TOWARDS SUBJECT INDEPENDENT SIGN LANGUAGE RECOGNITION: A SEGMENT-BASED PROBABILISTIC APPROACH

KONG WEI WEON

(B.Eng.(Hons.),M.Eng., NUS)

A THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE

2011

Acknowledgements

I owe my deepest gratitude to my supervisor, Prof. Surendra Ranganath for his unceasing support and persistence in guiding me through all these years to make this thesis possible. It is never an easy task to keep in close touch to work on the thesis across the miles. I am truly grateful for his constant encouragement and teachings during this long journey which is marked by many changes and obstacles. In addition to the valuable technical knowledge, I have also learned from him the importance of being patient, thoughtful and conscientious. I sincerely wish him happiness everyday.

Special thanks goes to my current supervisor Assoc. Prof. Ashraf Kassim who has granted me the opportunity to continue to work with the project smoothly. I am thankful for his assistance and advice.

I would like to express thanks to the members of the Deaf & Hard-of-Hearing Federation (Singapore) for providing the sign data. Also, a big thanks goes to Angela Cheng who has consistently offered her time and help for my thesis work.

On a personal note, I would like to thank my parents for their unlimited love and support. I wish to offer my heartfelt gratitude and appreciation to Tzu-Chia who has constantly supported and encouraged me at difficult times to work on completing my thesis. I am also grateful and thankful to A-Zi, Yuru and Siew Pheng who have reminded me that there is a real magic in enthusiasm. I would like to dedicate this thesis my loving niece Gisele, who has accompanied me throughout the writing process and helped me to stay lighthearted.

Lastly, I offer my regards and blessings to all of those who have showed me their kind gestures and supported me in any respect during the completion of the thesis, especially to my neighbour in Dharamsala who has encouraged me to have faith in myself.

Kong Wei Weon

18 July 2011

Contents

Summary

This thesis presents a segment-based probabilistic approach to recognize continuous sign language sentences which are signed naturally and freely. We aim to devise a recognition system that can robustly handle the inter-signer variations exhibited in the sentences. In preliminary work, we considered isolated signs which provided insight into inter-signer variations. Based on this experience, we tackled the more difficult problem of recognizing continuously signed sentences as outlined above. Our proposed scheme has kept in view the major issues in continuous sign recognition including signer independence, dealing with movement epenthesis, segmentation of continuous data, as well as scalability to large vocabulary.

We use a discriminative approach rather than a generative one to better handle signer variations and achieve better generalization. For this, we propose a new scheme based on a two-layer conditional random field (CRF) model, where the lower layer processes the four parallel channels (handshape, movement, orientation and location) and its outputs are used by the higher level for sign recognition. We use a phoneme-based scheme to model the signs, and propose a new PCA-based representation phoneme transcription procedure for the movement component. k-means clustering together with affinity propagation (AP) is used to transcribe phonemes for the other three components.

The basic idea of the proposed recognition framework is to first over-segment the continuously signed sentences with a segmentation algorithm based on minimum velocity and maximum change of directional angle. The sub-segments are then classified as sign or movement epenthesis. The classifier for labeling the sub-segments of an input sentence as sign or movement epenthesis is obtained by fusing the outputs of independent CRF and SVM classifiers through a Bayesian network. The movement epenthesis sub-segments are discarded and the recognition is done by merging the sign sub-segments. For this purpose, we propose a new decoding algorithm for the two-layer CRF-based framework, which is based on the semi-Markov CRF decoding algorithm and can deal with segment-based data, compute features for recognition on the fly, discriminate between possibly valid and invalid segments that can be obtained during the decoding procedure, and merge sub-segments that are not contiguous. We also take advantage of the information given by the location of movement epenthesis sub-segments to reduce the complexity of the decoding search.

A glove and magnetic tracker-based approach was used for the work and raw data was obtained from electronic gloves and magnetic trackers. The data used for the experiments was contributed by seven deaf native signers and one expert signer and consisted of 74 distinct sentences made up from a 107-sign vocabulary. Our proposed scheme achieved a recall rate of 95.7% and precision accuracy of 96.6% for unseen samples from seen signers, and a recall rate of 86.6% and precision accuracy of 89.9% for unseen signers.

List of Tables

List of Figures

The limits of my
language mean the
limits of my world.
Ludwig Wittgenstein
(1889-1951)

1
Introduction

Sign language is widely used by the deaf for communication and as the language of instruction in schools for the deaf. In recent years, there has been increasing interest in developing sign language systems to aid communication between the deaf and hearing people.

Sign language is a rich and expressive language with its own grammar, rhythm and syntax, and is made up of manual and non-manual signals. Manual signing involves hand and arm gestures, while non-manual signals are conveyed through facial expressions, eye gaze direction, head movements, upper torso movements and mouthing. Non-manual signals are important in many areas of sign language structure including phonology, morphology, syntax, semantics and discourse analysis. For example, they are frequently used in sentences that involve "yes-no questions", "wh-questions", "negation", "commands", "topicalization" and "conditionals". In manual signing, four key components are used to compose signs, namely, handshape, movement, palm orientation and location; the systematic change of these components produces a large number of different sign appearances. Generally, the appearance and meaning of basic signs are well-defined in sign language dictionaries. For example, when signing TREE, the rule is "The elbow of the upright right arm rests on the palm of the upturned left hand (this is the trunk) and twisted. The fingers of the right hand with handshape "5" wiggle to imitate the movement of the branches and leaves." [136]. Figure 1.1 shows the appearance of the sign.

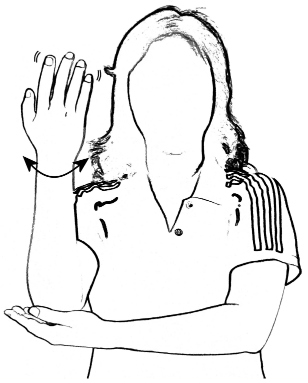


Figure 1.1: ASL sign: TREE.

Although rules are given for all basic signs, variations occur due to regional, social, and ethnic factors, and there are also differences which arise from gender, age, education and family background. This can lead to significant variations in manual signs performed by different signers, and poses challenging problems for developing robust computer-based sign language recognition systems.

In this thesis, we address manual sign language recognition that is robust to inter-signer variations. Most of the recent works in the literature have addressed the recognition of continuously signed sentences, with focus on obtaining high recognition accuracy and scalability to large vocabulary. Although these are important problems to consider, many works are based on data from only one signer. Some works attempted to demonstrate signer independence but they were mainly based on hand postures or isolated signs and hence limited in scope. This thesis considers the additional practical problem of recognizing continuous manually signed sentences that contain complex inter-signer variations which arise due to the reasons mentioned above. As part of this problem, we also consider approaches to deal with movement epenthesis (unavoidable hand movements between signs which carry no meaning) which presents additional complexity for sign recognition. The inter-signer variations in movement epentheses themselves are usually non-trivial and pose a challenge for accurate sign recognition. However, many works either neglect it or pay no special attention to the problem. In works that do consider it explicitly, the common approaches are either to model the movement epentheses explicitly, or assume that the movement epenthesis segments can be absorbed into their adjacent sign segments. In this thesis, we suggest that movement epenthesis needs to be handled explicitly, though without elaborate modeling these “unwanted” segments.

In the next section, the background of American sign language (ASL) is first presented followed by discussion of the nature of variations which arise in manual signing in Section 1.2. Section 1.3 describes movement epenthesis in more detail. Section 1.4 presents the motivation and Section 1.5 describes the research goals of this thesis.

1.1 Background of American Sign Language

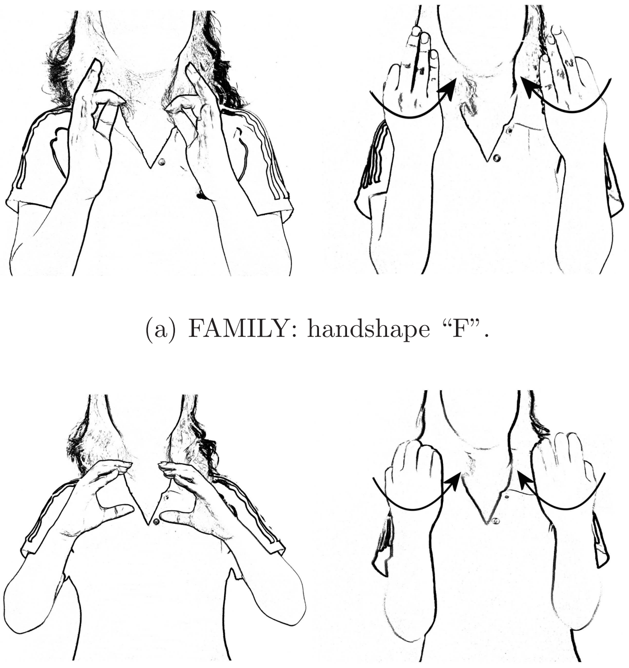
American Sign Language (ASL) is one of the most commonly used sign languages. It is a complex visual language that is based mainly on gestures and concepts. It has been recognized by linguists as a legitimate language in its own right and not a derivation of English. ASL has its own specific rules, syntax, grammar, style and regional variations, and has the characteristics of a true language. Analogous to words in spoken languages, signs are defined as the basic semantic units of sign languages [144]. ASL signs can be broadly categorized as static gestures and dynamic gestures. Handshape, palm orientation, and location are considered as static in the sense that they can be categorized at any given time instant. However, hand movement is dynamic and the full meaning can be understood after the hand motion is completed.

1.1.1 Handshape

Handshape is defined by the configuration of fingers and palm and is highly iconic. Bellugi and Klima [16] identify about 40 handshapes in ASL. In a static sign, the handshape usually contributes a large amount of information to the sign meaning. In dynamic signs, the handshape can either remain unchanged or make a transition from one handshape to another. Typically, the essential information given by the handshape is at the start and the end of the sign movement. Handshape becomes the distinguishable factor for signs that have the same movement. For example the signs FAMILY and CLASS shown in Figure 1.2 have the same movement and they are differentiated only by the handshape “F” and “C”. In addition, handshape is the major component when fingerspelling is required, for example, when proper names and words that are not defined in the lexicon are spelled letter by letter.

1.1.2 Movement

Twelve simple hand movements are identified in [16]. In ASL, many signs are characterized by different movements such as the signs CHEESE and SCHOOL in Figure 1.3. Hand movement in sign language is described through trajectory shape and direction. Straight-line motion, circular motion, parabolic motion

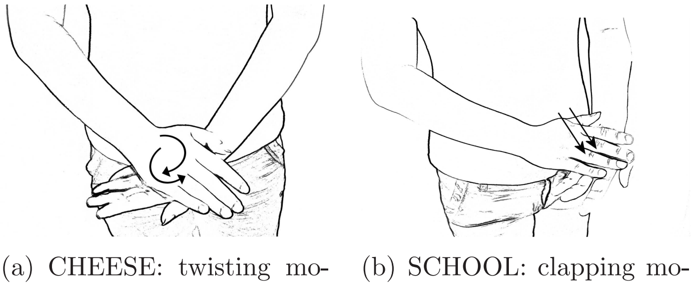


(a) FAMILY: handshape "F".

(b) CLASS: handshape "C".

Figure 1.2: ASL signs with different handshape.

etc, are some examples of trajectory shape. Direction is a crucial component of movement which is used to specify the signer and an addressee. For example, the hand movement in sign GIVE can be towards or away from the signer. The former indicates that an object is given to the signer while the later denotes that the signer gives an object to the addressee. This special group of signs, namely, the directional verbs will be discussed in more detail in Section 1.1.5.



(a) CHEESE: twisting motion.

(b) SCHOOL: clapping motion.

Figure 1.3: ASL signs with different movement.

Hand movement usually carries a large amount of information about sign meaning. Many signs are made with a single movement which conveys the basic meaning. Repetition of the movement, the size of the movement trajectory, the speed and intensity of the movement give additional or different meanings to a sign. Repetitive movement usually indicates the frequency of an action, the plurality of a noun, or the distinction between a noun and a verb; the size of the movement trajectory directly relates to the actual physical volume or size; speed and intensity of the movement convey rich adverbial aspects of what is being expressed [144].

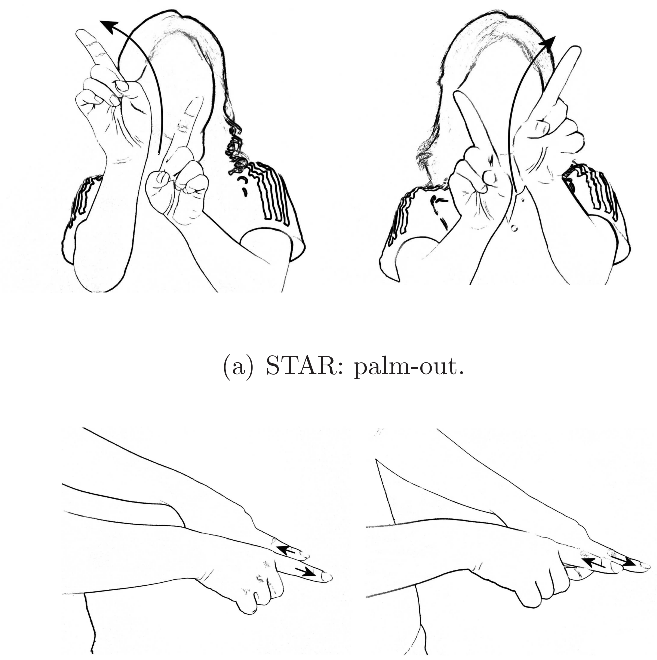
1.1.3 Orientation

This refers to the orientation of the palm in relation to the body or the degree to which the palm is turned. Due to physical restriction on human hand postures, palm orientations can be broadly classified into approximately 16-18 significant categories [16], e.g. palm upright facing in/out, palm level facing up/down, -45^o slanting up/down, etc. The signs STAR and SOCK are mainly differentiated by the orientation of the palm while handshape and movement trajectory remain the same for the two signs. Figure 1.4 shows the two signs.

1.1.4 Location

Location is described as the region where the sign is performed, relative to the signer's body, e.g. around the head, near the chin, around the chest etc. Many of the signs are formed near the head and chest area because they can be easily seen. The important location information is usually conveyed at the start and end of a sign. About 12 locations are identified in [16].

An example of a minimal pair that is distinguished only by the location consists of the signs MOTHER and FATHER which are shown in Figures 1.5(a) and 1.5(b). Very often, the location carries some meaning of the sign, for example,



(a) STAR: palm-out.

(b) SOCK: palm-down.

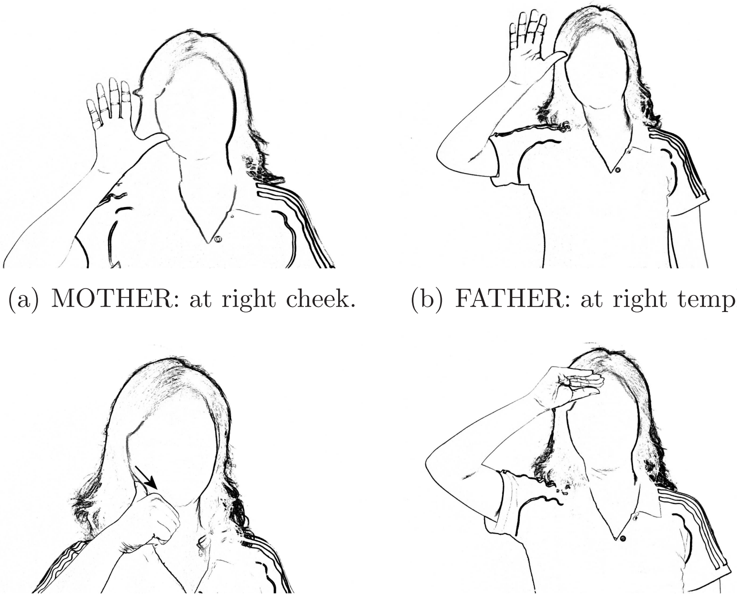
Figure 1.4: ASL signs with different palm orientation.

location is used to differentiate gender in some signs. Signs related to males are always signed at the upper part of the head while signs related to females are signed at the lower part of the head. Figure 1.5 shows the signs FEMALE and MALE as well as MOTHER and FATHER illustrating gender differentiation by location. In addition, the signs HAPPY and SORRY in Figures 1.6(a) and 1.6(b) are made near the heart showing that these are signs that are related to feelings while the sign IMAGINE 1.6(c) which is related to mind is made near the head.

1.1.5 Grammatical Information in Manual Signing

Some signs in ASL are made according to context and modified systematically to convey grammatical information. These "inflections" are conveyed by varying the size, speed, tension, intensity, and/or number of repetitions of the sign. These systematic variations are defined as inflections for temporal aspect.

In ASL, there is another important grammatical process called directional



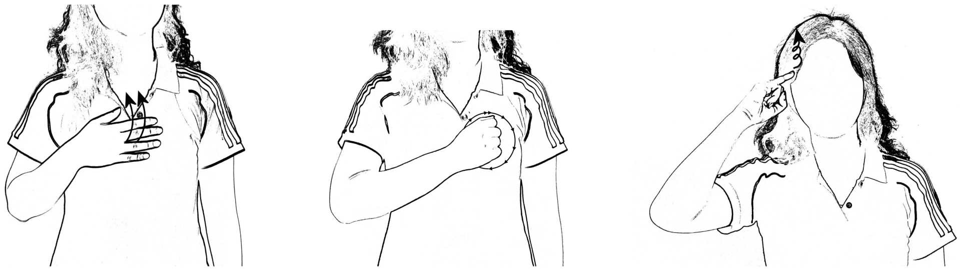
(a) MOTHER: at right cheek.

(b) FATHER: at right temple.

(c) FEMALE: at right jaw.

(d) MALE: at forehead.

Figure 1.5: Gender differentiation in ASL signs according to location.



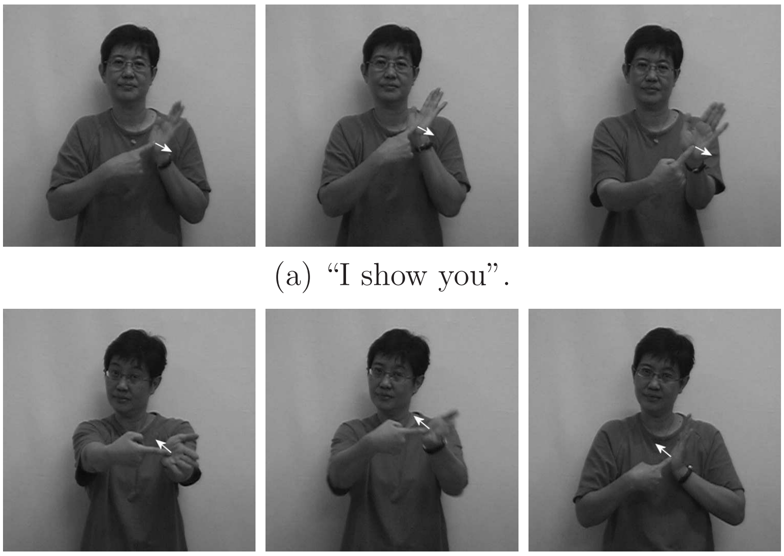
(a) HAPPY: near the heart.

(b) SORRY: near the heart.

(c) IMAGINE: near the forehead.

Figure 1.6: ASL signs denotes different meanings in different location.

verbs which makes use of the movement path direction to identify the subject and the object in a sentence. The subject is the doer of an action (signer) and the object is the recipient of the action (addressee). For instance, when the sentence, “I show you.” is signed, only SHOW is signed with the hand motion moving from the signer to the addressee, i.e. from I to YOU. On the other hand, when “You show me.” is signed, SHOW is signed with reversed hand movement direction, i.e. from YOU (the addressee) to I (the signer). Figure 1.7 illustrates two examples of the inflected sign SHOW. In directional verbs, the change in movement direction is usually accompanied by changes in location and/or palm orientation. Also, the directionality of directional verbs is not fixed as it depends on the location of the object or the addressee which can be anywhere with respect to the signer.



(a) "I show you".

(b) "You show me".

Figure 1.7: Directional verb SHOW.

1.1.6 Non-Manual Signals

Complete meaning in sign language cannot be conveyed without non-manual signals. For example, the sentences "The girl is at home." and "The girl is not at home." are manually signed as "GIRL HOME". The difference is conveyed through non-manual signals where head shaking and frowning denote the negation. Non-manual signals convey important grammatical information in ASL using facial expressions, mouthing when signing, raising the eyebrows, shaking the head, etc. For example, negative sentences are accompanied by a characteristic negative head shake; "yes/no questions" are accompanied by raised eyebrows, wide eyes, head forward; and "wh-questions" are marked by furrowed eyebrows, head forward.

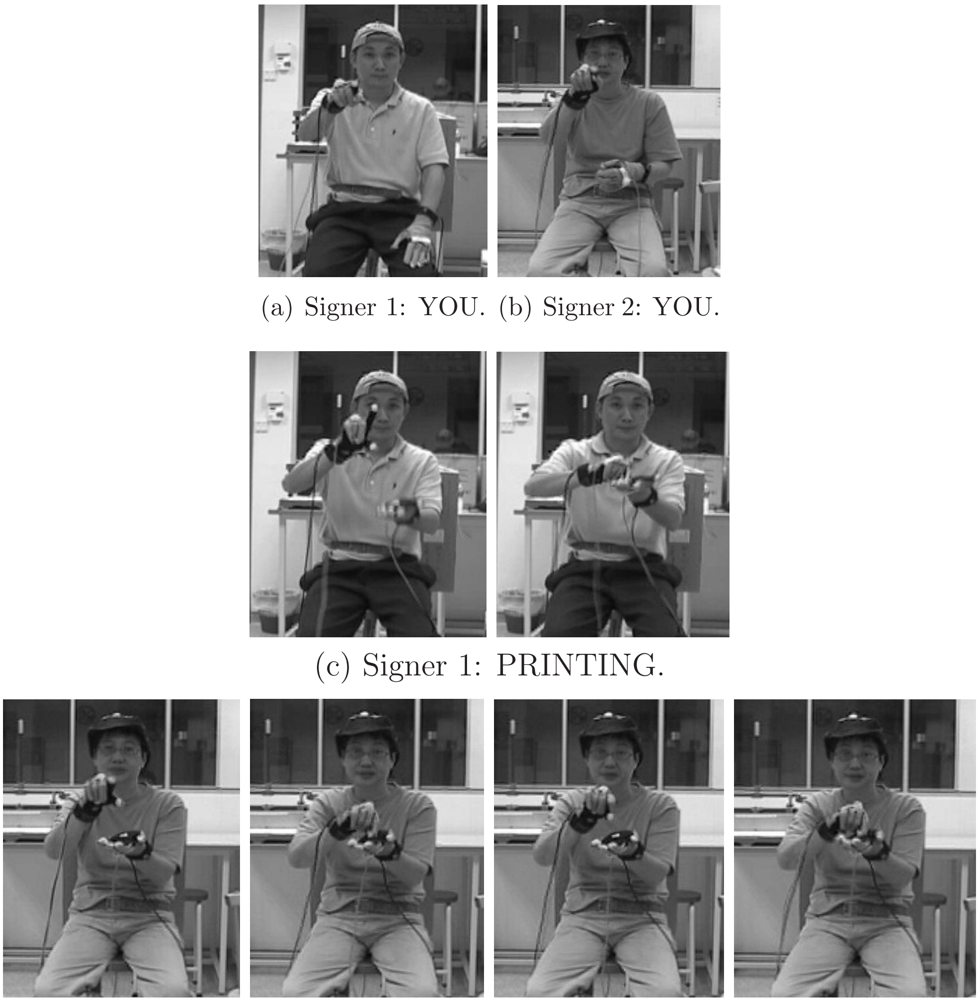
1.1.7 One-Handed Signs and Two-Handed Signs

Some signs in ASL require one hand while others require both hands. In [20], one hand is defined as the dominant hand and the other is defined as the dominated hand. For two-handed signs, the dominant hand is used to describe the main action while the dominated hand either serves as a reference or makes actions symmetric to the dominant hand. One-handed signs are made with the dominant hand only, and there is no restriction on the dominated hand in terms of handshape, orientation, and location, though it should not have significant movement. Its use depends on the preceding and following signs as well as the signer's habit.

1.2 Variations in Manual Signing

Variations occur naturally in all languages and sign language is no exception. Variations in language are not purely random; some are systematic variations with restricted dimensions, while some can vary in a greater range. These variations can be minor; a circle signed by two signers can never be exactly the same. Nonetheless, these variations are limited, i.e. a circle has to be signed to be "circle-like" and not as a square; a handshape "B" should not be signed as a handshape "A", etc. Figure 1.8 shows an example of two signers signing the sentence YOU PRINTING HELP$^{\text{I}\rightarrow\text{YOU}}$ (HELP$^{\text{I}\rightarrow\text{YOU}}$ denotes I-HELP-YOU; the annotation is explained in detail in Appendix A.) with some variations. It is observed that the position of the first sign YOU for signer 2 is relatively higher than that for signer 1 in relation to their bodies. In addition, signer 2 signs PRINTING twice while signer 1 signs it once.

Variations in sign appearance can be attributed to several factors. Sign language as any other language, evolves over time. For example, some two handed-



(a) Signer 1: YOU. (b) Signer 2: YOU.

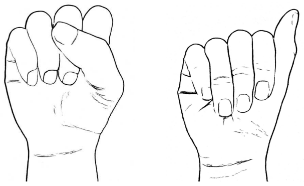
(c) Signer 1: PRINTING.

(d) Signer 2: PRINTING.

Figure 1.8: ASL sentence: YOU PRINTING HELP$^{\text{I}\rightarrow\text{YOU}}$.

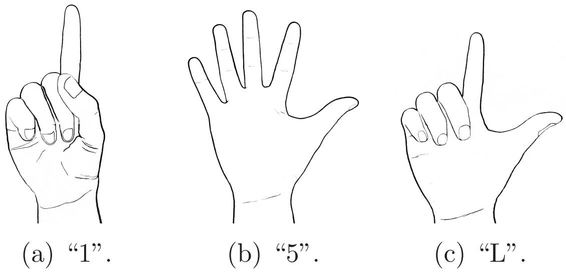
signs such as CAT and COW have slowly become one-handed over the years. This may lead to differences in the choice of signs being used by the younger and older generations. Regional variability is another factor. Deaf people from different countries use different sign languages, for example, ASL in America, British sign language in the UK, Taiwanese sign language in Taiwan, to name a few. However, even within a country, e.g. America, deaf people in California may sign differently from deaf people in Louisiana. Social and ethnic influences may also affect sign appearance. At the individual level, variation occurs simply because of the uniqueness of individuals. Differences in gender, age, style, habit, education, family background, etc contribute to variations in sign appearance.

In ASL, variations which appear in the basic components, i.e. handshape, movement, palm orientation and location, are classified as phonological variation by linguists. Some handshapes are naturally close to each other, for example, the signs with handshapes “S” and “A” as shown in Figure 1.9 can easily resemble each other when they are signed loosely. Also, some handshapes may be used interchangeably in certain signs, for example, signs such as FUNNY, NOSE, RED and CUTE are sometimes signed with or without thumb extension [11]. Studies in [101] show that signs with handshape “1” (index finger extended, all other fingers and thumb closed) are very often signed as signs with handshape “L” (thumb and index extended, all other finger closed) or handshape “5” (all fingers open) by deaf people in America. Figure 1.10 shows the three handshapes. Some examples of sign with handshape “1” are BLACK, THERE and LONG.



(a) “S”. (b) “A”.

Figure 1.9: Handshapes “S” and “A”.



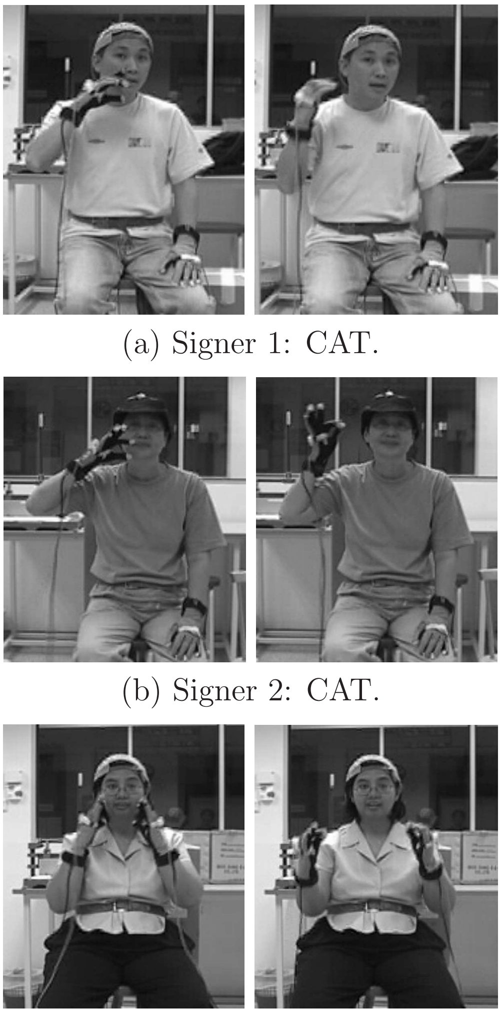
(a) “1”. (b) “5”. (c) “L”.

Figure 1.10: Handshapes “1”, “5” and “L”.

Locations of a group of signs may also change from one part of the body to another. For example, the sign KNOW is prescribed to be signed at the forehead in the ASL dictionary, but, it is frequently signed at a lower position near the cheek. In [101], it was found that younger signers tend to make these signs below the forehead more often than older signers. Also, men tends to lower the sign location more than women. The movement path and palm orientation of a sign may also be modified when making a sign; for example, signs with straight line movement can often be signed with arc-shaped movement or with palm orientation changing from palm-down to palm-left. Assimilation of handshape, movement, palm orientation and location also occurs in compound signs. It refers to a process when the two signs forming the compound sign begin to look more and more like one another. For example in the compound sign THINK MARRY which means "believe" in English translation, the palm orientation of THINK assimilates to the palm orientation of MARRY [101]. Some other phonological variations include deletion of one hand in a two-handed sign and deletion of hand contact.

Figure 1.11 shows the variations in the sign CAT when it is made by three signers. Signers 1 and 2 make a one-handed sign while signer 3 makes it two-handed. Also, the handshapes used by signer 1 and signer 2 are somewhat different. Signer 1 uses handshape "G" while signer 2 uses handshape "F" to make the sign for CAT. Naturally, this causes variation in the palm orientation as well. Variation in the movement component can also be observed in the same example. The straight line hand trajectory made by signer 3 is larger compared to signers 1 and 2. Figure 1.12 further illustrates the variation in movement direction where signer 1 signs GO with direction slightly towards his left but signer 2 moves her hands straight in front of herself.

There can be systematic variation present in the grammatical aspect of sign language and two of the grammatical processes were described briefly in 1.1.5. Typological variations concerning sign order also occur where signs are arranged



(a) Signer 1: CAT.

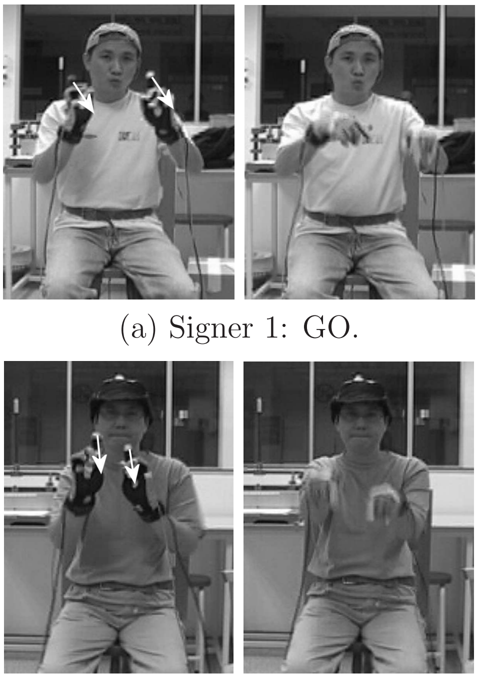
(b) Signer 2: CAT.

(c) Signer 3: CAT.

Figure 1.11: Signer variation: one-handed vs. two-handed, handshape and trajectory size.

differently in sentences. Lastly, some signs can be made in unpredictable ways. For example, in [101] it was reported that the sign for PIZZA was made in very different ways: some signers fingerspelled it, some signed it as a person taking a bite out of a piece of pizza, and some signed it as a round plate on which pizza is served. These variants of the sign do not share handshapes, locations, palm orientation and movement.

The above variations in sign language are related to the linguistic aspects, and a sign language recognition system involving multiple signers must robustly



(a) Signer 1: GO.

(b) Signer 2: GO.

Figure 1.12: Signer variation: movement direction.

handle these variations. In addition, physical variations (e.g. hand size, body size, length of the arm, etc of the signers) also contribute to the complexity of building a robust sign language recognition system.

1.3 Movement Epenthesis

Movement epenthesis refers to the transition segment which connects two adjacent signs. This is formed when the hands move from the ending location of one sign to the starting location of another sign, and does not carry any information of the signs. Linguistic studies of movement epenthesis in the literature are limited and there is no well-defined lexicon for movement epenthesis. Perlmutter [119] also showed that movement epenthesis had no phonological representation. Though movement epenthesis may not carry meaning, it can have a significant effect on computer recognition of continuously signed sentences, as the transition period of the segment can even be as long as a sign segment. This problem needs to be addressed explicitly for robust sign language recognition. It must be noted that movement epenthesis is a different phenomenon from co-articulation in speech; co-articulation does occur in sign language, and manifests itself in some signs as hold deletion, metathesis and assimilation [101].

There has not been much research in the phonology of movement epenthesis, and variations in movement epenthesis are not well-characterized. As it is a connecting segment between signs, its starting and ending locations would depend on the preceding and succeeding signs, respectively. Also, it can be conjectured that any variations in the adjacent signs may affect the movement epenthesis. Hence, it is conceivable that the variations seen in movement epentheses, are comparable to variations in sign. As there are no well-defined rules for making such transitional movements, dealing with movement epenthesis adds significant complexity to the task of recognizing continuous signs.

1.4 Research Motivation

The main motivation of our research is to develop a sign language recognition system which will facilitate communication between the deaf and hearing people. The deaf tend to be isolated from the hearing world, and face many challenges in integrating with hearing people who do not know sign language. Technologies may provide a solution to bridge the communication gap through a system for translating sign language to spoken language/text or vice versa. Such a system can be useful in many situations; for example, in an educational setting, a teacher-student translation system will be useful for communicating knowledge effectively; in emergencies, deaf people can make use of the translation system to seek help. There are several useful applications of this nature, e.g. TESSA [25], an application built by VISICAST, aims to aid transactions between a deaf person and a clerk in a Post Office by translating the clerk's speech to British sign language. VANESSA [55] is a newer and improved version of an application by VISICAT, which provides speech-driven assistance for eGovernment and other transactions in British sign language. VANESSA is an attempt to facilitate communication between the hearing and deaf people so the latter can be easily assisted in filling complicated forms, etc.

A practical sign language recognition system would need to recognize natural signing by different signers. In real communication, signs are not always performed according to textbook and dictionary specifications. Signing is not merely making rigidly defined gestures; it has to make communication effective and natural. This implies that sign recognition systems must be robust to signer variations. Analogous to speech recognition, we expect well-trained signer dependent systems to outperform signer independent systems. Typically, in speech recognition, the error rate of a well-trained speaker dependent speech recognition system is three times less than that of a speaker independent speech recognition system [66]. However, many hours worth of sign language sentences are required to train a signer dependent system well, obtaining this data could be difficult or even impossible. Hence, a signer independent system is definitely desirable in applications where signer-specific data is not available. Extensive work on speaker independence has been done in speech recognition, but it has yet to receive much attention in sign language recognition. In the latter, it is mostly considered in works related to hand postures or isolated signs but works on continuously signed sentences are limited. Many of the current "signer-independent" systems in the literature rely on an adaptation strategy, where a trained system is adapted to a new signer by collecting a small data from him/her. Adaptation is a promising approach but it has limitations; these are discussed in more detail in Chapter 2.

Thus, a signer independent system that uses no adaptation at all is ideal.

Although sign language has similarities with speech that can be exploited, sign language exhibits both spatial and temporal properties. Unlike speech which is a sequential process, the constituent components of sign language can occur simultaneously, and each of the manual components, i.e. handshape, movement, palm orientation and location can contribute differently to the variations in a sign. We will study and analyze the effects of the variations on these components, and develop an appropriate modeling framework to achieve robust recognition.

1.5 Research Goals

The main aim of this work is to devise a sign language recognition system to robustly handle signer variation in continuously signed sentences. Variation in sign language is a broad and complex issue as described in Section 1.2. Our focus is on the phonological variations in sign language, i.e. variations in handshape, movement, palm orientation and location. These are variations arising from different signers who sign naturally without restricting themselves to textbook definitions. We also include directional verbs which exhibit variation in grammatical aspect. Though phonological variation is our key focus, we also consider others such as variations in sign order which can occur in natural signing. However, signs made with completely different appearances are beyond the scope of this thesis.

To recognize continuously signed sentences, addressing the problem of movement epenthesis is crucial. Approaches in speech recognition which deal with co-articulation effects are not suited to handle the movement epenthesis problem. Often, the duration of movement epentheses can be comparable to that of signs and we cannot naïvely assume that movement epenthesis segments can be modeled as parts of the adjacent signs. Even locating the movement epenthesis segments is a difficult problem as there is no well-defined movement epenthesis lexicon for reference. This difficulty is compounded in natural signing, but must be addressed to successfully recognize signs. We aim to find a solution to handle movement epenthesis in continuous sign language recognition.

In linguistics, a phoneme is defined as the smallest phonetic unit of a language. However, there is no standard phoneme definition in sign language. Though handshape, movement, palm orientation and location are characterized as the phonological components of sign language, linguists define a variable number of units for each component. Due to this ambiguity, phonemes are often defined by using an unsupervised clustering algorithm in sign language recognition works. This is a reasonable approach for the static components, i.e. handshape, palm orientation and location, but it is difficult to cluster the dynamic movement component by static clustering algorithms. Thus, we propose a strategy to define "phonemes" for the movement component automatically from the data.

Although four components are commonly specified by sign linguists, many of current works in sign language recognition do not differentiate between movement and location explicitly. Frequently, either 2-D or 3-D positions of the hands are tracked and used to represent movement and location. For complete representation of sign language as suggested by linguists, we include the movement component unambiguously in our modeling. Movement component is made up of direction and trajectory shape which are heavily dependent on the start and end point of a hand gesture. The feature extraction process for the movement component is challenging in continuously signed sentences as information of the start and end point of hand motion is usually not clear. In this thesis, we seek a representation that can characterize direction and trajectory shape for the movement component and work out a procedure to extract the movement features from continuously signed sentences.

1.6 Thesis Organization

The rest of the thesis is organized as follows: Chapter 2 summarizes related works to give an overview of the recent state of the art in sign language recognition. The overall modeling concept and proposed strategy for handling signer variation is also described in this chapter.

Chapter 3 presents the framework and experimental results for recognition of isolated signs based on Signing Exact English (SEE). This was our preliminary investigation on variations in sign language and provides useful insights for subsequent works on continuous signing. Chapter 4 proposes an automatic phoneme transcription procedure which is based on Principal Component Analysis (PCA) for the movement component and standard clustering algorithm for the other components. We discuss the strategy to deal with movement epenthesis and present a conditional random field (CRF)/support vector machine (SVM) based modeling framework which discriminates between sign and movement epenthesis in Chapter 5. Chapter 6 describes the final recognition framework based on a two-layer CRF model. Experimental results for the different subsystems are presented in Chapter 7. This chapter also describes the data collected for the continuous signing recognition experiments using Cyberglove and magnetic trackers. For the final recognition framework, comparison experiments based on Hidden Markov Models (HMMs) are also given with results, analysis and discussion. Lastly, Chapter 8 gives the conclusions of this thesis and suggests possible extensions for future work.

Everything has its wonders,
even darkness and silence,
and I learn,
whatever state I may be in,
therein to be content.
Helen Keller (1880-1968)

2
Related Works and Overview of Proposed Approach

2.1 A Brief History

Research on computer recognition of sign language started during early 90's with recognizing static hand postures or fingerspelling handshapes. However, all four components, i.e. handshape, movement, palm orientation and location are needed for complete recognition of sign meaning. Promising results on handshape recognition motivated more researchers to study and examine the dynamic aspects of sign language, and many initial works addressed the simpler problem of recognizing isolated signs as a first step towards recognizing continuously signed sentences.

Sign language recognition experiments use either vision or glove and magnetic tracker-based input. One of the earliest works on recognizing static handshape is by Beale and Edwards [15] who used a vision-based approach to recognize hand postures. Artificial neural networks (ANN) have been widely used for fingerspelling handshape and isolated sign recognition, for example in [15, 30, 49, 64, 68, 74, 94, 147, 160, 171]. In more recent works, there has been a shift towards using HMMs for dynamic sign language recognition, due to their capability of handling spatio-temporal variations [1, 22, 59, 69, 70, 78, 103, 142, 145, 175, 179, 181]. Other approaches such as template matching [5, 57, 108], PCA-based techniques [28], decision tree [63], discriminant analysis [26], graph or shape transition networks [50, 60], dynamic programming [29, 31, 86, 98, 131], unsupervised clustering [112] were also explored. Most of these works used only one signer for their experiments, and the number of signs was typically not more than 200. Generally, ANN and HMM approaches provided better performance as compared to other approaches, and recognition accuracy is ranging from 85.0% to 99.9%. On the other hand, template matching approaches often yielded poorer accuracy.

Recently, recognizing continuously signed sentences has become the major focus. Many works started by devising algorithms to recognize the basic meaning of manual signs, but later, more researchers began to explore the grammatical aspects of sign language including non-manual signals. There are many issues to be addressed in continuous signing and many problems are yet to be solved. These include segmentation of continuously signed sentences, scalability to large vocabulary, dealing with movement epenthesis and co-articulation, robustness to noise, etc. A comprehensive review of sign language research was presented in [115]. Other good reviews can be found in [43, 100, 143]. In the subsequent sections, we describe the progressive development of the state of the art in sign language recognition and discuss the major issues in continuous signing.

2.1.1 Recognition of Continuous Signing

Sign language gesturing in sentences is continuous, and needs to be deciphered continuously. At the least, a practical sign language recognition system should recognize continuous signing; a fully functioning system should be capable of handling the grammatical aspects of sign language, including the non-manual components.

The transition from isolated sign recognition to continuous signing was made by Starner et al. [134, 135] who used HMMs to solve sentence-level ASL recognition with a 40 word lexicon in a vision-based approach. Strict grammar rules were applied in the system and the whole sign was taken as the smallest unit. The results demonstrated high recognition accuracy. Since this work in the late 90's, research in continuous sign recognition has increased tremendously. A good example is the SignSpeak project [33, 34] for translation of continuous sign language. They used a vision-based approach and tackled many problems in the recognition of continuous signing. Their works include extracting features in manual signs [39], tracking related techniques [35, 36, 38, 40], adapting speech recognition techniques to sign language recognition [32], providing benchmark databases [37], phonetic annotation [88] etc.

Good performance is certainly the ultimate goal of a sign language recognition system, but before this can be achieved, several problems need to be tackled successfully. As mentioned earlier, there are many noteworthy issues in continuous sign language recognition as compared to isolated signs, and thus, continuous signing recognition is discussed in more detail with respect to the major issues in the subsequent sections.

2.2 Issue 1: Segmentation in Continuous Signing

Unlike isolated signs, the start and end points of a sign are not well-defined in continuous signing. There are two ways to approach this problem, viz. explicit segmentation, where segmentation is performed prior to the classification stage and implicit segmentation, where segmentation is done along with classification.

In explicit segmentation the main concern is to choose the correct cues that will allow inferring the physical transition points. Harling and Edwards [62] used hand tension as a cue to perform segmentation on two British sign language sentences. This was based on the idea that intentional gestures are made from one position to another with a tense hand. They also pointed out that higher level inputs such as grammar of the gestural interaction is crucial for segmentation tasks. Minimum velocity of hand movement was used to indicate hand transition boundaries in [87, 111]. Sagawa and Takeuchi [125] proposed that velocity alone was inadequate to segment sign language sentences in general, and used a parameter defined as “hand velocity” which included changes in handshape, direction and position. Minimal “hand velocity” was used as a candidate for a border. In addition, a transition boundary was indicated when a change in the hand movement direction was above a threshold. Recognition was carried out according to the method presented in [126]. Wang et al. [164] also used a similar method for trajectory segmentation. In Liang and Ouhyoung [96], transition boundaries were identified with time-varying parameter (TVP) detections. They assumed a gesture stream was always a sequence of transitions and posture holdings. When the parameter TVP fell below a threshold, indicating a quasi-stationary segment, it was taken to be a sign segment. 250 signs in Taiwanese sign language were recognized with 80.4% accuracy by HMMs trained with 51 postures, 6 orientations and 8 motions. Gibet and Marteau [54] identified boundary points where the radius of curvature became small and there was a decrease in velocity. They used the product of velocity and curvature to detect boundary points. Rao et al. [123] used the spatio-temporal curvature of motion trajectory to describe a “dynamic instant”, which is taken to be an important change in motion characteristics such as speed, direction and acceleration. These changes were captured by identifying maxima of spatio-temporal curvature. Walter et al. [162] used a two-step segmentation algorithm for 2-D hand motion. They first detected rest and pause positions by identifying points where the velocity dropped below a preset threshold. After this, they identified discontinuities in orientation to recover strokes (movement and hold) by applying Asada and Brady’s Curvature Primal Sketch [8]. In [67], continuously fingerspelled signs consisting of 20 handshapes and 6 local small movements at the palm area were investigated. A distance-based hierarchical classifier was used for handshape and 1-NN or naïve Bayes classifiers with genetic algorithm were used for movement. The handshape segments followed by movement information was used to decode the meaning of the signs. However, the evaluation of their final framework was not clear. They only tested on two different spelled sentences and reported a total of 19 and 18 deletion errors in each type of sentence.

Generally, these approaches devise rules to characterize boundary points based on the selected features and appropriate tuning of threshold values. The effectiveness of the segmentation algorithm depends on the selected features and the chosen thresholds. Although velocity, change of directional angle, and curvature are commonly used for identifying boundary points, other features such as those used in [62, 72, 92] may also be useful. However, when more features are used, the rules become complex and difficult to formulate. In addition, it is difficult to set thresholds for the features when the sentences are signed naturally, as the variations are complex, and the signer's habits, rhythms and speed will affect the estimation of boundary points. Hence, it is necessary to have an effective algorithm to handle the problem. Fang and Gao [45] used a recurrent neural network to segment continuous Chinese sign language. The temporal data points were labeled as left boundary, right boundary or interior of a segment. The features for segmentation were automatically learned by a self-organizing map and the segmentation accuracy was reported to be 98.8%. However, the nature of the sentences used is not clear and, and training recurrent neural networks is not straightforward. Bashir et al. [10] detected discontinuities in the motion trajectory by using curvature to measure the sharpness of bend in 2-D curves. They used hypothesis testing to locate the points of maximum change of the curvature data. In [72], a hierarchical activity segmentation approach was proposed to segment dance sequences. Force, kinetic energy and momentum were computed from the velocity, acceleration and mass at the lowest level of the hierarchy, to represent activity. Each choreographer profile was represented by a trained naïve Bayesian classifier, and the average accuracy was 93.3%.

Besides the segmentation approach which relies on physical cues, other strategies for temporal segmentation have also been proposed. Santemiz et al. [127] aimed to extract isolated signs from continuous signing. They showed that modeling the signs with HMMs using the segmented results from DTW provided better performance than using HMMs or DTW separately, and they obtained an accuracy of 83.42%. Lichtenauer et al. [97] proposed that time warping and classification should be separated because of conflicting likelihood modeling demands. They used statistical DTW only for time warping and combined the warped features with combined discriminative feature detectors (CDFDs) and used quadratic classification on discriminative feature Fisher mapping (Q-

DFFM). They showed that their strategy outperformed HMM and statistical DTW in a proof-of-concept experiment. A unified spatial segmentation and temporal segmentation algorithm was proposed in [4]. It consisted of a spatiotemporal matching algorithm, a classifier-based pruning framework which rejected poor matches, and a sub-gesture reasoning algorithm that was able to identify the falsely matched parts. They evaluated their algorithm on hand-signed digits and continuous signing of ASL and good results were shown. In [95], continuous gestures were segmented and recognized simultaneously. They either applied motion detection and explicit multi-scale search to step through all possible motion segments, or used dynamic programming to detect the endpoints of a gesture. The best recognition rate for two arm and single hand gestures was 96.4%.

In schemes that implicitly segment and recognize, HMMs are a widely used solution. For continuous recognition, it is required to discover the most probable hidden state sequence which produced the observation sentence. The Viterbi algorithm in HMMs is a natural tool to find the single best state sequence for an observation sequence. As search is carried out along with recognition, the sentence is implicitly segmented. Some of the earliest works to use HMMs for continuous sign recognition was by Starner et al. [134, 135]. Bauer et al. [12] used task beam search along with continuous HMMs to recognize continuous signs from a single colour video camera. They obtained 91.7% recognition rate based on a lexicon of 97 signs in German sign language (GSL). With the addition of bigram language model [13], the recognition rate improved to 93.2%. Volger and Metaxas [152] also used HMMs to recognize a 53 sign vocabulary. They attempted a temporal segmentation of the data stream by coupling three-dimensional computer vision with HMMs. The continuous data was segmented into parts with minimal velocity and the segments were fitted to lines, planes or holds. A directed acyclic graph pooling the primitives was used to determine the sequence of signs and the results served as a backup to check the Viterbi algorithm outputs. It was pointed out in the work that requirement of large amount of training data and the lack of contextual information are some of the major problems of using HMMs alone. Furthermore, in practice, there is always the complexity of training HMMs. More recently, Holden [65] used HMMs with grammars to recognize colloquial Auslan phrases. Experiments were conducted with 163 sign phrases of varying grammatical formations, and recognition accuracies of about 97% and 99% were obtained at the sentence and sign levels, respectively. Maebatake et al. [102] applied multistream HMMs to recognize 25 signed sentences consisting of 81 signs. They used only location and movement components and hand position data was collected with Polhemus FASTRAK. Different weights were assigned to the two components to specify their importance. They achieved 75.6% accuracy with higher weight given to the movement component. In [76], a threshold HMM was used based on a parallel HMM network [153] and used to spot 8 signs from 240 video clips. 95.6% detection rate and 93.2% realiability was obtained. Although the accuracy was high, the experiment used only a small number of signs. In [140], dynamic Bayesian network (of which HMMs are a special case) was used to recognize continuous hand gestures with a recall rate of 84.00% and precision rate of 80.77%.

Recently, conditional random fields (CRFs) have provided a promising avenue to work with sequential data. Their major advantage over HMMs is the relaxation of the strong independence assumptions which are made in HMMs. CRFs were first used by Lafferty and McCallum [91] to segment and label sequence data for the parts-of-speech tagging problem. Subsequently, many works have used CRFs to solve problems involving sequential data to recognize text, speech and sign language. The original framework of CRFs is not geared to handle frame-based sequential features. Hence, many variants of CRFs have been proposed and one example is the hidden CRF [122] for dealing with frame-based data as encountered in speech or sign language recognition.

Applications of CRFs to sign language recognition have been relatively recent. Yang and Sarkar [172] attempted to use CRFs to detect co-articulation in ASL and obtained 85% detection accuracy. Lee et al. [139, 167, 168, 169, 170] used a threshold model based on CRFs in a series of works for sign spotting. They included an additional non-sign label in their CRF model but avoided using a fixed threshold to discriminate between sign and non-sign patterns. In their work, a CRF model was first trained without the non-sign samples. A threshold model with CRF was then built by adding the label for non-sign patterns in the trained CRF using the weights of the state and transition feature functions of the original CRF. They adopted a threshold model proposed by Dugad et al. [75] to compute the weights for the state feature functions of the non-sign patterns. For the weights of the transition feature functions, they conjectured that the frequency of non-sign patterns is larger than sign patterns, and formulated a way to calculate the weights. They tested the framework on continuous ASL sentences consisting of 48 signs and obtained 87.0% spotting rate and 93.5% recognition rate on the spotted isolated signs. Later, they presented extensions to spot signs and fingerspellings simultaneously using hierarchical CRFs [167, 168]; this was able to distinguish signs, fingerspellings and non-signs, yielding 83.0% spotting rate for signs and 78.0% spotting rate for fingerspellings. Besides hierarchical CRFs, they also used a conventional semi-Markov [23] CRFs to perform sign language spotting. Spotting rates for ASL and Korean sign language were 77.1% and 71.0%, respectively.

2.3 Issue 2: Scalability to Large Vocabulary

Another important objective is to accommodate a large vocabulary. There are usually a few thousand signs in sign language lexicons and the vocabulary size will keep increasing with the addition of new signs.

Analogous to speech recognition, signs can be broken down into phonemes which are defined as the smallest contrastive units in a language model. The rationale for this representation is that a limited number of phonemes can be used to represent a large number of signs. However, the major difficulty in sign language is that unlike phonemes in speech which are linguistically well defined and can be used without ambiguity to transcribe speech sentences, there is no consistent phoneme lexicon in sign language, and is dependent on the modeling approach and features used in different sign language recognition systems. There is also no standard way of defining phonemes in sign language, and different schemes have been used for phoneme transcription. Sign linguists such as Stokoe [137] and Liddell and Johnson [99] offer some guidelines to model phonemes by distinguishing the basic components of a sign gesture as consisting of the handshape, hand orientation, location, and movement. Stokoe emphasized the simultaneous organization of these components while Liddell and Johnson's Movement-Hold model emphasized sequential organization.

Currently, there are two main approaches for phoneme transcription viz. 1) transcription based on sign language models defined by sign linguists, and 2) transcription which is dependent on the data collected and features used. In the first approach, the sign components are quantized into limited categories and sign language models such as Stokoe's or Liddell's are used to label the signs. Vogler and Metaxas [154] adopted this approach and defined the phonemes for movement and location, using Liddell's model to recognize 22 ASL signs based on these phonemes. The small vocabulary size makes the transcription straightforward. They obtained an accuracy of 91.2%. Wang et al. [165] defined a phoneme as the smallest unit that has meaning and can distinguish one sign from another. They performed an extensive study of Chinese sign language (CSL) and explicitly defined a large number of phonemes, about 2400 for CSL. However, the transcription process and the resulting phonemes are not clearly described. Similarly, a subword approach was adopted in [180]. They used adaptive boosting (AdaBoost) and HMMs to recognize 102 single-handed subwords of Chinese sign language with an accuracy of 92.7%. A manual transcription process can be very laborious and time-consuming, and when the vocabulary size is large this approach is unreliable and impractical. Hence, it is important to devise an automatic method to define phonemes.

In the data dependent approach, many works use unsupervised learning to define phonemes automatically. Walter et al. [162] adopted a mixture density-based clustering approach for transcribing phonemes from gesture trajectory segments. Mixture parameters were determined using expectation maximization (EM) and minimum description length (MDL) was used as the criterion to automatically determine the number of clusters. Bauer and Kraiss [14] used k-means clustering to self-organize trajectory segments into fenones. In this approach, the fenones formed usually do not relate to phonetic concepts. Also, temporal segments may not be properly aligned when segments are obtained from continuously signed sentences. This poses problems for clustering algorithms such as k-means which use the Euclidean distance measure. Hence, a complex clustering algorithm is often required to handle the problem. Wang et al. [163] adopted dynamic programming (DP) to segment the data streams, and a hybrid of neural networks and k-means was used to cluster the segments. Fang et al. [46] proposed a temporal clustering algorithm to group segments using concatenated handshape, position and orientation features of both hands. The temporal clustering algorithm was based on a modified k-means algorithm proposed by Wilpon and Rabiner [166]. However, these are complex and computationally expensive. The work by Han et al. [61] explored the modeling and segmentation techniques of the subunits of sign language. Boundary points were identified by temporal discontinuities of the hand motion. A clustering procedure utilizing spatio-temporal features was used to obtain the subunits. They claimed that their proposed model was highly correlated to linguistic models though their primary motivation was to achieve scalability to large vocabulary. More recently, Nayak et al. [109, 110] attempted to find subunits at the sentence level and defined signemes as the common parts in signs. They first represented each sign sequence as a series of points in a low dimensional space of relational distributions, and then performed speed normalization to extract the signemes using a dynamic programming framework. Later in [110], they improved the framework by using iterated conditional modes (ICM) and more comprehensive experiments were conducted. They tested on 10 signs from 136 sentences and obtained 98 correct, 20 partially correct and 18 incorrect cases out of 136 signemes. One key limitation of their proposed model is that it works only with a single signeme instance, i.e. it requires several runs to extract different signemes. Fenemes were defined in [177] as subwords which were indistinguishable based on some discriminative features. They applied state tying to the inseparable states detected by segmentally-boosted HMMs and this provided them an intuitive indication of which segments came from the same feneme. Though they showed that the fenemes obtained were perceptually meaningful, the approach was sensitive to “bad signing”. In [61], a subunit was defined as a motion pattern with interrelated spatio-temporal features. They detected the subunit boundary points using hand motion discontinuity obtained from motion speed and trajectory. Temporal clustering based on DTW was then applied to obtain the subunits. From the works described so far, it is again apparent that there is no standard definition for the smallest linguistic units in sign language, and an automatic transcription procedure is required.

A multiple parallel channel framework is another strategy for scalability to large vocabulary. In many of Gao's works, e.g. [44, 45], the features from gloves and trackers were tightly coupled into a combined feature vector for recognition. However, in sign language, there are many simultaneous processes and thus, tight coupling is unsuitable, especially for a signer independent system. Another disadvantage of this approach is that a larger number of classes would need to be distinguished (i.e. the product of the number of classes in the individual channels). Bossard et al. [20] discussed the importance of simultaneous information in a sign language system, where many elements occur simultaneously and proposed a parallel channel framework. Several works of Vogler and Metaxes used this modeling approach. Parallel Hidden Markov Models (PaHMMs) were proposed in [153] to address the scalability problem. PaHMMs model N processes with N independent HMMs with separate outputs based on the assumption that the separate processes change independently and produce independent outputs. Their experimental results justified this assumption. In [155, 156], they further applied the PaHMMs to a modified Movement-Hold model. Simultaneous inputs were broken down into independent channels. One channel was used to describe the "dominant" hand movement-hold segments and the body locations while other channels were used to describe the "dominated" hand segments, handshape, hand orientation and wrist orientation.

2.4 Issue 3: Movement Epenthesis

In continuous sign language recognition, an important issue is to efficiently tackle movement epenthesis, the extra connecting hand movement between two successive signs. The presence of movement epentheses (these are not defined in the dictionary) adds to complexity in continuous sign recognition. Different signers could have different ways of making the connecting movements as there are no strict “rules” in the sign language lexicon which specify how a movement epenthesis should look. Also, co-articulation effects, similar to speech are present, where the appearance of the end and start points of the preceding and succeeding signs are influenced. Signers who sign slowly will have minimal co-articulation between signs while a fluent signer will sign faster with heavy co-articulation. An increase in the co-articulation effect results in an increase in the statistical variability of the signs, and a consequent decrease in the recognition rate.

Similar to speech recognition, implementing a context-dependent model was one of the common ways to handle movement epenthesis in the early years. Vogler and Metaxas [151] addressed the issue with bi-sign context-dependent HMMs. The number of models required to cover all possible contexts on a 53 sign vocabulary was originally $53^2 = 2809$, but this was reduced to 337 models after tying some of the HMM parameters together. Bigram language models were also included to improve the experimental results, and a recognition rate of 91.7% was obtained on the 53 sign vocabulary. A similar approach can be seen in Gao et al. [52]. However, Gao employed dynamic programming to segment the training sentence into basic subword units, and obtained a recognition rate of 95.2%. Wang et al. [164] pointed out that using a context-dependent model to solve the movement epenthesis and co-articulation problem is not efficient especially when the vocabulary size is large; the number of basic phonemes to be modeled is itself already large, and if movement epenthesis is also included, the number of possible context-dependent phonemes i.e. “TRIPHONE” will be too large to be handled efficiently. This approach is more suited for the co-articulation problem but not for handling movement epenthesis, which is a non-trivial segment between signs, and needs to be explicitly addressed. Movement epenthesis and co-articulation should be differentiated as two separate problems in their own right. Context-dependent models which are commonly used in speech recognition to solve the co-articulation problem are not suitable to solve the movement epenthesis problem in sign language.

The movement epenthesis problem was implicitly addressed in Yuan [178]. It was proposed that since linguistic restrictions were relatively loose in sign language, a context-independent model could be used. Connections between two subwords were classified into weak connections and strong connections. Strong connections indicated the connection of subwords in one sign word and were recognized with the use of subword trees to aid the Viterbi algorithm. Weak connections indicated the connections of words without such a relationship. They were recognized by constraining the Viterbi algorithm into every single model and end score at each frame was compared. 70% accuracy was shown. A clearer strategy to deal with movement epenthesis was to model the movement epenthesis explicitly [151]. However, it was assumed that the number of movement epentheses was finite and limited. Phonemes for movement epenthesis were obtained by k-means clustering. The main advantage of this approach was that the number of models required and the complexity were reduced. The accuracy with this method was 95.83%. Gao et al. [47, 53] also adopted explicit modeling of movement epenthesis and proposed transition movement models in their framework. They observed that transition movements could be grouped and proposed a more general algorithm for temporal clustering with DTW. With this algorithm, manual segmentation of the continuous signs was avoided as the algorithm could automatically segment the transition movement with a bootstrap iteration. The Viterbi algorithm serached for the final decoded path from the sign models and transition movement models. Assan and Grobel [9] modeled transitions with a separate single state HMM. Unigram language model was employed and 73% recognition rate was achieved. In [17], continuous hand gesture segmentation and gesture transition point detection were examined. The term "co-articulation" was loosely used in their work to describe the transition points. They first detected the gesture boundary points and the segments were matched with a finite state machine for co-articulation detection. One of the drawbacks of this work is that they assumed a clear "pause" between two gestures but this is not the case in natural continuous signing. Explicit modeling of movement epenthesis though apparently a feasible solution, adds complexity to the signer independent system as variations in movement epenthesis could be large as well. Often, there are more movement epenthesis models than phone models for signs even in a signer dependent system. For example, Vogler [150] defined 78 phone models for signs and 133 movement epenthesis models, which is 70% more movement epenthesis models than phone models. The common way to reduce the number of movement epenthesis is to cluster them, but this may result in a loss of modeling accuracy.

Recently, there has been more emphasis on the movement epenthesis problem in some works. Yang and Sarkar started to examine the movement epenthesis problem in [172]. They defined movement epenthesis as co-articulation with longer durations. Their aim was to segment the signs in continuous signing by using CRFs to detect the movement epenthesis frames in a video sequence. They obtained 85.0% accuracy for segmenting ASL signs. However, the CRF-based approach was not extended in their later works. Instead, they adopted an enhanced level building algorithm for handling movement epenthesis in [173, 174]. They trained models based on signs only. The basic idea was to have trained model signs using the space of probability functions (SoPF) with Mahalanobis distance measure. The enhanced level building algorithm was used to simultaneously segment and match signs to continuous sign language sentences in the presence of movement epenthesis. Movement epenthesis was automatically discarded along the matching process. They enhanced the classical level building (eLB) algorithm [106] based on dynamic programming and coupled it with a trigram grammar, to obtain 83.0% word level recognition rate in [173]. Further experiments were conducted in [174] where they tested their algorithm on ASL data sets and compared the performance of their method with CRFs and latent dynamic CRF-based approaches. They achieved 40% improvement over CRF-based approaches in terms of frame labeling rate, and 70% improvement in sign recogntion as compared to the unenhanced DP matching algorithm. The works by Lee et al. [139, 167, 168, 169, 170] mentioned in Section 2.2 adopted the strategy of spotting signs in continuously signed sentences and performed recognition on the signs. In their approach, movement epenthesis segments are bypassed automatically. The proposed CRF threshold models and hierarchical CRF-based model showed promising performance. Kelly et al. [76, 77] also proposed a parallel HMM threshold model to handle movement epenthesis in sign language based on the threshold HMM proposed by Lee and Kim [93]. The key idea in threshold HMM is that the likelihood can be used as an adaptive threshold for selecting the proper gesture model. Hence, a dynamic threshold based on the likelihood measure was computed in [76, 77] to distinguish between signs and movement epentheses. Experiments conducted in [77] showed that 100 different types of movement epenthesis and eight different signs were identified.

2.5 Issue 4: Signer Independence

A practical requirement for a sign language recognition system is to be signer independent. Presently, many of the continuous sign language recognition works report results on a single signer, e.g. [9, 96, 164, 165, 170]. Most works by Bauer and Vogler are also based on one signer. Other signer dependent works use more signers but usually the same signers are used to train and test the system, for example, [2, 28, 138]. Deng [28] used data from two non-native signers but tested the system with unseen data from the same signers used for training and reported an accuracy of 70.0%. A fuzzy-ruled approach was used in [138] to recognize 3-D arm movements in Taiwanese sign language. 10 persons were asked to make 10 different arm movements and each arm movement was made 10 times. They showed recognition rate of 96.6%. A large amount of data was collected from 60 volunteers in [2]. For each gesture, 40 out of the 60 samples were used for training while the remaining 20 samples were used for testing. A recognition rate of 87.0% was obtained; however the results were all on "seen" signers, i.e. those who contributed data to the training set.

Signer independent recognition is challenging due to variations in handshape, bodysize and gesturing habit or rhythm among signers as discussed in Chapter 1. Kadous [71] attempted to investigate signer independence based on instance-based learning and a decision-tree. In full round-robin tests by leaving one signer out each time, unsatisfactory results, ranging from 12.0% to 15.0%, were obtained. This demonstrated the difficulty of devising a signer independent system. The framework may fail to work for a new signer without a proper strategy. Generally there are two key strategies for signer independent recognition. One is to build a baseline recognition system using one signer or a few signers and adapt the system to new signer using a small amount of data from the new signer. Another approach is to devise a robust recognition algorithm that is designed to be tolerant to signer variations and thus anticipate good generalization.

At present, this aspect has not received much attention in the literature. Most works that considered signer independence dealt only with isolated signs recognition. Waldron and Kim [160] used data from one signer to train their system which consisted of 36 handshapes, 10 locations, 11 orientations and 11 hand movements. They separately collected samples for 14 signs from six persons including the signer who was used in training and used them for testing, a recognition rate of 86.0% was achieved. A multilayer classifier using HMM and SVM was used in [176] to recognize 4942 signs. They claimed to have a signer independent system and reported results from three signers with an average recognition rate of 89.40%. However, the training set was not clearly explained in their work. Fang et al. [46] demonstrated signer independence in Chinese sign language with large vocabulary. They used a fuzzy decision tree together with self-organizing feature map/hidden Markov model (SOFM/HMM) to recognize 5113 signs. They tested their framework with six signers and reported an average recognition rate of 83.7% for the unseen signer. Zieren and Kraiss [182] achieved a recognition rate of 99.3% for a person dependent system which recognized 232 isolated signs in a controlled environment. They attempted to normalize the extracted features to obtain person independent system, but this yielded only a recognition accuracy of 44.1% for 221 isolated signs. This again shows that performance can drop significantly due to strong interpersonal variance in signing. Aran and Akarun [6] adopted a multi-class classification strategy using Fisher score to devise a signer independent sign language system. 19 signs consisting of manual and non-manual components were used in their experiments. They performed eight-fold, leave-one-out cross validation experiments. Best recognition rate of 65.71% was obtained, outperforming the HMM approach which yielded 52.82% recognition rate. Ding and Martinez [31] adopted a vision-based approach to extract handshape, movement and location features. They used video sequences from 10 different persons with each person performing 38 signs. Each component was recognized separately with some generic approaches and a tree-based classifier was used to combine the information of the three components for recognizing the final sign. The average recognition rate on an unseen signer was 93.9%. In [3], 30 isolated signs were recognized with standard HMMs. They showed 96.74% and 93.80% recognition rates in signer-dependent experiments with offline and online mode. For signer independent experiment in offline mode, they used 1500 samples from eight signers to train the HMMs and tested with 3545 samples from another 10 signers. In online mode, they trained the HMMs with 1800 samples from eight signers and tested on 1500 samples from 10 different signers. The respective results were 94.2% and 90.6%. Caridakis et al. [22] proposed a self organizing Markov map to recognize hand gestures and targeted to tackle intra and inter-person variation. Their approach involved transformation of a gesture representation from a series of coordinates and movements to a symbolic form and classification was based on probabilistic models. The framework was tested on 30 artificial gestures but the number of persons involved in their experiments was not clear. They conducted 10-fold cross validation experiment to evaluate the generalization capability of the proposed method and reported recognition rate of 93.0%. The work by Lichtenauer et al. [98] was slightly different. Instead of recognizing different signs, they wanted to classify a sign as “correct” and “incorrect”. A set of one-class classifiers was built and each classifier was used to judge the correctness of one sign. They extracted 3D coordinates of the hands from stereo images and used statistical dynamic time warping to align the image features with a fixed length feature model. The likelihood of the selected features was calculated based on a Gaussian model. Their vocabulary consisted of 120 signs collected from 75 persons, and 75 examples were made for each sign. They tested their system using 5-fold cross-validation. For each sign in each cycle, samples from 60 persons was used for training and the remaining 15 for testing. Classification accuracy of 96.5% was obtained.

The performance of the signer independent schemes proposed in different works are generally good; however, these works are based only on hand postures or isolated signs. Works on signer independence for recognition of continuous signing are scarce. Fang et al. [44, 45] demonstrated some signer independent attributes in their continuous recognition systems. A divide-and-conquer approach was presented to recognize continuous signs from CSL. A recurrent network based approach was first applied to segment the continuously signed sequences into isolated signs. Outputs of the recurrent networks were then used as states of the HMMs and Viterbi algorithm was applied to perform sign sequence decoding. Three signers were asked to sign 100 sentences consisting of 208 signs twice. They used partial data from two signers for training and left out one signer as "unseen" by the system. They showed recognition accuracy of 85.0% for the unseen signer while the standard HMM approach showed 81.2%. Nevertheless, the nature of the signs and sentences in their works is not clear, nor how their methodology adequately addresses signer independence. Farhadi et al. [48] proposed a somewhat different approach based on transfer learning. The key idea of transfer learning is to allow information obtained from learning one task to be transferred to another. Their method relied heavily on the discriminative features which described the intrinsic properties of a sign. Logistic regression was used to spot the word boundaries, and discriminative feature spaces based on the dictionary were used to compare with the test image feature spaces. They trained their system to recognize 90 signs and tested it on a new signer signing 40 signs in frontal view as well as 3/4 view, obtaining 64.17% and 62.5%, accuracy, respectively. In [174], 10 sentences were collected from three signers and two experiments based on 5 and all 10 sentences were conducted to test their proposed enhanced level building recognition framework. Experiments by leaving one out was carried out in a round robin manner. The best result was 80% but the worst accuracy was lower than 30%.

The signer independent works we have described so far do not use adaptation. Adaptation is common in speech recognition to generalize to a new speaker. Some researchers have also examined this approach to sign language recognition. An adaptation scheme was applied in [6] and three randomly selected examples per sign were used for the adaptation. With the SFFS search strategy, the recognition accuracy without adaptation was 65.21% while the result with adaptation was 68.35%. Adaptation with MAP estimation was applied to Bayesian networks trained to recognize 20 simulated isolated sign gestures in [114]. Data from three signers was used to train the Bayesian networks, and one new signer was used for testing. Accuracies of 52.6% and 88.5% were obtained for experiments without adaptation and with adaptation, respectively. Agris et al. [159] devised a vision-based recognition system that adapted to unknown signers to recognize 153 isolated signs. Their adaptation algorithm was based on both maximum likelihood linear regression (MLLR) and maximum a posteriori (MAP) estimation. Three signers were used for training the signer independent model, and one signer was used for testing. Supervised adaptation with 80 adaptation sequences yielded a recognition accuracy of 78.6% while the signer independent system without adaptation gave 55.5% recognition accuracy. Though promising results are shown in the system, it recognizes only isolated signs. Also, only one signer was tested with the system. They further extended signer independence works to recognition of continuous signing in [157, 158]. In [158], a database was created to tackle signer independence for continuous signing in German sign language. The database comprised 450 basic signs making up 780 sentences signed by 20 different signers. Preliminary recognition experiments were conducted based on the HMM framework. For signer independent experiments with adaptation, accuracy of 70.4%, 67.8%, and 64.9% were reported for vocabulary size of 150, 300 and 450, respectively. A more comprehensive work was carried in [157]. They applied rapid adaptation for continuous sign language recognition with combined approach of eigenvoice (EV), maximum likelihood linear regression (MLLR), and maximum a posteriori (MAP). MLLR and MAP are two commonly used adaptation strategies. The eigenvoice approach [90] mainly provides some constraints and thus reduces the number of free parameters to be estimated during the adaptation process. They found from experiments that the EV+MLLR+MAP approach provided the best results. They tested on the full corpus consisting of 450 signs and 780 sentences, with each sentence performed by 25 native signers. HMMs were trained to perform classification and a baseline recognition accuracy of 65.3% was shown for a leaving-one-out test. They obtained an increase in recognition accuracy to 75.8% with supervised adaptation using the EV+MLLR+MAP approach.

2.6 Issue 5: Beyond Recognizing Basic Signs

Thus far, most of the efforts have been in recognizing basic signs. Often, results in the literature are reported on signs that are textbook based but signs vary according to situation. Moreover, meanings which are imparted by non-manual signs and inflections have not received much attention. These are essential features for clear, meaningful sign language communication and need to be considered for a sign language recognition system to be fully functional. With respect to grammatical aspects of manual signing, only Braffort [21] and Sagawa [124] have looked at inflected signs in terms of spatio-temporal aspects. In Braffort [21], signs were separated into conventional signs and non-conventional signs. The non-conventional signs were those created depending on context. They further distinguished variable signs as conventional signs that had one or more variable parameters, depending on context. HMMs were used for classification. There were two modules in the classifier, one for classifying the conventional signs and the other for the non-conventional signs and variable signs. The assumption made was that whenever a non-conventional sign or variable sign was input to a conventional classifier module, it would be identified as an "unknown sign" and would then be processed by the other module. After the classification, an interpreter module was used to give more information or meaning to the signs. A set of rules was devised to deal with the non-conventional and variable signs. However, the implementation was not demonstrated fully. Through Braffort brought out the issue of spatio-temporal inflections, the analysis of the problem was not deep enough and the implementation of the proposed ideas was unclear. Sagawa [124] made a more thorough analysis of directional verbs. They extracted parameters that represented the spatial relationship between the inflected signs and the basic sign. They investigated the difference between the direction of motion of the directional verb and the position where the basic sign related to it was represented. Parameters like start and end position, angle and distance were used in their analysis. Templates were made based on these parameters were used for recognition. Experiments showed a promising detection rate of 93.4% for sign language words related to the directional verb. In their work, the physical locations of the signer and the addressee were restricted, and thus analysis and creation of such templates was possible. However, in sign language the number of directional verbs related to a basic sign is usually large as they depend on the location of the subject that the signer is referring to. This method might be cumbersome in real situations. Recently, Ong [114, 116, 117], brought out the aspects of other modifications that affect grammatical information. Bayesian networks were employed to decipher the basic meanings and the layered meanings caused by intensity, rate and distance.

Non-manual signals are also important for the grammatical aspects of sign language. Works which use both the manual and non-manual components for the recognition of sign language are just beginning to appear. Fusion of non-manual signals such as facial expressions and head nods and manual signing was described [129]. They showed that the inclusion of non-manual information from faces could decrease both, deletion and insertion errors in recognition of continuously signed sentences. A belief-based sequential fusion approach for the non-manual signals and manual signs was used in [7]; the non-manual signals included facial expressions and head/shoulder motion. In summary, the grammatical aspects of sign language present is a fertile area for further research.

2.7 Limitations of HMM-based Approach

As HMMs are the most commonly used approach in sign language recognition, we provide more discussion on them here. Similar to speech recognition, HMMs have been extensively employed in continuous signing recognition mainly because of their ability to segment sequential data implicitly along with classification.

As generative models, HMMs rely on non-discriminative training methods such as maximum likelihood (ML) estimation or expectation maximization algorithm (EM) in which the model of the joint probability of each class is trained separately by using the samples that belong to the class. In generative modeling, parametric distribution of the observed data is always assumed. Though HMMs have shown successful performance in many applications of sign language, the main disadvantage of a generative model is the difficulty of verifying the correctness of the joint probability model of the observed data. The problem becomes obvious when the observed data exhibits large variation, and it becomes hard to train a representative model to fit the underlying distribution of the observed data. One may argue that if enough samples are available for training, the model will be able to handle the variations well. However, getting adequate number of representative training samples usually poses significant problems as in practice. In addition, when models are trained with variations that are too large, the models become less distinctive and errors occur. It is pointed out in [19] that the generalization performance of generative models is often found to be poorer than that of discriminative models due to differences between the model and the true distribution of the data. Hence, HMMs often require adaptation to new signer, e.g. in [157, 159] or use of complex hybrid models of HMMs with other classifiers. Works by Fang et al [44, 45] also possibly indicate that HMMs in combination with other classifiers may be better suited to achieve signer independence. From the above discussion, we conclude that standard HMMs are not the best approach to deal with large signer variation.

2.8 Overview of Proposed Modeling Approach

Based on the study of sign language works, we have identified that signer variations pose problems for developing a signer independent system, and have not been sufficiently explored. It is naïve to expect that systems that are trained on a single signer or a few signers will generalize well to new signers. Two indicative works that highlight this difficulty are [71, 182]. Kadous [71] trained their system on four signers and tested the system on an unseen signer and obtained only 12%-15% accuracy. Zieren and Kraiss [182] obtained a recognition accuracy of only 44.1% for a person independent system with 221 isolated signs. This shows that a good strategy to handle signer variations is important to recognize sign language sentences from new signers. Our works with isolated signs which are presented in Chapter 3 also indicate that variations in signing need to be handled with care to develop a signer independent system. In addition, our survey of related works has indicated several key issues in continuous signing to be segmentation, movement epenthesis, and scalability to large vocabulary.

Based on the research in linguistics and our previous works, we infer that variations in sign language occur differently in each component. Some signs tend to have larger variation in handshape while other signs exhibit variations in other components. Hence, it is easier to tackle phonological variation in each component by adopting a multichannel framework, where each component is modeled independently. A potential problem with this is that the assumption of channel independence may not be correct, however, it is an engineering tradeoff that makes the recognition problem tractable. More importantly, each component is handled separately according to its characteristics.

The physical variations can be handled at the feature level, by normalizing the individual component feature vectors appropriately or by selecting invariant features which are more robust to these variations. The range of linguistic variations is broader and more difficult to deal with. Minor linguistic variations can be dealt with at the feature level while most of the larger variation has to be tackled at a higher level where the semantic meaning of the signs is formed. Hence, we propose a multilayer framework to handle the variations. At the first level of the framework, suitable features are selected and normalized. The variations are expected to be handled statistically by a probabilistic model. At the higher level where the semantic meaning of the signs is formed, another model is trained for the linguistic variations which exhibit larger differences.

Variation in the movement component is more challenging to handle as compared to the other three static components. Direct normalization cannot be applied as the start and end points of a hand gesture are usually not known in continuous signing. Hence, a different strategy is needed for the movement. We devise a scheme which relies on a simple segmentation algorithm and a line fitting approach to define unit directional vectors that characterize the direction and trajectory shape.

In this thesis we also include directional verbs which exhibit variation in grammatical aspect of ASL. The grammatical variation in this type of inflected signs appears systematic, but these signs are very context dependent, i.e. the positions of the signer and addressee can vary. We treat directional verbs as basic signs whose variations occur in the location and movement components. Hence, the same modeling techniques used for basic signs are applied to the directional verbs. At the first level of the framework, they are treated no differently from the basic signs where they are modeled with four independent components. The meaning of the directional verbs is handled flexibly at the second level of the framework where they can either be decoded as a group of signs with the same basic meaning, e.g. HELP$^{\mathrm{I\rightarrow YOU}}$, HELP$^{\mathrm{YOU\rightarrow ME}}$ or HELP$^{\mathrm{YOU\rightarrow GIRL}}$ are recognized as a group of signs with action HELP or they can also be recognized separately as different signs. For simplicity in modeling, we choose the latter decoding scheme.

Due to the considerations given previously, rather than modeling the movement epenthesis explicitly, we train the recognition scheme based only on signs and deal with the movement epenthesis problem during the sign sequence decoding process. In this work, we propose a discriminative approach based on conditional random fields (CRFs) to achieve better generalization performance. CRFs are a useful alternative to HMMs in linear sequence structure modeling because they relax the strong independence assumption between the observation variables, which are made in HMMs. The CRF-based recognition framework is made up of two layers where each layer is designed to handle the signer variation specifically. The following section presents the framework with further details.

2.8.1 Continuous Signing Recognition Framework

We adopt a glove-based approach in this thesis and the data collected using the gloves and magnetic trackers are described in details in Chapter 7. We propose a two-layer multichannel methodology that allows independent analysis and processing of the input features of the components in the first level of the system. Further recognition using the higher level descriptive components is carried out at the second level of the system. The overall system is shown in Figure 2.1. Various terms with reference to the framework are defined below:

1. **Segment**: A segment is part of the continuous observation data from a sentence, which may or may not correspond to the start and end boundary points of a sign. A sign segment is one whose beginning and end corresponds to the start and end boundary points of a sign in a continuously signed

sentence.

2. **Sub-segment**: A sub-segment is part of a segment which is obtained by over-segmenting an observation sequence using a segmentation algorithm described in Chapter 4.

3. **Phoneme**: The smallest phonetic unit in a language; each component has its own defined phonemes. In our work, a phoneme is defined by a sequence of subphones.

4. **Subphone**: Subphones are the basic units that make up a phoneme. Each component has it own defined subphones, and they are obtained by clustering the component features.

5. **Sign**: A gesture that carries the meaning of a word to convey an idea and information. It consists of four components, namely handshape, hand movement, location and orientation. A sign is made up of a combination of phonemes from the four components.

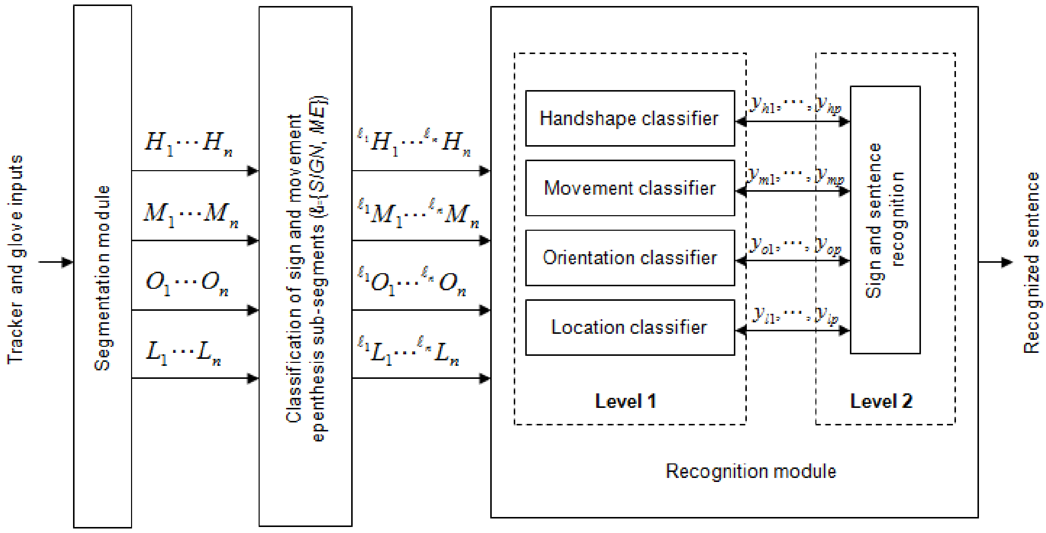


Figure 2.1: Proposed segment-based ASL recognition system which consists of a segmentation module, a classification of sign and movement epenthesis sub-segment module, and a recognition module.

Our proposed ASL recognition system is illustrated in Figure 2.1. We first segment the continuous input sequences using a segmentation algorithm based on minimum velocity and maximum directional angle change. This yields over segmented points which include most of the true boundary points. The resulting data stream consists of a sequence of sub-segments which can be part of signs or movement epentheses. The next step labels the sub-segments as belonging to sign or movement epenthesis with a CRF/SVM-based classifier. Ideally, all the sign and movement epenthesis sub-segments will be classified accurately and the movement epenthesis sub-segments will be discarded. In practice, this is difficult especially with data from signers who are not used to train the classifier. Correct detection of the movement epenthesis sub-segments is a valuable piece of information as it provides a clue to break the continuous sentence into smaller partial sequences making the problem easier to tackle. On the other hand, missed detection of the sign sub-segments may be problematic as dropping of the sign sub-segments will lead to loss of information which reduces the recognition accuracy. Hence, our aim here is not to achieve a perfect classification performance but to achieve a trade-off where as many movement epenthesis sub-segments as possible are identified without discarding too many sign sub-segments. After the sub-segments are labeled as sign or movement epenthesis, we retain only the detected sign sub-segments and discard those labeled as movement epenthesis. Subsequently, we work out a strategy to merge the remaining sub-segments to form segments and perform recognition on these segments to obtain the final sequence of the signs. We propose a CRF-based approach to merge the sub-segments efficiently during the recognition process. During training, we train the two-layer framework using only sign segments by removing all the movement epenthesis segments manually. The first level of the recognition module consists of four channels which independently recognize a sequence of phonemes for the four components. We define sequence of subphones as the input to the CRF-based recognition scheme to recognize the component phonemes. At the second level, phoneme output labels from each channel are combined and used as inputs to recognize the signs in the sentence. For testing, we modified the decoding algorithm based on the semi-Markov CRF proposed by Sarawagi and Cohen [128] to cope with our two-layer multichannel framework. In addition, we also modified the decoding algorithm to accommodate skip states so that it can deal with the incorrectly classified movement epenthesis sub-segments. During the decoding procedure, different combinations of the sub-segments are merged efficiently and features are extracted on the fly. The best path is decoded similar to the Viterbi algorithm.

One hears only those
questions for which one
is able to find answers.

Friedrich Nietzsche
(1844-1900)

3 Recognition of Isolated Signs in Signing Exact English

3.1 Scope and Motivation

In this chapter, signer variation is investigated using isolated signs from signing exact English (SEE). This is a preliminary step towards our final goal of recognizing continuously signed sentences in ASL. Isolated signs are examined because they are more straightforward to deal with as they do not involve segmentation or movement epenthesis problems. SEE is similar to ASL but it has more structured grammatical aspects. Thus, we chose SEE because of its similarity to ASL and closeness to spoken English.

Basically, SEE is based on ASL signs and expanded with words, tenses, suffixes and prefixes to give a clear and complete visual presentation of English. It takes much of its vocabulary of signs from ASL. However, it often modifies the handshapes used in ASL signs in order to incorporate the handshape used for the first letter of the English word that the SEE sign is meant to represent. Both SEE and ASL are characterized by handshape, orientation, location, hand motion, facial expression, gaze, eyebrow movement and lip motion. Generally, similar recognition strategies can be used for both, but the meanings of the recognized words and the formulation of sentences would be different as they follow different basic syntax and grammar rules.

We perform the investigation with a hierarchical classification approach that uses Fisher's linear discriminant (FLD) and a decision tree for handshape recognition, and a vector quantization principal component analysis (VQPCA) based method for isolated movement trajectory. We also devise a classifier for location and combine the results from the three component classifiers to recognize SEE signs at the sign level.

In the following, Section 3.2 describes our modeling framework for handshape recognition based on a decision tree-based classification scheme using FLD. In Section 3.3, we present the modeling framework for isolated movement trajectory recognition based on VQPCA. We present the schemes for location recognition and sign-level recognition in Section 3.4. Section 3.5 gives the experimental details with results, analysis and discussion, and Section 3.6 gives a summary of the work.

3.2 Handshape Modeling and Recognition

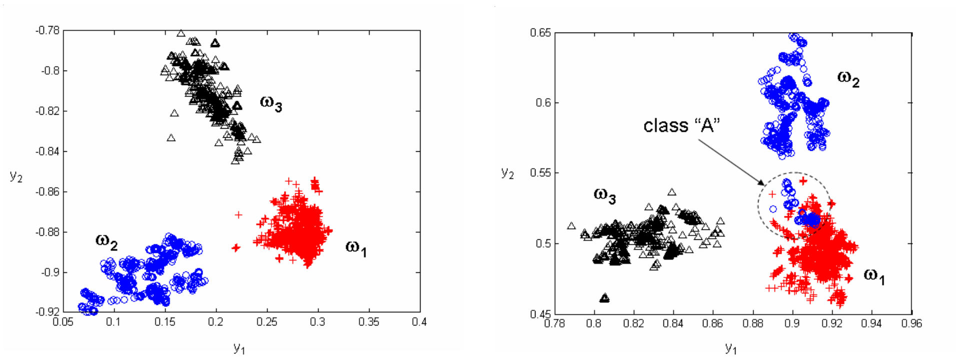
We classify 27 handshapes which are used most frequently in SEE including the 26 letters of the alphabet and 6 other important handshapes, i.e. BENT, FLAT, SMALL-C, I-L-HAND, BENT-V and L-1-HAND. The exact handshape appearances can be found in [58]. Among the letters, some have the same handshapes, and differ only in the orientation of the palm. Regardless of orientation, we group together letters that have the same handshape. For example, the letters “D” and “Z”, “G” and “Q”, “H” and “U”, “I” and “J” as well as “P” and “K”, are not differentiated and are grouped together. We use a decision-based approach using the optimal FLD-based classifier [42] at each node. With a decision tree approach, not only can a lower dimensional problem be solved at each level, but also the number of classes to be considered at each node is greatly reduced.

3.2.1 Handshape Classification with FLD-Based Decision Tree

The main issue in specifying the tree-structured classifier is to decide the number of classes at each level, and we do this by using prior knowledge of the 27 handshapes. For example, we specify three classes at the first level based on the second joint of the ring and middle fingers as follows:

- ω_1 (both fingers are bent): A, C, D, E, G, I, J, L, M, N, O, Q, S, T, X, Y, Z, SMALL-C, I-L-HAND, BENT-V, L-1-HAND
- ω_2 (the ring finger is bent while the middle finger is straight): H, K, P, R, U, V
- ω_3 (both fingers are straight): B, F, W, BENT, FLAT

The reasonableness of this grouping can be verified by studying the scatter distributions of projected handshape data. Figure 3.1(a) shows the scatter plot of the projected data of the three classes (ω_1, ω_2 and ω_3), where it is seen that the three classes are compact and well-separated. However, if the three groups had been incorrectly chosen, it is possible that they would have been non-compact and/or non-separable, and thereby suggest a different grouping. For example, if Class “A” which naturally belongs to ω_1 is put into ω_2, the scatter plot of the projected data changes to the one shown in Figure 3.1(b). Clearly, ω_1 and ω_2 are not separable in this case indicating that Class “A” is better grouped into ω_1.



(a) Scatter plot of the initial three-class data grouping for the decision tree-based handshape classifier.

(b) Scatter plot of the three-class data when class “A” is grouped into ω_2.

Figure 3.1: Scatter plots of FLD projected handshape data.

The subclasses at level 2 are obtained by further dividing each of the subclasses at level 1 into two subclasses. In the decision tree, the splitting of each class and the discrimination process are repeated until all the individual handshapes are specified at the leaf nodes of the tree. For example, the classification of the handshape “H”, follows the bold path shown in Figure 3.2, using a simple decision rule at each level. The subclasses at each level of the decision tree are summarized in Figure 3.3. The shaded boxes denote the final handshapes that will be recognized at the leaf nodes of the decision tree.

During testing, the feature vector is input at the root, and projection and classification are carried out at every intermediate node encountered in the classification path until the leaf node is reached to yield the final classification. At each node due to dimensionality reduction by FLD, the features are only 1-D or 2-D. Unlike NNs where key architectural parameters need to be estimated, the only parameter to set in our approach is the number of classes at each node of the

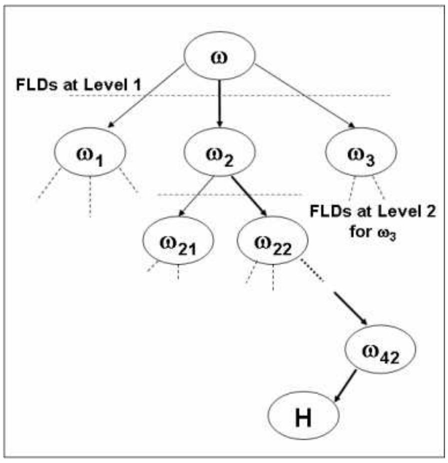
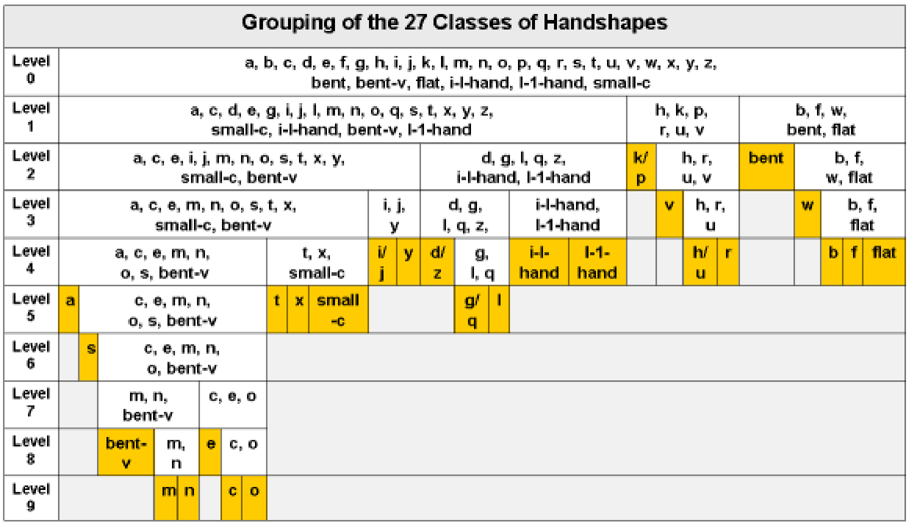


Figure 3.2: Handshape classification with decision tree and FLDs.



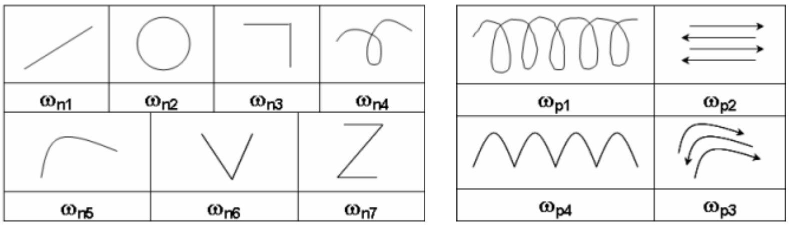
<table>
<tr><th colspan="28">Grouping of the 27 Classes of Handshapes</th></tr>
<tr><td>Level 0</td><td colspan="27">a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, bent, bent-v, flat, i-l-hand, l-1-hand, small-c</td></tr>
<tr><td>Level 1</td><td colspan="18">a, c, d, e, g, i, j, l, m, n, o, q, s, t, x, y, z, small-c, i-l-hand, bent-v, l-1-hand</td><td colspan="4">h, k, p, r, u, v</td><td colspan="5">b, f, w, bent, flat</td></tr>
<tr><td>Level 2</td><td colspan="13">a, c, e, i, j, m, n, o, s, t, x, y, small-c, bent-v</td><td colspan="5">d, g, l, q, z, i-l-hand, l-1-hand</td><td>k/p</td><td colspan="3">h, r, u, v</td><td>bent</td><td colspan="4">b, f, w, flat</td></tr>
<tr><td>Level 3</td><td colspan="11">a, c, e, m, n, o, s, t, x, small-c, bent-v</td><td colspan="2">i, j, y</td><td colspan="3">d, g, l, q, z,</td><td colspan="2">i-l-hand, l-1-hand</td><td></td><td>v</td><td colspan="2">h, r, u</td><td></td><td>w</td><td colspan="3">b, f, flat</td></tr>
<tr><td>Level 4</td><td colspan="8">a, c, e, m, n, o, s, bent-v</td><td colspan="3">t, x, small-c</td><td>i/j</td><td>y</td><td>d/z</td><td colspan="2">g, l, q</td><td>i-l-hand</td><td>l-1-hand</td><td></td><td></td><td>h/u</td><td>r</td><td></td><td></td><td>b</td><td>f</td><td>flat</td></tr>
<tr><td>Level 5</td><td>a</td><td colspan="7">c, e, m, n, o, s, bent-v</td><td>t</td><td>x</td><td>small-c</td><td colspan="3"></td><td>g/q</td><td>l</td><td colspan="11"></td></tr>
<tr><td>Level 6</td><td></td><td>s</td><td colspan="6">c, e, m, n, o, bent-v</td><td colspan="19"></td></tr>
<tr><td>Level 7</td><td colspan="2"></td><td colspan="3">m, n, bent-v</td><td colspan="3">c, e, o</td><td colspan="19"></td></tr>
<tr><td>Level 8</td><td colspan="2"></td><td>bent-v</td><td colspan="2">m, n</td><td>e</td><td colspan="2">c, o</td><td colspan="19"></td></tr>
<tr><td>Level 9</td><td colspan="3"></td><td>m</td><td>n</td><td></td><td>c</td><td>o</td><td colspan="19"></td></tr>
</table>

Figure 3.3: Subclasses of the handshapes at each level of the linear decision tree.

tree. Furthermore, unlike [26], our tree-based approach overcomes the problem of complete re-training if a new handshape needs to be included, as we group the handshapes using prior knowledge before performing FLD projection. Thus, if the new handshape falls into existing groups at the higher levels, there is no need to re-train at these higher levels; re-training may only be needed below a certain level when there is no match to existing groups. In the best possible case, we would only need to re-train the classifier at the lowest level node (leaf) that yields the actual class of the sign. In the worst case, the entire tree would need to be re-trained, but this is unlikely for handshapes used in SEE.

3.3 Movement Trajectory Modeling and Recognition

In the SEE lexicon there are 15-20 non-periodic and about 6 periodic trajectory shapes. Of these, it is found that 11 trajectories are used most often (about 97% of the time) [58], and these are considered here. The remaining trajectory shapes are quite distinct, and can easily be incorporated in our approach. Figure 3.4(a) shows seven non-periodic trajectories, $\omega_{n1} - \omega_{n7}$, in a planar view and Figure 3.4(b) shows four types of periodic trajectories, $\omega_{p1} - \omega_{p4}$. Category ω_{p1} represents a trajectory from a regular circular motion, and is periodic in nature. However, when signing, many people actually sign this as a spiral with variable radius. This latter gesture is not periodic but signers use these gestures loosely to convey the same meaning. Hence, in order to recognize this category reliably despite natural human variations, we added a non-periodic spiral trajectory class, ω_{n8}. If the final classification procedure labels the trajectory as ω_{n8} or ω_{p1}, it is taken as the periodic circular trajectory.



(a) Non-periodic trajectories. (b) Periodic trajectories.

Figure 3.4: Movement trajectories.

Here, we propose a novel scheme to recognize the hand motion trajectory of isolated gestures which can be both periodic and non-periodic [82]. It consists of a periodicity detection module followed by a classification module to recognize the two groups of gestures separately. This approach is useful because periodic gestures need to include a few cycles, and this takes longer to perform; on the other hand, if they were to be classified along with non-periodic gestures the feature vector length would need to be much longer resulting in increased computational times and complexity of the system.

We use a Fourier analysis approach related to [146] for periodicity detection, and the VQPCA clustering method proposed by Kambhatla and Leen [73] for trajectory recognition. VQPCA is a hybrid method of clustering and local PCA which makes no assumptions regarding the underlying data distribution, and finds locally linear representation of the data. The standard PCA-based approach for recognition is global in nature, and yields poor results when the data is subject to natural transformations such as rotation, translation and scaling. To overcome this, several works e.g. [118, 133] have proposed manually assigning data to different sets based on their transformation characteristics and then calculating different eigenspaces. This manual procedure is circumvented in the VQPCA approach which combines clustering with PCA, and we have used this to advantage for recognizing movement trajectories. Though the training process can be computationally expensive, we believe that its practical advantage - no tedious manual labeling process is necessary - significantly outweighs the increased computations.

3.3.1 Periodicity Detection

For periodicity detection, the raw 3-D position vectors $\mathbf{p}_t = [p_{xt}, p_{yt}, p_{zt}]^T$ from the tracker at time t are used to estimate the instantaneous speed along the trajectory as

$$|\mathbf{v}_t| = \sqrt{\mathbf{p}'_t \cdot \mathbf{p}'_t}, \tag{3.1}$$

where $\mathbf{p}'_t = [p'_{xt}, p'_{yt}, p'_{zt}]^T$ and $p'_{xt} = p_{xt} - p_{x(t-1)}$, etc.

To calculate the speed, the temporal sampling interval is assumed to be unity.

The speed along the trajectory retains the periodic nature of the signing, and is used for detecting periodicity. No normalization or scaling of the raw position data is done for computing speed. However, due to inertia of hand, starting and ending speeds at the two ends of a trajectory could be slower than normal. Using such non-representative samples in periodicity calculation could lead to errors. Hence, we discard samples at the beginning and end of a trajectory if the estimated hand speed is below a threshold. The steps of periodicity detection are described as follows:

i) As first difference is used to compute speed, smooth the speed vector $|\mathbf{v}_t|$ with a 5-point moving average filter to yield $|\mathbf{v}_{s,t}|$.

ii) Subtract the average value of the speed from the smoothed speed signal to obtain

$$|\tilde{\mathbf{v}}_{s,t}| = |\mathbf{v}_{s,t}| - |\bar{\mathbf{v}}_s|, \tag{3.2}$$

where $|\bar{\mathbf{v}}_s|$ is the average value of $|\mathbf{v}_{s,t}|$ over the complete trajectory. This is a useful step since the zero frequency component is usually quite large, and can overwhelm other peaks in the spectrum of the speed signal.

iii) Compute the autocorrelation function of the level shifted speed signal $|\tilde{\mathbf{v}}_{s,t}|$, and its discrete Fourier transform , $S(\hat{f})$, as the spectral estimate of the speed signal.

iv) If the trajectory is periodic, there will be a significant sharp peak at the fundamental frequency. If

$$S(\hat{f}_k) \geqslant \hat{\mu} + K\hat{\sigma}, \tag{3.3}$$

where $\hat{\mu}$ and $\hat{\sigma}$ are the mean and standard deviation of the spectral samples, and K is an empirically determined constant, it is taken as an indication of a periodic trajectory. We experimentally found that using $K > 2$ is too stringent and $K < 2$ gives too many false peaks. Hence, we use $K = 2$ in our implementation.

3.3.2 Movement Trajectory Classification with VQPCA

After periodicity detection, any given trajectory in 3-D space is re-sampled by linear interpolation to L samples or $2L$ samples, based on its classification as non-periodic or periodic, respectively (L is specified in Section 3.5.2). Trajectories for a given sign may have variations in location and size, arising from signer differences. Also, different signs can have the same trajectory shape but different trajectory sizes. However, the size component plays a role in only a subset of SEE signs, and can be classified separately. Hence, we do not consider it here, and normalize for translation and size by shifting the re-sampled trajectory to be centered at its mean, and normalizing it to unit length. These normalized 3-D gesture trajectories are used to form feature vectors of dimension $N = 3L$ or $N = 6L$, and used to classify trajectory shapes of non-periodic and periodic trajectories, respectively, by the VQPCA method.

Each SEE trajectory shape is characterized by an independent VQPCA model which naturally accommodates different orientations and directions - properties that make it easy to add new signs. Among the three descriptors of trajectories, viz. shape, orientation and direction, it is natural to categorize trajectories at the highest level by shape and then by orientation and direction. Hence, we use a VQPCA model to represent each gesture with different trajectory shape. The clusters formed within each VQPCA model can then be expected to represent different orientations and directions. For example, the signs AM (move slanting forwards), A (move sideways) and ABLE (move downwards) have the same trajectory shape (straight line in ω_{n1}) but with different orientations and directions; these will lie in different orientations and direction clusters. We trained eight different VQPCA models for each of the non-periodic hand trajectory shapes considered, and four models for periodic trajectory shapes.

During training, in each iteration, the VQPCA algorithm [73] first partitions the data, and then computes the local PCA of vectors in each cluster. The partitioning is done by assigning a vector to the cluster which gives minimum reconstruction error for that vector. After training, the representation obtained is the centroid and the leading eigenvectors in each cluster. To use reconstruction error as the clustering criterion in the training algorithm, two important parameters need to be specified. One is choice of the number of leading eigenvectors (m_i) to be selected in each cluster. This is specified to retain 95% of the energy of the subspace. The other parameter is the number of clusters (Q) in each VQPCA model; this is specified to be the number of trajectory orientations and directions with the same shape.

For subsequent classification, a trajectory vector is projected onto the local PCA subspaces of each VQPCA model and reconstructed. The vector is classified to the model which globally yields the smallest reconstruction error to yield the trajectory shape. The specific cluster within the VQPCA model gives the trajectory orientation and direction.

3.4 Sign-Level Recognition

We now integrate the component classifiers in order to recognize complete SEE signs from a vocabulary of 28 sign words, made up of 18 different handshapes and 9 different trajectory shapes. We use the previously trained handshape and movement trajectory classifiers, along with a location classifier, independently in each channel, and use table look-up to recognize the complete sign.

The start and end locations of the trajectories were used for classifying location. Here, the two dimensions of the tracker data representing the frontal plane of the signer were clustered into five groups to represent the five signing areas indicated in Figure 3.5. Thresholds (θ_i) were then set along the vertical axis to partition the signing areas.

In the handshape channel, signs can have fixed or dynamic handshapes. For the latter case, the starting and ending handshapes are important, while transition handshapes convey no meaningful information. We classify the handshapes at every time instant, and if more than 90% of the handshapes of a sign word are recognized as belonging to a single class, the sign word is taken to have the handshape of the majority class. Otherwise, the sign word is considered to have dynamic handshape, and the first and last 5% of the data are used to classify the starting and ending handshapes, respectively.

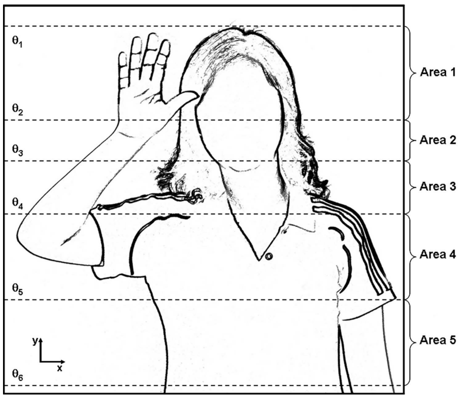


Figure 3.5: 5 signing spaces for hand location.

From the movement component, when we used the previously trained VQPCA models for classifying the new test data acquired for sign-level recognition, we found that the accuracy degraded for the circle trajectory compared to the others. This was found to be due to larger inter- and intra-person variation when signing the circle. Hence, to ensure good performance, more clusters were added to the circle VQPCA model previously obtained. This yielded better representation of the direction and orientation plane of the circle. For retraining the circle model, however, we used only the initial training data acquired for the trajectory classification experiment.

3.5 Experimental Results

The CyberGlove® and Polhemus FASTRACK® system as described in Chapter 7, Section 7.2 was used to acquire the finger angles for the handshape and hand positions for movement trajectory. The detailed configuration of the hardware and the data collected for this part of the experiment can be found in [84].

3.5.1 Handshape Recognition

The data for handshape recognition was provided by 12 signers, denoted as $P_1, P_2, \ldots, P_{12}$. The signers consisted of males and females as well as expert and non-expert signers. These details are given in Table 3.1. Each of the 12 signers performed each of the 27 handshapes 40-50 times. The training set included data from five signers, with all data from one signer and 70% of the data from the other four signers. The remaining 30% of the data from these four signers and all the data contributed by the seven unseen signers (i.e. signers whose data was not used for training) was used for testing. Of the four signers in the training set, three were expert signers while the fourth was not.

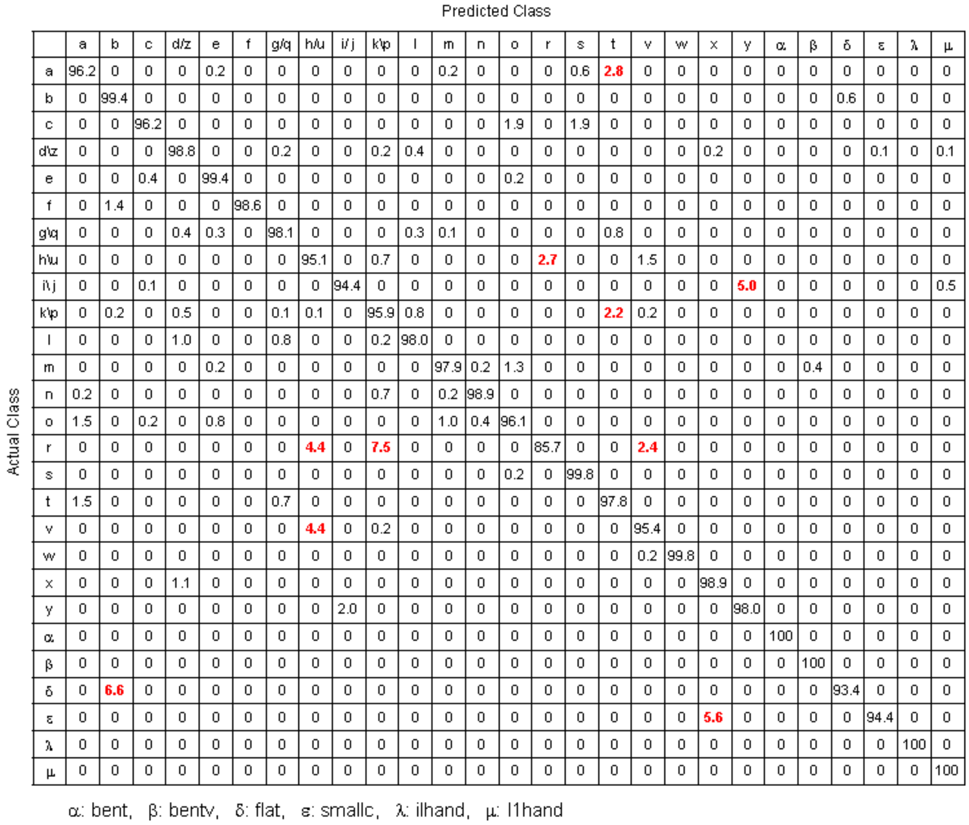
Table 3.1: Summary of the signers' status.

Person	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}
Gender	F	F	F	F	F	F	F	M	M	M	M	M
Expert signer	Yes	Yes	Yes	No	No	No	No	Yes	Yes	No	No	No

Figure 3.6 shows the confusion matrix obtained for the linear decision tree classifier for recognizing handshapes. Generally, good results were obtained for all

the classes except for handshape “R” which had a relatively low recognition rate of 85.7%. It is observed from the confusion matrix that handshape “R” is likely to be recognized as handshapes “H/U”, “K/P” or “V”. Generally, handshapes which are close in appearance and are loosely signed, are more likely to be confused during recognition. For example, “A” vs. “T”; “H/U” vs. “R”; “I/J” vs. “Y”; “V” vs. “H/U”; “FLAT” vs. “B” and “SMALLC” vs. “X” (see Figure 3.6). The sensors of the CyberGlove® do not give very distinctive measurements for some of these handshapes. For example, for “U” and “R”, only slight differences can be observed in the middle finger PIJ angle as well as the middle-ring abduction angle. As for handshapes “U” and “V”, the small difference is in the index-middle abduction angle. This accounts for the relatively high errors. The highest error arose from misclassification of “R” to “K/P” even though the two exemplar handshapes are somewhat different from each other. However, inter- and intra-signer handshape variations may have caused the two classes of handshapes to overlap.

Table 3.2 shows handshape recognition results for the test data of individual signers using the FLD-based tree classifier. As can be seen, the recognition results are very good across all signers. The average recognition rate for unseen signers is 96.1%, while it is 99.6% if the signers are included in the training set (seen signers). As a point of comparison, if the training set was changed to include two expert and two non-expert signers (one expert signer less compared to previous case), the average recognition rate for the unseen and seen signers dropped slightly to 94.9% and 97.6%, respectively. The good performance on unseen signers is encouraging. When acquiring data from the Cyberglove®, hand size which is likely to be different for males and females did not appear to affect the recognition results. On the other hand, whether a person is an expert signer or not affects the recognition results. It is observed from the table that recognition results for expert signers are extremely good, e.g. P_1, P_2, P_3 and P_9, while for non-expert signers the recognition results are lower, e.g. recognition rate for signer P_6 is 89.5%.


Predicted Class
Actual Class

	a	b	c	d/z	e	f	g/q	h/u	i/ j	k\p	l	m	n	o	r	s	t	v	w	x	y	α	β	δ	ε	λ	μ
a	96.2	0	0	0	0.2	0	0	0	0	0	0	0.2	0	0	0	0.6	**2.8**	0	0	0	0	0	0	0	0	0	0
b	0	99.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.6	0	0	0
c	0	0	96.2	0	0	0	0	0	0	0	0	0	0	1.9	0	1.9	0	0	0	0	0	0	0	0	0	0	0
d\z	0	0	0	98.8	0	0	0.2	0	0	0.2	0.4	0	0	0	0	0	0	0	0	0.2	0	0	0	0	0.1	0	0.1
e	0	0	0.4	0	99.4	0	0	0	0	0	0	0	0	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0
f	0	1.4	0	0	0	98.6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g\q	0	0	0	0.4	0.3	0	98.1	0	0	0	0.3	0.1	0	0	0	0	0.8	0	0	0	0	0	0	0	0	0	0
h\u	0	0	0	0	0	0	0	95.1	0	0.7	0	0	0	0	**2.7**	0	0	1.5	0	0	0	0	0	0	0	0	0
i\ j	0	0	0.1	0	0	0	0	0	94.4	0	0	0	0	0	0	0	0	0	0	0	**5.0**	0	0	0	0	0	0.5
k\p	0	0.2	0	0.5	0	0	0.1	0.1	0	95.9	0.8	0	0	0	0	0	**2.2**	0.2	0	0	0	0	0	0	0	0	0
l	0	0	0	1.0	0	0	0.8	0	0	0.2	98.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0.2	0	0	0	0	0	0	97.9	0.2	1.3	0	0	0	0	0	0	0	0	0.4	0	0	0	0
n	0.2	0	0	0	0	0	0	0	0	0.7	0	0.2	98.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0
o	1.5	0	0.2	0	0.8	0	0	0	0	0	0	1.0	0.4	96.1	0	0	0	0	0	0	0	0	0	0	0	0	0
r	0	0	0	0	0	0	0	**4.4**	0	**7.5**	0	0	0	0	85.7	0	0	**2.4**	0	0	0	0	0	0	0	0	0
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0.2	0	99.8	0	0	0	0	0	0	0	0	0	0	0
t	1.5	0	0	0	0	0	0.7	0	0	0	0	0	0	0	0	0	97.8	0	0	0	0	0	0	0	0	0	0
v	0	0	0	0	0	0	0	**4.4**	0	0.2	0	0	0	0	0	0	0	95.4	0	0	0	0	0	0	0	0	0
w	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.2	99.8	0	0	0	0	0	0	0	0
x	0	0	0	1.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98.9	0	0	0	0	0	0	0
y	0	0	0	0	0	0	0	0	2.0	0	0	0	0	0	0	0	0	0	0	0	98.0	0	0	0	0	0	0
α	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
β	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0
δ	0	**6.6**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	93.4	0	0	0
ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**5.6**	0	0	0	0	94.4	0	0
λ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
μ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

α: bent, β: bentv, δ: flat, ε: smallc, λ: ilhand, μ: l1hand

Figure 3.6: Confusion matrix for handshape recognition by the decision tree classifier.

3.5.2 Movement Trajectory Recognition

The experimental data was obtained from 10 signers with all performing the 11 classes of trajectory shapes. Each person contributed about 90 samples for each gesture class, where each trajectory shape was signed in at least three different orientations or directions, to correspond to SEE signs. All the collected data was used to test the periodicity detection algorithm described in Section 3.3. Figures 3.7 and 3.8 show plots of the speed and power spectrum of a periodic

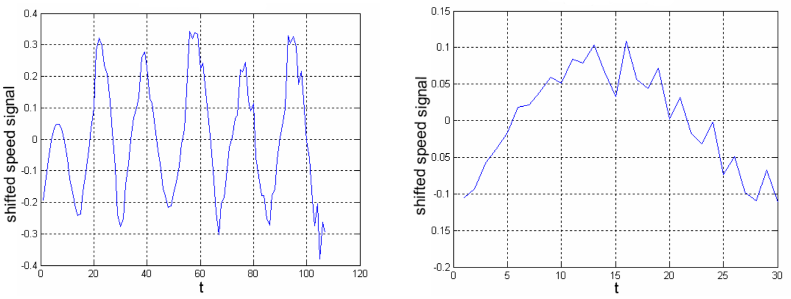
Table 3.2: Handshape recognition results for individual signers.

Recognition rate (%)			
Signer not in training set		Signer in training set	
P_4	95.9	P_1	99.3
P_5	99.7	P_2	99.0
P_6	89.5	P_3	100.0
P_7	94.4	P_{12}	100.0
P_9	99.9	-	-
P_{10}	98.9	-	-
P_{11}	94.7	-	-
Average = 96.1		Average = 99.6	

gesture (ω_{p4}) and a non-periodic gesture (ω_{n1}), respectively, where it can be seen that the spectral peak for periodic signals is considerably larger than the one for non-periodic signals, showing that they can be reliably discriminated.

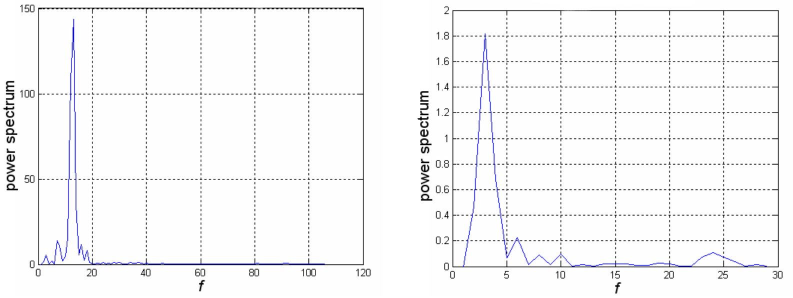
The periodicity detection results are summarized in Tables 3.3 and 3.4. The raw periodicity detection accuracy for ω_{n8}/ω_{p1} is not meaningful as the periodic circular motion (ω_{p1}) is also loosely signed by subjects as a spiral (ω_{n8}) which is not a periodic trajectory (see also Section 3.3). The recognition accuracy of this category can only be inferred in conjunction with trajectory classification accuracy for the nominal periodic circular gestures. On the whole, the average periodic/non-periodic detection rate is quite good at 97.2%. The periodicity detection algorithm works very well when strong periodicity is exhibited, i.e. there are sufficient number of repetitions. Error in detection increases when the periodicity cue is weak, e.g. when a gesture is repeated only twice. This explains the somewhat lower accuracy compared to non-periodic signals.

For trajectory classification, as mentioned in Section 3.3.2, before being input to the VQPCA algorithm, non-periodic and periodic trajectories were re-sampled to $L = 60$, and $L = 120$ samples, respectively, and were normalized for translation and size. Training data from only four signers was used for the VQPCA algorithm.



(a) Periodic (ω_{p4}). (b) Non-periodic (ω_{n1}).

Figure 3.7: Speed plots for a periodic and non-periodic movement trajectory.



(a) Periodic (ω_{p4}). (b) Non-periodic (ω_{n1}).

Figure 3.8: Power spectra for a periodic and non-periodic movement trajectory.

Table 3.3: Detection of non-periodic gestures by Fourier analysis.

Non-periodic movement trajectory							
	ω_{n1}	ω_{n2}	ω_{n3}	ω_{n4}	ω_{n5}	ω_{n6}	ω_{n7}
Accuracy (%)	100	99.9	100	92.7	100	99.9	99.0

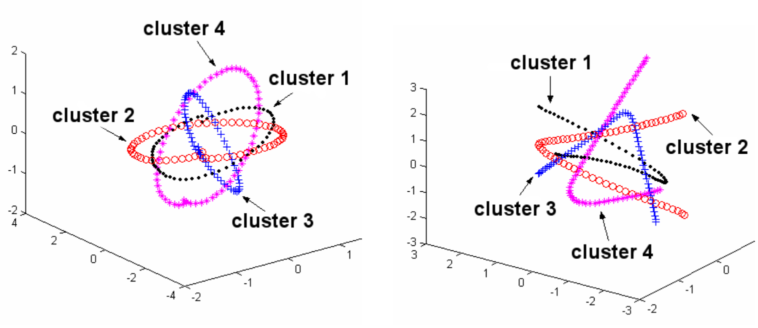
Table 3.4: Detection of periodic gestures by Fourier analysis.

Periodic movement trajectory				
	ω_{n8}/ω_{p1}	ω_{p2}	ω_{p3}	ω_{p4}
Accuracy (%)	-	92.0	91.3	96.8

From these signers, 70% of the data of each class was used for training and the remaining 30% was used for testing. In addition, all the data from the remaining six unseen signers was used for testing. Each trajectory shape was learned and represented by one VQPCA model. We used 4 to 7 clusters in each of the models.

The number of retained eigenvectors in each cluster ranged from 2 to 17.

Figure 3.9 shows the trained centroids of different clusters for different orientations and directions for two of the trajectories (circle and v-shape) obtained by VQPCA. Each VQPCA model consists of four clusters representing four different orientations or directions. Examination of the VQPCA model for each of the trajectories showed that clusters had correctly formed according to specific orientations and directions.



(a) Model 2 (circle).

(b) Model 6 (v-shape).

Figure 3.9: Centroids of clusters in VQPCA models for circle and v-shape trajectories.

Tables 3.5 and 3.6 give the average recognition results for all the gestures from the test set. $T_{average}$ represents results for the unseen 30% of the data of the four signers used for training while $S_{average}$ represents results for all the data contributed by the six unseen signers. As can be observed from the tables, VQPCA gives very good classification results for both the non-periodic gestures and periodic gestures. For the unseen test data of the four signers whose data was used for training the classifier ($T_{average}$), the total average recognition rate for both periodic and non-periodic gestures was 99.7%. The total average classification rate for the test data from unseen signers ($S_{average}$) was also good at 97.3% for non-periodic gestures and 97.0% for periodic gestures. From the high classification accuracy of the ω_{n8}/ω_{p1} class, we can infer that the periodicity classifier works reliably for the periodic circular gesture also.

Table 3.5: Average recognition rates with VQPCA for non-periodic gestures.

	w_{n1}	w_{n2}	w_{n3}	w_{n4}	w_{n5}	w_{n6}	w_{n7}	w_{n8}	$Average$
$T_{average}$ (%)	100	98.7	100	100	99.4	100	100	99.4	99.7
$S_{average}$ (%)	96.8	95.8	98.2	98.1	96.4	95.9	99.8	97.7	97.3

Table 3.6: Average recognition rates with VQPCA for periodic gestures.

	w_{p1}	w_{p2}	w_{p3}	w_{p4}	$Average$
$T_{average}$(%)	100	100	100	98.9	99.7
$S_{average}$(%)	99.8	93.9	99.1	95.3	97.0

3.5.3 Recognition of Complete SEE Signs

Here, the classifiers which were trained separately for handshape, trajectory and location recognition were integrated to recognize complete isolated SEE signs. To evaluate sign recognition performance, we acquired a new data set for 28 SEE signs from four signers. Each person was asked to sign each sign word 10 times. Some of the sign words were completely new, and did not appear in component classifier training. For example, the sign word OVAL which has a circular trajectory shape was used for testing, although the VQPCA classifier was trained with movement data from the sign words ABOUT, APPROXIMATE and TWIRL, representing different orientations and directions of the circle model.

A total of 1120 handshapes from the sign words were tested using the previously trained linear decision tree classifier. This yielded a recognition rate of 94.6%, which is slightly worse than the handshape recognition rate of 96.1% obtained for unseen signers in Section 3.5.1. This could be attributed to the fact that signers tend to be less conscious about handshape when signing a complete sign word which is more than just a handshape. Hence, handshapes like "A" and "S" might be signed with larger variability.

In movement recognition, when only trajectory shapes were considered, a recognition rate of 96.3% was obtained for the 1120 trajectories. This result is comparable to our previous trajectory shape experiment described in Section 3.5.2, showing that our VQPCA-based algorithm retains its good performance for trajectory shape recognition. The recognition rate was somewhat lower at 92.2% when direction of movement and orientation of the trajectory plane were included.

For recognizing location, the location space was divided into five areas as described in Section 3.4. The start and end locations of a trajectory were extracted for recognition, and an accuracy of 99.0% was obtained. For overall sign recognition, the results from the three classifiers were used in a table look-up procedure. Accuracy of the combined sign recognition system was computed by considering a sign word to be correctly recognized only if the recognition results from handshape, movement and location classifiers were all correct. This yielded a sign-level recognition rate of 86.8%.

We note here that even though a completely new set of data from new signers was acquired, separately trained classifiers as indicated above have yielded high accuracy for sign-level classification, indicating the feasibility of recognizing sign words with the component classifiers developed here.

3.6 Summary

We have presented a scheme to recognize isolated SEE signs based on combining the component classifiers for handshape, movement trajectory and location. On a 28-sign SEE vocabulary, the sign word recognition scheme yielded 86.8% accuracy.

Handshape and movement are the most important components of SEE signs, and we have proposed robust and effective methods for recognizing them. The proposed handshape recognition and trajectory recognition algorithms both show good generalization ability to signers who were not used to train the system, which is an important consideration in practice.

For the component classifiers, handshape recognition using a decision tree based classifier with Fisher's linear discriminant yielded an average recognition rate of 96.1% on unseen signers. Fourier analysis was used to detect periodic movement trajectories, and this yielded an average accuracy of 97.2%. Generally, the experiments show that signer independence is viable if the phonological variation in sign language is handled properly.

Besides demonstrating good classification results and generalization to unseen signers, some valuable observations for our subsequent work on continuous signing are also noted. From the experiments, we observe that variation in handshape data due to physical hand size variations are small, and are easily handled by appropriate normalization. Rather, a more noticeable impact is seen in the classification results when the handshape variation is caused by different ways of signing. For example, the confusion between "U", "V" and "R" etc as discussed in Section 3.5.1 can easily occur if the signers are asked to make the signs naturally, without constraints. Also, whether a person is an expert signer or not affects the classification performance. This further underscores the idea that linguistic variations which occur in sign language due to different signers' style, habit, education, family background etc as discussed in Chapter 1, must be robustly handled in developing a signer independent system.

Other useful observations can also be made from the movement component experiments. The unique habits of individual signers, give rise to variations in the trajectory shape, motion direction, size and shape, which must be handled robustly for recognition. Here, with isolated signs, the VQPCA method has been used under the assumption that the end points of a trajectory have been accurately identified, and that the number of sample points in the input trajectories to VQPCA are fixed. Hence, normalization for size and translation can be done easily and the experiments demonstrate satisfactory results on unseen signers. However, when the work is extended to continuously signed sentences, the start and end points of a sign are no longer known, and simple normalization is not feasible. This issue needs to be addressed carefully for extension to continuous signing.

What matters deafness
of the ears when the
mind hears?
The one true deafness,
the incurable deafness,
is that of the mind.
Victor Hugo (1802-1885)

4 Phoneme Transcription for Sign Language

4.1 Overview of Approach

An automatic phoneme transcription procedure is an essential step towards building practical sign language recognition systems that scale well with vocabulary size, and thus it is important to devise an efficient strategy for consistent phoneme transcription from continuously signed sentences. We propose such a scheme, designed to accommodate naturally signed ASL sentences rather than only textbook signs. A set of phonemes is defined for each of the four parallel components. Signed sentences can then be labeled with a sequence of these phonemes to infer the lexical meaning of signs. Here, we present a novel approach to transcribe phonemes for the trajectory in the hand movement channel and use simple clustering algorithms for the other three components, i.e. handshape, palm orientation, and location.

The remainder of this chapter is organized as follows. Background on Bayesian networks is given in Section 4.2. Section 4.3 describes our phoneme transcription procedure for the movement component which includes a segmentation algorithm and a PCA-based transcription method. In Section 4.4, the phoneme transcription procedure for the static components (handshape, palm orientation and location) is described. Section 4.5 summarizes our phoneme transcription scheme.

4.2 Bayesian Networks

A Bayesian network is a directed acyclic graph (DAG) where each node represents a random variable and each directed edge between nodes represents a probabilistic dependency. Absence of edges in the graph implies conditional independence and this allows the joint distribution of a set of random variables, $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)$, to be factored as a product of local conditional probabilities:

$$p(Z_1, Z_2, \ldots, Z_d) = \prod_{i=1}^{d} p(Z_i|\Gamma_{Z_i}), \tag{4.1}$$

where Γ_{Z_i} denotes the set of parents of random variable Z_i. Often, the structure of the network is manually defined using domain knowledge of the problem although the structure can be derived from training data. Thus, we only need to estimate network parameters from training data when the network structure is known. If all the node values are known at training time, the goal of learning is to find the values of network parameters using maximum likelihood (ML) estimation or Bayesian estimation. In the case of missing values, the EM algorithm can be used to find a locally optimal maximum-likelihood estimate of the parameters. After training, the network is used to infer the probability of a query node given the observed values of the evidence nodes in response to a test query.

The likelihood is written as

$$\begin{aligned} p(\mathcal{D}|\boldsymbol{\theta}) &= p(\mathbf{z}^1, \ldots, \mathbf{z}^{\mathcal{N}}|\boldsymbol{\theta}) \\ &= \prod_{r=1}^{\mathcal{N}} p(\mathbf{z}^r|\boldsymbol{\theta}) \\ &= \prod_{r=1}^{\mathcal{N}} \prod_{i=1}^{d} p(Z_i = z_i^r|\Gamma_{Z_i} = \gamma_{Z_i}^r, \boldsymbol{\theta}_i), \end{aligned} \tag{4.2}$$

where $\boldsymbol{\theta}_i$ is the vector of parameters for the distribution $p(Z_i|\Gamma_{Z_i})$. The log-likelihood is

$$\log p(\mathcal{D}|\boldsymbol{\theta}) = \sum_{r=1}^{\mathcal{N}} \sum_{i=1}^{d} \log p(Z_i = z_i^r|\Gamma_{Z_i} = \gamma_{Z_i}^r, \boldsymbol{\theta}_i), \tag{4.3}$$

which is maximized to obtain the parameters as

$$\begin{aligned} \hat{\boldsymbol{\theta}}_{ML} &= \operatorname*{argmax}_{\boldsymbol{\theta}} \log p(\mathcal{D}|\boldsymbol{\theta}) \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{r=1}^{\mathcal{N}} \sum_{i=1}^{d} \log p(Z_i = z_i^r|\Gamma_{Z_i} = \gamma_{Z_i}^r, \boldsymbol{\theta}_i). \end{aligned} \tag{4.4}$$

Hence, we have an independent estimation problem for each $\boldsymbol{\theta}_i$,

$$\hat{\boldsymbol{\theta}}_i = \operatorname*{argmax}_{\boldsymbol{\theta}_i} \sum_{r=1}^{\mathcal{N}} \log p(Z_i = z_i^r|\Gamma_{Z_i} = \gamma_{Z_i}^r, \boldsymbol{\theta}_i). \tag{4.5}$$

For a Bayesian network with discrete nodes, we have $\theta_{ijk} \triangleq p(Z_i = k|\Gamma_{Z_i} = j)$ and the ML parameter estimate is given as

$$\hat{\theta}_{ijk} = \frac{N_{ijk}}{\sum_{j'} N_{ij'k}}, \tag{4.6}$$

where N_{ijk} is the number of times $Z_i = k$ and $\Gamma_{Z_i} = j$ occur in the observation data set.

4.3 Phoneme Transcription for Hand Movement Trajectory

We propose an automatic phoneme transcription procedure for the movement component which saves time and intensive labor required for manual transcription. There are two steps in transcribing phonemes from continuously signed sentences, viz., segmentation of the hand trajectories, followed by phoneme transcription.

Several works have considered automatic trajectory segmentation for various purposes. In our work, we consider segmenting naturally signed ASL sentences by adopting Sagawa and Takeuchi's [125] approach of using minimum velocity and maximum change of directional angle as the basis for segmentation. This leads to considerable oversegmentation of the trajectories, however, such that the true segment boundary points are highly likely to be a subset of this initial segmentation. Simple thresholding to reduce the oversegmentation did not work well enough as many of the true boundary points were discarded as well. Hence, we investigated more refined methods to identify the true boundary points and minimize the false alarms; one was a rule-based method while the other was based on Bayesian networks. In order to further improve the detection accuracy of the true boundary points obtained for each sentence, we used majority voting using several examples of a sentence.

For phoneme transcription, we extracted PCA-based features from the segments and clustered them. Even though the hand movement trajectory of a complete sentence may be a complex 3-D curve, we can expect that the individual segments obtained will correspond to lines or planar curves. Hence, PCA of these segments will directly yield the directions of the lines and the planes of the curves. We applied PCA to each segment, and clustered features by k-means to define phonemes with geometric meaning. Our approach of using PCA features alleviates some of the problems in [14, 46, 162, 163] by allowing the use of simple k-means, rather than complex algorithms to cluster the temporal segments. Further, unlike the phonemes obtained in [14], the phonemes obtained by our approach are related to phonetic concepts which are more meaningful for describing sign language. Other works have also used PCA-based features to perform gesture or sign language recognition. Nam and Wohn [107] projected the 3-D hand trajectory to a plane found by PCA, and used a chain encoding scheme for representing the hand movement path for recognition. In related work, Vogler [154] used the first and the second eigenvalues from PCA to differentiate between lines and curves and used them as global features for sign language recognition. However, they did not explore this further. We believe that this is a good starting point to facilitate phoneme transcription.

4.3.1 Automatic Trajectory Segmentation

We investigated two schemes to classify the oversegmented points as true boundary points or false alarms. One is a rule-based scheme and the other is based on a naïve Bayesian network. For each scheme, the segmented trajectories from several examples of the same sentence are used in a majority voting scheme to further refine the results. Finally, one of the schemes is chosen for phoneme transcription based on performance characteristics.

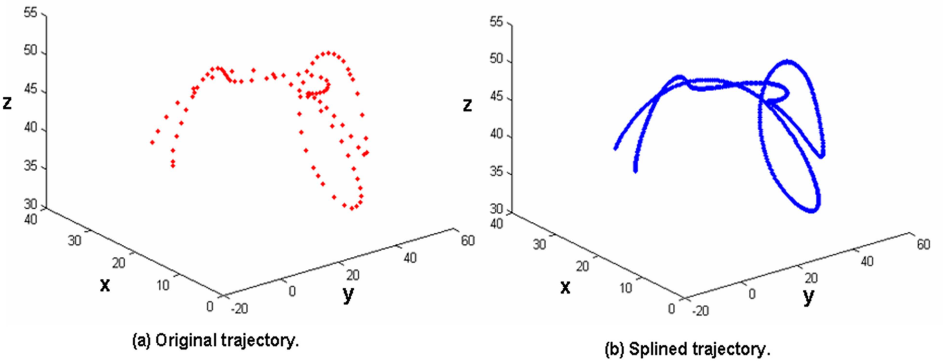
4.3.1.1 Initial Segmentation

Temporal segmentation is implemented by detecting points of minimal velocity and maximal change of directional angle. The continuous raw 3-D hand trajectory data is first interpolated and smoothed using splines. This step is useful for more accurate and reliable velocity and directional angle computation. Figure 4.1

shows an example of original and splined hand trajectories of a sentence. Velocity $\mathbf{v}_t$ is estimated as

$$\mathbf{v}_t = \mathbf{p}_{t+1} - \mathbf{p}_t, \tag{4.7}$$

where $\mathbf{p}_t = (x_t, y_t, z_t)$ is the 3-D position at time t.



(a) Original trajectory.

(b) Splined trajectory.

Figure 4.1: Original and splined trajectories.

The directional angle change, θ_t, is computed as the angle between two vectors formed by three consecutive 3-D positions as shown in Figure 4.2. Thus

$$\cos(\theta_t) = \frac{\mathbf{u}_1 \cdot \mathbf{u}_2}{\|\mathbf{u}_1\| \|\mathbf{u}_2\|}, \tag{4.8}$$

where $\mathbf{u}_1 = \mathbf{p}_t - \mathbf{p}_{t-1}$ and $\mathbf{u}_2 = \mathbf{p}_{t+1} - \mathbf{p}_t$.

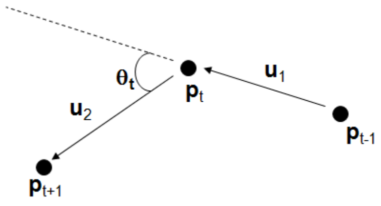


Figure 4.2: Directional angle.

The initial segment boundaries are marked at the points of local velocity minima and maxima of directional angle change. These are processed by i) rules or ii) naïve Bayesian network to identify true boundary points and minimize the false boundary points.

4.3.1.2 Rule-Based Classifier

The rules for the classifier are formulated based on observation and features that characterize the boundary points; these features are summarized in Table 4.1. **minVel** and **maxAng** are binary features to indicate a point of minimal velocity and a point of maximal change of directional angle, respectively. **normVel** is the normalized velocity with respect to the peak, lying in [0-1], and **dirAng** is the absolute directional angle change of a point in [0°-180°]. **lftValley**(P_{vl}/H_{vl}) and **rgtValley**(P_{vr}/H_{vr}) characterize the valley associated with a velocity minimum. Similarly, **lftPeak**(P_{al}/H_{al}) and **rgtPeak**(P_{ar}/H_{ar}) characterize the peak associated with an angle maximum. Figure 4.3 illustrates the definitions of these parameters.

Table 4.1: Features characterizing velocity minima and maxima of directional angle change.

Feature	Description
minVel	Binary indicator for a local minimum of velocity.
maxAng	Binary indicator for a local maximum of directional angle change.
normVel	Normalized velocity values.
dirAng	Absolute angle values.
lftValley	P_{vl}/H_{vl} (see Figure 4.3).
rgtValley	P_{vr}/H_{vr} (see Figure 4.3).
lftPeak	P_{al}/H_{al} (see Figure 4.3).
rgtPeak	P_{ar}/H_{ar} (see Figure 4.3).

The rules are summarized in Table 4.2. Rule 1 checks if a boundary point corresponds to a local minimum of velocity and maximum change of directional angle, and indicates a strong potential boundary point if both are true. Rules 2, 3 and 4 examine the characteristics of a valley in the velocity profile and a peak in the change of directional angle profile. A true detection should be characterized by a deep valley while a shallow valley is possibly a false alarm. A true maximal

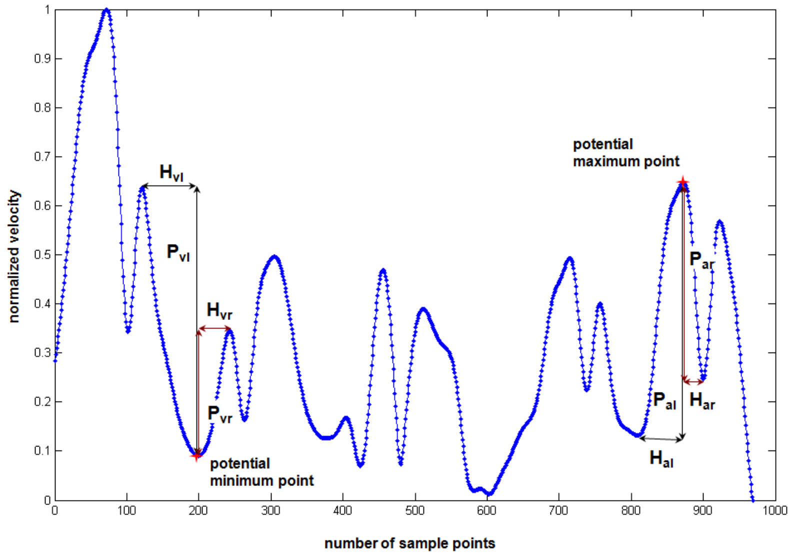


Figure 4.3: Definition of parameters for features described in Table 4.2.

Table 4.2: Formulated rules.

Rule	Description
Rule 1	§*if* (**minVel** = TRUE) *and* (**maxAng** = TRUE), check Rule 2
	elseif (**minVel** = TRUE) *and* (**maxAng** = FALSE), check Rule 3
	else check Rule 4
Rule 2	*if* (**lftValley** $> T_1$ *or* **rgtValley** $> T_2$)
	and (**lftPeak** $> T_3$ *or* **rgtPeak** $> T_4$), detection = TRUE POINT
	else detection = FALSE ALARM
Rule 3	*if* (**lftValley** $> T_1$ *or* **rgtValley** $> T_2$), check Rule 5
	else detection = FALSE ALARM
Rule 4	*if* (**lftPeak** $> T_3$ *or* **rgtPeak** $> T_4$), check Rule 5
	else detection = FALSE ALARM
Rule 5	*if* (**normVel** $<= T_5$ *and* **dirAng** $>= T_6$)
	or (**dirAng** $>= T_7$ *and* **normVel** $<= T_8$), detection = TRUE POINT
	else detection = FALSE ALARM

note: T_i, i = 1,2,. . .,8, are thresholds found empirically, and ($T_5 < T_8$), ($T_7 > T_6$).

§the condition "(**minVel** = FALSE) *and* (**maxAng** = FALSE)" will not occur.

angle change is characterized by relatively sharp peak. Rule 5 checks the values of the normalized velocity and directional angle change. A point with a high velocity value, and a low directional angle change is likely to be a false alarm, while a point with a low velocity value and a high directional angle change is a potential boundary point. However, we relax these conditions and accept a point with a very low velocity (T_5) but moderately high directional angle change (T_6) as a true boundary point. On the other hand, if this condition is not met, but the point exhibits a very high directional angle change (T_7) and moderately low velocity (T_8), we also consider it as a true boundary point. The threshold values (T_i) are found empirically as described in Section 7.3.

4.3.1.3 Naïve Bayesian Network Classifier

The naïve Bayesian network classifier has the structure shown in Figure 4.4. The query node, **Detection** is the node whose value is to be inferred, and the four observed nodes are **maxAng**, **minVel**, **normVel** and **dirAng**. The description of each node is given in Table 4.3.

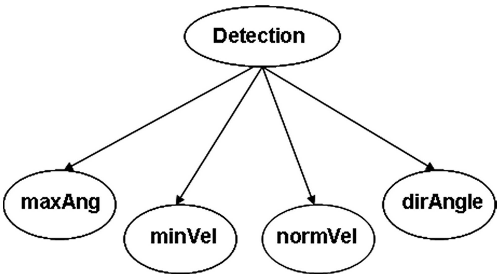


Figure 4.4: Naïve Bayesian network for classifying segmentation boundary points.

All the nodes have a finite number of discrete states, and their distribution is represented by a conditional probability table (CPT). During training, as the values of all the nodes are known, the CPTs are learned by maximum likelihood estimation as described in Section 4.2. When a test point is queried, the trained network is used to infer the probability of the query node (TruePoint or FalseAlarm) given the observed values of the evidence nodes. The detection rule is

$$S_{detect} = \underset{j=\{TruePoint, FalseAlarm\}}{\text{argmax}} \; p(\Gamma = j) \prod_{i=1}^{m} p(Z_i = z_i | \Gamma = j), \tag{4.9}$$

where S_{detect} will be assigned as a true boundary point or false alarm.

Table 4.3: Summary of the naïve Bayesian network nodes and their values.

Node	Variable	State	Description
Detection	Γ	TruePoint, FalseAlarm	True boundary point or false alarm.
maxAng	Z_1	Yes, No	Indicator for a local maximum of directional angle.
minVel	Z_2	Yes, No	Indicator for a local minimum of velocity.
normVel	Z_3	Low, Medium, High	Discretized normalized velocity values.
dirAng	Z_4	Low, Medium, High	Discretized absolute angle values.

4.3.1.4 Voting Algorithm

The segment boundary points obtained from trajectories of different samples belonging to the same sentence may not be consistent, as Figure 4.5 shows. Hence, we further refined performance by using majority voting to increase confidence in a point if it appeared consistently in all the trajectories. In order to find corresponding points, we first aligned the sample trajectories belonging to the same sentence by dynamic time warping (DTW). Each point then votes for the neighborhood it belongs to. For example, in Figure 4.5, at location R_1, F_2 is missing and the number of votes for a true boundary point is two out of three; thus, the point at R_1 is taken as a true boundary point. On the other hand, at R_2, G_1 and G_2 are missing, and the point at R_2 is taken to be a false alarm.

4.3.2 Phoneme Transcription

The segmented sentences consist of sign and movement epenthesis segments and the latter are discarded by inspection. Only the remaining sign segments are used for phoneme transcription. The segments obtained can have different lengths, lo-

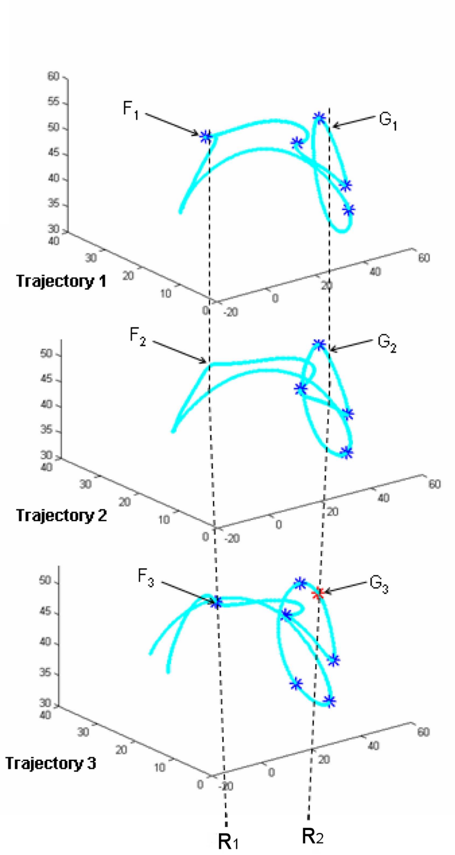


Figure 4.5: Three sample trajectories from the same sentence to illustrate majority voting process.

cations, orientations and directions of motion in the 3-D signing space. The segment boundary points may also be noisy due to slight deviations in segment boundaries from the segmentation algorithm. Co-articulation and movement epentheses in naturally signed continuous sentences also contributes to this. For example, Figure 4.6 shows a segment which is essentially a straight line, but has a small extraneous part that arises from co-articulation and movement epenthesis.

4.3.2.1 Descriptors for Trajectory Segments

The variations in the segments of naturally signed sentences make direct clustering difficult. Hence, we suggest a better representation to enable the use of simple clustering algorithms. For this, we characterize a trajectory segment by

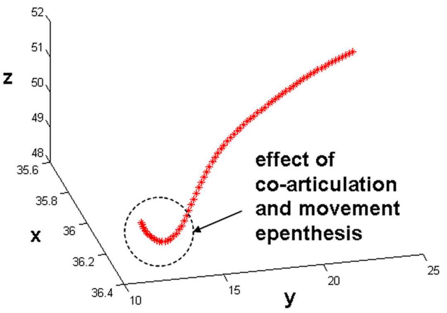


Figure 4.6: Straight line segment with a small portion arising from co-articulation and movement epenthesis.

the plane in which it lies, its shape, direction of motion, size and position. Curves are described by all the above features, while lines are described only by their direction, size and position. PCA of the position vectors can easily differentiate lines (1-D) from curves (2-D) based on eigenvalues. For a line, the first eigenvalue (when ordered from largest to smallest) greatly exceeds the second, and we use this fact to easily separate lines and curves. Based on normalized eigenvalues

$$E_i = \frac{\lambda_i}{\sum_{j=1}^{3} \lambda_j}, \quad i = 1, 2, 3, \tag{4.10}$$

a segment is determined to be a line if $E_1 > 0.95$, and a 2-D curve, otherwise. Following this determination, a set of features is extracted as described below.

1) *Plane of the Trajectory Segment*: The normal to the plane in which the curve lies in 3-D space can be obtained by the vector cross product

$$\mathbf{n}_i = \mathbf{e}_1^i \times \mathbf{e}_2^i, \tag{4.11}$$

where $\mathbf{n}_i$ is the normal to the plane, and $(\mathbf{e}_1^i, \mathbf{e}_2^i)$ are the first and second principal components (PCs) of the i^{th} segment. As there are two possible directions for $\mathbf{n}_i$ in 3-D, we adopt a fixed convention to choose its direction. Also, since two combinations of $\pm\mathbf{e}_1^i$ and $\pm\mathbf{e}_2^i$ correspond to the normal direction chosen, we use one of the pairs as our first and second PCs.

2) *Direction of Motion*: We use dominant motion direction to describe direction for lines, and clockwise/anticlockwise sense for circles. As for arcs, both are used.

Dominant Direction. Though the direction of a line can be simply computed from the starting position and the ending position of the trajectory segment, to reduce sensitivity to noise, the dominant direction is obtained based on the first PC, $\mathbf{e}_1^i$ which is along the direction of the largest variance in the data. As both $\mathbf{e}_1^i$ and $-\mathbf{e}_1^i$ can be considered to be valid directions of maximum variance, we resolve this ambiguity as follows:

i) Compute a unit vector from the starting point to the ending point of the segment as

$$\mathbf{w}_i = \frac{\mathbf{p}_n^i - \mathbf{p}_1^i}{\|\mathbf{p}_n^i - \mathbf{p}_1^i\|}, \tag{4.12}$$

where $\mathbf{p}_1^i$ and $\mathbf{p}_n^i$ are the starting and ending points of the i^{th} segment, respectively.

ii) Compute

$$\theta_1 = \cos^{-1}(\mathbf{w}_i \cdot \mathbf{e}_1^i), \tag{4.13}$$

$$\theta_2 = \cos^{-1}(\mathbf{w}_i \cdot -\mathbf{e}_1^i). \tag{4.14}$$

The dominant direction is chosen to point in the PC direction that is closer to $\mathbf{w}_i$ by choosing $\mathbf{e}_1^i$ if $\theta_1 < \theta_2$, and $-\mathbf{e}_1^i$, otherwise.

Clockwise and Anticlockwise Motion. We project the curve onto the plane defined by (4.11) to determine whether the motion is clockwise or anticlockwise as follows:

i) The first turning point, $\mathbf{q}$, of the curve is located, for example, as in Figure 4.7(a) or 4.7(b). The curve is then rotated so that $\mathbf{q}$ lies on the positive horizontal axis. The corresponding rotated trajectories are as shown in Figures 4.7(c) and 4.7(d), respectively.

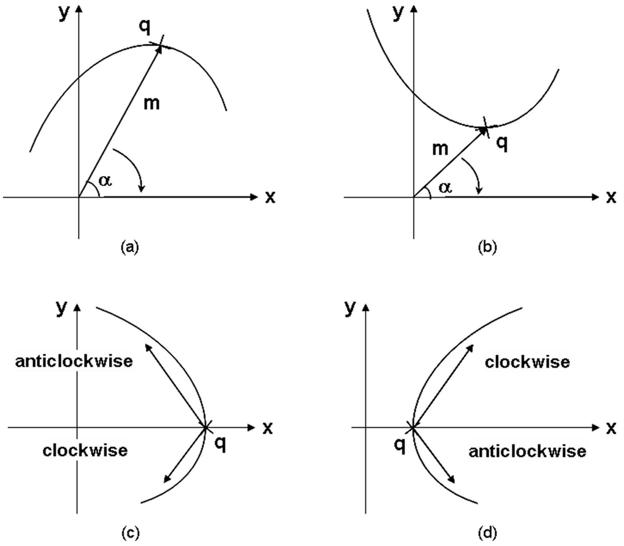


Figure 4.7: (a), (b) Projected trajectories and (c), (d) corresponding rotated trajectories.

ii) Clockwise and anticlockwise motion sense can then be found by the following rule:

$$\text{motion} = \begin{cases} \textit{clockwise} & \text{if } (x \uparrow, y \uparrow) \text{ or } (x \downarrow, y \downarrow) \text{ as } t \uparrow \\ \textit{anticlockwise} & \text{if } (x \uparrow, y \downarrow) \text{ or } (x \downarrow, y \uparrow) \text{ as } t \uparrow \end{cases}. \tag{4.15}$$

3) *Shape*: Both arcs and circles are initially classified as curves, but need to be distinguished based on shape of the segments in the 2-D principal subspace. This is done with Fourier descriptors, which are extracted following the steps below.

i) The trajectory segment is resampled to a fixed number of samples, N, equally spaced in arc length. N is chosen to be a power of 2 to facilitate the application of the fast Fourier transform (FFT). We used $N = 64$.

ii) The projected 2-D curve coordinates are used to define a complex signal

$$z_t = x_t + iy_t, \quad t = 0, 1, ..., N-1, \tag{4.16}$$

where x and y are the x- and y-coordinates in the projected plane.

iii) The motion direction of the projected trajectory segment (clockwise or anticlockwise) affects the ordering of the Fourier descriptors. To remove this sensitivity, we re-ordered the projected segment from the last sample to the first, if its motion sense was found to be anticlockwise.

iv) The DFT of $\mathbf{z} = [z_0, z_1, ..., z_{N-1}]$ is obtained as $\hat{\mathbf{F}} = [\hat{f}_0, \hat{f}_1, ..., \hat{f}_{N-1}]$.

v) Invariance to translation is obtained by removing the first element (DC component) in $\hat{\mathbf{F}}$. Rotation invariance is achieved by removing the phase information, i.e. using only the absolute values of $\hat{f}_k$. Scale normalization is obtained by dividing the Fourier coefficients by $|\hat{f}_1|$. The final Fourier descriptors are given as

$$\tilde{\mathbf{F}} = \left[\frac{|\hat{f}_2|}{|\hat{f}_1|}, \frac{|\hat{f}_3|}{|\hat{f}_1|}, ..., \frac{|\hat{f}_{N-1}|}{|\hat{f}_1|} \right]. \tag{4.17}$$

For discriminating only between circles and arcs, the first and last n elements in $\tilde{\mathbf{F}}$ were used, and $n = 5$ was found to be sufficient.

4) *Size and Position*: The maximum range in each of the x-, y-, z-coordinates is found, and the largest range is taken to represent the size. Position is described by using only the starting and ending positions of the segments. As these can be noisy, we represent the start and end positions of the segment by the mean values of the first and last 5% of the segment points.

4.3.2.2 Transcribing Phonemes with k-means

There are two alternatives for defining phonemes by clustering. We can either concatenate the extracted features and cluster these vectors or cluster each feature separately. We adopted the latter approach as it is simpler and allows simple geometric interpretation of the clusters. Figure 4.8 shows the transcription procedure. The 3-D trajectory segments are first segregated into lines or curves based on the principal eigenvalue found by PCA of each segment. The features used for lines are dominant direction, size and position. All the features described in Section 4.3.2.1 are used for arcs and circles, with the exception of dominant direction for circle. The individual features are clustered by k-means. Table 4.4 summarizes possible clusters for each feature and this serves as a guideline to determine the number of clusters for each feature. The actual number of clusters is found empirically.

The phonemes are then defined by grouping the trajectory segments which have the same geometric feature descriptions. For example, all the trajectory segments which are identified as lines with Dominant Direction = “down”, Size = “small”, and Position = “mouth”, are considered as a cluster (phoneme).

Table 4.4: Possible clusters for the descriptors.

Descriptors	Clusters
Plane	xy-,yz-,xz-,±45°-planes
Shape	circles, arcs
Dominant Direction	up,down,left,right,away,toward
Motion Sense	clockwise,anticlockwise
Size	large,small
Position	12 positions (refer to [16])

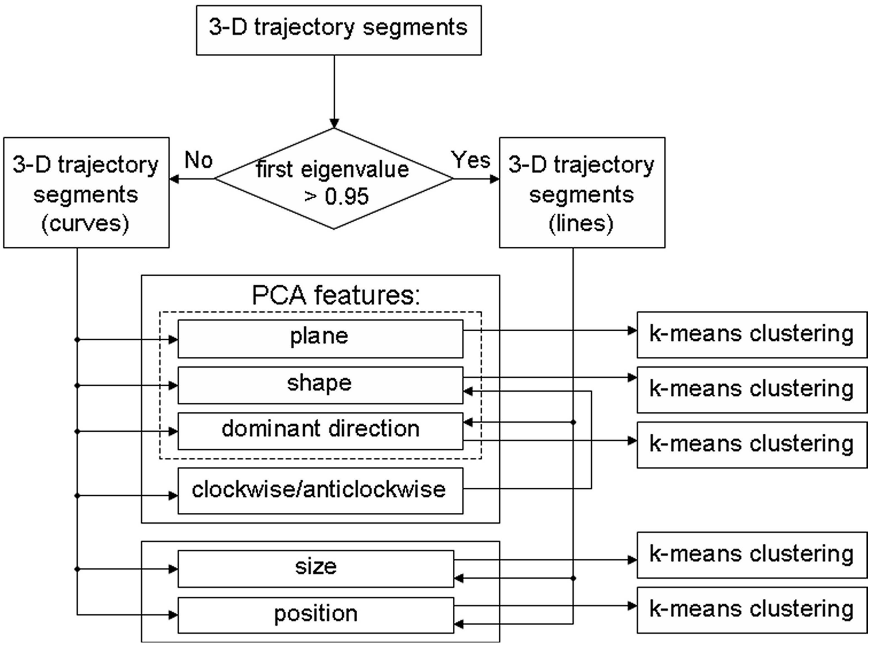


Figure 4.8: Phoneme transcription procedure for the hand movement component.

4.4 Phoneme Transcription for Handshape, Palm Orientation and Location

Phonemes for these static components are obtained by clustering. As the glove and trackers are synchronized, the four components are expected to be aligned, and we assume that the segments in the static components coincide with the segments in the movement component obtained by the automatic segmentation algorithm. The raw data described in Section 7.2 was used, i.e. 16-D handshape, 9-D palm orientation and 3-D location data. We normalized the handshape features to unit length to discount variations in hand size. The phonemes of the three components are defined independently based on the segments in the individual channels but the same procedure described below was applied to them.

We divided each segment into M intervals with equal arc length, and used only the means of the first and last intervals to form the feature vectors for clustering. We also tried dividing a segment into M equal time intervals, i.e. equal number of data points in each interval. However, we found that this approach was affected by the signing habit and speed variations of different signers as some signers tended to remain longer at the start and end of a hand movement trajectory. Hence, we adopted the "equal arc length" approach which is less sensitive to these variations.

The means of the starting and ending intervals of a segment were concatenated to form the feature vectors, which were clustered by k-means to define phonemes. To decide the optimum number of clusters ($\hat{k}$) for each component, we can try using guesses for $\hat{k}$ using guidelines given by linguists. However, this may not correspond to the true distribution of the data. Also, another problem with k-means clustering is that random initialization can cause slow convergence and difficulty in finding a good solution on a large data set. Hence, we used the affinity propagation algorithm to estimate $\hat{k}$ and provide a good initial starting partition for the k-means algorithm.

4.4.1 Affinity Propagation

Affinity propagation (AP) proposed by Frey and Dueck [51] is a message-passing clustering algorithm in which all data points are considered as potential exemplars, and form the nodes of a network. Real-valued messages are transmitted recursively along edges of the network until a good set of exemplars and corresponding clusters emerges. The input to AP consists of a collection of real-valued similarities between data points. When the objective is to minimize squared error, the similarity between each pair of points is set to the negative squared Euclidean distance between them. The similarity $\tilde{s}(i, k)$ indicates how well the data point with index k is suited to be the exemplar for data point i. Each data point k is given a "preference" value $\tilde{s}(k, k)$ and data points with larger values of $\tilde{s}(k, k)$ are more likely to be chosen as exemplars. "Preference" values influence the number of identified exemplars (number of clusters). A common value is used if all the data points are equally suitable as exemplars. Median of the input similarities results in a moderate number of clusters and minimum of the similarities leads to a small number of clusters.

There are two kinds of messages exchanged between data points, viz. "responsibility" and "availability". The "responsibility" $r(i,k)$, sent from point i to candidate exemplar k, reflects the accumulated evidence for how well-suited point k is to serve as the exemplar for point i, taking into account other potential exemplars for point i. The availability $a(i,k)$, sent from candidate exemplar point k to point i, reflects the accumulated evidence for how appropriate it would be for point i to choose point k as its exemplar, taking into account the support from other points that point k should be an exemplar. The AP algorithm is summarized in Table 4.5.

Table 4.5: Affinity propagation algorithm.

AP algorithm:

1) Initialize the "availabilities" $a(i,k) = 0$.

2) Update the "responsibilities" using rule:

$$r(i,k) \longleftarrow \tilde{s}(i,k) - \max_{k' s.t. k' \neq k} a(i,k') + \tilde{s}(i,k').$$

3) Update the "availabilities" using rule:

$$a(i,k) \longleftarrow min\{0, r(k,k) + \sum_{i' s.t. i' \neq i,k} max\{0, r(i',k)\}\}.$$

The "self-availability" $a(k,k)$ is updated differently:

$$a(k,k) \longleftarrow \sum_{i' s.t. i' \neq k} max\{0, r(i',k)\}.$$

4) Terminate the message-passing procedure after the maximum number of iterations is met or after changes in the messages fall below a threshold or after the local decisions stay constant for some number of iterations.

4.4.2 Transcription Procedure for the Static Components

The same phoneme transcription procedure is used for the handshape, palm orientation, and location as the three static components exhibit similar characteristics. The primary limitation of affinity propagation is the requirement of a large memory space. The method requires four $N_p \times N_p$ matrices, where N_p is the number of data points to be clustered. In our problem, the total number of sign segments obtained was 10852. Though only training segments are involved in the clustering, the number is still large. Hence, we chose to run the AP algorithm several times with a subset of segments which were randomly selected from the training pool by keeping all other parameters of AP fixed. The final exemplars were used as the initial conditions for k-means. The set of segments that provided minimum k-means clustering error was selected. The parameter "preference" in AP affects the number of clusters obtained. We will describe the parameter selection in Chapter 7 where the experimental results are presented and discussed. The phoneme transcription procedure is summarized below:

i) Pick N_p sign segments randomly from the entire training data set.

ii) Divide each segment into M intervals of equal arc length and compute the mean vectors of the samples within the starting and ending intervals. These are concatenated together to form the feature vector for clustering. For example, the 16-D starting and ending handshape mean vectors are concatenated to form a 32-D feature vector for the handshape component.

iii) Compute the similarity measure based on Euclidean distance and run the AP algorithm with the N_p data points.

iv) Use the "exemplars" and $\hat{k}$ found by the AP algorithm as the initial centroids and number of clusters, respectively, for k-means clustering.

v) Run k-means with all the training samples using the initialization parameters obtained from the AP algorithm. The final centroids obtained are used as the templates for the phonemes. A phoneme label j is given to a sample if it is found closest to the j^{th} cluster.

4.5 Summary

We devised an automatic procedure to temporally segment naturally signed ASL hand trajectories. Two segmentation algorithms based on rules and naïve Bayesian network classifiers were proposed for obtaining true segmentation points and eliminating false alarms. Experimental results presented in Chapter 7 show that the Bayesian network segmentation performed better and it is thus chosen to segment the hand movement trajectories in our automatic transcription procedure. This transcription scheme relied on effective feature representation. PCA was used to simplify the problem significantly by projecting 3-D hand trajectory segments to 1-D (lines) or 2-D (curves). High level features which described the geometry of the segments were extracted in the projected space. The same segmentation points obtained for the movement component were used for the static components (handshape, orientation and location). The clustering procedure described in Section 4.4 was used to define the phonemes for the static components. Experimental results for phoneme transcription are described in detail in Chapter 7 along with other results.

Silence is as deep
as eternity; speech,
shallow as time.
Thomas Carlyle
(1795-1881)

5

Segment-Based Classification of Sign and Movement Epenthesis

5.1 Overview of Approach

Continuously signed sentences include sign information as well as movement epentheses, and these need to be distinguished from each other for sign recognition. Explicit modeling of movement epentheses may not be the best approach for this purpose due to two important reasons. Firstly, there is limited study by linguists on movement epentheses and hence appropriate linguistic models are lacking. Secondly, the idea of modeling “unwanted” segments is moot, especially as large variations due to different signers can be expected. Hence, we propose an approach which uses only signs to train the recognition framework and deals with movement epentheses during the decoding process.

Some works, for example, [77, 170, 173, 174] also adopted a similar view. However, [173, 174] only used a single channel for processing and recognition, making it vulnerable to signer variations, and limiting generalization to new signers. Indeed, in their experiments with three signers, inconsistency was observed in the recognition results for a new signer. They reported recognition results of 80%, slightly more than 50% and less than 30% for three rounds of leave-one-out experiments with 10 sentences. The limited generalization could also be due to the generative modeling they used for signs. Furthermore, their sign based modeling approach may not be scalable to large vocabulary compared to a phoneme-based approach. Both of the other works [77, 170] were based on threshold models trained with only one signer, and the threshold parameters were required to be derived from the training data. However, finding good threshold values may be difficult when the problem is extended to several signers and the recognition framework may not perform robustly with new signers.

If a recognition framework for continuously signed sentences is trained only with sign information, then the sign segments must be obtained as accurately as possible for input to the decoder during recognition. This implies the need for a segmentation algorithm and an accurate classifier to label the segments as belonging to sign (*SIGN*) or movement epenthesis (*ME*). In this chapter, we focus on these two aspects.

Our classifier is based on CRFs and SVMs, and the required background on these models is given for completeness in Sections 5.2 and 5.3, respectively. In Section 5.4 we describe the rationale for the segmentation algorithm chosen. Section 5.5 describes the original frame-based features obtained in the four component, and the higher level features extracted for classification. Section 5.6

describes the classification algorithm for sign and movement epenthesis sub-segments. Here, the sub-segments are classified independently by a CRF model and a SVM, and their outputs are fused with a Bayesian network for improved accuracy. The chapter is summarized in Section 5.7.

5.2 Conditional Random Fields

A conditional random field (CRF) is a discriminative probabilistic model whose underlying conditional structure allows relaxing the strong independence assumptions between the observed variables which are made in HMMs, and thereby simplifies the problem. CRFs also avoid the label bias problem [91] which occurs in maximum entropy Markov models (MEMMs) [104] and other conditional Markov models based on directed graphical models. CRFs have shown success in many works including parts-of-speech (POS) tagging [91], shallow parsing [130], name entity recognition [80], morphological analysis [89], gene prediction [27], and speech recognition [105], to name a few. We provide a brief introduction and discussion of CRFs in the next section based on [18, 81, 161]. Excellent comprehensive tutorials of CRFs can be found in [81, 141, 161].

Probabilistic graphical models are schematic representations of probability distributions [18]. A graph is made up of nodes representing random variables, which are connected by edges denoting the relationships between the variables. Conditional independence allows complex probability distributions to be factorized into a product of factors. This reduces the complex learning or inferencing computations significantly. Based on the definition of conditional independence, the absence of an edge between two random variables implies that the random variables are conditionally independent given all other random variables in the model. Notationally, conditional independence is denoted as $p(a|b, c) = p(a|c)$, where a and b are independent given a random variable c. The conditional independence properties can be represented by the directed graph shown in Figure 5.1. Generally, graphical models can be represented as directed or undirected graphs. Naïve Bayes and HMMs are two common examples of directed graphical models. Examples of undirected graphical models are maximum entropy models and CRFs. More detail on graphical models can be found in [18]. Here, we provide the background for training an undirected graphical model, viz., the linear-chain CRF, and the associated inferencing technique.

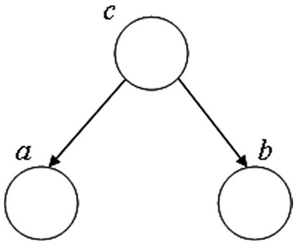


Figure 5.1: Graph to represent conditional independence properties.

5.2.1 Linear-Chain CRFs

A conditional random field is an undirected graphical model, globally conditioned on $\mathbf{X}$, the random vector representing observation sequences. Formally, the model allows computing the probability $p(\mathbf{y}|\mathbf{x})$, where $\mathbf{y} = (y_1, \ldots\ldots, y_n) \in \mathbf{Y}$ are the possible outputs and $\mathbf{x} = (x_1, \ldots\ldots, x_n) \in \mathbf{X}$ are the input observations. A linear-chain CRF whose graphical structure is illustrated in Figure 5.2 has a specialized linear structure, modeling the output variables as a sequence. In linear-chain CRFs, the joint distribution of $\mathbf{y}$ is factorized into real-valued potential functions. Each potential function operates on pairs of adjacent label variables y_j and y_{j+1}. The conditional probability of $\mathbf{y}$, in a linear-chain CRF can be

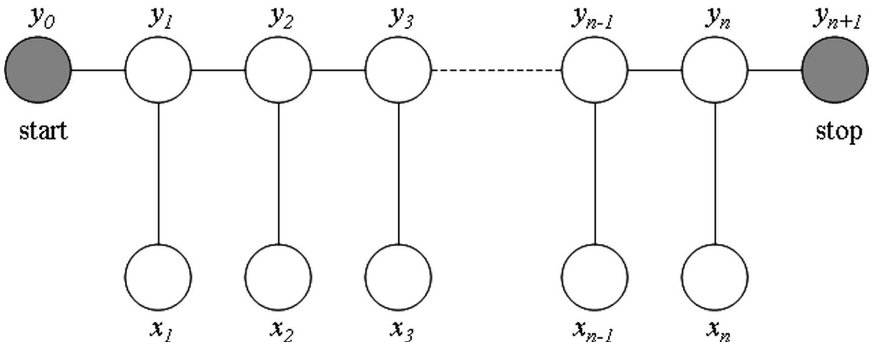


Figure 5.2: Graphical model of a linear-chain CRF.

written as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{\mathbf{Z}(\mathbf{x})}\exp\Bigg(\sum_{j=1}^{n}\sum_{i=1}^{m}\lambda_i f_i(y_{j-1}, y_j, \mathbf{x}, j)\Bigg), \tag{5.1}$$

where $\mathbf{Z}(\mathbf{x})$ is a normalization term in the range of $[0,1]$, given as

$$\mathbf{Z}(\mathbf{x}) = \sum_{\mathbf{y}\in\mathbf{Y}}\exp\Bigg(\sum_{j=1}^{n}\sum_{i=1}^{m}\lambda_i f_i(y_{j-1}, y_j, \mathbf{x}, j)\Bigg). \tag{5.2}$$

The index j specifies the position in the input sequence $\mathbf{x}$ and λ_i are weight parameters to be estimated from training data. Each $f_i(y_{j-1}, y_j, \mathbf{x}, j)$ is either a state function $f_i^s(y_j, \mathbf{x}, j)$ or a transition function $f_i^t(y_{j-1}, y_j, \mathbf{x}, j)$, so that we can write

$$\sum_{i=1}^{m}\lambda_i f_i(y_{j-1}, y_j, \mathbf{x}, j) = \sum_{i=1}^{t}\nu_i f_i^s(y_j, \mathbf{x}, j) + \sum_{i=t+1}^{m}\mu_i f_i^t(y_{j-1}, y_j, \mathbf{x}, j), \tag{5.3}$$

where the state and transition functions are weighted by the parameters ν_i and μ_i, respectively. This formulation is used in our framework that uses linear-chain CRFs. For simplicity in notation, the training and inferencing procedures are explained with respect to (5.1).

5.2.2 Parameter Estimation

In linear-chain CRFs, the parameters $\theta = \{\lambda_1, \lambda_2, \ldots, \lambda_m\}$ are estimated from the training data $\mathcal{D} = \{\mathbf{x}^k, \mathbf{y}^k\}_{k=1}^{N}$ where $\mathbf{x}^k = \{x_1^k, x_2^k, \ldots, x_n^k\}$ is the k^{th} input sequence and $\mathbf{y}^k = \{y_1^k, y_2^k, \ldots, y_n^k\}$ is the k^{th} predicted output sequence. Parameter estimation for CRFs is typically performed by using maximum-likelihood where the conditional log-likelihood, $\mathcal{L}$, is maximized based on the training data $\mathcal{D}$. The conditional log-likelihood is given as

$$\mathcal{L}(\theta) = \sum_{k=1}^{N} \log p(\mathbf{y}^k | \mathbf{x}^k). \tag{5.4}$$

By substituting (5.1) into (5.4), we get,

$$\begin{aligned}\mathcal{L}(\theta) =& \sum_{k=1}^{N} \log \left[\frac{\exp(\sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i f_i(y_{j-1}^k, y_j^k, \mathbf{x}^k, j))}{\sum_{\hat{\mathbf{y}} \in \mathbf{Y}} \exp(\sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i f_i(\hat{y}_{j-1}^k, \hat{y}_j^k, \mathbf{x}^k, j))} \right] \\ =& \sum_{k=1}^{N} \sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i f_i(y_{j-1}^k, y_j^k, \mathbf{x}^k, j) - \sum_{k=1}^{N} \log \mathbf{Z}(\mathbf{x}^k).\end{aligned} \tag{5.5}$$

Regularization is often applied to avoid overfitting. There are two common choices of penalty, viz. L1-norm (5.6) proposed by Goodman [56] and the L2-norm (5.7). C is a free parameter that determines the weight of the penalty term and is introduced to allow tuning for best performance. L1-norm is used to encourage sparsity in the trained parameters, but the maximization lacks a closed form solution, and thus numerical optimization is required.

$$\mathcal{L}(\theta) = \sum_{k=1}^{N} \sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i f_i(y_{j-1}^k, y_j^k, \mathbf{x}^k, j) - \sum_{k=1}^{N} \log \mathbf{Z}(\mathbf{x}^k) - C \sum_{i=1}^{m} \frac{|\lambda_i|}{2}, \tag{5.6}$$

$$\mathcal{L}(\theta) = \sum_{k=1}^{N} \sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i f_i(y_{j-1}^k, y_j^k, \mathbf{x}^k, j) - \sum_{k=1}^{N} \log \mathbf{Z}(\mathbf{x}^k) - C \sum_{i=1}^{m} \frac{|\lambda_i|^2}{2}. \tag{5.7}$$

For the optimization of (5.7), the derivative of the first term yields the expected value of a feature f_i under the empirical distribution,

$$\tilde{E}(f_i) = \sum_{k=1}^{N} \sum_{j=1}^{n} f_i(y_{j-1}^k, y_j^k, \mathbf{x}^k, j). \tag{5.8}$$

The derivative of the second term gives the expectation of f_i with respect to the model distribution,

$$E(f_i) = \sum_{k=1}^{N} \sum_{\hat{\mathbf{y}} \in \mathbf{Y}} p(\hat{\mathbf{y}}^k | \mathbf{x}^k) \sum_{j=1}^{n} f_i(\hat{y}_{j-1}^k, \hat{y}_j^k, \mathbf{x}^k, j). \tag{5.9}$$

$\tilde{E}(f_i)$ is computed by counting the frequency of each feature f_i which occurs in the training data. It is impractical to compute $E(f_i)$ directly as there is a combinatorial explosion of output sequence labels in evaluating the summation. Hence, a forward-backward algorithm as used in HMMs with slight modification is used to compute $E(f_i)$.

5.2.3 Inference

Inferencing in CRFs is formulated as finding the most likely output label sequence $\mathbf{y}^*$ given the observations $\mathbf{x}$. The recursive Viterbi algorithm is applied to efficiently find the most probable path as

$$\mathbf{y}^* = \underset{\mathbf{y}}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{x}). \tag{5.10}$$

The Viterbi recursion is given as

$$\delta_j(q) = \max_{\hat{\mathbf{q}} \in Q} \delta_{j-1}(\hat{q}) \Psi_j(\hat{q}, q, \mathbf{x}), \tag{5.11}$$

where

$$\Psi_j(\hat{q}, q, \mathbf{x}) = \exp\left(\sum_{i=1}^{m} \lambda_i f_i(y_{j-1} = \hat{q}, y_j = q, \mathbf{x}, j) \right). \tag{5.12}$$

The essence of the Viterbi algorithm lies in the recursive nature of the term $\delta_j(q)$ which stores the highest score along any path through point j which ends in state q. Table 5.1 summarizes the Viterbi algorithm for output path decoding in CRFs. $\Omega_j(q)$ is an array introduced to keep track of the most probable path to state q

at position j.

Table 5.1: Viterbi algorithm.

```
Viterbi algorithm:

1) Initialize $\Omega_1(q)$ to zeros and $\delta_1(q)$ to the corresponding start
   state probability values.
2) Perform recursion steps on $\delta_j(q)$ and update the $\Omega_j(q)$ as
   follows,
       $\delta_j(q) = \max_{\hat{q} \in Q} \delta_{j-1}(\hat{q}) \Psi_j(\hat{q}, q, \mathbf{x}), \quad \forall\, q \in Q,\ 1 \leq j \leq n,$
       $\Omega_j(q) = \operatorname{argmax}_{\hat{q} \in Q} \delta_{j-1}(\hat{q}) \Psi_j(\hat{q}, q, \mathbf{x}).$
3) At the end of the recursive steps, keep the end state with
   highest probability:
       $p^* = \max_{\hat{q} \in Q} \delta_n(\hat{q}),$
       $y_n^* = \operatorname{argmax}_{\hat{q} \in Q} \delta_n(\hat{q}).$
4) Perform backtracking as follows,
       $y_t^* = \Omega_{t+1}(y_{t+1}^*).$
```

5.3 Support Vector Machines

Support vector machines (SVMs) are motivated with a view to train linear machines with large margins. The underlying concept in SVMs is to use an appropriate nonlinear mapping to project data to high dimension so that two-class data can be separated by a hyperplane with minimal error. The commonly used SVMs were first proposed by Cortes and Vapnik [24] and are formulated as follows.

Given instances $\check{\mathbf{x}}_i, i = 1, \ldots, \mathcal{M}$ with labels $\check{y}_i \in \{1, -1\}$, the training of SVMs begins by choosing the nonlinear mapping functions that map the input to a higher dimension. The choice of the mapping function is often problem dependent and examples of common kernels are linear, sigmoid, polynomial and Gaussian radial basis functions. In the training of SVMs, the goal is to minimize the weight vectors $\mathbf{a}$,

$$\min_{\mathbf{a},b,\xi} \quad \frac{1}{2}\mathbf{a}^T\mathbf{a} + \tilde{C}\sum_{i=1}^{\mathcal{M}} \xi_i \tag{5.13}$$

$$\text{subject to} \quad \check{y}_i(\mathbf{a}^T\varphi(\check{\mathbf{x}}_i) + b) \geq 1 - \xi_i, \tag{5.14}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, \mathcal{M}, \tag{5.15}$$

where $\varphi(\check{\mathbf{x}}_i)$ maps $\check{\mathbf{x}}_i$ into a higher dimensional space and $\tilde{C} > 0$ is a scalar constant for regularization. Given the training data, this problem is often solved using quadratic programming though other schemes have also been devised.

In standard SVMs, the output is a distance measure between a test pattern and the decision boundary. When SVMs are used with other probabilistic models such as HMMs or CRFs, it is important for the SVM output to represent posterior class probability. Platt [120] modeled the class conditional densities $p(f(\check{\mathbf{x}})|\check{y} = +1)$ and $p(f(\check{\mathbf{x}})|\check{y} = -1)$ using Gaussians of equal variance and computed the posterior probability of a class given the SVM output as

$$p(\check{y} = +1|f(\check{\mathbf{x}})) = \frac{1}{1 + \exp(Af(\check{\mathbf{x}}) + B)}, \tag{5.16}$$

where $f(\check{\mathbf{x}})$ is the SVM output, $\check{\mathbf{x}}$ is the input pattern, $\check{y}$ is the class variable, and the parameters A, B are found by maximum likelihood estimation from the training set (see [120]).

5.4 Segmentation

Ideally, if test sentences could be perfectly segmented and labeled as *SIGN* or *ME* segments, the sign segments could be decoded to recognize the signs. However, perfect segmentation is difficult. Experimental results for the Bayesian network and rule-based segmentation algorithms of Chapter 4 on the movement channel showed a best accuracy of about 95% for boundary point detection. Though our experiments indicated that this level of accuracy is tolerable for automatic phoneme transcription, the decoding performance may degrade. On the other hand, the initial segmentation algorithm of Section 4.3.1.1, which marks boundary points at locations of minimum hand velocity and maximal change of directional angle, finds 99.9% of the true boundary points, but yields a high false alarm rate, i.e. the sequences are over-segmented. Hence, we tried other methods to maintain this high accuracy while reducing the false alarm rate, such as merging the sub-segments obtained from the initial segmentation algorithm by using penalized likelihood estimation, using criteria such as minimum description length (MDL) or Bayesian information criterion (BIC), to trade off model accuracy and complexity. However, the performance did not improve.

Hence, our strategy is to use only the initial segmentation algorithm in view of its high detection rate, and deal with the over-segmentation problem in a different way. That is, with the high detection rate, we anticipate that if the sub-segments can be correctly labeled as sign or movement epenthesis and merged properly, then the sign sequence in a sentence can be decoded with high accuracy. This is the crucial starting point for our proposed idea which adopts a segment-based approach to recognize the continuously signed sentences, rather than the usual frame-based sequences (the term “sequence” is used in the rest of this thesis to refer to sequential data in a sentence). Here, we aim to classify as many movement epenthesis sub-segments as possible without mislabeling any sign sub-segment in the sequence. The locations of the detected movement epenthesis sub-segments provide useful information for the final decoding algorithm, and thereby simplify the sign recognition problem significantly.

The initial segmentation algorithm is applied to the movement channel, and the boundary points obtained are also applied to the other three channels, as we assume that information in all channels evolves synchronously. Next we propose an appropriate representation for the sub-segments in the movement channel and extract appropriate features from it for classifying the sub-segments into *SIGN* and *ME*.

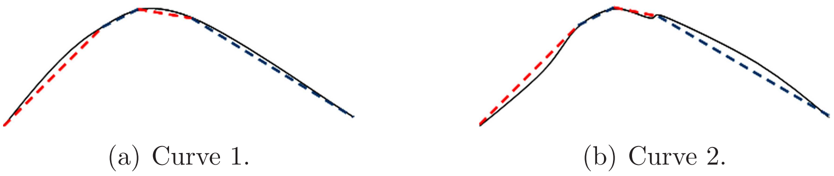
5.5 Representation and Feature Extraction

The raw data obtained from the glove and trackers are described in Section 7.2, and consist of handshape (16-D vectors), palm orientation (9-D vectors) and position (3-D vectors), obtained at frame rates. These raw vectors need to be appropriately normalized prior to feature extraction. The 16-D handshape vectors are normalized to unit length to discount hand size variations. The raw position vectors yield data for the movement channel and location channel. Usually a few vectors at the beginning and end of a segment are used to obtain location channel information.

For the movement channel, the basic descriptors are the movement direction and trajectory shape, rather than raw position vectors. These descriptors need to be invariant to location and size of trajectory, and hence care is required when obtaining them from the raw position data. For example, the position vectors for a circular hand movement made in the chest area, will be different for the same movement made in the head area, and hence, normalization is necessary to simply the recognition step. However, in continuous signing, direct normalization based on the entire signed sentence is not appropriate as there may be variations from sign to sign. Normalization based on segments is also not straightforward as the start and end points of a movement segment are unknown. Hence, we adopt a representation described below for the sub-segment trajectories in the movement channel, that can reduce these sensitivities.

5.5.1 Representation

A possible approach to represent the direction and trajectory shape is to normalize the difference between the current previous position vectors to a unit vector and represent the trajectory by a sequence of unit vectors, as for example in [79]. However, this process can be sensitive to noise and other variations that may be present from sequence to sequence. Hence, for robustness, we fit lines to the trajectory in each sub-segment using the end-point fitting algorithm [41]. Figure 5.3 shows two samples of the movement trajectory of a sign that exhibits obvious differences. By line fitting, some of the variations can be eliminated to make the piecewise linear representation of the two curves become more consistent. The line fitting procedure is applied to each sub-segment and is described as follows.


(a) Curve 1.

(b) Curve 2.

Figure 5.3: Fitting lines to curves.

i) Fit each sub-segment with lines using the iterative end-point fitting algorithm shown in Table 5.2. Figure 5.4 illustrates an example where a line through the two ends points (A and B) is first fitted, and distances between the points on the curve and the line are computed. The maximum distance is then located (point C). If this maximum distance (D) exceeds a preset threshold, point C is used to find new lines through A and B. The process continues until all maximum distances are smaller than a preset threshold. Figure 5.5 shows an example of the lines fitted to a 3-D hand movement trajectory.

ii) After the line fitting process, the unit directional vectors of the lines computed at every sample point represent the sub-segment trajectory (and their collection over all sub-segments represent the entire trajectory).

Table 5.2: Iterative end-point fitting algorithm.

```
Iterative end-point fitting algorithm:

1) Given a curve, fit an initial line by
   connecting the end points of the curve.
2) Compute the distances from each point on the
   curve to this line, and check
         if all distances < $\varepsilon$, stop,
              where $\varepsilon$ is a tolerance measure,
         else go to step 3.
3) Find the point furthest from the line and break
   the curve into two segments representing two
   new curves. Fit lines to the two new curves
   separately with their end points. Repeat Step 2.
```

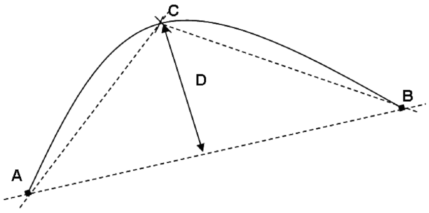


Figure 5.4: End point fitting algorithm.

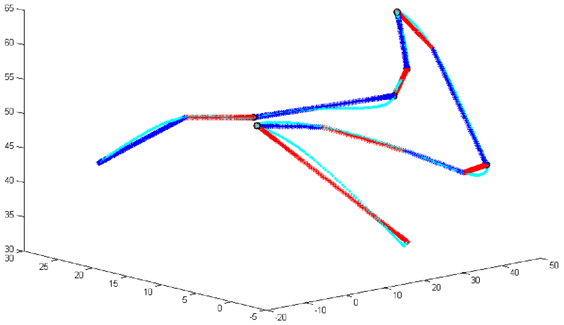


Figure 5.5: 3D hand movement trajectory fitted with lines.

At the end of the processing described above, we have a sequence of frame-based vectors in the four channels, as follows: 1) handshape (normalized for hand size), and 2) movement component represented by a sequence of unit vectors at every frame instant, extracted as described above, 3) no change in orientation vectors, and 4) location channel represented by position vectors (no change). We denote this as the set of "normalized" vectors.

5.5.2 Feature Extraction for Classification

We extract effective features from the sub-segments (consisting of the normalized vectors) and train a classifier to label a sub-segment as sign or movement epenthesis. The inputs to the classifier consist of features extracted from the four parallel handshape (H), movement (M), orientation (O) and location (L) subsequences. Figure 5.6 illustrates an example for the sentence GO MY HOME consisting of 10 sub-segments in each channel. The superscript "ME" denotes movement epenthesis sub-segments while others denote sign sub-segments (sign sub-segments can also be labeled generically as "SIGN"). The goal here is to classify the sub-segments as *SIGN* or *ME*. We define ${}^{l}S_j$ as the set of sub-segments from the four channels, i.e. ${}^{l}S_j = \{{}^{l}H_j, {}^{l}M_j, {}^{l}O_j, {}^{l}L_j\}$, $l = \{SIGN, ME\}$ is the label variable and $j = 1, 2, \ldots, T$, where T is the number of sub-segments in a sequence.

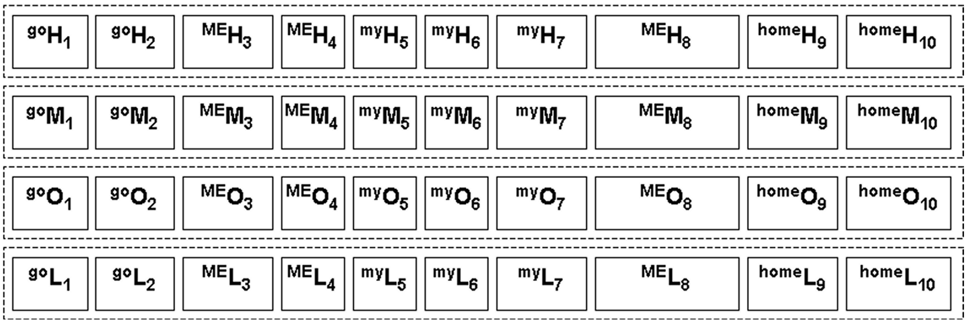

${}^{go}H_1$ ${}^{go}H_2$ ${}^{ME}H_3$ ${}^{ME}H_4$ ${}^{my}H_5$ ${}^{my}H_6$ ${}^{my}H_7$ ${}^{ME}H_8$ ${}^{home}H_9$ ${}^{home}H_{10}$
${}^{go}M_1$ ${}^{go}M_2$ ${}^{ME}M_3$ ${}^{ME}M_4$ ${}^{my}M_5$ ${}^{my}M_6$ ${}^{my}M_7$ ${}^{ME}M_8$ ${}^{home}M_9$ ${}^{home}M_{10}$
${}^{go}O_1$ ${}^{go}O_2$ ${}^{ME}O_3$ ${}^{ME}O_4$ ${}^{my}O_5$ ${}^{my}O_6$ ${}^{my}O_7$ ${}^{ME}O_8$ ${}^{home}O_9$ ${}^{home}O_{10}$
${}^{go}L_1$ ${}^{go}L_2$ ${}^{ME}L_3$ ${}^{ME}L_4$ ${}^{my}L_5$ ${}^{my}L_6$ ${}^{my}L_7$ ${}^{ME}L_8$ ${}^{home}L_9$ ${}^{home}L_{10}$

Figure 5.6: The sub-segment sequences in the four parallel channels.

The features extracted from the component sub-segments are explained below. Tables 5.3 and 5.4 summarize the details of the state and transition features, respectively, for the CRF. Table 5.5 summarizes details of the features extracted for the SVM.

i) **hand_start**, **hand_end**, **orien_start**, **orien_end**, **loc_start**, **loc_end**, **mov_start**, **mov_end**: The "start" and "end" features of each component are computed by taking the mean of the first or last 5% of data from the sub-segments.

ii) **hand_msdif**, **orien_msdif**: Given a c point sub-segment, this feature is the mean of $\mathbf{r}_t - \mathbf{r}_{t-1}$, where $\mathbf{r}$ is the raw feature vector of the component and $t = 2, 3 \ldots, c$.

iii) **loc_mean**: This is computed by taking the mean of the position vectors of the sub-segment.

iv) **mov_dom**: The dominant direction is obtained based on the first eigenvector as described in Chapter 4.

v) **arc_length**: The arc length of the sub-segment is computed as $\sum_{t=2}^{c} \|(\mathbf{p}_t - \mathbf{p}_{t-1})\|$, where $\mathbf{p_t} = (x_t, y_t, z_t)$ is the t^{th} 3-D position vector.

vi) **tri_feature**: The trigram features denote triplets that use not only the feature at the current time instant but also features from previous (or future) two time instants. For example, for a handshape sequence, $\mathbf{H} = (H_1, H_2, H_3, H_4, \ldots, H_T)$, for $j = 3$, "$H_1H_2H_3$" and "$H_3H_4H_5$" form the **tri_feature**. This applies to all the features listed in Table 5.3 except for arc length.

vii) **diff_strhand**, **diff_strorien**: These features are computed by taking the difference of the ending handshape (or palm orientation) of the previous sub-segment and the starting handshape (or palm orientation) of the current sub-segment.

viii) **diff_mloc**, **diff_mdom**: These are computed by taking the difference of the mean hand positions (or dominant directions) of current and previous sub-segments.

ix) **comb_arc**: The current and previous sub-segments are merged and their combined arc length is calculated.

The number of discrete symbols for CRF features was obtained experimentally. We clustered the data points for each feature (e.g. **hand_start** which is 16-D) into $\hat{k}$ clusters using k-means. To determine the best value for $\hat{k}$, we trained a CRF based on the target feature (i.e. only one feature is used such as **hand_start**), and repeated the procedure for different k values. The highest classification accuracy from the CRF for different k's was used to decide the "best" $\hat{k}$ to represent a feature (these values are shown in Tables 5.3 and 5.4). For the SVM, all the individual component features were cascaded to form a 126-D feature vector for input to the classifier.

5.6 Sub-Segment Classification

In practice, perfect labeling of sign and movement epenthesis sub-segments is not possible, but we aim to detect as many movement epenthesis sub-segments as possible with minimal mislabeling of any sign sub-segments in the sequence, as a good starting point for the decoding algorithm. With this view we trained both SVMs and CRFs independently for sub-segment classification as *SIGN* or *ME*, with extracted features from the four component sub-segments (${}^{l}H_j, {}^{l}M_j, {}^{l}O_j, {}^{l}L_j$). The CRF and SVM both use the same set of basic features except that the features for the SVM are raw features while the features for the CRF are discretized into a finite symbol set. The settings and procedures for training the CRF and SVM are described in Section 7.4.

Table 5.3: State features for CRF.

Component	State feature	Description	No. of symbols
Handshape	**hand_start**	Starting handshape of each sub-segment.	70
	hand_end	Ending handshape of each sub-segment.	70
	hand_msdif	Mean of the adjacent handshape differences of each sub-segment.	70
Orientation	**orien_start**	Starting palm orientation of each sub-segment.	50
	orien_end	Ending palm orientation of each sub-segment.	50
	orien_msdif	Mean of the adjacent palm orientation differences of each sub-segment.	80
Location	**loc_mean**	Mean of the hand positions of each sub-segment.	50
	loc_start	Starting hand position of each sub-segment.	50
	loc_end	Ending hand position of each sub-segment.	50
Movement	**mov_dom**	Dominant direction of hand motion of each sub-segment.	60
	mov_start	Starting direction of hand motion of each sub-segment.	60
	mov_end	Ending direction of hand motion of each sub-segment.	60
Others	**arc_length**	Arc length of each sub-segment.	10
	tri_features	Trigram features.	-

We observed that while the CRF and SVM classifiers provided good accuracy, there was scope for improvement by fusing their outputs. For this purpose, we computed the SVM and CRF output probabilities and fused them with other useful features using a Bayesian network.

Table 5.4: Transition features for CRF.

Transition feature	Description	No. of symbols
label	*SIGN* or *ME*.	2
diff_strhand	Difference of the end handshape of previous sub-segment and the start handshape of the current sub-segment.	70
diff_strorien	Difference of the end palm orientation of previous sub-segment and the start palm orientation of the current sub-segment.	80
diff_mloc	Difference of the mean of the hand positions of current and previous sub-segments.	60
diff_mdom	Difference of the mean of the dominant direction of the hand motion of current and previous sub-segments.	70
comb_arc	Combined arc length of current and previous sub-segments.	10

5.6.1 Fusion with Bayesian Network

The fusion of different classifiers aims to yield a more accurate classification decision than any single classifier. We used a Bayesian network to combine the outputs of the CRF and SVM. We defined three query nodes viz. **fLabel**, **svmErr**, and **crfErr**, and seven observed nodes viz. **svmProb**, **svmLab**, **svmPos**, **crfProb**, **crfLab**, **crfPos** and **arcLen**. The structure of the network is shown in Figure 5.7 and was specified using prior knowledge; all the nodes have finite discrete states which are described in Table 5.6. The observed nodes **svmPos** and **crfPos** are defined to have four states which are illustrated by the following example. Given a SVM or a CRF detected sequence $\hat{\mathbf{S}} = \{^{SIGN}\hat{S}_1, {}^{ME}\hat{S}_2, {}^{ME}\hat{S}_3, {}^{SIGN}\hat{S}_4, {}^{SIGN}\hat{S}_5, {}^{SIGN}\hat{S}_6\}$, we identify four cases for the positions of the sub-segments as follows. We group together sub-segments which have

Table 5.5: Features for SVM.

Component	Feature	Description	Length
Handshape	**hand_start**	Starting handshape of each sub-segment.	16-D (each feature)
	hand_end	Ending handshape of each sub-segment.	
	hand_msdif	Mean of the adjacent handshape differences of each sub-segment.	
	diff_strhand	Difference of the end handshape of previous sub-segment and the start handshape of the current sub-segment.	
Orientation	**orien_start**	Starting palm orientation of each sub-segment.	9-D (each feature)
	orien_end	Ending palm orientation of each sub-segment.	
	orien_msdif	Mean of the adjacent palm orientation differences of each sub-segment.	
	diff_strorien	Difference of the end palm orientation of previous sub-segment and the start palm orientation of the current sub-segment.	
Location	**loc_mean**	Mean of the hand positions of each sub-segment.	3-D (each feature)
	loc_start	Starting hand position of each sub-segment.	
	loc_end	Ending hand position of each sub-segment.	
	diff_mloc	Difference of the mean of the hand positions of current and previous sub-segments.	
Movement	**mov_dom**	Dominant direction of hand motion of each sub-segment.	3-D (each feature)
	mov_start	Starting direction of hand motion of each sub-segment.	
	mov_end	Ending direction of hand motion of each sub-segment.	
	diff_mdom	Difference of the mean of the dominant direction of the hand motion of current and previous sub-segments.	
Others	**arc_length**	Arc length of each sub-segment.	1-D (each feature)
	comb_arc	Combined arc length of current and previous sub-segments.	

the same consecutive labels: $\{{}^{SIGN}\hat{S}_1\}$, $\{{}^{ME}\hat{S}_2, {}^{ME}\hat{S}_3\}$, $\{{}^{SIGN}\hat{S}_4, {}^{SIGN}\hat{S}_5, {}^{SIGN}\hat{S}_6\}$. The aim is to use the error pattern based on the positions of the sub-segments within a segment to improve the accuracy. The first group consists of only one sub-segment and ${}^{SIGN}\hat{S}_1$ is labeled as "single position". To distinguish sub-segments at the group edges, ${}^{ME}\hat{S}_2$ and ${}^{SIGN}\hat{S}_4$ are labeled as "left position" while ${}^{ME}\hat{S}_3$ and ${}^{SIGN}\hat{S}_6$ are labeled as "right position". The last case is ${}^{SIGN}\hat{S}_5$ which lies between edges and is labeled as "other position". Given a set of training data $\mathcal{D} = \{\mathbf{z}^1, \mathbf{z}^2, \ldots, \mathbf{z}^N\}$ and network structure as shown in Figure 5.7, the network parameters $\boldsymbol{\theta}$ (conditional probability table (CPT) of the nodes) are estimated by ML estimation, as described in Chapter 4.

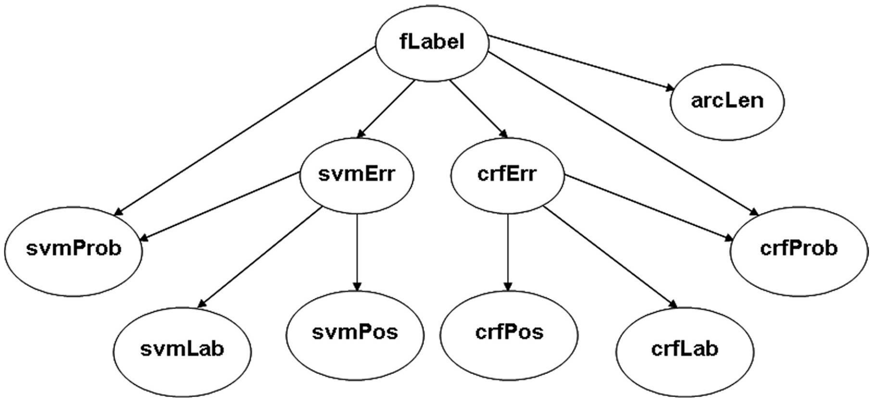


Figure 5.7: Bayesian network for fusing CRF and SVM outputs.

Experiments were conducted based on the classifier described above and the results are given in Chapter 7. However, in order to prevent too much sign information from being lost by misclassification which is important in the final recognition step, we relaxed the probability threshold used in the Bayesian network classifier, i.e. instead of using 0.5 as the threshold for *SIGN* and *ME* sub-segments, we adjusted the threshold to minimize the number of missed *SIGN*s at the expense of having more false alarms (*ME* sub-segments being classified as *SIGN*). Hence, at the end of this subsystem, the sub-segment labels are obtained

Table 5.6: Summary of the Bayesian network.

Node	State	Description
fLabel	*SIGN*, *ME*	Sign or movement epenthesis sub-segment.
svmErr	Yes, No	Detection by SVM is an error or not.
crfErr	Yes, No	Detection by CRF is an error or not.
svmProb	1-10	Quantized SVM output probabilities.
svmLab	*SIGN*, *ME*	Label from SVM.
svmPos	1-4	Position of the sub-segment from SVM classifications.
crfProb	1-10	Quantized CRF output probabilities.
crfLab	*SIGN*, *ME*	Label from CRF.
crfPos	1-4	Position of the sub-segment from CRF classifications.
arcLen	1-10	Quantized arc length.

based on the relaxed threshold for the *SIGN*/*ME* decision, and all sub-segments labeled as *ME* are discarded and the positions are recorded.

5.7 Summary

The hand movement trajectory was segmented by marking the point of minimum velocity and maximum directional angle change (the initial segmentation algorithm of Chapter4). The same segment boundary points were used for all the other channels as well. Appropriate features were extracted from corresponding sub-segments in each channel. In the movement channel, a piecewise linear representation of the sub-segment trajectories was obtained before feature extraction, with a view to reduce noise and sample to sample variations in the curves. SVMs and CRFs were investigated individually to classify the sub-segments as *SIGN* or *ME*. Based upon the performance characteristics, it appeared that accuracy could be improved by fusing the results. This was done by using a Bayesian network. The labeled sub-segments are incorporated into the recognition framework as described in the next chapter.

Kindness is the language
which the deaf can hear
and the blind can see.
Mark Twain (1835-1910)

6 Segmental Sign Language Recognition

6.1 Overview of Approach

To recognize continuously signed sentences that exhibit variations caused by different signers, we propose a discriminative CRF model to yield better generalization compared to generative models. We use a two-layered CRF model as shown in Figure 6.1, where the lower layer performs basic phoneme recognition independently in the four component channels, each with its unique set of subphones and phonemes. The second layer fuses the four component phonemes together to recognize the signs. For training, all the movement epenthesis segments are discarded and only the sign segments are used. During testing, different combinations of sub-segments (as obtained in Chapter 5) are merged and evaluated for recognition. For this, we modify the decoding algorithm of the semi-Markov

CRFs proposed by Sarawagi and Cohen [128] and devise an efficient approach to decode the sign sequences in sentences.

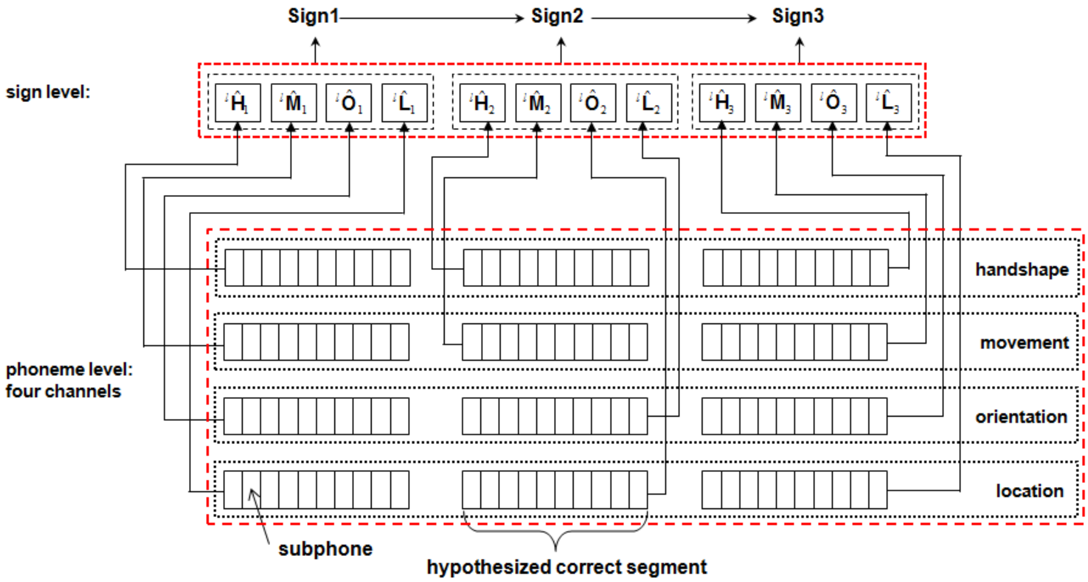


Figure 6.1: Overall recognition framework.

A semi-Markov CRF is a sequence model which relaxes the Markov assumption. It is used with the motivation that segments and features extracted from them can be more meaningful and expressive and thus yield better discriminative performance. Here, all the samples in a segment share the same label. Semi-Markov CRFs were first used in [128] for name entity recognition. They were also used for gesture and activity recognition [132, 148]. Another noteworthy application was in speech recognition where Zweig and Nguyen [183] made use of interesting segment-level features to recognize continuous speech sentences. Their approach outperformed an HMM-based approach by 2%.

In our CRF-based recognition framework, we start with segmented sentences (or sequences) where each sequence $\mathbf{S} = \{S_1, \ldots, S_n\}$ consists of n sub-segments comprising the normalized feature vectors as described in Section 5.5.1. A convenient approach to model the sequence of sub-segments is the semi-Markov CRFs.

For this modeling, we first need to extract features from the sub-segments to form the corresponding input observation sequence $\mathbf{x} = \{x_1, \ldots, x_n\}$. In this formulation, the term sub-segment is used to refer to an x_j in the input observation sequence $\mathbf{x}$, and the term "segment" is used to refer to a collection of contiguous x_j's in $\mathbf{x}$. Segment length is the number of sub-segments in a segment. We use u_t and v_t to indicate the start and end positions of a segment in $\mathbf{x}$, where u_t and v_t correspond to the sub-segment index in the input observation sequence, and $1 \leq u_t \leq v_t \leq |\mathbf{x}|$. The length of the output label sequence $\mathbf{y}$ depends on the final number of segments obtained by combining sub-segments in $\mathbf{x}$. Now, let $\mathbf{s} = \{s_1, s_2, \ldots, s_p\} \in \mathcal{S}$ denote a sequence of segments of $\mathbf{x}$, where $s_t = \{u_t, v_t, y_t\}$ is a segment with start position u_t, end position v_t, and a label $y_t \in \mathbf{Y}$. A segment feature function is defined as $g_i(\mathbf{s}, \mathbf{x}, t) = g_i(y_{t-1}, y_t, \mathbf{x}, u_t, v_t)$, and $\boldsymbol{\theta} = \{\gamma_1, \gamma_2, \ldots, \gamma_h\}$ are the parameters to be estimated in

$$p(\mathbf{s}|\mathbf{x}) = \frac{1}{\mathbf{Z}(\mathbf{x})} \exp\left(\sum_{t=1}^{p} \sum_{i=1}^{h} \gamma_i g_i(\mathbf{s}, \mathbf{x}, t)\right), \tag{6.1}$$

where

$$\mathbf{Z}(\mathbf{x}) = \sum_{\mathbf{s} \in \mathcal{S}} \exp\left(\sum_{t=1}^{p} \sum_{i=1}^{h} \gamma_i g_i(\mathbf{s}, \mathbf{x}, t)\right). \tag{6.2}$$

Given training data, $\mathcal{D} = \{\mathbf{x}^k, \mathbf{s}^k\}_{k=1}^{\mathcal{K}}$, the log-likelihood with L2-norm regularization is written as

$$\mathcal{L}(\theta) = \sum_{k=1}^{\mathcal{K}} \sum_{t=1}^{p} \sum_{i=1}^{h} \gamma_i g_i(\mathbf{s}^k, \mathbf{x}^k, t) - \sum_{k=1}^{\mathcal{K}} \log \mathbf{Z}(\mathbf{x}^k) - C \sum_{i=1}^{h} \frac{|\gamma_i|^2}{2}. \tag{6.3}$$

The parameter estimation procedure for semi-Markov CRFs is similar to the conventional linear-CRFs as described in Section 5.2. The main difference is that the start and end positions of the segments are taken into consideration during training in semi-Markov CRFs. More details of the training procedure of semi-Markov CRFs can be found in [128].

Inferencing in semi-Markov CRFs is formulated to find the best segment path given $\boldsymbol{\theta}$ and $\mathbf{x}$ as

$$\begin{aligned}\mathbf{s}^* &= \operatorname*{argmax}_{\mathbf{s}} p(\mathbf{s}|\mathbf{x}) \\ &= \operatorname*{argmax}_{\mathbf{s}} \sum_{t=1}^{p} \sum_{i=1}^{h} \gamma_i g_i(\mathbf{s}, \mathbf{x}, t) \\ &= \operatorname*{argmax}_{\mathbf{s}} \sum_{t=1}^{p} \sum_{i=1}^{h} \gamma_i g_i(y_{t-1}, y_t, \mathbf{x}, u_t, v_t).\end{aligned} \tag{6.4}$$

Let L be the upper bound on segment length and let ${}^{q}\mathbf{s}_{1:r}$ denote the set of all possible segments in $\mathbf{x}' = \{x_1, \ldots, x_r\}$, where $\mathbf{x}'$ is the input observation sequence from position 1 to $r \leq n$, such that the last segment has label q and ending position r. Let $\eta_r(q)$ denote the largest value of $p(\mathbf{s}'|\mathbf{x})$ for any $\mathbf{s}' \in {}^{q}\mathbf{s}_{1:r}$. The recursive formulation in semi-Markov CRFs is similar to the Viterbi algorithm and is written as:

$$\eta_r(q) = \begin{cases} \max\limits_{\hat{q}, d=1,\ldots,L} \eta_{r-d}(\hat{q}) \Phi'_{(r-d+1):r}(\hat{q}, q, \mathbf{x}) & \text{if } r > 0, \\ 1 & \text{if } r = 0, \\ 0 & \text{if } r < 0, \end{cases} \tag{6.5}$$

where

$$\Phi'_{(r-d+1):r}(\hat{q}, q, \mathbf{x}) = \exp\left(\sum_{i=1}^{h} \gamma_i g_i(y_{t-1} = \hat{q}, y_t = q, \mathbf{x}, r-d+1, r) \right). \tag{6.6}$$

The best segment path is traced by $\max_{\hat{q} \in \mathbf{Y}} \eta_{|\mathbf{x}|}(\hat{q})$.

However, one of the disadvantages in using the above (conventional) semi-Markov CRF for our problem is that it is highly dependent on the initial segmentation algorithm which yields the sub-segment sequences. Generally, we cannot expect the same break points or the same number of sub-segments to occur in two samples of the same sentence. For example, the segment $\tilde{S}_1$ in Figure 6.2 may have three sub-segments with different break points when the initial segmentation algorithm is applied to another sample of the sentence (consisting of the same underlying sign segments which need to be recovered). If features are extracted directly from these sub-segment sequences, we may have very different representation for the input observation sequences used to train and test the semi-Markov CRF. Also, the features extracted from individual sub-segments may not be representative enough to characterize the signs when they are combined together. More importantly, semi-Markov CRF cannot be straightforwardly used for phoneme-based modeling. If it is used to model the four parallel component channels independently, different segmentation may be obtained in different channels after decoding. This makes the sign sequence decoding procedure difficult as a more complex algorithm is needed to combine the information for final sign level segmentation.

In our case, we need a strategy that allows merging the original sub-segments together and recomputing features from the merged sub-segments. Our approach is to use only complete sign segments extracted from the signed sentences to train the recognition framework based on conventional linear CRFs as described in Section 5.2, and propose a new decoding algorithm for our two-layered CRF based on the decoding procedure used in semi-Markov CRFs, that can merge sub-segments optimally to recover signs. We describe the training of the proposed two-layered CRF framework in the next section (Section 6.2). The key recognition and decoding procedures are given in Section 6.3. The chapter ends with a summary presented in Section 6.4.

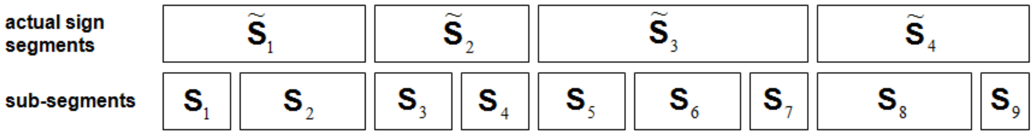


Figure 6.2: The test sub-segments and their corresponding clean segments.

6.2 Training the Two-Layered CRF Framework

To obtain appropriate training data from the signed sentences, the naïve Bayesian network segmentation algorithm given in Chapter 4 is first used to identify the boundary points. The few errors that result from automatic segmentation are manually corrected with the help of simultaneously recorded video sequences. Based on the identified boundary points and the video sequences, the movement epenthesis segments are easily discarded, and only the remaining clean sign segments are used for training.

An input observation consists of a sequence of feature vectors extracted from the sign segments of a sentence that remain after the movement epenthesis segments have been removed. We use ${}^{\ell_{sj}}\tilde{S}_j$ to denote a sign segment, where ${}^{\ell_{sj}}\tilde{S}_j = \{{}^{\ell_{hj}}\tilde{H}_j, {}^{\ell_{mj}}\tilde{M}_j, {}^{\ell_{oj}}\tilde{O}_j, {}^{\ell_{lj}}\tilde{L}_j\}$, comprises the corresponding segments for handshape, movement, orientation and location components at the phoneme level, ℓ_{sj} is a label associated with the sign level segment, $\{\ell_{hj}, \ell_{mj}, \ell_{oj}, \ell_{lj}\}$ are respective component labels at the phoneme level and j is the position of the segment in the sequence. For the k^{th} sentence with c sign segments, we have $\tilde{\mathbf{S}}^k = \{{}^{\ell_{s1}}\tilde{S}_1^k, \ldots, {}^{\ell_{sc}}\tilde{S}_c^k\}$, $\tilde{\mathbf{H}}^k = \{{}^{\ell_{h1}}\tilde{H}_1^k, \ldots, {}^{\ell_{hc}}\tilde{H}_c^k\}$, $\tilde{\mathbf{M}}^k = \{{}^{\ell_{m1}}\tilde{M}_1^k, \ldots, {}^{\ell_{mc}}\tilde{M}_c^k\}$, $\tilde{\mathbf{O}}^k = \{{}^{\ell_{o1}}\tilde{O}_1^k, \ldots, {}^{\ell_{oc}}\tilde{O}_c^k\}$, $\tilde{\mathbf{L}}^k = \{{}^{\ell_{l1}}\tilde{L}_1^k, \ldots, {}^{\ell_{lc}}\tilde{L}_c^k\}$. Features are extracted based on these sign segments and used to train the conventional CRFs in each channel independently. For clarity and convenience, we will drop the subscripts or superscripts whenever they are not required for discussion. Figure 6.3 illustrates the features extracted at each layer and layer outputs, and the details are explained in the following sections.

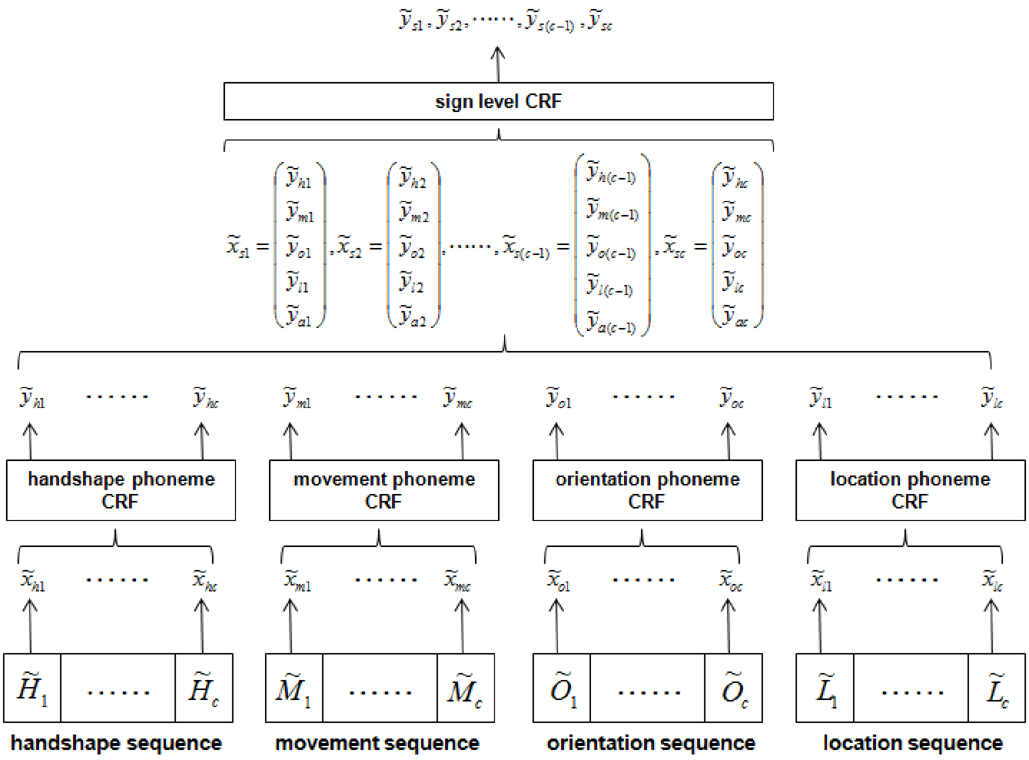


Figure 6.3: Input feature vectors extracted and their respective outputs at each level.

6.2.1 Training at the Phoneme Level

The lower level of Figure 6.1, shows the four independent channels which use CRFs to recognize the phonemes of the four components. The boundary points obtained by using the Bayesian network of Chapter 4 from the movement component are used for other components as well, and thus segments in all channels are aligned in time. For phoneme recognition, we define subphones as inputs to the lower level CRFs and represent a phoneme by a sequence of subphones. The handshape, movement, orientation and location training sequences at the phoneme level are denoted as $\mathcal{D}_h = \{\tilde{\mathbf{x}}_h^k, \tilde{\mathbf{y}}_h^k\}$, $\mathcal{D}_m = \{\tilde{\mathbf{x}}_m^k, \tilde{\mathbf{y}}_m^k\}$, $\mathcal{D}_o = \{\tilde{\mathbf{x}}_o^k, \tilde{\mathbf{y}}_o^k\}$, $\mathcal{D}_l = \{\tilde{\mathbf{x}}_l^k, \tilde{\mathbf{y}}_l^k\}$, for $k = 1, \ldots, N$, where $\tilde{\mathbf{x}}_h^k = \{\tilde{x}_{h1}^k, \ldots, \tilde{x}_{hc}^k\}$, $\tilde{\mathbf{x}}_m^k = \{\tilde{x}_{m1}^k, \ldots, \tilde{x}_{mc}^k\}$, $\tilde{\mathbf{x}}_o^k = \{\tilde{x}_{o1}^k, \ldots, \tilde{x}_{oc}^k\}$, and $\tilde{\mathbf{x}}_l^k = \{\tilde{x}_{l1}^k, \ldots, \tilde{x}_{lc}^k\}$ are the k^{th} input observation sequences, and $\tilde{\mathbf{y}}_h^k = \{\tilde{y}_{h1}^k, \ldots, \tilde{y}_{hc}^k\}$, $\tilde{\mathbf{y}}_m^k = \{\tilde{y}_{m1}^k, \ldots, \tilde{y}_{mc}^k\}$, $\tilde{\mathbf{y}}_o^k = \{\tilde{y}_{o1}^k, \ldots, \tilde{y}_{oc}^k\}$, and

$\tilde{\mathbf{y}}_l^k = \{\tilde{y}_{l1}^k, \ldots, \tilde{y}_{lc}^k\}$ denote the corresponding label sequences which are obtained from the phoneme transcription procedures described in Chapter 4. $\tilde{x}_{hj}^k$, $\tilde{x}_{mj}^k$, $\tilde{x}_{oj}^k$, $\tilde{x}_{lj}^k$ represents the j^{th} input feature vector in the respective component and the elements of the vectors correspond to the subphone labels.

The above is clarified by the illustration in Figure 6.4. To extract the features for subphones from the $\tilde{H}_j, \tilde{M}_j, \tilde{O}_j, \tilde{L}_j$ segments, each segment is divided into $M = 10$ sub-segments with equal arc length to yield $\tilde{H}_j = \{h_{j1}, h_{j2}, \ldots, h_{j10}\}$, $\tilde{M}_j = \{m_{j1}, m_{j2}, \ldots, m_{j10}\}$, $\tilde{O}_j = \{o_{j1}, o_{j2}, \ldots, o_{j10}\}$, $\tilde{L}_j = \{l_{j1}, l_{j2}, \ldots, l_{j10}\}$. Features extracted from these sub-segments are clustered to define subphone labels using a procedure similar to that used to define the static component phonemes (Section 4.4). The only difference is that feature vectors extracted from the individual sub-segments are directly used for clustering (instead of concatenating the starting and ending feature vectors of the sub-segments). The same procedure is applied to the four components to extract the respective subphone labels and is described below.

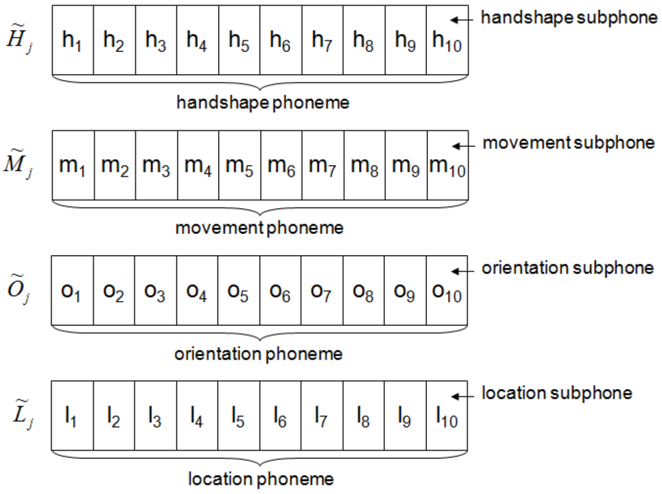


Figure 6.4: Phonemes and subphones.

i) Pick N_q sign segments randomly from the entire training data set ($N_q \gg$ the number of signs in the database).

ii) Divide each segment into M intervals and compute the mean values of the vectors within the interval, e.g. for handshape component, the mean of the 16-D handshapes in the interval is computed. The total number of vectors for clustering is thus $M \times N_q$.

iii) Compute the similarity measure based on Euclidean distance and run the AP algorithm with the $M \times N_q$ data points.

iv) Use the $\hat{k}$ exemplars found by the AP algorithm as the initial centroids for k-means.

v) Run k-means with all the mean vectors using the initial centroids obtained from the AP algorithm. The final centroids obtained are used as the templates for the subphones. A subphone label $\hat{j}$ is given to an interval if its mean vector is found closest to the $\hat{j}^{th}$ cluster.

The basic state features in each CRF are the 10 subphone labels extracted from each segment in the component sequences. We also extract n-gram features from across the segments as well as from within the segments in a sentence. Here, we choose $n = 3$ for both type of features. For example, in a sequence of handshape segments in a sentence, $\tilde{\mathbf{H}} = \{\tilde{H}_1, \tilde{H}_2, \tilde{H}_3, \tilde{H}_4, \tilde{H}_5\}$, each segment $\tilde{H}_j$ consists of a sequence of subphones, $\tilde{H}_j = \{h_{j1}, h_{j2}, \ldots, h_{j10}\}$, $j = 1, \ldots, 5$. Due to the causal nature of the decoding algorithm, the trigram features across segments are computed using only the previous segments, i.e. for a handshape subphone at time instant j and subphone location i, the trigram state feature is obtained by using the subphone labels of "$h_{(j-2)i}h_{(j-1)i}h_{ji}$". However, for the n-gram feature within segments, it is obtained by using the subphone labels of "$h_{j(i-1)}h_{ji}h_{j(i+1)}$". These are illustrated in Figure 6.5. The transition features are the labels of the adjacent segments. These features are used to train the four phoneme level linear CRFs independently with the conventional approach.

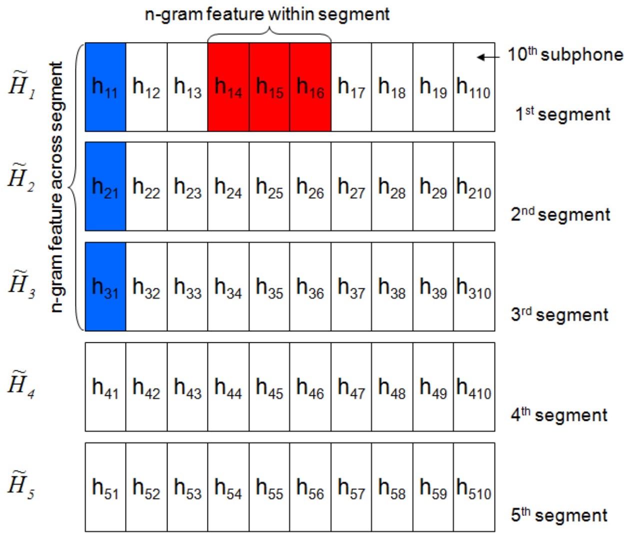

n-gram feature within segment
n-gram feature across segment
10th subphone
1st segment
2nd segment
3rd segment
4th segment
5th segment
$\tilde{H}_1$
$\tilde{H}_2$
$\tilde{H}_3$
$\tilde{H}_4$
$\tilde{H}_5$
h11 h12 h13 h14 h15 h16 h17 h18 h19 h110
h21 h22 h23 h24 h25 h26 h27 h28 h29 h210
h31 h32 h33 h34 h35 h36 h37 h38 h39 h310
h41 h42 h43 h44 h45 h46 h47 h48 h49 h410
h51 h52 h53 h54 h55 h56 h57 h58 h59 h510

Figure 6.5: N-gram features based on the respective sub-segments.

6.2.2 Training at the Sign Level

After the phoneme level training, all the component sequences in the training sentences are input to the parallel CRFs at the lower level independently for decoding using conventional CRF decoding algorithm. The decoded output phoneme sequences are used to train the sign level CRF. The training samples, $\mathcal{D}_s = \{\tilde{\mathbf{x}}_s^k, \mathbf{y}_s^k\}$, $k = 1, \ldots, N$ are used to train the sign level CRF, where $\tilde{\mathbf{x}}_s^k = \{\tilde{x}_{s1}^k, \ldots, \tilde{x}_{sc}^k\}$ is an input observation sequence at the sign level and $\tilde{\mathbf{y}}_s^k = \{\tilde{y}_{s1}^k, \ldots, \tilde{y}_{sc}^k\}$ is the corresponding output sequence where $\tilde{y}_{sj}^k$ is one of the $\mathcal{R}$ sign labels $\{SIGN_\kappa\}$ for $\kappa = 1, \ldots, \mathcal{R}$. We define (see Figure 6.3) $\tilde{x}_{sj}^k = (\tilde{y}_{hj}^k, \tilde{y}_{mj}^k, \tilde{y}_{oj}^k, \tilde{y}_{lj}^k, \tilde{y}_{aj}^k)$ as the j^{th} input feature vector of the k^{th} sequence to the sign level CRF consisting of phoneme label outputs from the four parallel CRFs and a segment arc length label, where the subscripts h, m, o and l denote the CRF outputs from the handshape, movement, orientation and location channels, respectively. We quantized the segment arc length into six symbols by simple thresholding and the j^{th} segment's arc length label in the k^{th} sequence is denoted as $\tilde{y}_{aj}^k$. For sign level recognition, we computed the n-gram features across segments and used the adjacent sign labels as the transition features. The arc length feature, $\tilde{y}_a \in 1, \ldots, 6$, and the number of symbols for $\tilde{y}_h$, $\tilde{y}_m$, $\tilde{y}_o$ and $\tilde{y}_l$ are determined experimentally as described in Section 7.5.

6.3 Modified Segmental Decoding Algorithm

After the CRF-based classifier is trained, it is used to recognize test sentences which have been segmented by the initial segmentation algorithm, and whose sub-segments have been classified as *SIGN* or *ME* (Chapter 5). Several issues need to be addressed during decoding. Firstly, the start and end points of the sign segments are unknown and need to be efficiently recovered by proper merging of the sub-segments. Secondly, though the test sequence has been segmented and the sub-segments have been classified as *SIGN* or *ME*, the classification scheme cannot be expected to be error free especially on data from new signers. Hence, simply discarding the sub-segments labeled as *ME* at this stage, and presenting only the sub-segments labeled as *SIGN* to the decoder will be suboptimal.

Hence, we develop a new decoding procedure which modifies the semi-Markov CRF decoding algorithm described in Section 6.1, and it is devised to have the following characteristics:

i) Ability to efficiently evaluate different merging combinations of sub-segments for recognition.

ii) When a hypothesized segment is recognized by the CRF-based classifier as a particular sign, there should provide a mechanism to further indicate whether the classification is one of the valid sign labels, or it is an unknown sign.

iii) A sub-segment can merge not only with its adjacent sub-segments, but also with sub-segments which are separated from it by M_s number of sub-segments.

The merging procedure based on the modified semi-Markov CRFs is described in Section 6.3.1. Section 6.3.2 describes the inclusion of a two class SVM to provide an *UNCLASSIFIED* label for use in the recognition procedure. Section 6.3.3 presents a method to allow the CRFs to skip states followed by the complete decoding procedure. The complexity of the decoding procedure is discussed in Section 6.3.4.

6.3.1 The Basic Algorithm

We start by presenting the merging and classification procedure used in the proposed decoding algorithm. For simplicity of explanation, we assume that the movement epenthesis sub-segments have been removed correctly and that we only have a sequence of sign sub-segments. If they are merged correctly, the sign segments would be available for sign recognition. Figure 6.2 shows an example of a sequence of test sub-segments with respect to the actual sign segments. Segmentation of this test sentence has yielded nine sub-segments; the sign segment and their boundary, however are unknown. Hence, the decoding algorithm must find the merging of the sub-segments that yields the most likely sign sequence. In this example $\{S_1, S_2\}$,$\{S_3, S_4\}$,$\{S_5, S_6, S_7\}$ and $\{S_8, S_9\}$ would need to be merged together for recognizing the sign sequence.

Our strategy is to consider merging adjacent sub-segments, hypothesize that the merged sub-segments form correct sign segments, and select the labels for the hypothesized segments with a Viterbi-like procedure to compute the most likely sign sequence. In our algorithm, the features are computed dynamically based on the merged sub-segments during the decoding process, i.e. every hypothesized segment is divided into M intervals and features are extracted from them. As different merging combinations are considered at each step, the length of the final decoded sign labels may be different from the length of the input sub-segment sequence. To reflect this, we modify the recursion in (6.5) as follows.

To modify the decoding algorithm in Section 6.1, we need to redefine some terms used in the formulation. Here, $\mathbf{s} = \{s_1, s_2, \ldots, s_p\}$ denotes a sequence of segments formed by merging the sub-segments in the sub-segment sequence $\mathbf{S} = \{S_1, \ldots, S_n\}$, and $\hat{\mathbf{x}}$ denotes an arbitrary input observation sequence consisting of feature vectors extracted from the merged sub-segments in $\mathbf{S}$. Hence, u_t and v_t in $s_t = (u_t, v_t, y_t)$ are the sub-segment positions in $\mathbf{S}$ which describe the start and end positions of the t^{th} segment, and y_t is the label assigned to the segment. The inferencing is formulated to find the best segment path

$$\mathbf{s}^* = \operatorname*{argmax}_{\mathbf{s}} p(\mathbf{s}|\hat{\mathbf{x}}). \tag{6.7}$$

Similar to (6.5), we define L as the upper bound on the number of sub-segments to be merged to form a segment, and use ${}^{q}\mathbf{s}_{1:r}$ to denote the set of all possible partial merging in $\mathbf{S}' = \{S_1, \ldots, S_r\}$, such that the last segment which is yielded by merging the sub-segments, has label q and ending position r. To further facilitate the decoder formulation, we define $\hat{\mathbf{x}}_{j_1:j_2}$ as an input observation sequence with sub-segments merged from position j_1 to j_2. Let $\delta_r(q)$ denote the largest value of $p(\mathbf{s}'|\hat{\mathbf{x}}')$ for any $\mathbf{s}' \in {}^{q}\mathbf{s}_{1:r}$. The modified recursive term is written as:

$$\delta_r(q) = \begin{cases} \max\limits_{\hat{q}, d=1,\ldots,L} \delta_{r-d}(\hat{q}) \Phi_r(\hat{q}, q, \hat{\mathbf{x}}_{(r-d+1):r}) & \text{if } r > 0, \\ 1 & \text{if } r = 0, \\ 0 & \text{if } r < 0, \end{cases} \tag{6.8}$$

where

$$\Phi_r(\hat{q}, q, \hat{\mathbf{x}}_{(r-d+1):r}) = \exp\left(\sum_{i=1}^{h} \lambda_i f_i(y_{t-1} = \hat{q}, y_t = q, \hat{\mathbf{x}}_{(r-d+1):r}\right), \tag{6.9}$$

and $\hat{\mathbf{x}}_{(r-d+1):r}$ denotes an input observation sequence with merged sub-segments from position $r-d+1$ to position r. Though (6.5) and (6.8) appear to be similar, the important distinction is that in the latter case the sub-segments are actually merged and feature vectors are then extracted from these merged sub-segments. In our formulation, r denotes the position of the sub-segment in the sequence $\mathbf{S}$ and d denotes the length of a hypothesized segment (the number of sub-segments to be merged).

Now, we extend the algorithm to the two-layered CRFs. One way to perform decoding in the two-layered CRFs is to first decode the component phoneme sequences independently in the four channels and use the outputs from the phoneme level CRFs to further decode the sign sequence. The main problem with this approach is that different sub-segment merging combinations may form in the different channels as the four components are decoded independently. This leads to higher complexity for sign level recognition due to inconsistent segment lengths. Furthermore, independent merging of sub-segments in each channel may not be reliable. Hence, we adopt an approach which allows the phoneme level and sign level decoding procedures to proceed simultaneously, where the decoded component phonemes are fused for sign recognition as the decoding proceeds. In this scheme, the proposed decoding algorithm is used for the sign level CRF only and the conventional decoding algorithm is used for the phoneme level CRFs. At the phoneme level, parallel decoding of the partial phoneme sequences is carried out and the output information from the partial sequences is combined and used to compute the recursive term for sign level recognition.

Given a sentence consisting of n sub-segments, with $\mathbf{S} = \{S_1, \ldots, S_n\}$ at the sign level and $\mathbf{H} = \{H_1, \ldots, H_n\}$, $\mathbf{M} = \{M_1, \ldots, M_n\}$, $\mathbf{O} = \{O_1, \ldots, O_n\}$ and $\mathbf{L} = \{L_1, \ldots, L_n\}$ at the phoneme level, the decoding procedure for the two-layered CRFs is described below. For illustration, we also provide a simple example to explain the proposed decoding procedure. Assume that we have four sub-segments in a sequence consisting of two sign segments as shown in Figure 6.6 and let $L = 2$.

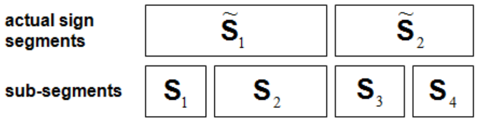


Figure 6.6: A sequence with four sub-segments.

i) For $r = 1$, we need to compute the recursive term $\delta_1(q)$ with $\{S_{1:1}\}$ at the sign level, where the notation $S_{r1:r2}$ denotes segments merged from position r_1 to r_2. However, this requires the phoneme labels for the corresponding merged sub-segments from the component channels, $\{H_{1:1}, M_{1:1}, O_{1:1}, L_{1:1}\}$. The merged sub-segments are hypothesized as correct sign segments and each segment $\{H_{1:1}, M_{1:1}, O_{1:1}, L_{1:1}\}$ is divided into 10 intervals from which the mean features are extracted. These features from the hypothesized segment in each channel are decoded independently by the respective CRFs using the standard CRF decoding algorithm. The phoneme label outputs from the CRFs are concatenated to form an input feature vector to compute $\delta_1(q)$. In the example of Figure 6.6, $\delta_1^1(q)$ and $\delta_1^2(q)$ are computed for $d = 1$ and $d = 2$, respectively as shown in Figure 6.7. The highest value of $\delta_1(q)$ given by $d = 1$ and some value of $\hat{q}$, is selected and it is highlighted in Figure 6.7.

ii) For $r = k$, $k > 1$, $\delta_r(q)$ is computed by first finding the "best" possible

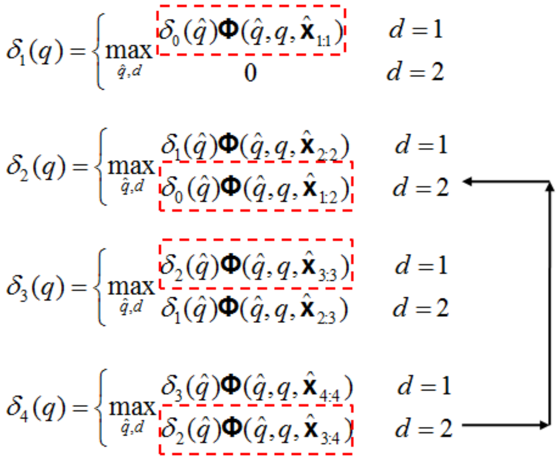

$\delta_1(q) = \max_{\hat{q},d} \delta_0(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{1:1})$ $d=1$
0 $d=2$
$\delta_2(q) = \max_{\hat{q},d} \delta_1(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{2:2})$ $d=1$
$\delta_0(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{1:2})$ $d=2$
$\delta_3(q) = \max_{\hat{q},d} \delta_2(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{3:3})$ $d=1$
$\delta_1(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{2:3})$ $d=2$
$\delta_4(q) = \max_{\hat{q},d} \delta_3(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{4:4})$ $d=1$
$\delta_2(\hat{q})\mathbf{\Phi}(\hat{q},q,\hat{\mathbf{x}}_{3:4})$ $d=2$

Figure 6.7: An example to illustrate the decoding procedure.

merging combination for the sub-segments in the partial sign sequence from position $r = 1$ to $r = k - d$ at the sign level. This is done by performing a backtracking for the partial sign sequence $\mathbf{S} = \{S_1, \ldots, S_{k-d}\}$ using the modified decoding algorithm as described earlier to form the hypothesized sequence $\hat{\mathbf{S}} = \{S_{1:k'_1}, \ldots, S_{k'_s:k-d}, S_{k-d+1:k}\}$. The same sub-segment merging that is hypothesized at the sign level is applied to the component phoneme level and features are extracted from these merged sub-segments. These features are input to the respective parallel CRFs for decoding using the conventional approach and the last phoneme label outputs from the decoded segments are combined and used to compute $\delta_r(q)$. For the example in Figure 6.6, at $r = 3$ we need to compute the recursive term for $\delta_3(q)$. The partial sequences required for computation at the phoneme level are $\mathbf{H} = \{H_1, H_2, H_3\}$, $\mathbf{M} = \{M_1, M_2, M_3\}$, $\mathbf{O} = \{O_1, O_2, O_3\}$, $\mathbf{L} = \{L_1, L_2, L_3\}$, corresponding to the sign level sub-segment sequence, $\mathbf{S} = \{S_1, S_2, S_3\}$. To compute $\delta_3(q)$ (at $r = 3$), two possible ways of merging sub-segments need to be considered, i.e. $S_{3:3}$ for $d = 1$ and $S_{2:3}$ for $d = 2$. For $S_{2:3}$, the only possible merging is $\hat{\mathbf{S}} = \{S_{1:1}, S_{2:3}\}$, for which a likelihood $\delta_3^1(q)$ can be calculated. For computing this, sub-segments 2 and 3 from each component need to be merged, features extracted from them and classified to phoneme labels by the four CRFs. For $d = 1$, segment 3 is not merged with previous segments, and so the possibilities to consider are $\hat{\mathbf{S}}_1 = \{S_{1:1}, S_{2:2}, S_{3:3}\}$ and $\hat{\mathbf{S}}_2 = \{S_{1:2}, S_{3:3}\}$. However, $\delta_2(q)$ at $r = 2$ already records which of the mergings $\{S_{1:1}, S_{2:2}\}$ or $\{S_{1:2}\}$ has higher likelihood. Suppose $\{S_{1:2}\}$ has the higher likelihood. Then, the likelihood of the merged sequence $\{S_{1:2}, S_{3:3}\}$ is calculated as $\delta_3^2(q)$. If $\delta_3^1(q)$ is larger than $\delta_3^2(q)$, then the merged sub-segment sequence at $r = 3$ is $\{S_{1:1}, S_{2:3}\}$; else,it is $\{S_{1:2}, S_{3:3}\}$.

iii) At the end of the recursive steps, backtracking is performed to retrieve the sign sequence. In the example given in Figure 6.7, the best value of $\delta_4(q)$ is obtained when $d = 2$ for some $\hat{q}$. Backtracking from here leads to $\delta_2(q)$ which also has a best value when $d = 2$ for some $\hat{q}$. With this approach, we merge $\{S_1, S_2\}$ at step 2 and $\{S_3, S_4\}$ at step 4 and along with this the signs are also recovered by tracing the best possible state sequence.

6.3.2 Two-Class SVMs

Thus far, the modified decoding algorithm described in Section 6.3.1 makes a force assignment of one of the 107 sign labels (with the maximum likelihood) to the hypothesized segments regardless of whether the segments correspond to sign labels or not. Also, test sentences will consist of sign and movement epenthesis sub-segments which have been misclassified. Thus, hypothesized segments may actually contain movement epenthesis segments for which the modified CRF algorithm has not been trained. Hence, it is desirable to have a mechanism to provide additional information to the decoding algorithm to ascertain whether a hypothesized segment can be one of the valid signs or not.

A possible approach is to add an *UNCLASSIFIED* class to the CRFs, and train the CRFs to recognize this in addition to the valid signs. However, it is difficult to obtain representative *UNCLASSIFIED* samples to discriminate against a large number of signs. Often, a very large number of negative samples is required which can easily overwhelm the number of the sign samples. In [170] the invalid patterns are added to the trained CRF by using some threshold models to compute new weights for the state and transition feature functions of the *UNCLASSIFIED* class. However, a threshold-based approach may not be suitable to handle signer variations and it is difficult to use simple classifier to discriminate an *UNCLASSIFIED* class against a large number of sign classes. In addition, the feature function weights for the invalid class are computed based on all the sign patterns. Thus, every new sign that is added to the system will affect the thresholds, and the new feature function weights for the *UNCLASSIFIED* class need to be found.

A simpler approach that we adopt is to use two-class SVMs for each sign to discriminate between valid and invalid segments. This approach reduces the complexity of the problem significantly from a large multi-class problem to a set of two-class problems. Also, if a new sign is added to the system, the trained SVMs will not be affected and only one new SVM needs to be trained for the new sign. The main task in this approach is to generate samples of the *UNCLASSIFIED* class. The positive samples can be obtained easily by using the sign segments from a particular class. To generate negative samples, the sub-segment sequences of the training sentences can be used to form all possible segments by incorrect merging, but the number of generated negative samples will be too large, leading to a large imbalance in training the two-class SVMs. Hence, we propose a simple approach to generate a subset of representative negative samples for each class of signs, as follows. Given a partial sequence of correct segments, find the class label of the next hypothesized segment if it is obtained from wrongly merged sub-segments. Then, decode the segment formed by incorrect merging and use the segment as a negative sample with respect to its identified class label. This step is repeated for all training sequences with and without movement epenthesis segments. The following gives an example to explain the procedure to generate the negative samples.

i) Given a sequence of n sub-segments (for example, $n = 6$) as $\mathbf{S} = \{{}^{SIGN_1}S_1, {}^{SIGN_1}S_2, {}^{ME}S_3, {}^{SIGN_2}S_4, {}^{SIGN_3}S_5, {}^{SIGN_3}S_6\}$, and an upper bound L for the length of the invalid segments (L is the same as the maximum segment length used in the decoding algorithm). Suppose $L = 5$ in this example.

ii) Starting from $r = 1$, where r is the index for the position of the sub-segment in the sequence, $\{S_{1:2}\}$ is used as a positive sample for class $SIGN_1$. The segments $\{S_{1:1}, S_{1:3}, S_{1:4}, S_{1:5}\}$ which do not correspond to valid signs are input to the two-layer CRF to obtain a sign label for each segment. These segments serve as negative examples for the respective decoded signs. Here, we use only the conventional linear-CRF decoding algorithm for decoding as the merged sub-segments are already given.

iii) For $r > 1$, we attempt to generate negative samples only at positions where the preceding partial sequence forms a valid segment formed by correctly merged sub-segments. Thus, at $r = 2$ we do not generate any negative samples as $\{S_{1:1}\}$ is an invalid partial sequence. At $r = 3$, the negative samples are generated as $\{S_{3:3}, S_{3:4}, S_{3:5}, S_{3:6}\}$ and the sequences of $\hat{\mathbf{S}}_1 = \{S_{1:2}, S_{3:3}\}$, $\hat{\mathbf{S}}_2 = \{S_{1:2}, S_{3:4}\}$, $\hat{\mathbf{S}}_3 = \{S_{1:2}, S_{3:5}\}$, $\hat{\mathbf{S}}_4 = \{S_{1:2}, S_{3:6}\}$ are formed and decoded. The class labels for the negative samples are obtained from the decoded sign sequence. The same procedure is applied for the remaining sub-segments.

iv) We repeat the steps for the same sequence with movement epenthesis sub-segments removed, i.e. ${}^{ME}S_3$ for this example. We discard repeated negative samples if they are identical to samples generated in the sequence with movement epenthesis.

In general, there is a possibility that no negative samples are generated for some signs in which case, these signs need to be handled separately. However, there was no such problem in our data set. All the generated negative samples for a sign are used with the positive samples to train a two-class SVM. As the SVMs are incorporated into our proposed CRF-based decoding procedure, SVMs with probabilistic outputs as described in Section 5.3, are used. The features used for training the SVMs are similar to the features used for training the *SIGN*/*ME* classifier of Chapter 5 and are listed in Table 6.1. Here, the only difference is that we do not use the transition features described in Chapter 5 as the two-class SVMs need to discriminate using only local features of given segments. The features listed in Table 6.1 are concatenated to form 126-D real-valued feature vectors for input to the two-class SVMs.

The two-class SVMs are integrated into the decoding algorithm by having additional steps in the maximization of the likelihood. Instead of directly obtaining the maximum and making a forced choice for sign label, the most likely sign label for each hypothesized segment is first determined, and the two-class SVM for the selected class is used to check if the hypothesized segment is valid. Only the valid hypothesized segments are passed to the usual decoding procedure to obtain the most likely sign label. In case all hypothesized segments are declared as invalid by the two-class SVMs, we fall back on the usual decoding procedure using simple maximization. We denote this function as “svmmax()”.

Table 6.1: Features for SVM.

Component	Feature	Description	Length
Handshape	**hand_start**	Starting handshape of each segment.	16-D (each feature)
	hand_end	Ending handshape of each segment.	
	hand_msdif	Mean of the adjacent handshape differences of each segment.	
	hand_std	Handshape standard deviation of each segment.	
Orientation	**orien_start**	Starting palm orientation of each segment.	9-D (each feature)
	orien_end	Ending palm orientation of each segment.	
	orien_msdif	Mean of the adjacent palm orientation differences of each segment.	
	orien_std	Palm orientation standard deviation of each segment.	
Location	**loc_mean**	Mean of the hand positions of each segment.	3-D (each feature)
	loc_start	Starting hand position of each segment.	
	loc_end	Ending hand position of each segment.	
	loc_std	Location standard deviation of each segment.	
Movement	**mov_dom**	Dominant direction of hand motion of each segment.	3-D (each feature)
	mov_start	Starting direction of hand motion of each segment.	
	mov_end	Ending direction of hand motion of each segment.	
	mov_std	Movement standard deviation of each segment.	
Others	**arc_length**	Arc length of each segment.	1-D
	num_seg	Number of sub-segments merged.	1-D

6.3.3 Modified Decoding Algorithm with Skip States

As the *SIGN* and *ME* classifier in Chapter 5 is not error free, the final test sequences may include sub-segments which are actually movement epenthesis and be missing some of the actual sign sub-segments which may have been erroneously classified as *ME* and discarded. The modified decoding algorithm described in Section 6.3.1 assumed for simplicity that sub-segment classification is perfect, so that sub-segments were merged with their immediate neighbors without skipping any of the sub-segments. However, if a sequence consists of movement epenthesis sub-segments erroneously labeled as *SIGN*, it would be desirable to skip these sub-segments. To accommodate this, we modify the decoding algorithm as follows. Let M_s be the maximum number of states that can be skipped. Together with the inclusion of the two-class SVMs in the decoding algorithm, the recursive term can be rewritten as

$$\delta_r(q) = \begin{cases} \underset{\substack{\hat{q},\, d=1,\ldots,L,\\ t_s=0,\ldots,M_s}}{\text{svmmax}} \delta_{r-d-t_s}(\hat{q})\Phi_r(\hat{q}, q, \hat{\mathbf{x}}_{(r-d+1):r}) & \text{if } r > 0, \\ 1 & \text{if } r = 0, \\ 0 & \text{if } r < 0, \end{cases} \tag{6.10}$$

where t_s denotes the number of states to be skipped and the remaining terms are the same as in (6.8) and (6.9).

Suppose $\mathbf{S} = \{S_1, S_2, S_3\}$ is a test sequence with a classification error only in sub-segment 2 (i.e. it is actually not *SIGN*, but *ME*). If no skip state is allowed, S_2 has to be included to evaluate the most likely path even though it is in error, and three possible segment sequences $\hat{\mathbf{S}}_1 = \{S_1, S_2, S_3\}$, $\hat{\mathbf{S}}_2 = \{S_{1:2}, S_3\}$ and $\hat{\mathbf{S}}_3 = \{S_1, S_{2:3}\}$ would need to be evaluated. In the extended decoding algorithm with skip state, direct transition from S_1 to S_3 becomes possible. Then, if the sequence $\hat{\mathbf{S}}_4 = \{S_1, S_3\}$ is found to have higher likelihood, it will be selected as the most likely segment sequence, excluding the incorrectly classified S_2.

The *SIGN* and *ME* labels of the sub-segments provided by the classifier of Chapter 5 can also be used to break the complete sequence into partial sequences; i.e. the positions of the *ME* sub-segments which have been discarded can be used as indicators of potential boundary points across which merging is not necessary. Hence, when an *ME* label is encountered, the sequence is broken into two independent partial sequences and sub-segments from the two partial sequences are never merged. This reduces the decoding computations for the sign sequence significantly. The complete modified decoding algorithm is given as follows.

i) Given a sequence of sub-segments $\mathbf{S} = \{S_1, \ldots, S_n\}$, with S_j classified as *SIGN* or *ME*.

ii) The recursive term in (6.10) is computed for all partial sequences with an appropriate choice of M_s. Increasing $M_s > 0$ increases the computational cost. When *ME* sub-segment is encountered, the search ends and further sub-segment merging is stopped. The next partial sequence is treated as a new sequence and merging is only done with succeeding sub-segments.

iii) The computation of the recursive term is continued until the end of the sequence or until an *ME* label is encountered, and backtracking retrieves the sign sequence.

6.3.4 Computational Complexity

The inferencing computation for semi-Markov CRFs is more expensive than conventional linear-CRFs as it needs to consider several potential segment lengths d to maximize the likelihood. However, the additional cost is only linear in L [128] which is no more expensive than order-L linear CRFs. The additional computational cost for order-L CRFs is exponential in L. In our case, computational cost of the basic decoder is similar to semi-Markov CRFs, but additional computational cost is incurred to maximize over possible skip states t_s. Again, the additional cost is linear in $(M_s + 1)$. Thus, the proposed inferencing procedure incurs additional cost which is linear in $L \times (M_s + 1)$ as compared to conventional linear-CRFs. We observed in our data that a movement epenthesis segment often consists of one to six *ME* sub-segments. After discarding the *ME* sub-segments using the *SIGN*/*ME* classifier, the maximum number of consecutive *ME* sub-segments that are erroneously classified as *SIGN* in a sign segment is about two. Hence, we chose $M_s = 2$ for our problem. Compared to linear-CRFs and semi-Markov CRFs, the increase in computational cost by using the two-class SVMs (to classify segments as valid or invalid signs) and extracting features on the fly along with the inferencing procedure is minor.

6.4 Summary

In this important part of the work, we devised a two-layer CRF-based framework to recognize continuously signed sentences from multiple signers. The first layer of the framework was used to recognize component phoneme sequences independently and their output labels were combined to decode sign sequence at the sign level. For training, the phoneme transcription procedure described in Chapter 4 was used to train the phoneme level CRFs, and the phonemes outputs were used to train the sign level CRF. For decoding, sub-segments and their labels as obtained from the classifier proposed in Chapter 5 were used as inputs to the proposed segmental CRF decoding algorithm, where subphones were used as basic elements. The decoding algorithm was devised to merge the sub-segments to form sign segments for robust sign sequence recognition. The algorithm was extended to include two-class SVMs which were used to classify hypothesized segments during decoding as valid or invalid, and thus relieve the decoder from making an incorrect forced choice. For handling possible erroneous movement epenthesis sub-segments remaining in sentences, the decoder was modified to consider sub-segments that were non-contiguous.

True creativity often starts
where language ends.

Arthur Koestler
(1905-1983)

7 Experimental Results and Discussion

7.1 Experimental Schemes

We present experimental results in this Chapter for the different subsystems in our approach to recognize naturally signed continuous sentences, viz. 1) the segmentation algorithms and phoneme transcription procedures of Chapter 4 (subsystem 1), 2) the *SIGN/ME* sub-segment classifier of Chapter 5 (subsystem 2), and 3) the two-layer CRF recognition framework of Chapter 6 (subsystem 3). For subsystems 2 and 3, we used the complete data set, but for subsystem 1, we used only a subset of the data as this part of the experiments served as a preliminary evaluation of the segmentation algorithms and the phoneme transcription procedures. In addition, we conducted preliminary evaluation of phoneme transcription procedure only for the movement component as this is a novel approach, unlike the straightforward approach that was adequate for the static components. The selected segmentation algorithms as well as the proposed phoneme transcription procedures for all components were then applied to the entire data in subsystem 3 where they were used to train the phoneme classifiers in the recognition framework.

In the following, the data collected for the experiments is described in Section 7.2. Details of the experiments and results for subsystems 1, 2 and 3 are presented in Sections 7.3, 7.4 and 7.5, respectively. The experiments for obtaining the final phonemes in the four channels based on the complete set of data are also presented in Section 7.5. Section 7.6 summarizes the results obtained.

7.2 Data Collection for Continuous ASL

We used a CyberGlove® [149] and Polhemus FASTRACK® system [121] to acquire the handshape, palm orientation and hand position data. The tracker and glove data were synchronized at a frame rate of about 31.10 ms. The 18 sensors in the glove yielded readings of the joint and abduction angles of the fingers as well as the wrist pitch and yaw. The trackers were attached to the back of each of the signer's hands and a third to the base of the signer's spine, to serve as a reference for discounting the variation in the position, and facing orientation of the signer relative to the transmitter. We adopted the same procedures given in [113] (Appendix C) to calculate the relative readings for the position and orientation of the hands. This raw data consists of 16-D, 9-D and 3-D feature vectors for handshape, palm orientation and location, respectively. We used only 16-D glove angles and discarded the wrist pitch and yaw which are not related to handshape. The 9-D feature vector for palm orientation is made up of three unit vectors corresponding to the x-axis, y-axis and z-axis which measure the hand rotations in the x-y plane as well as the palm direction (z-direction). The 3-D position vector of the hand is used to estimate location. All the experiments in this work were conducted using the four components from the right hand, obtained from tracker and glove data. For each sentence, a corresponding video sequence of the frontal view of the signer was also recorded. This served as a useful visual aid during manual processing of glove and tracker data to generate appropriate training sets for various experiments.

The data for the continuous sign recognition experiments was obtained from eight signers including seven deaf persons and a hearing person. The seven deaf subjects were native signers of the local sign language and the last was an expert signer. The signed sentences were performed continuously, without pauses between signs and closely followed ASL grammar. There were in total 74 distinct sentences from a 107-sign vocabulary that included basic signs and signs with directional verb inflections. Each sentence was made up of 2 to 6 signs. The average number of samples per signer for each distinct sentence was between 3 and 10, providing a total of 2393 sentences and 7786 signs. Table A.1 in Appendix A lists the 107-sign vocabulary which consists of 72 different basic lexical words and Table A.2 lists the 42 inflected directional verbs. The ASL sentences collected contained most of the variations in sign language that were described in Chapter 1, the exception being lexical variation, where a sign can be performed with completely different appearances.

7.3 Subsystem 1: Experiments and Results

In this part of the work, we used only the 3-D position (x,y,z) coordinates of the right hand obtained from the Polhemus FASTRACK® trackers. In addition, a video of the frontal view of the signer was used to facilitate manual segmentation of the sentences and phoneme transcription by visual observation. We conducted several experiments to evaluate the automatic rule-based trajectory as well as the naïve Bayesian network segmentation procedures and the phoneme transcription process. The evaluations are based on 25 ASL sentences signed 5-10 times by a deaf signer (one of the eight signers described in Section 7.2). Various subsets of this data were used for different experiments described below.

7.3.1 Automatic Trajectory Segmentation

To assess the performance of automatic trajectory segmentation, "ground truth" segment boundary points were manually marked by an expert signer. We use the terms "true segment/boundary point" and "false alarm" only in relation to points manually marked by the expert signer. Manual segmentation involves difficult judgements and guesses, and it would be optimistic to label this as "ground truth". In the segmentation algorithms, the initial segments were obtained from all samples of the 25 sentences by automatically locating the points of minimal velocity and maximal change of directional angle. This yielded 1996 initial segment boundary points.

We conducted two experiments on these segmented points with the naïve Bayesian network classifier (Experiment NB) as well as with the rule-based classifier (Experiment RB1). To use the rules in Table 4.2 for processing this initial segmentation in Experiment RB1, threshold values (T_i) are needed for the features. To obtain these thresholds, we picked two training samples each from 13 randomly picked sentences, which yielded 330 initial segment points. Of these 188 points were false alarms and 142 true boundary points in relation to the manually marked points. The required features (velocity and change of directional angle) were then extracted from this training data, and the threshold value for each feature was set by examining its distribution. We used the same training data to extract features described in Section 4.3.1.3 to train the naïve Bayesian network classifier in Experiment NB.

We divided the data into three groups to assess the rule-based and naïve Bayesian network classifiers. Experiment EP1 assessed performance only on the training samples; Experiment EP2 assessed performance on the same training sentences but with unseen samples. Experiment EP3 used completely unseen sentences. Table 7.1 summarizes the classification accuracy of the naïve Bayesian network classifier and rule-based classifier (in square parenthesis). For naïve Bayesian network classifier, the false alarms in Experiments EP1, EP2 and EP3 were 10.6%, 8.7% and 13.7%, respectively, showing that the classifier is able to keep the false alarms to a relatively low level. In addition, the classification accuracies in the three experiments were 94.4%, 89.1% and 88.1%, respectively, showing that the classifier is also robust to unseen samples and sentences. The corresponding results for the rule-based classifier were 6.9%, 7.7% and 11.4% (false alarms), and 87.3%, 88.0% and 83.1% (detection accuracies), respectively. Generally, the two sets of results are comparable; the naïve Bayesian classifier yielded better detection accuracy though with somewhat more false alarms compared to the rule-based classifier.

For further comparison between the rule-based scheme and the Bayesian network scheme, we devised a set of rules shown in Table 7.2, which used the same features as the Bayesian network, i.e. **maxAng**, **minVel**, **normVel** and **dirAng** (Experiment RB2). Comparative results are given in Table 7.1 (in round parenthesis). Although the detection accuracies obtained in RB2 are better in EP1, EP2 and EP3 compared to those in NB and RB1, there is a significant increase in false alarm rates, and hence the simplified rule-based scheme is not viable. The

Table 7.1: Classification accuracies of Experiment NB, Experiment RB1 (in square parenthesis) and Experiment RB2 (in round parenthesis).

Experiment	EP1 (Seen sentence and sample)	EP2 (Seen sentence, unseen sample)	EP3 (Unseen sentence and sample)
Total no of points	330	674	992
True boundary points	142	284	379
Detected points	134, 94.4% [124, 87.3%] (134, 94.4%)	253, 89.1% [250, 88.0%] (268, 94.4%)	334, 88.1% [315, 83.1%] (349, 92.1%)
False alarms	20, 10.6% [13, 6.9%] (53, 28.2%)	34, 8.7% [30, 7.7%] (102, 26.2%)	84, 13.7% [70, 11.4%] (212, 34.6%)
Missed points	8, 5.6% [18, 12.7%] (8, 5.6%)	31, 10.9% [34, 12.0%] (16, 5.6%)	45, 11.9% [64, 16.9%] (30, 7.9%)

rule-based classifier in RB2 detects the true segmentation points about 8% and 4% better than the rule-based and naïve Bayesian network classifiers in RB1 and NB, respectively, but it is about 21% and 19% worse in its ability to discard false alarms, respectively.

Table 7.2: Formulated rules.

Rule	Description

Rule 1	§*if* (**minVel** = TRUE) *and* (**maxAng** = TRUE),
	detection = TRUE POINT
	else, check Rule 2

Rule 2	*if* (**normVel** $<=$ T_5 *and* **dirAng** $>=$ T_6)
	or (**dirAng** $>=$ T_7 *and* **normVel** $<=$ T_8),
	detection = TRUE POINT
	else detection = FALSE ALARM

note: T_i, i = 5,6,7,8, are the same thresholds as in Table 4.2.

§the condition "(**minVel** = FALSE) *and* (**maxAng** = FALSE)" will not occur.

Between the rule-based classifier of Table 4.2 and the naïve Bayesian network classifier, we chose the latter for our final segmentation scheme, as it dispensed with the need to manually set several thresholds that are required in the rule-based scheme. Using the naïve Bayesian network classifier to classify all samples of each of the 25 sentences, and the voting algorithm of Section 4.3.1.4 to consistently specify the final segmentation points for each set of sentences, the results in Table 7.3 were obtained, with an overall detection accuracy of 92.5%.

Table 7.3: Final classification accuracies for 25 sentences.

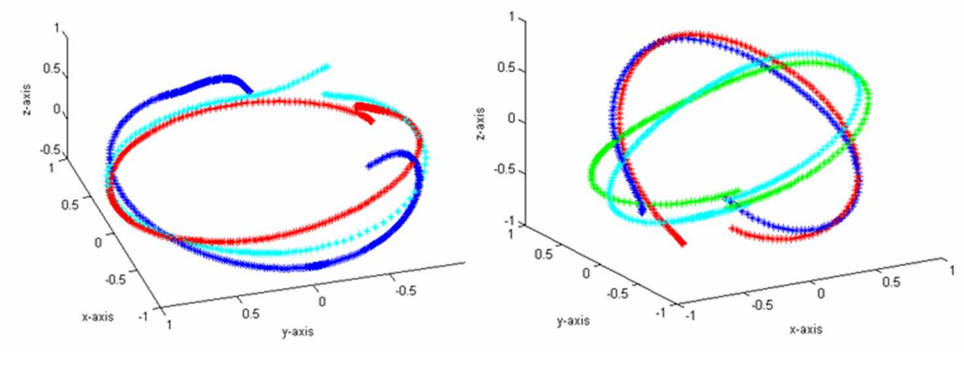
Category	No of points
Labeled true boundary points	133
Detected true boundary points	123
False alarms	9
Missed points	10

7.3.2 Phoneme Transcription

For purposes of comparison, we obtained phoneme transcriptions in two different ways. In Experiment PT1, the transcription process was manual. For this experiment, an expert signer specified the trajectory segments in one sample of each of the 25 sentences according to sign linguistics, in conjunction with the initial segments obtained at points of velocity minima and/or maxima of directional angle change, and the video of the signer as visual aid. Based on this collective information, the expert signer identified 173 segments consisting of 84 sign segments in the 25 sentences.

As a basis of comparison for the automatic phoneme transcription procedure of Section 4.3.2, the expert signer also manually transcribed these 84 sign segments into phonemes by visual observation. The video was used together with an ASL dictionary for manual transcription. This manual approach yielded 33 phonemes, and when the same 84 sign segments were automatically transcribed by the procedure of Section 4.3.2 (Experiment PT2), we obtained 36 phonemes.

The phoneme clusters obtained by both approaches were checked by plotting the trajectories of cluster members. We observed that the clusters obtained by the automatic procedure were generally more consistent and the cluster members were closer in appearance. On the other hand, some phoneme clusters specified by manual transcription were poorly formed. This can be expected as it is difficult to maintain consistency in manual transcription. Also, relying on the video clips during this process could have led to errors when there were visual occlusions. In the automatic transcription process, the PCA process separates the segments into lines, or planar curves and circles, and each feature of these categories is individually clustered. This simplifies checking the validity of the clusters as the number of clusters obtained for each feature is greatly reduced. Figures 7.1(a) and 7.1(b) show one of the clusters (phonemes) obtained by the automatic and manual phoneme transcription processes, respectively. It can be seen that the cluster formed by automatic phoneme labeling is more consistent. Another attractive benefit of automatic phoneme transcription is the significant reduction in time and labor as compared to manual transcription.



(a) Automatic transcription. **(b) Manual transcription.**

Figure 7.1: Clusters obtained (trajectories are normalized).

From this part of the experiment, we found that the naïve Bayesian network classifier yielded the best performance. We have also verified that the automatic phoneme transcription procedure yields comparable number of sign phonemes when compared to the manual procedure. In addition, the automatic transcription procedure yielded more consistent clusters showing promising performance. Hence, we adopted the naïve Bayesian network scheme for segmenting training sentences and the PCA-based automatic transcription procedure for transcribing the movement phonemes used in the subsequent experiments. More details for the automatic segmentation algorithm and phoneme transcription works can be found in [83, 85].

7.4 Subsystem 2: Experiments and Results

All the samples from the eight signers as described in Section 7.2, consisting of 74 distinct sentences and 107-sign vocabulary were used for the experiments to evaluate sub-segment labeling. A total of 47086 sub-segments were obtained from the initial segmentation algorithm. The training and testing were done using a full round robin procedure by leaving one signer out as an unseen signer (i.e. signer whose data was not used for training) each time. We also used 80% of the data from seven signers for training and the remaining 20% as unseen samples from seen signers for testing. We first trained CRFs and SVMs to classify the sub-segments and investigated their performance independently, followed by the Bayesian network fusion scheme.

7.4.1 Results with Conditional Random Fields

Based on the extracted state features and transition features, we trained a linear-chain CRF, where the features listed in Tables 5.3 and 5.4 were used to specify a set of feature functions ($func1, func2, \ldots, funcN$), examples of which are shown in Table 7.4. If N_c is the number of symbols for each feature and L_c is the number of output classes, then $(N_c \times L_c)$ state feature functions and $(N_c \times L_c \times L_c)$ transition feature functions can be specified. The important settings for training and testing the CRF are summarized in Table 7.5. We conducted three different sets of experiments. The first experiment was to estimate $\hat{k}$, the number of discrete symbols used for each feature. The second experiment evaluated the performance and effect of L1 and L2-norm regularization methods on the classification rates. The last set experiments used the best performing CRF to obtain the final CRF classification result.

Table 7.4: Example of CRF state feature functions.

$func1$ = if (output = *SIGN* and **hand_start** = "1")	return 1 else return 0.
$func2$ = if (output = *ME* and **hand_start** = "1")	return 1 else return 0.
$func3$ = if (output = *SIGN* and **hand_start** = "2")	return 1 else return 0.
$func4$ = if (output = *ME* and **hand_start** = "2")	return 1 else return 0.
.	
.	
.	
.	

Table 7.5: Settings used for CRFs.

Setting	Description
Output labels	Two class: *SIGN* and *ME*
No. of state features	25
No. of transition features	6
Optimization	Quasi-newton algorithm (LBFGS)
Regularization	L2-norm
Decoding	Viterbi algorithm

7.4.1.1 Determination of $\hat{k}$ Discrete Symbols

The original real-valued features based on handshape, movement, orientation and location components were converted to discrete symbols by k-means before being used as inputs to CRFs. We used only the training data to find the clusters, and the trained centroids were labeled as $1, 2, \ldots, \mathcal{C}$. All cluster members were assigned the same numeric symbol as their cluster centroid. For test data, the feature vector was given a number symbol corresponding to its closest cluster. To determine the optimal number of clusters $\hat{k}$ for each feature, we searched in a range of potential values based on the number specified by linguists. For every k, we trained a CRF to classify the sub-segments as *SIGN* or *ME* based only on the single feature using the training data, and selected the $\hat{k} = k$ which yielded the best accuracy. For example, if there were about 40 handshapes defined by linguists, we searched for $\hat{k}$ in the range of 30-80 in steps of 5 or 10 for the state or transition features related to handshape. Table 7.6 shows the best $\hat{k}$ values for each feature and their corresponding classification results.

Table 7.6: Best $\hat{k}$ for state and transition features.

Feature	$\hat{k}$	Accuracy (%)	Feature	$\hat{k}$	Accuracy (%)
hand_start	70	65.8	**mov_dom**	60	66.8
hand_end	70	64.8	**mov_start**	60	65.5
hand_msdif	70	67.6	**mov_end**	60	65.6
orien_start	50	66.9	**arc_length**	10	59.1
orien_end	50	64.5	**diff_strhand**	70	61.2
orien_msdif	80	64.3	**diff_strorien**	80	66.7
loc_mean	50	64.4	**diff_mloc**	60	67.8
loc_start	50	67.5	**diff_mdom**	70	71.6
loc_end	50	64.4	**comb_arc**	10	59.8

7.4.1.2 L1-Norm and L2-Norm Regularization

Regularization was applied to CRFs to avoid overfitting. We used the L1-norm (5.6) and L2-norm (5.7) and compared their results. L1-norm is typically used to obtain sparse parameters making the problem more interpretable and computationally manageable. However, the main problem with L1-norm is that it is not differentiable at zero, and the standard gradient-based optimization algorithms such as the LBFGS quasi-Newton methods cannot be applied. Rather, it requires numerical optimization which often leads to instability in computation. We randomly selected a subset of data from one signer and split the data into training and testing samples. We used all the features as described in Tables 5.3 and 5.4 to train a CRF to classify the sub-segments as *SIGN* or *ME*. Table 7.7 shows the classification results obtained by using the L1-norm and L2-norm by varying C, a free parameter which weights the penalty term.

Table 7.7: Performance of L1-norm and L2-norm.

	Classification accuracy (%)					
	L1-norm			L2-norm		
C	Training sample	Testing sample	No. of features	Training sample	Testing sample	No. of features
0.001	57.4	58.8	8	76.6	74.0	497326
0.01	82.1	80.4	7049	99.0	87.3	497326
0.20	83.8	81.8	14775	99.8	87.5	497326
0.40	89.1	85.8	34434	100.0	87.4	497326
0.60	88.9	85.7	56571	100.0	87.4	497326
0.80	89.3	85.8	91640	100.0	87.2	497326
1.00	91.4	86.7	114196	100.0	87.3	497326
5.00	94.2	87.3	493646	100.0	87.2	497326
10.0	94.5	87.0	494563	100.0	87.1	497326

From Table 7.7, it is observed that the L2-norm performs better than the L1-norm at the expense of using more features. The L2-norm accuracy is consistently around 87% on testing samples, while the L1-norm accuracies increase with the number of features retained. As this gets closer to the total number of features, the accuracy increases close to that of the L2-norm. Though the L1-norm offers a tradeoff between accuracy and computational cost, its accuracies are below that of the L2-norm. As we did not face any computational problems in working with the total number of features, we chose to use the L2-norm.

7.4.1.3 Classification with CRFs

We conducted two different experiments for classifying the *SIGN* and *ME* sub-segments. All sentences were used for this part of the experiment. Experiment C1 used 70% of the data from each signer and tested on the remaining data from the same signer. This was to evaluate the performance of the classifier on unseen samples from the same signer used to train the model. Experiment C2 evaluated the performance on samples from an unseen signer. For this, we used a full round robin procedure by leaving one signer out in each round as described in Section 7.4. Tables 7.8 and 7.9 summarize the classification results obtained by CRFs in Experiments C1 and C2, respectively.

Table 7.8: Experiment C1 (single signer) - Classification of *SIGN* and *ME*.

	Classification accuracy with CRFs and SVMs (in parenthesis) (%)	
Signer	Seen signer, seen sample	Seen signer, unseen sample
S_1	100.0 (99.5)	92.9 (94.7)
S_2	100.0 (99.3)	92.1 (94.4)
S_3	100.0 (98.4)	87.2 (89.9)
S_4	100.0 (99.2)	88.0 (92.2)
S_5	100.0 (99.1)	91.1 (92.8)
S_6	100.0 (99.3)	92.3 (95.3)
S_7	100.0 (99.8)	92.8 (96.3)
S_8	100.0 (99.5)	91.4 (94.3)
S_{ave}	100.0 (99.3)	91.0 (93.7)

A good value of C was determined experimentally as $C = 0.085$; however, the classification results were not very sensitive to the parameter C. Generally, accuracy was consistently good for unseen samples from seen signers and averaged 91%. An average classification accuracy of 85.7% was obtained for unseen signers in Experiment C2. This shows that the trained CRFs can generalize quite well

Table 7.9: Experiment C2 (multiple signer) - Classification of *SIGN* and *ME*.

	Classification accuracy with CRFs and SVMs (in parenthesis) (%)		
Round	Seen signer, seen sample	Seen signer, unseen sample	Unseen signer unseen sample
R_1	99.4 (95.6)	90.4 (92.2)	85.0 (86.0)
R_2	99.5 (95.9)	90.5 (92.4)	87.1 (87.5)
R_3	99.5 (96.1)	91.5 (93.0)	86.3 (87.0)
R_4	99.5 (96.0)	91.0 (92.9)	82.5 (83.8)
R_5	99.5 (95.9)	90.6 (92.6)	86.0 (84.2)
R_6	99.5 (95.7)	90.8 (92.3)	84.8 (84.5)
R_7	99.4 (95.6)	90.4 (92.0)	87.6 (86.6)
R_8	99.4 (95.8)	90.4 (92.3)	86.0 (85.3)
R_{ave}	99.5 (95.8)	90.7 (92.5)	85.7 (85.6)

to data from new signers.

7.4.2 Results from Support Vector Machines

The features listed in Table 5.5 were concatenated to form 126-D real-valued feature vectors as inputs to the SVMs. The elements of the feature vectors were normalized to have zero mean and unit variance. We used Gaussian radial basis functions as the kernels for the SVMs. The regularization parameter $\tilde{C}$ was tuned experimentally and we used $\tilde{C} = 5$. We conducted two experiments similar to the CRF classification experiments as described in Section 7.4.1.3. The SVM classification results are also summarized in Tables 7.8 and 7.9 (in parenthesis).

The classification results of SVMs and CRFs are comparable and consistent. SVMs performed about 2% better than CRFs for unseen samples from seen signers. For unseen signers, both classifiers yielded rather consistent classification accuracies for all eight rounds with CRFs and SVMs yielding average accuracies of 85.7% and 85.6%, respectively. The main disadvantage of SVMs as compared to CRFs is that they require longer training time because the feature vectors are high dimensional.

Generally, SVMs treat every sub-segment as an individual isolated input while CRFs encode the transition characteristics of the sub-segments in a sentence. The errors made by both approaches are different although they show similar classification accuracies. Figure 7.2 shows an example of the CRF and SVM classification outputs for a sentence COME WITH ME made by an unseen signer. Except for identical errors made in sub-segment S_{17}, the errors from the SVM and CRF were different. The CRF made an error in S_8 while the SVM detected the sub-segment correctly. On the other hand, when the SVM made errors in S_7 and S_{15}, the CRF classified the sub-segments correctly. Hence, we conjectured that the classification accuracy may be further improved by combining the two experts if they are able to complement each other.

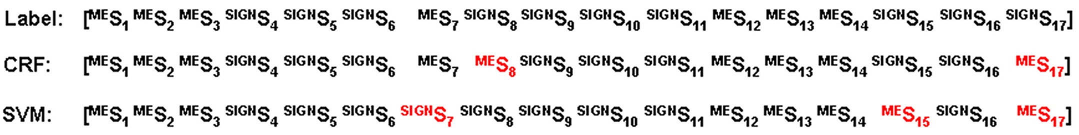

Label: [MES1 MES2 MES3 SIGNS4 SIGNS5 SIGNS6 MES7 SIGNS8 SIGNS9 SIGNS10 SIGNS11 MES12 MES13 MES14 SIGNS15 SIGNS16 SIGNS17]
CRF: [MES1 MES2 MES3 SIGNS4 SIGNS5 SIGNS6 MES7 MES8 SIGNS9 SIGNS10 SIGNS11 MES12 MES13 MES14 SIGNS15 SIGNS16 MES17]
SVM: [MES1 MES2 MES3 SIGNS4 SIGNS5 SIGNS6 SIGNS7 SIGNS8 SIGNS9 SIGNS10 SIGNS11 MES12 MES13 MES14 MES15 SIGNS16 MES17]

Figure 7.2: CRF and SVM outputs for the sentence COME WITH ME.

7.4.3 Fusion Results with Bayesian Networks

We repeated Experiment C2 by using the Bayesian network shown in Figure 5.7. As observed from Table 5.6, we need the error outputs of the CRF and SVM for training the Bayesian Network. However, the classification accuracies for the training samples with CRF and SVM were 99.5% and 95.8%, respectively. Thus, there were too few error samples from the CRF to train the Bayesian network. Hence, we lowered the value of the parameter C in (5.7) to $C = 0.015$, to obtain 4-5% errors (Table 7.10 summarizes the classification results thus obtained) and trained the Bayesian network. The real-valued features were quantized as described in Table 5.6. The conditional probability table (CPT) of the discrete nodes were learned by maximum likelihood estimation. During testing, the nodes **crfErr**, **svmErr** and **fLabel** were inferred based on the input observations. The final classification accuracy shown in Table 7.11, is improved compared to CRF and SVM results, for both unseen samples from seen signers as well as unseen signers. For the unseen samples from seen signer, an average improvement of 2.4% was obtained when compared to the CRF, and a marginal improvement of 0.6% was obtained when compared to the SVM. However, for unseen signers, the Bayesian network gave an average improvement of about 2.5% in classification accuracy as compared to both CRF and SVM. More importantly, the Bayesian network was more consistent across the unseen signers. From Table 7.11, it is observed that the worst accuracy of 86.9% was obtained in $R4$ when Signer 4 was tested as a new signer. In comparison, if only the CRF or SVM was used individually, the result for $R4$ would have been as low as 82.5% (CRF) or 83.8% (SVM). Accuracy for unseen signers is important for the later part of our work when we need to incorporate the classified *SIGN* and *ME* labels into the final recognition scheme.

Table 7.10: Classification with less overfitted CRFs.

	Classification accuracy (%)		
Round	Seen signer, seen sample	Seen signer, unseen sample	Unseen signer, unseen sample
R_1	95.1	89.3	84.2
R_2	94.9	88.8	87.3
R_3	95.4	90.0	84.1
R_4	95.4	89.7	82.9
R_5	95.0	88.9	85.0
R_6	95.1	89.2	84.1
R_7	95.0	89.2	87.2
R_8	95.0	89.2	85.8
R_{ave}	95.1	89.3	85.1

Table 7.11: Classification with Bayesian network.

	Classification accuracy (%)		
Round	Seen signer, seen sample	Seen signer, unseen sample	Unseen signer, unseen sample
R_1	98.8	92.8	87.4
R_2	98.9	92.9	89.5
R_3	99.0	93.7	88.9
R_4	98.9	93.2	86.9
R_5	98.9	93.2	87.4
R_6	98.8	92.9	87.3
R_7	98.8	92.7	89.5
R_8	98.8	93.1	88.3
R_{ave}	98.9	93.1	88.2

Here, we also analyzed the errors made by the Bayesian network. Among the errors, the number of false alarms (*ME* sub-segments classified as *SIGN*) is slightly more than the number of missed *SIGN*s. Table 7.12 presents the proportion of false alarms and misses in the errors obtained from all the experiments, indicating an average of $6 - 12\%$ more false alarms than misses. In our work, missed *SIGN*s are more of a concern as they entail permanent loss of information; false alarms can still be dealt with in the decoding algorithm. For analysis, we categorized the errors into four groups, viz. 1) left edge error ($\mathcal{E}_1$), 2) right edge error ($\mathcal{E}_2$), 3) single segment error ($\mathcal{E}_3$) and 4) random error ($\mathcal{E}_4$). Figure 7.3 illustrates these errors and Table 7.13 shows the proportion of the different errors obtained by the Bayesian network. Often, the random errors create complexity for the decoder as they split a segment into arbitrary fragments which may change the intrinsic characteristics of the original sign. Single segment errors cause complete loss of the sign. The edge errors are less severe as they imply partial loss of information and there is still a possibility of correct decoding. Fortunately, from Table 7.13, the percentage of random errors is relatively small, and most of the errors appear at the edges of the segments.

Table 7.12: Error analysis of false alarms and misses from the Bayesian network.

	Number of false alarms and misses in the errors (%)					
Round	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer unseen sample	
	False alarm	Misses	False alarm	Misses	False alarm	Misses
R_1	57.4	42.6	52.9	47.1	57.9	42.1
R_2	55.3	44.7	54.2	45.8	54.6	45.4
R_3	57.1	42.9	52.4	47.6	58.9	41.1
R_4	57.9	42.1	53.6	46.4	56.7	43.3
R_5	52.9	47.1	53.4	46.6	72.6	27.4
R_6	55.0	45.0	52.6	47.4	40.0	60.0
R_7	56.2	43.8	53.0	47.0	53.3	46.7
R_8	55.9	44.1	52.6	47.4	45.9	54.1
R_{ave}	56.0	44.0	53.1	46.9	55.0	45.0

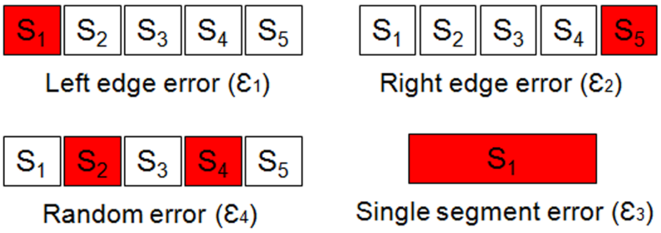

S_1 S_2 S_3 S_4 S_5
Left edge error (ε_1)
S_1 S_2 S_3 S_4 S_5
Right edge error (ε_2)
S_1 S_2 S_3 S_4 S_5
Random error (ε_4)
S_1
Single segment error (ε_3)

Figure 7.3: Error types.

7.5 Subsystem 3: Experiments and Results

For the sign recognition experiments, we used all the samples from eight signers as described in Section 7.2, consisting of 74 distinct sentences from a 107-sign vocabulary, comprising in total 2393 sentences and 10852 sign instances. The trained two-layered CRF classifier was used to recognize the continuously signed sentences. We tested the classifier in a full round robin procedure by leaving one signer out as an unseen signer in each round. For training, we selected 80% of the data randomly from each of the seven signers and used the remaining 20% as

Table 7.13: Error types.

	Proportion of error types (%)											
Round	Seen signer, seen sample				Seen signer, unseen sample				Unseen signer unseen sample			
	$\mathcal{E}_1$	$\mathcal{E}_2$	$\mathcal{E}_3$	$\mathcal{E}_4$	$\mathcal{E}_1$	$\mathcal{E}_2$	$\mathcal{E}_3$	$\mathcal{E}_4$	$\mathcal{E}_1$	$\mathcal{E}_2$	$\mathcal{E}_3$	$\mathcal{E}_4$
R_1	28.8	46.6	6.1	18.4	40.2	38.8	5.1	15.9	31.4	39.2	5.8	23.5
R_2	25.7	48.0	6.1	20.3	39.7	40.3	6.3	13.7	33.7	39.6	5.5	21.2
R_3	29.0	53.1	3.4	14.5	37.8	44.5	3.0	14.7	29.5	31.9	5.8	32.9
R_4	27.4	51.9	3.0	17.8	39.7	39.0	5.1	16.2	40.7	39.9	14.6	4.9
R_5	26.7	51.6	3.7	18.0	38.7	40.2	6.7	14.4	44.9	37.8	4.1	13.3
R_6	28.6	50.9	5.6	14.9	39.2	40.3	5.7	14.8	33.8	37.0	10.4	18.8
R_7	29.0	45.6	4.7	20.7	39.7	41.1	4.5	14.8	30.9	42.7	9.3	17.1
R_8	26.3	52.5	5.0	16.3	41.1	39.3	5.3	14.4	43.2	30.0	4.5	22.2
R_{ave}	27.7	50.0	4.7	17.6	39.5	40.4	5.2	14.9	36.0	37.3	7.5	19.2

unseen samples from seen signers for testing. The following sections describe the experiments to verify the various subsystems and the final classifier.

7.5.1 Phoneme and Subphone Extraction

The phonemes of the four components were defined and extracted mainly for training the linear-CRFs in the four parallel channels at the phoneme level. The transcription procedures described in Chapter 4 were used to define the component phonemes using the training data in each round of the experiments. The phonemes for the movement component were obtained by using the transcription procedure with PCA-based representations. For the other three components, we randomly selected 5000 segments (i.e. $N_p = 5000$) for AP clustering. Features from the starting and ending intervals of each segment were concatenated, resulting in 5000 feature vectors for AP clustering. The "preference" parameter in the AP algorithm affects the number of phonemes obtained. It was set to $\rho\tilde{s}_{min}$, where $\tilde{s}_{min}$ is the minimum value of pairwise data point similarities based on Euclidean distance and ρ is a scaling constant for parameter tuning. The

AP clustering algorithm was run with different scale factors for 10 sets of 5000 randomly chosen segments to find the best number of clusters for representing the phonemes. We used the exemplars obtained from AP clustering to initialize the k-means algorithm for final clustering. The scale factors which led to the smallest errors from k-means were chosen. The different scale factors that were tried for handshape, orientation and location were from the sets $\{1.50, 1.60, 1.70\}$, $\{3.50, 4.00, 4.50\}$ and $\{1.20, 1.30, 1.40, 1.50, 1.60, 1.70\}$ from which we chose 1.70, 4.50 and 1.30, respectively.

The subphones for each component were extracted by using the method described in Section 6.2.1 which is similar to the phoneme extraction procedure for the static components. We randomly selected 1000 segments ($N_q = 1000$) and clustered them by AP. Each segment was divided into 10 intervals and the mean feature vector was obtained for each interval. Thus, the total number of feature vectors for clustering the subphones using AP was 10000. The different scale factors tried for handshape, movement, palm orientation and location were from the sets $\{1.25, 1.50, 1.75\}$, $\{0.50, 0.75, 1.00, 1.25, 1.50, 1.75\}$, $\{1.00, 1.25, 1.50, 1.75, 2.00\}$, and $\{0.30, 0.40, 0.50, 0.60\}$, respectively. Following the procedure in phoneme definition, the final scale factors were chosen as 1.25, 1.00, 2.00, 0.30 for handshape, movement, orientation and location, respectively.

The “best” number of phonemes and subphones was obtained experimentally and Table 7.14 summarizes the number of phonemes and subphones selected for the components in each round. The numbers of phonemes and subphones that we obtained were comparable to those defined by linguists.

Table 7.14: Number of phonemes and subphones for handshape, movement, orientation and location components.

	Number of phonemes				Number of subphones			
Round	H	M	O	L	H	M	O	L
R_1	33	39	30	25	40	50	41	46
R_2	30	39	32	25	40	48	43	43
R_3	30	39	30	28	39	48	40	47
R_4	31	39	31	30	46	52	40	48
R_5	32	40	29	28	41	51	43	43
R_6	31	39	33	28	38	52	40	42
R_7	35	39	32	25	40	49	40	46
R_8	37	41	30	26	37	53	41	45

7.5.2 Sign vs. Non-Sign Classification by SVM

The procedure of Section 6.3.2 was used to generate the $SIGN_\kappa$, $\kappa = 1, \ldots, 107$ and *UNCLASSIFIED* training segments from the training sentences, and test segments from the test sentences. The features listed in Table 6.1 were concatenated to form 126-D real-valued feature vectors for input to two-class SVMs with Gaussian radial basis functions as the kernels. The elements of the feature vectors were normalized to zero mean and unit variance. The SVMs provided probability outputs and a threshold of 0.5 was used to differentiate the $SIGN_\kappa$, $\kappa \in \{1, \ldots, 107\}$ and *UNCLASSIFIED* classes. Different cost functions $\tilde{C}$ were tuned experimentally for each SVM.

Table 7.15 shows the overall classification results for all the eight rounds of experiments. We obtained average accuracies of 94.8% and 93.4% for unseen samples from seen signers and unseen signers respectively. Though these test samples do not represent the actual pool of *UNCLASSIFIED* and $SIGN_\kappa$ segments that will be formed during the decoding procedure, it is a subset of the possible samples, and is a promising indicative result. These SVMs were integrated into the decoding algorithm and their functionality was further verified in the recognition experiments.

Table 7.15: Overall sign vs. non-sign classification accuracy with two-class SVMs.

	Classification accuracy (%)		
Round	Seen signer, seen sample	Seen signer, unseen sample	Unseen signer, unseen sample
R_1	97.4	95.0	94.2
R_2	97.4	95.0	92.6
R_3	96.8	94.7	92.5
R_4	97.5	95.1	93.6
R_5	97.1	95.0	93.3
R_6	96.6	94.6	93.7
R_7	96.6	94.4	93.5
R_8	96.7	94.5	93.8
R_{ave}	97.0	94.8	93.4

7.5.3 Continuous Sign Recognition Results

The proposed recognition framework was designed to be robust to variations in sign language from the feature level to the sign level, as described in Chapter 1. Since the training of CRFs is supervised, we need to address word order variations in sentences, i.e. test sentences which have different word order from training sentences. To deal with this type of variation, identified pairs of signs that tended to be swapped in order and duplicated the same weights for the sign label transition features of these sign pairs such that the sign pairs were equally likely to transit from one to the other. This strategy was applied to the sign level CRF only. We conducted several progressive experiments to evaluate the two-layered CRF-based classifier. The settings for training the phoneme level and sign level CRFs are summarized in Table 7.16. The recognition performance of the continuously signed sentences was evaluated based on substitution, deletion and insertion errors. We used “recall” and “precision” to measure recognition

Table 7.16: Settings used for training phoneme and sign level CRFs.

Setting	Phoneme level	Word level
Output label	The phonemes defined for each component by the transcription procedure	107 signs
No. of state features	30	10
No. of transition features	1	1
Optimization	Quasi-newton algorithm (LBFGS)	
Regularization	L2-norm	
Decoding	Modified segmental decoding algorithm	

accuracy, computed as

$$\text{Recall} = \frac{N_c}{N_c + N_s + N_d}, \tag{7.1}$$

$$\text{Precision} = \frac{N_c}{N_c + N_s + N_i}, \tag{7.2}$$

where N_c, N_s, N_d, N_i are the number of correct classification, substitution, deletion and insertion errors, respectively.

For a basic comparison with our method, we trained standard left to right HMMs to recognize the 107 signs for each experiment. The observation sequences to train the HMMs were obtained by concatenating the normalized vectors (described in Section 5.5.1) from all the components to form 31-D feature vectors. We modeled each sign with 3-5 states and each state was represented by a single Gaussian with full covariance matrix. The Viterbi algorithm was used to decode the sign sequences. For training, the transition probabilities were set to be equiprobable except for the invalid transitions, whose probabilities were set to zero. We divided the sign segments from the training data into 3-5 sub-segments with equal arc length and used them to initialize the Gaussian parameters in each state. We attempted to use the same approach used in the CRF-based framework to tackle the word order issue in the sentences. However, the performance was not as good as the naturally trained parameters. Hence, we did not adjust the HMM parameters. We used standard HMMs as benchmarks for all the experiments conducted, rather than sophisticated HMM-based algorithms, as work in the literature shows that standard HMM performance does not deviate substantially from the latter. For example, Vogler [153] compared their PaHMMs to standard HMMs and obtained sign recognition accuracies of 94.23% and 93.27%, respectively.

7.5.3.1 Clean Sign Segment Recognition

To systematically evaluate the two-layered CRF framework, we first checked performance by assuming known boundary points of the segments in test sequences, i.e. a sequence of isolated sign segments, obtained after discarding the movement epenthesis segments. The features were extracted directly from the sign segments. The decoding procedure is straightforward for purely sign segment sequences. The component phonemes were decoded independently in the four parallel channels and together with the arc length labels of the sign segments, the phoneme label outputs of the four components were concatenated at every time instant. There were input to the sign level CRF and the standard CRF decoding algorithm was applied to obtain the sign sequences. The regularizing parameters C were tuned experimentally in the range $C = 0.1 - 0.8$ for both phoneme and sign level CRFs during training. Table 7.17 shows the recognition results for the test sentences consisting of sequences of clean segments. These results serve to indicate the upper bound for the accuracy of our CRF-based classifier, assuming perfect segmentation. We obtained an average accuracy of 98.0% for unseen samples from seen signers and 90.8% for samples from unseen signers indicating good performance and good generalization. Table 7.18 presents sign recognition accuracy based on only one component. As expected, the accuracies are much lower if only one component is used especially for unseen signers. This demonstrates the importance of using all four components for sign language recognition, though in our work, handshape seems to be the more influential among the four components.

Table 7.17: Recognition accuracy for clean segment sequences using two-layered CRFs.

	Recognition accuracy (%)		
Round (R_i)	Seen signer, seen sample (SS)	Seen signer, unseen sample (SU)	Unseen signer, unseen sample (UU)
R_1	100.0	97.8	91.8
R_2	100.0	97.5	90.8
R_3	100.0	98.4	88.9
R_4	100.0	97.8	92.8
R_5	100.0	97.4	96.1
R_6	100.0	98.0	88.0
R_7	100.0	98.3	89.1
R_8	99.9	98.6	89.1
R_{ave}	100.0	98.0	90.8

Table 7.18: Recognition accuracy based on individual components.

	Recognition accuracy (%)											
	Handshape			Movement			Orientation			Location		
R_i	SS	SU	UU	SS	SU	UU	SS	SU	UU	SS	SU	UU
R_1	91.5	86.8	62.2	89.3	68.6	48.1	92.7	83.9	49.9	93.1	82.5	40.5
R_2	91.6	86.4	67.0	90.2	70.6	27.8	94.9	84.5	10.5	93.6	81.6	23.6
R_3	91.2	85.8	67.5	89.7	69.1	46.7	93.0	84.5	44.5	93.2	84.1	36.8
R_4	91.0	86.0	74.3	90.2	70.4	50.5	92.9	84.1	56.2	94.5	82.7	55.2
R_5	90.8	86.1	85.4	89.1	67.6	45.6	90.2	80.6	61.7	91.8	81.3	50.2
R_6	90.5	85.9	64.7	89.7	69.4	50.5	92.1	83.6	47.9	93.2	81.1	45.5
R_7	91.5	87.1	22.3	89.5	69.2	47.2	93.1	83.6	40.2	92.2	81.1	40.9
R_8	91.9	87.8	58.7	89.3	70.6	56.5	92.2	82.4	44.8	92.7	83.3	47.0
R_{ave}	91.3	86.5	62.8	89.6	69.4	46.6	92.6	83.4	44.5	93.0	82.2	42.5

7.5.3.2 Recognition of Sign Sentences with Unknown Boundary Points

The next set of experiments considered continuously signed sentences which were automatically segmented, and movement epenthesis sub-segments in all sentences were manually discarded. This left only sign segments in the testing sentences, though with unknown segment boundary points. We conducted three experiments to verify the proposed decoding algorithm as described in Sections 6.3.1 and 6.3.2 (without skip states). This is sufficient to deal with sign sentences that do not contain movement epenthesis; the main task for the decoding algorithm is to merge the sub-segments and recognize the sign sequences correctly without the need to skip any sub-segments. The first two experiments were based on trained CRFs while the third used HMMs (trained as described in Section 7.5.3). For the first experiment, we used the modified segmental CRF decoding procedure without two-class SVMs and skip states, while two-class SVMs were used in the decoding procedure for the second experiment.

Table 7.19: Recognition accuracy with modified segmental CRF decoding procedure without two-class SVMs and skip states.

Recognition accuracy (%)						
	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer, unseen sample	
Round	Recall	Precision	Recall	Precision	Recall	Precision
R_1	98.4	98.4	96.7	96.5	87.7	91.1
R_2	98.9	99.1	97.2	97.3	88.8	91.0
R_3	98.7	98.7	96.5	97.4	84.2	87.1
R_4	98.3	98.5	96.3	96.2	88.7	89.4
R_5	98.2	98.2	96.3	96.5	93.8	93.5
R_6	98.2	98.4	96.7	96.9	87.6	88.0
R_7	98.6	98.7	96.9	96.8	88.4	88.4
R_8	98.6	98.6	96.9	97.2	84.2	87.6
R_{ave}	98.5	98.6	96.7	96.9	87.9	89.5

The recognition results shown in Tables 7.19 and 7.20 for the first and second

Table 7.20: Recognition accuracy with modified segmental CRF decoding procedure with two-class SVMs but without skip states.

	Recognition accuracy (%)					
	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer, unseen sample	
Round	Recall	Precision	Recall	Precision	Recall	Precision
R_1	98.9	98.4	97.4	96.9	89.8	91.3
R_2	99.3	99.2	97.9	97.4	89.7	92.0
R_3	98.9	98.4	97.8	97.8	86.4	89.6
R_4	98.8	98.5	97.2	96.9	88.4	87.6
R_5	98.1	97.9	97.4	96.9	93.9	93.0
R_6	98.2	98.1	97.2	97.2	89.4	88.1
R_7	98.9	98.5	97.2	96.0	88.9	89.3
R_8	98.7	98.1	97.8	97.3	86.7	89.1
R_{ave}	98.7	98.4	97.5	97.1	89.2	90.0

Table 7.21: Recognition accuracy with HMM-based approach.

	Recognition accuracy (%)					
	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer, unseen sample	
Round	Recall	Precision	Recall	Precision	Recall	Precision
R_1	99.0	98.9	98.9	98.3	77.8	62.6
R_2	99.0	99.0	99.2	98.4	53.4	57.7
R_3	99.3	98.7	99.3	97.2	77.7	54.5
R_4	99.0	99.0	99.1	97.6	78.7	62.0
R_5	98.8	99.0	98.2	96.9	77.9	57.8
R_6	98.7	99.1	98.2	96.1	66.2	48.4
R_7	97.7	99.5	97.4	97.5	56.7	65.3
R_8	99.5	98.9	99.6	98.3	68.0	50.1
R_{ave}	98.9	99.0	98.7	97.5	69.6	57.3

experiments are comparable. The modified segmental decoding algorithm with two-class SVMs performed slightly better and showed about 1% improvement on the recall rates for both unseen samples from seen and unseen signers. The recall rates of the modified segmental decoding algorithm with two-class SVMs were 97.5% and 89.2% for unseen samples from seen signer and unseen signers, respectively. These are close to the clean segment recognition accuracies of 98.0% and 90.8%, respectively. Table 7.21 shows the recognition performance with the standard HMM approach. The recognition accuracies are very good for unseen samples from seen signers and even outperforms our proposed framework slightly. However, when the HMM-based framework was tested with samples from unseen signers, the accuracy dropped drastically, yielding an average recall rate of 69.6% and precision accuracy of 57.3%. Our decoder outperformed it by 19.6% and 32.7% for recall and precision, respectively. This shows that as generative models, HMMs may not be good for generalization to new sign sequences from unseen signers.

We further tested the HMM-based framework by using a single signer, a protocol that has been widely used in many works in the literature. Table 7.22 summarizes the results for the eight individual signers. Good recognition accuracies for the unseen testing samples are obtained and this further shows that HMMs are good for recognizing sequences from subjects seen during training. Comparing results from Table 7.21 and Table 7.22, we observe an obvious drop

Table 7.22: HMM recognition accuracy with single signer.

Recognition accuracy (%)				
	Seen signer, seen sample		Seen signer, unseen sample	
Signer	Recall	Precision	Recall	Precision
S_1	99.3	94.5	99.1	92.8
S_2	98.5	96.7	95.5	96.3
S_3	97.3	94.1	95.5	91.1
S_4	100.0	100.0	99.6	98.8
S_5	96.0	92.4	95.1	89.9
S_6	94.7	90.2	91.1	86.3
S_7	96.0	93.1	93.9	90.5
S_8	95.4	91.6	93.4	87.4
S_{ave}	97.2	94.1	95.4	91.6

in recall rate from 98.7% to 95.4% and precision accuracy from 97.5% to 91.6%. This can be largely attributed to the decrease in the number of training samples. For the single signer experiment, the number of samples used to train the 107 signs is roughly seven times less than the number of samples used in the experiments where the training samples were obtained from seven signers. This again shows the sensitivity of HMMs to training sample size.

7.5.3.3 Recognition of Sentences with Movement Epenthesis

Thus far, all experiments considered recognizing signs from continuously signed sentences after manually discarding movement epenthesis segments. The last part of the experiments was to evaluate performance on the complete problem where movement epenthesis sub-segments may present in the test sequences due to automatic segmentation and sub-segment classification errors. Here, the complete decoding algorithm with all the proposed features is used.

Before we present the final results, we note the difficulty of applying the conventional HMM-based approach to this problem. As discussed earlier, one possible approach to deal with movement epenthesis segments is to model them explicitly, which is arguably wasteful. Another common approach allows movement epenthesis segments to be part of their adjacent sign segments. We tried this strategy to train HMMs for recognizing the 107 signs using samples from seven signers. However, it was difficult to train the HMMs correctly due to large variations in the movement epenthesis segments. Hence, we used a very simplified data set. We randomly selected five sentences from the 74 distinct sentences and trained HMMs to recognize these five sentences. We conducted two sets of experiments based on this simple data set, where we trained HMMs to recognize the five sentences with and without movement epenthesis. The HMMs modeled only signs and three states were used to represent each sign segment. For the first experiment, movement epenthesis segments were manually discarded and not used during training and testing. For the second experiment, we used training sequences consisting of both sign and movement epenthesis to train the HMMs. We started with training the HMMs using 80% of the samples from one signer and tested on the remaining unseen samples from the same signer. For single signer, results for the five sentences were consistent and we obtained 100.0% accuracy for both experiments using sentences with and without movement epenthesis. We conducted further experiments by adding samples from another six signers to train the HMMs, and Table 7.23 summarizes the results. In this case, the recall rate decreased. The decrease in recall rate is more obvious in the experiment with movement epenthesis showing the potential problems with the HMM-based approach. Though this data set is quite small and simple, the 9% drop in accuracy is quite large, and again indicates the inability of the HMM-based approach to handle variation from new signers.

Table 7.23: Recognition of five sentences with and without movement epenthesis using HMMs.

	Recognition accuracy (%)					
	SS		SU		UU	
	Recall	Precision	Recall	Precision	Recall	Precision
Without ME	97.3	98.7	96.1	98.3	95.0	81.0
With ME	91.0	100.0	91.1	98.8	91.3	81.9

For our proposed strategy, we started by investigating the effects of false alarms and misses obtained from the sub-segment classifier of Chapter 5. Two experiments were conducted, viz. 1) the outputs from the sub-segment classifier were used directly (Experiment D1) 2) the Bayesian network sub-segment classifier was tuned to recover more missed points at the expense of having more false alarms (Experiment D2). We discarded the sub-segments labeled as *ME* and used the remaining sub-segments in the sequences, assuming that they were correct, and hence, needed to be merged. Hence, we used our modified segmental CRF decoding algorithm without the two-class SVMs and skip states. For comparison, we also used the previously trained HMMs based on only sign data to decode the sequences. Tables 7.24 and 7.25 present the recognition accuracy of experiments D1 and D2, respectively, where HMM accuracies are given in parenthesis.

Table 7.24: Recognition accuracy for Experiment D1.

	Recognition accuracy with our approach and HMMs (parenthesis) (%)					
	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer, unseen sample	
R_i	Recall	Precision	Recall	Precision	Recall	Precision
R_1	98.7 (98.9)	98.2 (98.1)	95.7 (98.2)	95.2 (95.3)	84.2 (76.1)	85.7 (58.5)
R_2	98.2 (98.7)	98.1 (97.7)	95.7 (98.0)	95.3 (93.2)	78.7 (54.2)	82.5 (53.4)
R_3	98.3 (99.0)	97.9 (97.4)	95.6 (98.4)	95.6 (90.2)	77.3 (75.6)	83.8 (48.1)
R_4	97.8 (98.8)	97.7 (97.8)	95.5 (98.4)	95.2 (91.7)	84.2 (77.8)	83.6 (55.7)
R_5	97.8 (98.5)	97.8 (97.3)	94.9 (97.3)	94.7 (90.5)	90.5 (73.4)	89.6 (51.5)
R_6	97.9 (98.7)	97.9 (98.2)	94.9 (97.6)	95.2 (92.2)	81.0 (63.6)	83.3 (45.4)
R_7	98.6 (97.7)	98.1 (98.8)	95.9 (96.9)	95.3 (94.1)	77.0 (55.2)	77.9 (60.8)
R_8	98.0 (99.3)	97.8 (97.2)	95.2 (98.6)	95.0 (91.7)	81.3 (64.3)	86.4 (44.7)
R_{ave}	98.2 (98.7)	97.9 (97.8)	95.4 (97.9)	95.2 (92.4)	81.8 (67.5)	84.1 (52.3)

Table 7.25: Recognition accuracy for Experiment D2.

	Recognition accuracy with our approach and HMMs (parenthesis) (%)					
	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer, unseen sample	
R_i	Recall	Precision	Recall	Precision	Recall	Precision
R_1	98.1 (98.9)	98.2 (99.0)	95.5 (98.3)	96.0 (97.1)	82.6 (76.9)	86.7 (62.9)
R_2	98.7 (98.9)	99.0 (99.0)	95.3 (98.5)	96.2 (97.2)	80.2 (51.7)	86.1 (55.3)
R_3	98.6 (99.3)	98.6 (98.7)	95.6 (98.4)	96.7 (95.9)	81.0 (74.3)	87.0 (51.3)
R_4	98.2 (98.9)	98.4 (98.9)	94.9 (98.3)	95.1 (96.2)	85.4 (77.5)	87.4 (61.6)
R_5	98.0 (98.7)	98.1 (98.9)	94.6 (97.8)	95.4 (95.8)	90.1 (75.1)	91.7 (58.1)
R_6	97.9 (98.6)	98.1 (98.9)	95.2 (97.4)	96.0 (94.4)	81.4 (63.2)	85.1 (47.6)
R_7	98.3 (97.6)	98.6 (99.3)	95.7 (97.3)	95.9 (96.6)	80.7 (54.3)	82.8 (63.4)
R_8	98.5 (99.4)	98.4 (98.8)	95.3 (98.8)	96.4 (97.4)	80.8 (67.6)	87.0 (51.3)
R_{ave}	98.3 (98.8)	98.4 (98.9)	95.3 (98.1)	96.0 (96.3)	82.8 (67.6)	86.7 (56.4)

The results in Table 7.24 show that both the proposed approach and the HMM-based approach perform well for unseen samples from seen signers. This is not surprising as the average classification accuracy of the sub-segment classifier for unseen samples from seen signers is 93.1% and we can expect that the recognition accuracies will not deviate appreciably from the experiments without movement epenthesis shown in Tables 7.19 and 7.21. However, there are obvious decreases in the recognition accuracy from 87.9% to 81.8% and 89.5% to 84.1% for recall and precision in samples from unseen signers with our approach. Similarly in the HMM experiment, the recall rate dropped from 69.6% to 67.5% and the precision dropped from 57.3% to 53.3%. When we relaxed the decision boundary threshold of the Bayesian network, we obtained some improvement for both our approach and the HMM-based approach as shown in Table 7.25. In Experiment D1, the recalls and precisions for unseen samples, seen signer and unseen signer were (95.4%, 95.2%) and (81.8%, 84.1%), respectively, and the corresponding results in Experiment D2 were (95.3%, 96.0%) and (82.8%, 86.7%), respectively.

To improve performance, we applied the modified segmental CRF decoding procedure with the two-class SVMs and skip states to the sequences obtained from the sub-segment classifier as in Experiment D2. We tested for two different values of maximum number of skip states, $M_s = 1, 2$. We found experimentally that both $M_s = 1$ and 2 yielded comparable results. Hence, we chose $M_s = 1$ for less computational cost. The final recognition accuracy is presented in Table 7.26. With our decoding algorithm with two-class SVMs and skip states, we were able to achieve a recall rate of 95.7% and precision accuracy of 96.6% for unseen samples from seen signers. As for unseen signers, we obtained 86.6% recall rate and 89.8% precision accuracy. These results are close to the clean segment results where we obtained 98.0% for unseen samples from seen signers and 90.8% for

Table 7.26: Recognition accuracy with modified segmental CRF decoding procedure with two-class SVMs and skip states

	Recognition accuracy (%)					
	Seen signer, seen sample		Seen signer, unseen sample		Unseen signer, unseen sample	
Round	Recall	Precision	Recall	Precision	Recall	Precision
R_1	98.4	98.7	95.9	96.4	88.3	91.8
R_2	98.3	99.1	96.5	97.1	85.2	89.7
R_3	97.7	98.6	95.3	96.3	83.6	90.2
R_4	97.7	98.3	94.5	96.0	88.5	88.0
R_5	97.5	98.2	96.0	96.4	92.6	92.5
R_6	97.4	98.1	95.7	97.0	85.2	89.4
R_7	97.9	98.5	95.6	96.9	84.2	86.2
R_8	97.4	98.2	95.9	96.5	85.1	90.3
R_{ave}	97.8	98.5	95.7	96.6	86.6	89.8

unseen signers. These results are also close to the experiments where we assumed the movement epenthesis segments were removed perfectly. The corresponding recognition accuracies shown in Table 7.20 for the experiments using sentences without movement epenthesis are 97.5%, 97.1%, 89.2% and 90.0%. The experimental results demonstrate that our CRF-based framework can cope with signer variations in sign language, showing good generalization to new signers. Compared to many of the signer independent systems without adaptation surveyed in the literature, we have obtained high recognition accuracy for continuously signed sentences by unseen signers. Even works with adaptation as presented in [157], achieved a recognition accuracy of only 75.8%. Hence, our results show good performance with respect to the state of the art.

7.6 Summary

For subsystem 1, the naïve Bayesian network classifier was selected as the most efficient segmentation algorithm. It yielded 88.1% accuracy and 13.7% false alarms on unseen, naturally signed sentences. The trajectory segmentation approach based on Bayesian network is promising. The results of the automatic phoneme transcription approach show that the segments obtained are useful for sign language phoneme transcription and recognition. Further, the experimental results show that our automatic approach is more accurate than manual trajectory segmentation and phoneme transcription, while significantly saving time consuming human labour required in the manual approach. An automatic approach will be even more important for large vocabulary systems where manual transcription is impractical.

For subsystem 2, the CRF yielded 85.7% sub-segment classification accuracy on naturally signed sentences by unseen signers while the SVM also yielded comparable accuracy of 85.6%. Based on observations of the errors obtained by the CRF and SVM, we used a Bayesian network to fuse the outputs from the two classifiers. The experimental results showed an improvement in accuracy to 88.2% for unseen signers. From the experimental results, we also observed that the results on unseen signers obtained by the Bayesian network were more consistent than the results yielded individually by the CRF and SVM. This outcome is essential for our subsequent works.

For subsystem 3, the best recognition accuracy was achieved by incorporating the two-class SVMs and the state skipping function in the proposed decoding algorithm. We obtained a recall rate of 95.7% and precision accuracy of 96.6% for unseen samples from seen signers as well as a recall rate of 86.6% and precision accuracy of 89.8% for unseen signers. In comparison, the HMM-based approach failed to provide good results for unseen signers in our experiments. Our strategy works well for sentences that have been seen by the system and also yields good performance for recognizing naturally signed sentences from new signers.

We can do anything
we want to do if we
stick to it long enough.
Helen Keller (1880-1968)

8
Conclusions

The key contribution of this thesis was in devising a segment-based sign recognition approach that is robust to different variations which arise in continuous sign language sentences. Current works in the literature have largely ignored the problem of signer independent continuous sign language recognition, a crucial aspect necessary for using these systems in practice. We believe our work contributes to progress in this area.

Moving away from the conventional HMM-based recognition approach which is mainly generative, we have demonstrated that the underlying characteristics of a discriminative model are more suitable to deal with variations in sign language and provide better generalization. For this purpose, we devised a robust CRF-based recognition framework which shows high degree of generalization to unseen signers without using any adaptation algorithm. More specifically, the thesis makes the following contributions:

i) The framework was designed to tolerate variations in continuous sign recognition. We demonstrated the viability of using CRFs in the parallel channels which is commonly modeled with HMMs. The multi-channel methodology with CRFs allows variations in the components to be tackled independently and more efficiently. We also adopted two-layer CRFs for phoneme level and sign level recognition allowing feature level variation as well as sign level variation to be handled separately. All these characteristics contribute to the good recognition performance obtained in our experiments for recognizing continuously signed sentences by unseen signers. In addition, the parameters of the model pertaining to the parallel phoneme level CRFs as well as the word level CRF are learned independently making the training tractable and fast.

ii) We have devised a novel and efficient decoding algorithm for the two-layered CRFs by modifying the semi-Markov CRF inferencing algorithm. We proposed a method to include an "UNCLASSIFIED" class in the CRF-based framework by using two-class SVMs. This reduces the complexity in the decoder, where sub-segments need to be merged, and their likelihoods evaluated. We also introduced skip states into the decoding algorithm to accommodate merging of sub-segments that are not contiguous, with the motivation to compensate for errors in the sub-segment classification algorithm. Unlike many layered models where decoding is performed separately in different channels or levels, producing either top-down or bottom-up information flow, the data streams at different channels and levels of our proposed framework are modeled simultaneously. The final inferencing results are obtained based on instantaneous fusion of information from the phoneme and sign levels and this leads to natural synchronization between the streams of data. Our CRF-based decoding framework has yielded high recognition accuracy for decoding continuously signed sentences by unseen signers, yielding 86.6% recall and 89.8% precision rates. These high rates have been achieved without any adaptation procedure for new signers.

iii) Our view is that modeling movement epentheses is not a good idea for developing signer independent systems as it may include large variations that are not systematic. Hence, in our proposed framework, movement epenthesis segments are differentiated from sign, and discarded from test sentences, before decoding for signs. Any errors in this are dealt with dynamically during decoding. Further, as opposed to the conventional perception of movement epenthesis as a non-informative segment which is merely for a transition from one sign to another, we suggest that it provides information that is useful to the decoder. We presented a strategy to identify the locations of movement epentheses in a sentence by training a classifier for *SIGN* and *ME* sub-segments. This helps to break down a sentence into smaller sequences, and thereby reduce the decoding complexity significantly. This classifier was a Bayesian network which fused outputs from a SVM and a CRF.

iv) We presented a new phoneme transcription procedure for the movement component using a PCA-based representation, and used AP and k-means clustering algorithms to define and extract phonemes for the other three (static) components. The phonemes were automatically extracted to facilitate training of the parallel CRFs at the phoneme level. As there is still no consensus among linguists on a phonological model for signs, our approach offers a systematic way to transcribe phonemes in sign language. Though it is a data-driven approach, the phonemes defined were observed to have linguistic meaning.

v) Though four components are commonly described in sign language, most of the previous works do not model them fully in four separate channels. Our work closely follows the sign language model defined by linguists, and includes all the four components, viz. handshape, movement, palm orientation and location in our recognition framework. Modeling the movement component as a separate channel is difficult as it requires features that evolve as a quasi-stationary process and at the same time are invariant to position [113]. We proposed simple and efficient line fitting procedure to extract features for the movement component in continuously signed sentences, and demonstrated that the movement features are effective for continuous sign recognition.

8.1 Future Works

Although the dominant hand carries more information, there are several two handed signs in sign language. In future work, it would be useful to include data from both hands for continuous sign recognition. Normally, the non-dominant hand expresses relative spatial relation and is also used in symmetrical signs. Including the non-dominant hand data may entail additional channels in the phoneme level of our CRF-based framework.

Currently, the segmentation errors made by the naïve Bayesian network classifier were corrected manually before phoneme transcription is carried out. It would be desirable to further improve the classification rate by using more efficient features to further reduce the manual work required. The main disadvantage of the decoding algorithm for the two-layer CRF is that it runs much slower than the linear CRF for large number of sub-segments in the sequence. Hence, it would be useful to devise a search algorithm which can decode the path of the sign sequence more efficiently such as [148].

Grammatical information of sign language is another important aspect which should not be neglected. Sign language communication will not be complete without the grammatical information provided by inflected signs and non-manual signs. Inflections give layered or added meanings to a sign. Though we have included directional verbs in our experiments, there are other important inflections in sign language which can be investigated.

The non-manual channel is another important aspect for complete understanding of sign language communication. Often, gesture researchers neglect this aspect which is related to face recognition, expression recognition, lip motion tracking etc. Extracting information from the non-manual channel and fusing it with the manual channel present many challenging research problems.

Publication List

1) W. W. Kong and S. Ranganath. Sign language phoneme transcription with PCA-based representation. In *Proceedings of the International Conference on Information, Communications & Signal Processing*, pages 1-5, Singapore, Dec 2007.

2) W. W. Kong and S. Ranganath. Signing exact English (SEE): Modeling and recognition. *Pattern Recognition*, 41(5):1638-1652, 2008.

3) W. W. Kong and S. Ranganath. Automatic hand trajectory segmentation and phoneme transcription for sign language. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 1-6, Amsterdam, The Netherlands, Sep 2008.

4) W. W. Kong and S. Ranganath. Sign language phoneme transcription with rule-based hand trajectory segmentation. *Signal Processing Systems*, 59(2):211-222, 2010. Invited paper.

5) W. W. Kong and S. Ranganath. Subject independent sign language recognition using probabilistic models and segmentation. Manuscript in preparation for submission to *Pattern Recognition.*

Bibliography

[1] S. Akyol and U. Canzler. An information terminal using vision based sign language recognition. In *Proceedings of International ITEA Workshop on Virtual Home Environments*, pages 61–68, Paderborn, Germany, Feb 2002.

[2] O. Al-Jarrah and A. Halawani. Recognition of gesture in Arabic sign language using neuro-fuzzy systems. *Artificial Intelligence*, 133:117–138, 2001.

[3] M. Al-Rousan, K. Assaleh, and A. Tala'a. Video-based signer-independent Arabic sign language recognition using hidden Markov models. *Applied Soft Computing*, 9:990–999, 2009.

[4] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1685–1699, sep 2009.

[5] A. Alvi et al. Pakistan sign language recognition using statistical template matching. *World Academy of Science, Engineering and Technology*, 3:52–55, 2005.

[6] O. Aran and L. Akarun. A multi-class classification strategy for Fisher

scores: Application to signer independent sign language recognition. *Pattern Recognition*, 43:1776–1788, 2009.

[7] O. Aran et al. A belief-based sequential fusion approach for fusing manual signs and non-manual signals. *Pattern Recognition*, 42:812–822, 2008.

[8] H. Asada and M. Brady. The curvature primal sketch. Technical Report 758, MIT AI Memo, 1984.

[9] M. Assan and K. Grobel. Video-based sign language recognition using hidden Markov models. In *Proceedings of Gesture Workshop*, pages 97–109, Bielefeld, Germany, Sep 1997.

[10] F. I. Bashir, A. A. Khokhar, and D. Schonfeld. Object trajectory-based activity classification and recognition using hidden Markov models. *IEEE Transaction on Image Processing*, 16(7):1912–1919, 2007.

[11] R. Battison, H. Markowicz, and J. Woodward. A good rule of thumb: Variable Phonology in American sign language. In R. W. Fasold and R. W. Shuy, editors, *Analyzing Variation in Language*, pages 291–302. Georgetown University Press, 1975.

[12] B. Bauer and H. Hienz. Relevant features for video-based continuous sign language recognition. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 440–445, Washington, DC, USA, Mar 2000.

[13] B. Bauer, H. Hienz, and K.-F. Kraiss. Video-based continuous sign language recognition using statistical methods. In *Proceedings of International Conference on Pattern Recognition*, pages 463–466, Barcelona, Spain, Sep 2000.

[14] B. Bauer and K.-F. Kraiss. Towards an automatic sign language recognition system using subunits. In *Proceedings of Gesture Workshop*, pages 64–75, London, UK, Apr 2001.

[15] R. Beale and A. D. N. Edwards. Recognizing postures and gestures using neural networks. In *Neural Networks and Pattern Recognition in Human-Computer Interaction*, pages 163–169, Ellis Horwood, New York, 1992.

[16] U. Bellugi and E. S. Klima. Aspects of sign language and its structure. In J. F. Kavanagh and J. E. Cutting, editors, *The Role of Speech in Language*, pages 171–203. The MIT Press, 1975.

[17] M. K. Bhuyan, D. Ghosh, and P. K. Bora. Continuous hand gesture segmentation and co-articulation detection. In *Proceedings of Conference on Computer Vision, Graphics and Image Processing*, pages 564–575, Madurai, India, 2006.

[18] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, first edition, 2006.

[19] C. M. Bishop and J. Lasserre. Generative or discriminative? Getting the best of both worlds. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian Statistics*, pages 3–24. Oxford University Press, 2007.

[20] B. Bossard, A. Braffort, and M. Jardino. Some issues in sign language processing. In *Proceedings of Gesture Workshop*, pages 90–100, Genova, Italy, Apr 2003.

[21] A. Braffort. ARGo: An architecture for sign language recognition and

interpretation. In *Proceedings of Gesture Workshop*, pages 17–30, York, UK, Mar 1996.

[22] G. Caridakis et al. SOMM: Self organizing Markov map for gesture recognition. *Pattern Recognition Letters*, 31:52–59, 2009.

[23] S.-S. Cho, H.-D. Yang, and S.-W. Lee. Sign language spotting based on semi-Markov conditional random field. In *Proceedings of Workshop on Applications of Computer Vision*, pages 1–6, Snowbird, UT, Dec 2009.

[24] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[25] S. Cox et al. TESSA, a system to aid communication with deaf people. In *Proceedings of International ACM SIGCAPH conference on Assistive Technologies*, pages 205–212, Edinburgh, Scotland, 2003.

[26] Y. Cui and J. Weng. Appearance-based hand sign recognition from intensity image sequences. *Computer Vision and Image Understanding*, 78(2):157–176, 2000.

[27] D. DeCaprio et al. Gene prediction using conditional random fields. *Genome Research*, 17(9):1389–1396, 2007.

[28] J.-W. Deng and H. T. Tsui. A novel two-layer PCA/MDA scheme for hand posture recognition. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 294–299, Washington, DC, USA, 2002.

[29] K. G. Derpanis, R. P. Wildes, and J. K. Tsotsos. Definition and recovery of kinematic features for recognition of American sign language movements. *Image and Vision Computing*, 26:1650–1662, Apr 2008.

[30] D. B. Dias et al. Hand movement recognition for Brazilian sign language: A study using distance-based neural networks. In *Proceedings of International Conference on Neural Networks*, pages 697–704, Atlanta, Georgia, USA, Jun 2009.

[31] L. Ding and A. M. Martinez. Modelling and recognition of linguistic components in American sign language. *Image and Vision Computing*, 27:1826–1844, Feb 2009.

[32] P. Dreuw et al. Spoken language processing techniques for sign language recognition and translation. *Technology and Disability*, 20(2):121–133, Jul 2008.

[33] P. Dreuw et al. SignSpeak - Understanding, recognition, and translation of sign languages. In *Workshop on the Representation and Processing of Sign Languages: Copora and Sign Language Technologies*, Valletta, Malta, May 2010.

[34] P. Dreuw et al. The SignSpeak project - Bridging the gap between signers and speakers. In *International Conference on Language Resources and Evaluation*, Valletta, Malta, May 2010.

[35] P. Dreuw, J. Forster, T. Deselaers, and H. Ney. Efficient approximations to model-based joint tracking and recognition of continuous sign language. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 1–6, Amsterdam, The Netherlands, Sep 2008.

[36] P. Dreuw, J. Forster, and H. Ney. Tracking benchmark databases for video-based sign language recognition. In *ECCV International Workshop on Sign, Gesture, and Activity*, Crete, Greece, Sep 2010.

[37] P. Dreuw, C. Neidle, V. Athitsos, S. Sclaroff, and H. Ney. Benchmark databases for video-based automatic sign language recognition. In *Proceedings of International Conference on Language Resources and Evaluation*, pages 1115–1120, Marrakech, Morocco, Jun 2008.

[38] P. Dreuw and H. Ney. Visual modeling and feature adaptation in sign language recognition. In *International Computer Vision Summer School*, Sicily, Italy, Jul 2008.

[39] P. Dreuw, D. Stein, and H. Ney. Enhancing a sign language translation system with vision-based features. In *Proceedings of Gesture Workshop*, pages 108–113, Lisbon, Portugal, May 2007.

[40] P. Dreuw, P. Steingrube, T. Deselaers, and H. Ney. Smoothed disparity maps for continuous American sign language recognition. In *Proceedings of International Conference on Pattern Recognition and Image Analysis*, pages 24–31, Póvoa de Varzim, Portugal, Jun 2009.

[41] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, first edition, 1973.

[42] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* John Wiley and Sons, second edition, 2001.

[43] A. D. N. Edwards. Progress in sign language recognition. In *Proceedings of Gesture Workshop*, pages 13–21, Bielefeld, Germany, 1998.

[44] G. Fang et al. Signer-independent continuous sign language recognition based on SRN/HMM. In *Proceedings of Gesture Workshop*, pages 76–85, London, UK, Apr 2001.

[45] G. Fang and W. Gao. A SRN/HMM system for signer-independent continuous sign language recognition. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 312 – 317, Washington, D.C., May 2002.

[46] G. Fang, W. Gao, and D. Zhao. Large vocabulary sign language recognition based on fuzzy decision trees. *IEEE Transactions on Systems, Man, and Cybernertics - Part A: Systems and Humans*, 34(3):305–314, May 2004.

[47] G. Fang, W. Gao, and D. Zhao. Large-vocabulary continuous sign language recognition based on transition-movement models. *IEEE Transactions on Systems, Man, and Cybernertics - Part A: Systems and Humans*, 37(1):1–8, Jan 2007.

[48] A. Farhadi, D. Forsyth, and R. White. Transfer learning in sign language. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN, Jun 2007.

[49] S. S. Fels and G. E. Hinton. Glove-talk: A neural network interface between a data-glove and speech synthesizer. *IEEE Transactions on Neural Networks*, 4(1):2–8, Jan 1993.

[50] M. Flasiński and S. Myśliński. On the use of graph parsing for recognition of isolated hand postures of Polish sign language. *Pattern Recognition*, 43:2249–2264, Jan 2010.

[51] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[52] W. Gao et al. Sign Language Recognition based on HMM/ANN/DP. *Pattern Recognition and Artificial Intelligence*, 14(5):587–602, 2000.

[53] W. Gao et al. Transition movement models for large vocabulary continuous sign language recognition. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 553–558, Seoul, Korea, May 2004.

[54] S. Gibet and P.-F. Marteau. Approximation of curvature and velocity using adaptive sampling representations - Application to hand gesture analysis. In *Proceedings of Gesture Workshop*, May 2007.

[55] J. R. W. Glauert, R. Elliott, S. J. Cox, J. Tryggvason, and M. Sheard. VANESSA - A system for communication between Deaf and hearing people. *Technology and Disability*, 18(4):207–216, Dec 2006.

[56] J. Goodman. Exponential priors for maximum entropy models. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 305–312, Boston, Massachusetts, May 2004.

[57] L. Gupta and S. Ma. Gesture-based interaction and communications: Automated classification of hand gesture contours. *IEEE Transactions on Systems, Man, and Cybernertics - Part C: Applications and Reviews*, 31:114–120, 2001.

[58] G. Gustason and E. Zawolkow. *Signing Exact English.* Modern Sign Press, Inc., 1995.

[59] H. Haberdar and S. Albayrak. A two-stage visual Turkish sign language recognition system based on global and local features. In *Proceedings of International Symposium on Methodologies for Intelligent Systems*, pages 29–37, Bari, Italy, Sep 2006.

[60] Y. Hamada, N. Shimada, and Y. Shirai. Hand shape estimation under complex backgrounds for sign language recognition. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 589–594, Seoul, Korea, May 2004.

[61] J. Han, G. Awad, and A. Sutherland. Modeling and segmenting subunits for sign language recognition based on hand motion analysis. *Pattern Recognition Letters*, 30:623–633, 2009.

[62] P. A. Harling and A. D. N. Edwards. Hand tension as a gesture segmentation cue. In *Proceedings of Gesture Workshop*, York, UK, Mar 1996.

[63] J. L. Hernandez-Rebollar, N. Kyriakopoulos, and R. W. Lindeman. A new instrumented approach for translating American sign language into sound and text. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 547–552, Seoul, Korea, May 2004.

[64] H. Hienz, K. Grobel, and G. Beckers. Video-based handshape recognition using artificial neural networks. In *European Congress on Intelligent Techniques and Soft Computing*, pages 1659–1663, 1996.

[65] E.-J. Holden, G. Lee, and R. Owens. Australian sign language recognition. *Machine Vision and Applications*, 16(5):312–320, 2005.

[66] X. Huang and K. Lee. On speaker-independent, speaker-dependent, and speaker-adaptive speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 877–880, Apr 1991.

[67] A. Ibarguren, I. Maurtua, and B. Sierra. Layered architecture for real time sign recognition: Hand gesture and movement. *Engineering Applications of Artificial Intelligence*, 23:1216–1228, 2010.

[68] J. Isaacs and S. Foo. Hand pose estimation for American sign language recognition. In *Proceedings of Southeastern Symposium on System Theory*, pages 132–136, Atlanta, Georgia, Mar 2004.

[69] A. Just and S. Marcel. A comparative study of two state-of-the-art sequence processing techniques of hand gesture recognition. *Computer Vision and Image Understanding*, 113:532–543, 2008.

[70] T. Kadir et al. Minimal training, large lexicon, unconstrained sign language. In *British Machine Vision Conference*, London, UK, Sep 2004.

[71] M. W. Kadous. Machine recognition of Auslan signs using powergloves: Towards large-lexicon recognition of sign languages. In *Proceedings of Workshop on the Integration of Gestures in Language and Speech*, Wilmington Delaware, Oct 1996.

[72] K. Kahol, P. Tripathi, and S. Panchanathan. Automated gesture segmentation from dance sequences. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 883–888, May 2004.

[73] N. Kambhatla and T. K. Leen. Dimension reduction by local principal component analysis. In *Neural Computation*, volume 9, pages 1493–1516, 1997.

[74] A. Karami, B. Zanji, and A. K. Sarkaleh. Persian sign language (PSL) recognition using wavelet transform and neural networks. *Expert Systems with Applications*, 38:2661–2667, 2011.

[75] R. Kasturi and R. Jain. *Computer Vision: Principles.* IEEE Computer Society Press, 1991.

[76] D. Kelly, J. McDonald, and C. Markham. Continuous recognition of motion based gestures in sign language. In *Proceedings of International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1073–1080, Kyoto, Japan, Sep 2009.

[77] D. Kelly, J. McDonald, and C. Markham. Recognizing spatiotemporal gestures and movement epenthesis in sign language. In *Proceedings of International Conference on Machine Vision and Image Processing*, pages 145–150, Dublin, Ireland, Sep 2009.

[78] C. Keskin and L. Akarun. STARS: Sign tracking and recognition system using input-output HMMs. *Pattern Recognition Letters*, 30:1086–1095, 2009.

[79] I.-C. Kim and S.-I. Chien. Analysis of 3D hand trajectory gestures using stroke-based composite hidden Markov models. *Applied Intelligence*, 15:131–143, 2001.

[80] R. Klinger et al. Identifying gene specific variations in biomedical text. *Journal of Bioinformatics and Computational Biology*, 5(6):1277–1296, 2007.

[81] R. Klinger and K. Tomanek. Classical probabilistic models and conditional random fields. Algorithm Engineering Report TR07-2-013, Department of Computer Science, Dortmund University of Technology, 2007.

[82] W. W. Kong and S. Ranganath. 3-D Hand Trajectory Recognition for Signing Exact English. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 535–540, Seoul, Korea, May 2004.

[83] W. W. Kong and S. Ranganath. Automatic hand trajectory segmentation and phoneme transcription for sign language. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 1–6, Amsterdam, The Netherlands, Sep 2008.

[84] W. W. Kong and S. Ranganath. Signing exact English (SEE): Modeling and recognition. *Pattern Recognition*, 41(5):1638–1652, 2008.

[85] W. W. Kong and S. Ranganath. Sign language phoneme transcription with rule-based hand trajectory segmentation. *Signal Processing Systems*, 59(2):211–222, 2010.

[86] V. Kosmidou and L. J. Hadjileontiadis. Sign language recognition using intrinsic-mode sample entropy on sEMG and accelerometer data. *IEEE Transactions on Biomedical Engineering*, 56(12):2879–2890, 2009.

[87] J. Kramer and L. Leifer. The "talking glove": An expressive and receptive "verbal" communication aid for the deaf, deaf-blind, and nonvocal. In *Proceedings of Conference on Computer Technology/Special Education/Rehabilitation*, pages 335–340, California, Northridge, Oct 1987.

[88] B. J. Kröger et al. An action-based concept for the phonetic annotation of sign language gestures. In *Elektronische Sprachsignalverarbeitung*, Berlin, Germany, Sep 2010.

[89] T. Kudo, K. Yamamoto, and Y. Matsumoto. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 230–237, Barcelona, Spain, Jul 2004.

[90] R. Kuhn et al. A rapid speaker adaptation in eigenvoice space. *IEEE Transactions on Speech and Audio Processing*, 8(6):695–707, 2000.

[91] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of International Conference on Machine Learning*, pages 282–289, 2001.

[92] C. Lee et al. The control of avatar motion using hand gesture. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 59–65, Nov 1998.

[93] H.-K. Lee and J. H. Kim. An HMM-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973, Oct 1999.

[94] Y.-H. Lee and C.-Y. Tsai. Taiwan sign language (TSL) recognition based on 3D data and neural networks. *Expert Systems with Applications*, 36:1123–1128, 2009.

[95] H. Li and M. Greenspan. Model-based segmentation and recognition of dynamic gestures in continuous video streams. *Pattern Recognition*, 44(8):1614–1628, Aug 2011.

[96] R.-H. Liang and M. Ouhyoung. A real-time continuous gesture recognition system for sign language. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 558–565, 1998.

[97] J. F. Lichtenauer, E. A. Hendriks, and M. J. Reinders. Sign language recognition by combining statistical DTW and independent classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):2040–2046, 2008.

[98] J. F. Lichtenauer, G. A. Holt, M. J. Reinders, and E. A. Hendriks. Person-independent 3D sign language recognition. In *Proceedings of Gesture Workshop*, pages 69–80, Lisbon, Portugal, 2007.

[99] S. K. Liddell and R. E. Johnson. *America Sign Language: The Phonological Base*, chapter 64, pages 195–277. Sign Language Studies, 1989.

[100] B. Loeding et al. Progress in automated computer recognition of sign language. In *Proceedings of International Conference on Computers Helping People with Special Needs*, pages 1079–1087, Paris, France, Jul 2004.

[101] C. Lucas, R. Bayley, and C. Valli. *What's Your Sign for Pizza?: An Introduction to Variation in American Sign Language.* Gallaudet University Press, 2003.

[102] M. Maebatake et al. Sign language recognition based on position and movement using multi-stream HMM. In *Proceedings of International Symposium on Universal Communication*, pages 478–481, Osaka, Japan, Dec 2008.

[103] T. Matsuo, Y. Shirai, and N. Shimada. Automatic generation of HMM topology for sign language recognition. In *Proceedings of International Conference on Pattern Recognition*, pages 1–4, Tampa, FL, Dec 2008.

[104] A. McCallum, D. Freitag, and F. C. N. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of International Conference on Machine Learning*, pages 591–598, Stanford University, Stanford, CA, USA, Jun 2000.

[105] J. Morris and E. Fosler-Lussier. Conditional random fields for integrating local discriminative classifiers. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(3):617–628, 2008.

[106] C. Myers and L. Rabiner. A level building dynamic time warping algorithm for connected word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(2):284–297, 1981.

[107] Y. Nam and K. Wohn. Recognition of hand gestures with 3D, nonlinear arm movement. *Pattern Recognition Letters*, 18:105–113, 1997.

[108] J. Naoum-Sawaya et al. A real-time continuous gesture recognition system for sign language. In *Proceedings of International Symposium on Communications, Control and Signal Processing*, Marrakech, Morocco, Mar 2006.

[109] S. Nayak, S. Sarkar, and B. Loeding. Unsupervised modeling of signs embedded in continuous sentences. In *Proceedings of CVPR Workshop on Vision for Human-Computer Interaction*, San Diego, CA, USA, Jun 2005.

[110] S. Nayak, S. Sarkar, and B. Loeding. Automated extraction of signs from continuous sign language sentences using iterated conditional modes. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 2583–2590, Miami, FL, USA, Jun 2009.

[111] E. Ohira, H. Sagawa, T. Sakiyama, and M. Ohki. A segmentation method for sign language recognition. *IEICE Transactions on Information and Systems*, E78-D(1):49–57, 1995.

[112] E.-J. Ong and R. Bowden. A boosted classifier tree for hand shape detection. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 889–894, Seoul, Korea, May 2004.

[113] S. C. W. Ong. *Beyong Lexical Meaning: Probabilistic Models for Sign Language Recognition.* PhD thesis, National University of Singapore, 2007.

[114] S. C. W. Ong and S. Ranganath. Deciphering gestures with layered meanings and signer adaptation. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 559–564, Seoul, Korea, May 2004.

[115] S. C. W. Ong and S. Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):873–891, Jun 2005.

[116] S. C. W. Ong and S. Ranganath. A new probabilistic model for recognizing signs with systematic modulations. In *Proceedings of International Workshop on Analysis and Modeling of Faces and Gestures*, pages 16–30, Rio de Janeiro, Brazil, Oct 2007.

[117] S. C. W. Ong, S. Ranganath, and Y. V. Venkatesh. Understanding gestures with systematic variations in movement dynamics. *Pattern Recognition*, 39(9):1633–1648, 2001.

[118] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 84–91, Seattle, WA, USA, Jun 1994.

[119] D. M. Perlmutter. On the segmental representation of transitional and bidirectional movements in ASL phonology. In S. D. Fischer and P. Siple, editors, *Theoretical Issues in Sign Language Research: Volume 1*, pages 67–80. The University of Chicago Press, 1990.

[120] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett,

B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. The MIT Press, 2000.

[121] Polhemus, Inc. *3SPACE® FASTRAK® USER'S MANUAL*, rev. c edition, Nov 2002.

[122] A. Quatonni et al. Hidden conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1848–1852, 2007.

[123] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *International Journal of Computer Vision*, 50(2):203–226, 2002.

[124] H. Sagawa and M. Takeuchi. A method for analyzing spatial relationships between words in sign language recognition. In *Proceedings of Gesture Workshop*, pages 197–209, Gif-sur-Yvette, France, Mar 1999.

[125] H. Sagawa and M. Takeuchi. A method for recognizing a sequence of sign language words represented in a Japanese sign language sentence. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 434–439, Grenoble, France, Mar 2000.

[126] H. Sagawa, M. Takeuchi, and M. Ohki. Methods to describe and recognize sign language based on gesture components represented by symbols and numerical values. *Knowledge-Based Systems*, 10:287–294, 1998.

[127] P. Santemiz et al. Automatic sign segmentation from continuous signing via multiple sequence alignment. In *Proceedings of International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2001–2008, Kyoto, Japan, Sep 2009.

[128] S. Sarawagi and W. W. Cohen. Semi-Markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, page 11851192, Vancouver, British Columbia, Canada, Dec 2004.

[129] S. Sarkar, B. Loeding, and A. S. Parashar. Fusion of manual and non-manual information in American sign language recognition. In C. H. Chen, editor, *Handbook of Pattern Recognition and Computer Vision*, pages 477–495. Imperial College Press, 2010.

[130] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Edmonton, Canada, May 2003.

[131] T. Shanableh, K. Assaleh, and M. Al-Rousan. Spatio-temporal feature-extraction techniques for isolated gesture recognition in Arabic sign language. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 37(3):641–650, 2007.

[132] Q. Shi et al. Human action segmentation and recognition using discriminative semi-Markov models. *Computer Vision*, 93(1):22–32, Sep 2010.

[133] S. Srinivasan and K. L. Boyer. Head pose estimation using view based eigenspaces. In *Proceedings of International Conference on Pattern Recognition*, pages 302–305, Quebec, Canada, Aug 2002.

[134] T. Starner and A. Pentland. Visual recognition of American sign language using hidden Markov models. In *Proceedings of International Conference on Automatic Face and Gesture Recognition*, pages 189–194, Zurich, Switzerland, 1995.

[135] T. Starner and A. Pentland. Real-time American sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1371–1375, Dec 1998.

[136] M. L. Sternberg. *American Sign Language: A Comprehensive Dictionary.* Harper and Row, 1981.

[137] W. C. Stokoe. *Sign Language Structure: An Outline of the Visual Communication System of the American Deaf, Studies in Linguistics: Occasional Papers 8.* Linstok Press, 1960., Silver Spring, MD, 1978.

[138] M.-C. Su et al. A fuzzy rule-based approach to recognizing 3-d arm movements. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 9:191–201, Jun 2001.

[139] H.-I. Suk, S.-S. Cho, H.-D. Yang, M.-C. Roh, and S.-W. Lee. Real-time human-robot interaction based on continuous gesture spotting and recognition. In *Proceedings of International Symposium on Robotics*, pages 120–123, Seoul, Korea, Oct 2008.

[140] H.-I. Suk, B.-K. Sin, and S.-W. Lee. Hand gesture recognition based on dynamic Bayesian network framework. *Pattern Recognition*, 43:3059–3072, 2010.

[141] C. Sutton and A. McCallum. An introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning.* MIT Press, 2006.

[142] N. Tanibata and N. Shimada. Extraction of hand features for recognition of sign language words. In *Proceedings of International Conference on Vision Interface*, pages 391–398, Calgary, Canada, May 2002.

[143] G. ten Holt, P. Hendriks, and T. Andringa. Why don't you see what I mean? Prospects and limitations of current automatic sign language research. *Sign Language Studies*, 6(4):416–437, 2006.

[144] R. A. Tennant and M. G. Brown. *The American Sign Language Handshape Dictionary*. Gallaudet University Press, 1998.

[145] S. Theodorakis, A. Katsamanis, and P. Maragos. Product-HMMs for automatic sign language recognition. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 1601–1604, Taipei, Taiwan, Apr 2009.

[146] P. S. Tsai et al. Cyclic motion detection. Technical Report CS-TR-93-08, University of Central Florida, Orlando, FL, 1993.

[147] P. Vamplew. Recognition of sign language gestures using neural networks. In *European Conference on Disabilities, Virtual Reality and Associated Technologies*, pages 27–33, Maidenhead, England, Jul 1996.

[148] L. T. Vinh et al. Semi-Markov conditional random fields for accelerometer-based activity recognition. *Applied Intelligence*, 33(1), Mar 2010.

[149] Virtual Technologies, Inc. *CyberGlove ® Reference Manual*, Aug 1998.

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

[151] C. Vogler and D. Metaxas. Adapting hidden Markov models for ASL recognition by using three-dimensional computer vision methods. In *Proceedings of International Conference on Systems, Man and Cybernetics*, pages 156–161, Orlando, FL, Oct 1997.

[152] C. Vogler and D. Metaxas. ASL recognition based on a coupling between HMMs and 3D motion analysis. In *Proceedings of International Conference on Computer Vision*, pages 363–369, Mumbai, India, Jan 1998.

[153] C. Vogler and D. Metaxas. Parallel hidden Markov models for American sign language recognition. In *Proceedings of International Conference on Computer Vision*, pages 22–25, Kerkyra, Greece, 1999.

[154] C. Vogler and D. Metaxas. Towards scalability in ASL recognition: Breaking down sign into phonemes. In *Proceedings of Gesture Workshop*, pages 211–224, Gif-sur-Yvette, France, Mar 1999.

[155] C. Vogler and D. Metaxas. Handshapes and movements: Multiple-channel ASL recognition. In *Proceedings of Gesture Workshop*, pages 247–258, Genova, Italy, Apr 2003.

[156] C. Vogler, H. Sun, and D. Metaxas. A framework for motion recognition with applications to American sign language and gait recognition. In *Proceedings of Workshop on Human Motion*, pages 33–38, Austin, TX, Dec 2000.

[157] U. von Agris, , C. Blömer, and K.-F. Kraiss. Rapid signer adaptation for continuous sign language recognition using a combined approach of eigenvoices, MLLR, and MAP. In *Proceedings of International Conference on Pattern Recognition*, pages 1–4, Tampa, FL, Dec 2008.

[158] U. von Agris and K.-F. Kraiss. Towards a video corpus for signer-independent continuous sign language recognition. In *Proceedings of Gesture Workshop*, Lisbon, Portugal, May 2007.

[159] U. von Agris, D. Schneider, J. Zieren, and K.-F. Kraiss. Rapid signer adaptation for isolated sign language recognition. In *Proceedings of Conference on Computer Vision and Pattern Recognition Workshop*, pages 159–164, New York, Jun 2006.

[160] M. B. Waldron and S. Kim. Isolated ASL sign recognition system for deaf persons. *IEEE Transactions on Rehabilitation Engineering*, 3(3):261–271, Sep 1995.

[161] H. M. Wallach. Conditional random fields: An introduction. Technical Report MS-CIS-04-21, Department of Computer and Information Science, University of Pennsylvania, 2004.

[162] M. Walter, A. Psarrou, and S. Gong. Auto clustering for unsupervised learning of atomic gesture components using minimum description length. In *Proceedings of International Conference on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (ICCV Workshops)*, pages 157–162, Vancouver, BC, Canada, 2001.

[163] C. Wang et al. An approach to automatically extracting the basic units in Chinese sign language recognition. In *Proceedings of International Conference on Signal Processing*, pages 855–858, Beijing, China, Aug 2000.

[164] C. Wang, W. Gao, and Z. Xuan. A real-time large vocabulary continuous recognition system for Chinese sign language. In *Pacific Rim Conference on Multimedia*, pages 150–157, Beijing, China, Oct 2001.

[165] C. Wang, S. Shan, and W. Gao. An approach based on phonemes to large vocabulary Chinese sign language recognition. In *Proceedings of Interna-*

tional Conference on Automatic Face and Gesture Recognition, pages 411–416, Washington, DC, USA, May 2002.

[166] J. Wilpon and L. Rabiner. A modified k-means clustering algorithm for use in isolated work recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(3):587–594, 1985.

[167] H.-D. Yang and S.-W. Lee. Robust sign language recognition with hierarchical conditional random fields. In *Proceedings of International Conference on Pattern Recognition*, pages 2202–2205, Istanbul, Turkey, Aug 2010.

[168] H.-D. Yang and S.-W. Lee. Simultaneous spotting of signs and fingerspellings based on hierarchical conditional random fields and boostmap embeddings. *Pattern Recognition*, 43(1):2858–2870, Jan 2010.

[169] H.-D. Yang, S. Sclaroff, and S.-W. Lee. Garbage model formulation with conditional random fields for sign language spotting. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, Jun 2008.

[170] H.-D. Yang, S. Sclaroff, and S.-W. Lee. Sign language spotting with a threshold model based on conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(7):1264–1277, Jul 2009.

[171] M.-H. Yang and N. Ahuja. Recognizing hand gesture using motion trajectories. In *Computer Vision and Pattern Recognition*, volume 1, pages 1466–1472, Ft. Collins, CO, USA, Jun 1999.

[172] R. Yang and S. Sarkar. Detecting coarticulation in sign language using conditional random fields. In *Proceedings of International Conference on Pattern Recognition*, pages 108–112, Hong Kong, China, Aug 2006.

[173] R. Yang, S. Sarkar, and B. Loeding. Enhanced level building algorithm for the movement epenthesis problem in sign language recognition. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN, Jun 2007.

[174] R. Yang, S. Sarkar, and B. Loeding. Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):462–477, Mar 2010.

[175] X. Yang et al. Visual sign language recognition based on HMMs and auto-regressive HMMs. In *Proceedings of Gesture Workshop*, pages 80–83, Berder Island, France, May 2005.

[176] J. Ye, H. Yao, and F. Jiang. Based on HMM and SVM multilayer architecture classifier for Chinese sign language recognition with large vocabulary. In *Proceedings of International Conference on Image and Graphics*, pages 377–380, Dec 2004.

[177] P. Yin et al. Learning the basic units in American sign language using discriminative segmental feature selection. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, pages 4757–4760, Taipei, Taiwan, Apr 2009.

[178] Q. Yuan et al. Recognition of strong and weak connection models in continuous sign language. In *Proceedings of International Conference on Pattern Recognition*, pages 75–78, Quebec City, QC, Canada, Aug 2002.

[179] M. M. Zaki and S. I. Shaheen. Sign language recognition using a combina-

tion of new vision based features. *Pattern Recognition Letters*, 32:572–577, 2010.

[180] L.-G. Zhang et al. Recognition of sign language subwords based on boosted hidden Markov models. In *Proceedings of International Conference on Multimodal Interfaces*, pages 282–287, Trento, Italy, Oct 2005.

[181] J. Zieren and K.-F. Kraiss. Non-intrusive sign language recognition for human-computer interaction. In *IFAC/IFIP/IFORS/IEA Symposium Analysis, Design, and Evaluation of Human-Machine Systems*, pages CD–paper 49, Atlanta, GA, USA, Sep 2004.

[182] J. Zieren and K.-F. Kraiss. Robust person-independent visual sign language recognition. In *Proceedings of Iberian Conference on Pattern Recognition and Image Analysis*, pages 520–528, Estoril, Portugal, Jun 2005.

[183] G. Zweig and P. Nguyen. A segmental CRF approach to large vocabulary continuous speech recognition. In *Workshop on Automatic Speech Recognition and Understanding*, pages 152–157, Merano, Italy, Dec 2009.

Appendix A

Table A.1 lists the 72 basic signs used in the experiments. Seven verbs are used to form 42 directional verbs as shown in Table A.2. The annotation VERB$^{P1 \rightarrow P2}$ is explained as follows. VERB refers to the root verb in the basic signs, and P1 is the subject and P2 is the object. For example, HELP$^{I \rightarrow YOU}$ is denoted as "I help you" in English sentence. "I" used in the annotation refers to the signer and the positioning of the addressees "YOU", and two other non-present referents "GIRL" and "JOHN" is shown in Figure A.1. "YOU" is assumed to be right in front of the signer; "GIRL" is roughly to the right of the signer; "JOHN" is roughly to the left of the signer.

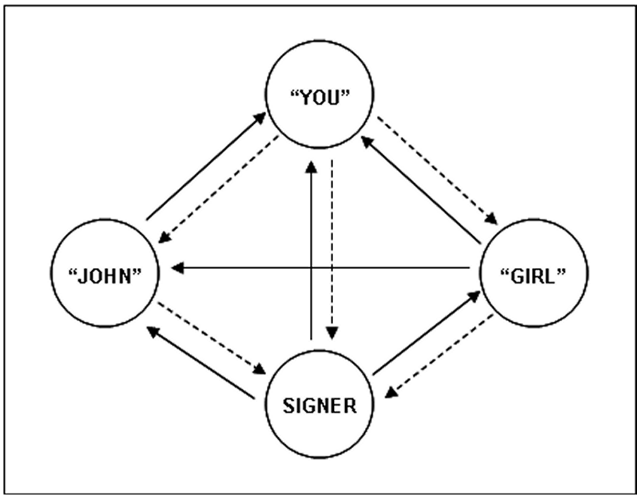


Figure A.1: Positions of the signer and addressees.

Table A.1: Basic signs.

Category	Basic signs
Nouns	BABY, BIBLE, BOOK, BOX, BUILDING, CAT, EMAIL,, GIRL, FISH, HOME, HORSE, JOHN, LECTURE, PAPER, PEN, PICTURE, PLACE, SIGN-LANGUAGE, STONE, WINTER
Pronouns	I, ME, MY, YOU, YOUR
Verbs	BLAME, COME, DRIVE, EAT, GIVE, GO, HELP, KEEP, LOOK, MEET, PREACH, PRINT, SEND, SHOW, TAKE, TEACH, WORK
Modal verbs	MUST
Adjectives	AFRAID, A-LOT, BEAUTIFUL, BLACK, BORING, CLEAN, CLEAR, COLD, DIRTY, EVERYDAY, FAST, HEAVY, HOT, HUNGRY, IMPORTANT, MAD, OLD, SICK, SLOW, UGLY, WRONG
Adverbs	HERE, LATER, NONE, THAT, THERE
Prepositions	WITH
Interjections	PLEASE, WOW

Table A.2: Directional verbs.

Basic verbs	Inflected directional verbs
TEACH	TEACH$^{\text{I}\rightarrow\text{YOU}}$, TEACH$^{\text{I}\rightarrow\text{GIRL}}$, TEACH$^{\text{I}\rightarrow\text{JOHN}}$
BLAME	BLAME$^{\text{I}\rightarrow\text{YOU}}$, BLAME$^{\text{I}\rightarrow\text{GIRL}}$, BLAME$^{\text{I}\rightarrow\text{JOHN}}$ BLAME$^{\text{YOU}\rightarrow\text{ME}}$, BLAME$^{\text{GIRL}\rightarrow\text{ME}}$, BLAME$^{\text{JOHN}\rightarrow\text{ME}}$
GIVE	GIVE$^{\text{I}\rightarrow\text{YOU}}$, GIVE$^{\text{I}\rightarrow\text{GIRL}}$, GIVE$^{\text{I}\rightarrow\text{JOHN}}$ GIVE$^{\text{YOU}\rightarrow\text{ME}}$, GIVE$^{\text{GIRL}\rightarrow\text{ME}}$, GIVE$^{\text{GIRL}\rightarrow\text{YOU}}$ GIVE$^{\text{JOHN}\rightarrow\text{YOU}}$, GIVE$^{\text{GIRL}\rightarrow\text{JOHN}}$
HELP	HELP$^{\text{I}\rightarrow\text{YOU}}$, HELP$^{\text{I}\rightarrow\text{GIRL}}$, HELP$^{\text{I}\rightarrow\text{JOHN}}$ HELP$^{\text{YOU}\rightarrow\text{ME}}$, HELP$^{\text{YOU}\rightarrow\text{GIRL}}$, HELP$^{\text{YOU}\rightarrow\text{JOHN}}$, HELP$^{\text{GIRL}\rightarrow\text{ME}}$, HELP$^{\text{GIRL}\rightarrow\text{YOU}}$, HELP$^{\text{JOHN}\rightarrow\text{ME}}$, HELP$^{\text{JOHN}\rightarrow\text{YOU}}$, HELP$^{\text{GIRL}\rightarrow\text{JOHN}}$
SEND	SEND$^{\text{I}\rightarrow\text{YOU}}$, SEND$^{\text{I}\rightarrow\text{GIRL}}$, SEND$^{\text{I}\rightarrow\text{JOHN}}$
TAKE	TAKE$^{\text{I}\rightarrow\text{YOU}}$, TAKE$^{\text{I}\rightarrow\text{GIRL}}$, TAKE$^{\text{I}\rightarrow\text{JOHN}}$
SHOW	SHOW$^{\text{I}\rightarrow\text{YOU}}$, SHOW$^{\text{I}\rightarrow\text{GIRL}}$, SHOW$^{\text{I}\rightarrow\text{JOHN}}$ SHOW$^{\text{YOU}\rightarrow\text{ME}}$, SHOW$^{\text{GIRL}\rightarrow\text{ME}}$, SHOW$^{\text{GIRL}\rightarrow\text{YOU}}$ SHOW$^{\text{JOHN}\rightarrow\text{ME}}$, SHOW$^{\text{JOHN}\rightarrow\text{YOU}}$