Intl. Conf. Language Resources and Evaluation, vol. III, pp. 927-930, 2002.

[Kong et al. 04] Kong, S.G., Heo, J., Abidi, B.R., Paik, J., and Abidi, M.A., Recent Advances in Visual and Infrared Face Recognition A Review, Computer Vision Image Understanding, 2004.

[Kramer and Leifer 87] Kramer, J., and Leifer, L., The Talking Glove: An Expressive and Receptive Verbal Communication Aid for the Deaf, Deaf-Blind, and Nonvocal, Proc. Third Ann. Conf. Computer Technology, Special Education, Rehabilitation, pp. 335-340, Oct. 1987.

[Kruppa et al. 02] Kruppa, H., Bauer, M. A. and Schiele, B., Skin patch detection in real-world images, In Annual Symposium for Pattern Recognition of the DAGM 2002, Springer LNCS 2449, pp. 109-117.

[Lee and Xu 00] Lee, C., and Xu, Y., Trajectory fitting with smoothing splines using velocity information, Proc. of IEEE Conf. on Robotics and Automation, pp. 2796-2801, 2000.

[Lee and Yoo 02] Lee, J. Y., and Yoo, S. I., An elliptical boundary model for skin colour detection. In proc. of the Intl. Conf. on Imaging Science, Systems, and Technology, 2002.

[Liang and Ouhyoung 98] Liang, R.-H, and Ouhyoung, M., A Real-time Continuous Gesture Recognition System for Sign Language, Proc. of the Third Intl Conf. on Automatic Face and Gesture Recognition, pp. 558-565, 1998.

[Liddell and Johnson 89] Liddell, S. and Johnson, R., American Sign Language: The phonological base, Sign Language Studies, vol. 64, pp. 195-277, 1989.

[Lien et al. 98] Lien, J. J., Kanade T., Cohn, J.F., and Li, C-C., Automated Facial Expression Recognition Based on FACS Action Units, Proc. of FG'98, pp.390-395, 1998.

[Lockton and Fitzgibbon 02] Lockton, R., and Fitzgibbon, A., Real-time Gesture Recognition Using Deterministic Boosting, in Proc. of British Machine Vision Conf., pp. 817-826, 2002.

[Lu et al. 03] Lu, S., Metaxas, D., Samaras, D., and Oliensis, J., Using Multiple Cues for Hand Tracking and Model Refinement, Proc. of Conf. on Computer Vision and Pattern Recognition, pp. 443-450, 2003.

[Lv and Nevatia 06] Lv, F., and Nevatia, R., Recognition and Segmentation of 3-D Human Action Using HMM and Multi-class Adaboost, in Proc. of European Conference on Computer Vision, pp. 359-372, 2006.

[Mammen et al 01] Mammen, J., Chaudhuri, S., and Agrawal, T., Simultaneous Tracking of Both Hands by Estimation of Erroneous Observations, Proc. of British Machine Vision Conf., pp. 83-92, 2001.

[Martin et al. 98] Martin, J., Devin, V., and Crowley, J., Active Hand Tracking, Proc. of IEEE Intl Conf. Automatic Face and Gesture Recognition, pp. 573-578, 1998.

[Matsuo et al. 97] Matsuo, H., Igi, S., Lu, S., Nagashima, Y., Takata, Y., and Teshima, T., The Recognition Algorithm with Non-Contact for Japanese Sign Language Using Morphological Analysis, Proc. Gesture Workshop, pp. 273-285, 1997.

[McAllister et al. 02] McAllister, G., McKenna, S.J., and Picketts, I.W., Hand Tracking for Behaviour Understanding, Image and Vision Computing, vol. 20, pp. 827-840, 2002.

[McGuire et al. 04] McGuire, R.M., Hernandez-Rebollar, J., Starner, T., Henderson, V., Brashear, H., and Ross, D.S., Towards a One-Way American Sign Language Translator, Proc. Intl Conf. Automatic Face and Gesture Recognition, pp. 620-625, 2004.

[Mckenna et al. 99] Mckenna, S. J., Raja, Y., Gong, S., Tracking Colour Objects using Adaptive Mixture Models, Image and Vision Computing, vol. 17, pp. 225-231, 1999.

[Min et al. 97] Min, B. W., Ho-Sub, Y., Jung, S., Yun-Mo, Y., and Toshiaki, E., Hand Gesture Recognition Using Hidden Markov Models, IEEE Intl. Conf. On Systems, Man, And Cybernetics: 'Computational Cybernetics And Simulation', vol. 5, pp. 4232-4235, 12-15 Oct. 1997.

[Murakami and Taguchi 91] Murakami, K., and Taguchi, H., Gesture Recognition Using Recurrent Neural Networks, Proc. SIGCHI Conf. Human Factors in Computing Systems, pp. 237- 242, 1991.

[Ng and Gong 02] Ng, J., and Gong, S., Learning Intrinsic Video Content using Levenshtein Distance in Graph Partitioning, Proc. of European Conf. on Computer Vision, pp. 670-684, 2002.

[Ong and Bowden 04] Ong, E.-J., and Bowden, R., A Boosted Classifier Tree for Hand Shape Detection, Proc. Intl Conf. Automatic Face and Gesture Recognition, pp. 889-894, 2004.

[Ong and Ranganath 05] Ong, S. C. W., and Ranganath, S., Automatic Sign Language Analysis: A Survey and the Future beyond Lexical Meaning, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 27, no. 6, June 2005.

[Opelt-CVPR 06] Opelt, A., Pinz, A., and Zisserman, A., Incremental learning of object detectors using a visual alphabet. In Proc. CVPR, 2006.

[Opelt et al. 06] Opelt, A., Pinz, A., and Zisserman, A., A Boundary-Fragment-Model for Object Detection, in Proc. of European Conf. on Computer Vision, pp. 575-588, 2006.

[Opelt-PAMI 06] Opelt, A., Fussenegger, M., Pinz, A., and Auer, P., Generic object recognition with boosting. PAMI, 28(3), 2006.

[Pakchalakis and Lee 99] Pakchalakis, S., and Lee, P., Pattern recognition in gray level images using moment based invariant features, IEE Conf. Publication on Image Processing and its Applications, no.465, pp.245-249, 1999.

[Pantic and Rothkrantz 00] Pantic, M., and Rothkrantz, L.J.M., Automatic Analysis of Facial Expressions: The State of the Art, IEEE Trans. Pattern Analysis Machine Intelligence, vol. 22, no. 12, pp. 1424-1445, Dec. 2000.

[Papageorgiou et al. 98] Papageorgiou, C., Oren, M., and Poggio, T., A general framework for objec detection, In Proc. of the Intl. Conf. on Computer Vision, 1998.

[Pavlovic et al. 97] Pavlovic, V.I., Sharma, R., and Huang, T.S., Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review, IEEE Trans. Pattern Analysis Machine Intelligence, vol. 19, no. 7, pp. 677-695, July 1997.

[Peer 03] Peer, P., Kovac, J., and Solina, F., Human skin colour clustering for face detection. In Intl. Conf. on Computer as a Tool, EUROCON 2003.

[Phung et al. 02] Phung, S. L., Bouzerdoum, A. and Chai, D., A novel skin colour model in ycbcr colour space and its application to human face detection, In IEEE Intl. Conf. on Image Processing ICIP 2002, vol. 1, 289 292.

[Phung et al. 05] Phung, S. L., Bouzerdoum A., Chai, D., Skin segmentation using colour pixel classification: analysis and comparison, In IEEE Trans. on Pattern analysis and Machine Intelligence, vol. 27, no. 1, Jan. 2005.

[Reeves et al. 88] Reeves, A. P., Prokop, R. J., Andrews, S. E., and Kuhl, F., Three-dimensional shape analysis using moments and Fourier descriptors, IEEE Transacfion on Pattern Analysis and Machine Intelligence, vol.10, no.6, pp.937-943, Nov. 1988.

[Rehg and Kanade 94] Rehg J., and Kanade, T., Visual Tracking of High DOF Articulated Structures: an Application to Human Hand Tracking, Proc. of Third European Conf. On Computer Vision, pp. 35-46, 1994.

[Rosten and Drummong 06] Rosten, E., and Drummong, T., Machine learning for high-speed corner detection, Proc. European Conf. on Computer Vision, 2006.

[Sato et al. 00] Sato, Y., Kobayashi, Y., and Koike, H., Fast Tracking of Hands and Fingertips in Infrared Images for Augmented Desk Interface, Proc. of IEEE Intl Conf. Automatic Face and Gesture Recognition, pp. 429-434, 2000.

[Schohn and Cohn 00] Schohn, G. and Cohn, D., Less is More: Active Learning with Support Vector Machines, Proc. of Intl Conf. on Machine Learning, pp. 839-846, 2000.

[Shamaie and Sutherland 05] Shamaie, A., and Sutherland, A., Hand Tracking in Bimanual movements, Image and Vision Computing, vol. 23, pp. 1131-1149, 2005.

[Sherrah 00] Sherrah, J., and Gong, S., Resolving Visual Uncertainty and Occlusion through Probabilistic Reasoning, Proc. British Machine Vision Conf., pp. 252-261, 2000.

[Sherrah and Gong 00] Sherrah, J., and Gong, S., Resolving Visual Uncertainty and Occlusion through Probabilistic Reasoning, Proc. European Conf. Computer Vision, vol. 2, pp. 150-166, 2000.

[Sigal et al. 04] Sigal, L., Sclaroff, S., and Athitsos, V., Skin Color-Based Video Segmentation under Time-Varying Illumination, IEEE Trans. On Pattern Anal. Machine Intell., vol. 26, no. 7, pp. 862-877, Jul. 2004.

[SLR group] Sign Language Recognition research group, http://www.ee.surrey.ac.uk/Personal/R.Bowden/sign.

[Soriano et al. 03] Soriano, M., Martinkauppi, B., Huovinen, S., Laaksonen, M., Adaptive Skin Color Modeling using the Skin Locus for Selecting Training Pixels, Pattern Recognition, vol. 36, pp. 681-690, 2003.

[Starner et al. 98] Starner, T., Weaver, J., Pentland, A., Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video, IEEE Trans. on Pattern Anal. and Machine Intell., vol. 20, no. 12, pp. 1371-1375, Dec. 1998.

[Stenger et al. 01] Stenger, B., Mendonca, P.R.S., and Cipolla, R., Model-Based 3D Tracking of an Articulated Hand, Proc. of Conf. on Computer Vision and Pattern Recognition, pp. 310-315, 2001.

[Stokoe 78] Stokoe, W., Sign Language Structure: An Outline of the Visual Communication System of the American Deaf, Studies in Linguistics: Occasional papers 8, Linstok Press, MD, 1960, revised 1978.

[Strickon and Paradiso 98] Strickon, J., and Paradiso, J., Tracking Hands Above Large Interactive Surfaces with a Low-Cost Scanning Laser Rangefinder, Proc. of ACM CHI Conf., pp. 231-232, 1998.

[Sturman and Zeltzer 94] Sturman, D.J., and Zeltzer, D., A Survey of Glove-Based Input, IEEE Computer Graphics and Applications, vol. 14, pp. 30-39, 1994.

[Sutherland 96] Sutherland, A., Real-Time Video-Based Recognition of Sign Language Gestures Using Guided Template Matching, Proc. Gesture Workshop, pp. 31-38, 1996.

[Sweeney and Downton 96] Sweeney, G.J., and Downton, A.C., Towards Appearance-Based Multi-Channel Gesture Recognition, Proc. Gesture Workshop, pp. 7- 16, 1996.

[Tanibata et al. 02] Tanibata, N., Shimada, N., and Shirai, Y., Extraction of Hand Features for Recognition of Sign Language Words, Proc. Intl. Conf. Vision Interface, pp. 391-398, 2002.

[Terrillon 00] Terrillon, J.-C., Shirazi, J.-C., Fukamachi, H., and Akamatsu, S., Comparative Performance of Different Skin Chrominance Models and Chrominance Spaces for the Automatic Detection of Human Faces in Colour Images, Proc. IEEE Intl Conf. Automatic Face and Gesture Recognition, pp. 54-61, Mar. 2000.

[Terrillon et al. 02] Terrillon, J.-C., Piplr, A., Niwa, Y., and Yamamoto, K., Robust Face Detection and Japanese Sign Language Hand Posture Recognition for Human-Computer Interaction in an Intelligent Room, Proc. Intl Conf. Vision Interface, pp. 369-376, 2002.

[Tieu and Viola 04] Tieu, K., and Viola, P., Boosting Image Retrieval, International Journal of Computer Vision, vol. 56, pp. 17-36, 2004.

[Tong and Chang 01] Tong, S. and Chang, E., Support Vector Machine Active Learning for Image Retrieval, Proc. of ACM Multimedia, pp. 107-118, 2001.

[Tong and Koller 01] Tong, S. and Koller, D., Support Vector Machine Active Learning with Applications to Text Classification, Journal of Machine Learning Research, pp. 45-66, 2001.

[Torralba et al. 04] Torralba, A., Murphy, K. P., and Freeman, W. T., Sharing features: efficient boosting procedures for multiclass object detection. In Proc. CVPR, 2004.

[Vamplew and Adams 98] Vamplew, P., and Adams, A., Recognition of Sign Language Gestures Using Neural Networks, Australian J. Intelligence Information Processing Systems, vol. 5, no. 2, pp. 94-102, 1998.

[Vapnik 95] Vapnik, V., The Nature of Statistical Learning Theory, Springer, New York, 1995.

[Vapnik 98] Vapnik, V., Statistical Learning theory. Wiley, 1998.

[Veltman and Prasad 94] Veltman, S. R., and Prasad, R., Hidden Markov Models Applied to On-line Handwritten Isolated Character Recognition, IEEE Transactions on Image Processing, 314-318, 1994.

[Vezhnevets et al. 03] Vezhnevets, V., Sazonov, V. and Andreeva, A, A Survey on Pixel-Based Skin Colour Detection Techniques, Graphicon2003, In 13th Intl. Conf. on the Computer Graphics and Vision, Moscow, Russia, Sept.2003.

[Viola and Jones 01] Viola, P., and Jones, M.J., Robust real-time object detection. In Proc. of IEEE Workshop on Statistical and Theories of Computer Vision, 2001.

[Vogler 03] Vogler, C., American Sign Language Recognition: Reducing the Complexity of the Task with Phoneme-Based Modeling and Parallel Hidden Markov Models, PhD thesis, Univ. of Pennsylvania, 2003.

[Vogler and Metaxas 97] Vogler, C., and Metaxas, D., Adapting Hidden Markov Models for ASL Recognition by Using Three-Dimensional Computer Vision Methods, Proc. Intl Conf. Systems, Man, Cybernetics, vol. 1, pp. 156- 161, 1997.

[Vogler and Metaxas 98] Vogler, C., and Metaxas, D., ASL Recognition Based on A Coupling between HMMs and 3D Motion Analysis, in Proc. Intl. Conf. on Computer Vision, pp. 363-369, 1998.

[Vogler and Metaxas 99] Vogler, C., and Metaxas, D., Toward scalability in ASL recognition: Breaking down signs into phonemes, Proc. of the Gesture Workshop, pp. 211-224, 1999.

[Waldron and Kim 95] Waldron, M.B., and Kim, S., Isolated ASL Sign Recognition System for Deaf Persons, IEEE Trans. Rehabilitation Eng., vol. 3, no. 3, pp. 261-271, Sept. 1995.

[Wang et al. 02] Wang, C., Gao, W., and Shan, S., An Approach Based on Phonemes to Large Vocabulary Chinese Sign Language Recognition, Proc. Intl. Conf. Automatic Face and Gesture Recognition, pp. 393-398, 2002.

[Wang 03] Wang, L., Chan, K. L., and Zhang, Z., Bootstrapping SVM Active Learning by Incorporating Unlabelled Images for Image Retrieval, Proc. Conf. Computer Vision Pattern Recognition, vol. 1, pp. 629-634, 2003.

[Wang et al. 03] Wang, L., Hu, W., and Tan, T., Recent Developments in Human Motion Analysis, Pattern Recognition, vol. 36, pp. 585- 601, 2003.

[Wang and Brandstein 99] Wang, C. and Brandstein, M., Multi-Source Face Tracking with Audio and Visual Data. IEEE MMSP, pp. 168, 1999.

[Wu and Huang 00] Wu, Y., and Huang, T.S., View-Independent Recognition of Hand Postures, Proc. Conf. Computer Vision Pattern Recognition, vol. 2, pp. 88-94, 2000.

[Wu et al 00] Wu, Y. and Huang, T. S., Color Tracking by Transductive Learning, Proc. Conf. On Computer Vision and Pattern Recognition, pp. 133-138, 2000.

[Xiang and Gong 04] Xiang, T., and Gong, S., Activity Based Video Content Trajectory Representation and Segmentation, Proc. Conf. BMVC, 2004.

[Yang et al. 98] Yang, J., Lu, W., and Waibel, A., Skin-color Modeling and Adaptation, Proc. of Asia Conf. Computer Vision, pp. 687-694, 1998.

[Yang et al. 02] Yang, M.-H., Ahuja, N., and Tabb, M., Extraction of 2D Trajectories and Its Application to Hand Gesture Recognition, IEEE Trans. On Pattern Anal. and Machine Intell., vol. 24, no. 8, pp. 1061-1074, Aug. 2002.

[Yang and Ahuja 98] Yang, M. H., and Ahuja, N., Detecting human faces in colour images, In Intl. Conf. on Image Processing ICIP, vol.1, pp.127-130, 1998.

[Yang and Waibel 96] Yang, J. and Waibel, A., A Real-Time Face Tracker, Proc. IEEE Workshop Applications of Computer Vision, pp. 142-147, Dec. 1996.

[Yeasin and Chaudhuri 00] Yeasin, M., and Chaudhuri, S., Visual understanding of dynamic hand gestures, Pattern Recognition, vol. 33, pp. 1805-1817, 2000.

[Yoon et al. 99] Yoon, H., Soh, J., Ming, B., and Yang, H., Recognition of alphabetical hand gestures using hidden markov model. IEEE Trans. Fundamentals, vol. 82(7), pp. 1358-1366, Jul 1999.

[Yuan et al. 02] Yuan, Q., Gao, W., Yao, H., and Wang, C., Recognition of Strong and Weak Connection Models in Continuous Sign Language, Proc. Intl Conf. Pattern Recognition, vol. 1, pp. 75-78, 2002.

[Zarit et al. 99] Zarit, B. D., Super, B. J., AND Quek, F. K. H., Comparison of five colour models in skin pixel classification. In ICCV Intl Workshop on recognition, analysis and tracking of faces and gestures in Real-Time systems, pp. 58 - 63, 1999.

[Zhang and Lu 01] Zhang, D. S., and Lu, G. J., A Comparative Study on Shape Retrieval Using Fourier Descriptors with Different Shape Signatures, In Proc. Intl. Conf. on Multimedia and Distance Education. Fargo, ND, USA, pp.1-9, June 2001.

[Zhu et al. 00] Zhu, X., Yang, J., and Waibel, A., Segmenting Hands of Arbitrary Colour, Proc. IEEE Intl Conf. Automatic Face and Gesture Recognition, pp. 446-453, Mar. 2000.

[Zhu et al. 04] Zhu, Q., Wu, C. T., Cheng, K. T., and Wu, Y. L., An Adaptive Skin Model and Its Application to Objectionable Image Filtering, Proc. of ACM Multimedia, pp. 56-63, 2004.

[Zieren et al. 02] Zieren, J., Unger, N., and Akyol, S., Hands Tracking from Frontal View for Vision-Based Gesture Recognition, Proc. 24th DAGM Symp., pp. 531-539, 2002.

List of publications from this work

1. H. Jeff, A. George, S. Alistair, Subunit Boundary Detection for Sign Language Recognition Using Spatio- temporal Modelling, Proc. 5th International Conference on Computer Vision systems ICVS2007, Germany, 21-24 March, 2007. (first co-author, poster presentation).

2. C. Thommas, A. George, H. Jeff, S. Alistair, Real Time Hand Gesture Recognition Including Hand Segmentation and Tracking, Proc. 2nd International Symposium on Visual Computing, ISVC06, Nevada, USA, Nov. 6-8, 2006. (oral presentation).

3. H. Jeff, A. George, S. Alistair, ASST: Automatic skin segmentation and tracking for sign language recognition, Submitted to IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans. (under review).

4. A. George, H. Jeff, S. Alistair, A unified system for segmentation and tracking of face and hands in sign language recognition, Proc. 18th International Conference on Pattern Recognition, ICPR2006, Hong Kong, August 20-24, 2006. (poster presentation).

5. A. George, C. Tommy, H. Jeff, S. Alistair, Real-Time Hand Gesture Segmentation, Tracking and Recognition, 9th European Conference on Computer Vision, ECCV2006, Graz, Austria, May 7 - 13, 2006. (Demo presentation).

6. H. Jeff, A. George, S. Alistair, Automatic Skin Segmentation for Gesture Recognition Combining Region and Support Vector Machine Active Learning, Proc. 7th International Conference on Automatic Face and Gesture Recognition, FG2006, Southampton, UK, April 10-12, 2006. (oral presentation).