AMERICAN SIGN LANGUAGE RECOGNITION: REDUCING THE COMPLEXITY OF THE TASK WITH PHONEME-BASED MODELING AND PARALLEL HIDDEN MARKOV MODELS

Christian Philipp Vogler

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2003

Dimitris N. Metaxas
Supervisor of Dissertation

Benjamin C. Pierce
Graduate Group Chair

To my loving parents Gesa and Dietrich Vogler, without whose unwavering support my quest for education and knowledge would have run into a dead end long ago.

Acknowledgments

I am indebted to a great many people, who all contributed to my time at Penn and this dissertation, directly or indirectly. Even beginning to list all of them seems like a Sisyphean task, but I will try my best. First and foremost, thanks go to my thesis advisor Professor Dimitris Metaxas, for his unflinching support of my dissertation topic throughout the years, and also for providing me with the opportunity to become intimately familiar with the topic of 3D tracking with deformable models. Special thanks go to my dissertation committee, Professors Norman Badler, Jean Gallier, Martha Palmer, and Carol Neidle, for their enthusiastic support of and feedback on my work. I am particularly indebted to Professor Carol Neidle and Robert Lee for their invaluable feedback on the sections on ASL. Additional thanks go to Ben Bahan, Lana Cook, and Mike Schlang for posing in the sign language sequences used throughout this thesis.

I also would like to thank some of the faculty at my department who contributed valuable feedback, advice, discussions, and generally helped me find my place at Penn: Professors Ruzena Bajcsy, Kostas Daniilidis, Sampath Kannan, Mitchell Marcus, Max Mintz, Scott Nettles, and CJ Taylor. The graduate program coordinator, Mike Felker, went out of his way countless times to help me with any problems or concerns that I had. Gail Shannon at the business office helped me handle financial matters with exemplary courtesy and efficiency.

The people at Penn's Office of International Programs hold a special place in my heart. Andrew Adair, Shalini Bhutani, Renee Boroughs, Lisa Felix, and James Fine all did a job

miles above and beyond their requisite duties in helping me adjust and deal with life in the United States, and made the international community at Penn feel much more like a big family. I consider myself fortunate to have been part of it. Alice Nagle, the coordinator of the student disabilities service, also provided outstanding support and put me in touch with Janet Keen, Donna Ellis, and Nancy Levine, all of whom were wonderful at helping me out during my first year at Penn.

During my time at the HMS and VAST laboratories at Penn I had the privilege of being placed with the most amazing group of fellow students imaginable: Harold Sun, Dimitris Samaras, Matt Beitler, and Siome Goldenstein. The experience of talking to them, working with them, and playing with them is something that I will always treasure. During the last few years, I have been working closely with Siome on deformable model tracking, which provided valuable practical experience and insights into teamwork, and large programming projects. Most important, it also founded and cemented a close friendship that helped me a lot to pull through the final hard years of the dissertation.

My time at Penn would not have been the same without the Deaf community in and around Philadelphia. Michelle Nashleanas, Michael Janger, Christina Clementi, Mark Drolsbaugh, Shiri Hornik, Sangeeta Parekh, Steve Florio, Dennis Dillahunt, Sean Crist, and Newby Ely all were there when it mattered, and caused me more than once to reexamine my beliefs and values about deafness. Many thanks also go to Becky Stevens and Stephen Jane for just being themselves, and for providing a wonderful place to stay over the various holidays.

My path toward education and the Ph.D. would have been a much rockier ride, had it not been for the love and dedication of my parents, Gesa and Dietrich Vogler, who staunchly supported me and helped me tear down barriers when it mattered the most. I am grateful to my grandparents, Renate and Rudolf Vogler, and my sister Constanze, for having shaped much of my outlook on life, and for constantly reminding me of the enjoyable aspects of life. I also would like to thank Barbara Torwegge for her support and

being instrumental to my philosophy about learning and education, Wolfgang Schmidt for introducing me to the Deaf community, and Bernd Rehling for letting me stay involved with the German Deaf community. This dissertation reflects much of all their accomplishments, not only mine.

Last, but not least, my heartfelt thanks go out to Krystallo Tziallila, for her love, her understanding, her support, and for being there for me, even when it was 6,000 miles away from Philadelphia. No matter whether it was at a high or a low point of the Ph.D. program, she has always provided me with something to look forward to, and a purpose beyond education and work.

ABSTRACT

AMERICAN SIGN LANGUAGE RECOGNITION: REDUCING THE COMPLEXITY OF THE TASK WITH PHONEME-BASED MODELING AND PARALLEL HIDDEN MARKOV MODELS

Christian Philipp Vogler

Supervisor: Dimitris N. Metaxas

In this thesis I present a framework for recognizing American Sign Language (ASL) from 3D data. The goal is to develop approaches that will scale well with increasing vocabulary sizes.

Scalability is a major concern, because the computational treatment of ASL is a very complex undertaking. Two points particularly stand out: First, ASL is a highly inflected language, resulting in too many appearances of inflectional variants to model them all separately. Second, in ASL events occur both sequentially and simultaneously. Unlike speech recognition, ASL recognition cannot consider all possible combinations of simultaneous events explicitly, because of their sheer number. As a result, the computational treatment of ASL is much more complex than the computational treatment of spoken languages.

Reducing the complexity of the task requires a two-pronged approach, which encompasses work on both the modeling and the computational sides. On the modeling side, I tackle the many appearances by breaking the signs down into their constituent phonemes, which are limited in number. I use the Movement-Hold phonological model for ASL as a guideline, and extend the parts of it that are not directly applicable to recognition systems. In addition, I recast it to describe simultaneous events in independent channels, so that it is no longer necessary to consider all their possible combinations. The result is a significant reduction of the modeling complexity.

On the recognition side, I pose parallel hidden Markov models (PaHMMs) as an extension to conventional hidden Markov models. I develop a PaHMM recognition algorithm

specifically geared toward the properties of sign languages. PaHMMs are the computational counterpart to modeling simultaneous events in independent channels, and allow putting them together on the fly at recognition time, instead of having to consider them *a-priori.*

I validate the modeling approach and the PaHMM recognition algorithm in a pilot study with with experiments on 53-sign and 22-sign data sets. In the PaHMM experiments, the independent channels consist of the hand movements of both hands, and the handshape of the strong hand. The results demonstrate the viability of both the phoneme modeling and the description of simultaneous events in independent channels.

Contents

List of Tables

List of Algorithms

List of Figures

Glossary

5-hand(shape) the handshape with all fingers extended and spread.

abduction angle the spread angle between two fingers.

agreement the presence of morphological inflection that marks correspondence of one word with another in gender, number, or person.

articulatory features the decomposition of a sign's articulation (such as handshape, hand orientation, location) for a single movement or hold segment.

channel a single aspect of the sign's configuration, such as the handshape, or hand movement.

citation form the form in which a sign is listed in the dictionary.

configuration the handshape, hand orientation, and location that describe a sign at a particular time.

feature vector a vector of numeric values that represents the data signal at each frame. It constitutes the input to the HMM recognition algorithm and can, for example, contain the 3D positions of the hands, the velocities, or a more elaborate representation.

hand configuration the handshape and hand orientation.

HMM acronym for hidden Markov model, a type of statistical model used for recognition.

hold the hand is held stationary for a brief amount of time; no aspect of the configuration changes.

inflection a morphological affix that indicates a grammatical function, such as number, case, gender, and tense.

neutral space the space in front of the signer's torso, where signs with an indeterminate location are articulated.

metacarpophalangeal joint the finger joint closest to the palm.

morpheme minimal unit of meaning. Among others, these can be whole words or signs without affixes, or can be affixes by themselves.

movement segment where some part of the configuration changes, such as location, or handshape change.

MPJ see metacarpophalangeal joint.

observation sequence a sequence of data frames (e.g., from speech or sign language); these are modeled with HMMs. Equivalent to a sequence of feature vectors.

PaHMM parallel HMMs; multiple HMMs are modeled independently from one another, whose probabilities are multiplied.

phoneme the smallest contrastive articulatory unit in a language. Phonemes compose and distinguish morphemes (minimal units of meaning).

PIJ see proximal interphalangeal joint.

proximal interphalangeal joint the finger joint closest to the MPJ.

strong hand the hand that performs the one-handed signs and the major component of two-handed signs. Typically the right hand of right-handed people. Also called the dominant hand.

token passing algorithm an alternative formulation of the Viterbi algorithm, which obtains the most probable state sequence through an HMM.

V-hand(shape) the handshape with the index and middle fingers spread and extended in the shape of a "V," with all other fingers closed.

weak hand the opposite of the strong hand. Also called the nondominant hand.

X segment segment that functions *conceptually* like a hold, but the hand need not actually be held stationary.

List of Symbols

a_{ij}	the transition probability from one HMM state to the next one.
$a_{ij}^{(c)}$	the transition probability from one HMM state to the next one in channel c.
$a_{Q_t Q_{t+1}}$	the transition probability between the states Q_t and Q_{t+1}.
$a_{Q_t^{(c)} Q_{t+1}^{(c)}}$	the transition probability between the states $Q_t^{(c)}$ and $Q_{t+1}^{(c)}$.
$b_i(k)$	the output probability distribution of state S_i.
$b_{Q_t}(k)$	the output probability distribution of the state denoted by Q_t.
C	the number of channels.
λ	an HMM.
$\lambda^{(c)}$	an HMM in channel c.
M	the number of hypotheses used in the PaHMM recognition algorithm.
N	the number of states in an HMM.
O_t	the tth observation frame in an observation sequence.
$\mathbf{O}$	an observation sequence.
$\mathbf{O}^{(c)}$	an observation sequence in channel c.

$\mathbf{O}_w^{(c)}$	the observation sequence in channel c for segment w.
$O_t^{(c)}$	the tth observation frame in channel c.
$\omega_w^{(c)}$	the weight of segment w in channel c.
π_i	the probability of an HMM starting in state S_i.
Q_t	the tth state in a sequence that traces a path through an HMM.
$Q_t^{(c)}$	the tth state in a sequence that traces a path through an HMM in channel c.
$\mathbf{Q}$	a state sequence that generates $\mathbf{O}$.
$\mathbf{Q}^{(c)}$	a state sequence that generates $\mathbf{O}^{(c)}$.
$\mathbf{Q}_w^{(c)}$	the state sequence in channel c for segment w that generates $\mathbf{O}_w^{(c)}$.
S_i	a state in an HMM.
$S_i^{(c)}$	a state in an HMM in channel c.
T	the number of frames in an observation sequence.
$\text{tok}_t^{(c)}(i)$	the token in HMM state $S_i^{(c)}$ in channel c at frame t.
W	the number of segments into which an observation sequence is split.

Chapter 1

Introduction

Computers still have a long way to go before they can interact with users in a truly natural fashion. From a user's perspective, the most natural way to interact with a computer would be through a speech and gesture interface. Although speech recognition has made significant advances in the past ten years, gesture recognition has been lagging behind. Yet, gestures are an integral part of human-to-human communication and convey information that speech alone cannot [46]. A working speech and gesture interface could entail a major paradigm shift away from point-and-click user interfaces toward a natural language dialog-and spoken command-based interface.

Sign language recognition enters the picture in four ways. First, a speech-based interface would leave behind those deaf people, who depend on sign language as their primary mode of communication. There is a sense of urgency because of the recent, steady improvements in speech recognition. Unless sign language recognition arrives at the same level of performance as speech recognition, accessibility of computers will become a major issue for the deaf.

Second, a working sign language recognition system could help make deaf-hearing interaction easier. Particularly public functions, such as the courtroom, conventions, meetings, and so on, could become much more accessible to the deaf.

Third, sign language recognition systems could help with the automation of transcriptions. Almost always one of the first steps that needs to be taken with data for linguistic research is to provide a transcription. This task is tedious at best, and often it is very difficult for humans to spot fine nuances consistently. Current sign language recognition systems are still a long way off from being able even to match human transcription performance, but if they arrive at that point, they could provide some very useful backup.

Fourth, gesture recognition in itself is a difficult problem, because gestures are much less constrained than signed languages. Still, gestures take place in the same visual modality as sign languages, and the latter possess a high degree of structure, thanks to their status as full-fledged languages. There is a unique opportunity for sign language recognition systems to bridge the gap to gesture recognition systems, especially because sign language does make use of gesture. A prime example that highlights the relationship between gesture and sign language is the concept of *classifier signs* [71]. These are signs with a distinct handshape, such as one representing a person, which then trace out a path describing the movements or actions that the referent takes. The handshape is a linguistic element of sign languages, subject to the grammatical constraints thereof, whereas the path is free-form, and thus provides an example of blending gesture with sign language.

As a result, a fully capable sign language recognition system must be able to solve the problems of distinguishing among signs, gestures, and meaningless movements, as well as the problem of interpreting what the gestural aspects of signs mean. These two problems are among the hardest to be found in the entire gesture recognition field. From this point of view, gesture recognition can benefit from sign language recognition, because the structure of signed languages provides some anchors that may help solve these problems in sign language recognition first.

In this thesis I present a framework for continuous sign language recognition that is intended to scale well to large vocabulary sizes. To put the contributions and the goals of this

thesis into perspective, I now first give a high-level overview of what sign language recognition is all about, and explore the parallels with speech recognition, before discussing the goals in detail.

1.1 High-Level Overview of Sign Language Recognition

Sign language recognition, as described in this thesis, constitutes only a specific part of a much larger system. The goals of this larger system — which people often refer to as "sign-to-text" — are very similar to speech-to-text systems. On the one end, a person expresses something in sign language, and on the other end, the systems delivers a written language rendition of what the person signed. Superficially, such a system seems to be no different from a speech-to-text system (see also the next section), but in reality it has to contend with three disparate major tasks across multiple stages, as schematically shown in Figure 1.1 on the following page. Note that this system is an ideal; the current state of the art is nowhere near what is shown in this figure.

At the beginning, in the first stage, stands the signer, who is captured on video. This video is fed into the computer in the second stage. The third stage then poses the first major task: extracting the salient information about the signer from this video (such as the 3D positions of the hands, the joint angles of the fingers, and facial expressions), sampling it at discrete time intervals, and condensing the samples into a time-varying data signal. This information about the signer is called the **feature vector** in recognition-speak. Consequently, the data signal is a series of feature vectors, each of which represents a sample of the utterance at a particular time.

The fourth stage poses the second major task: recognizing the signs that the data signal actually represents. The result of this stage is some kind of annotation of what sign took place at what time in the signal. It is important to realize that this annotation has nothing to do with English, or any other spoken language. At this point all that the system has is

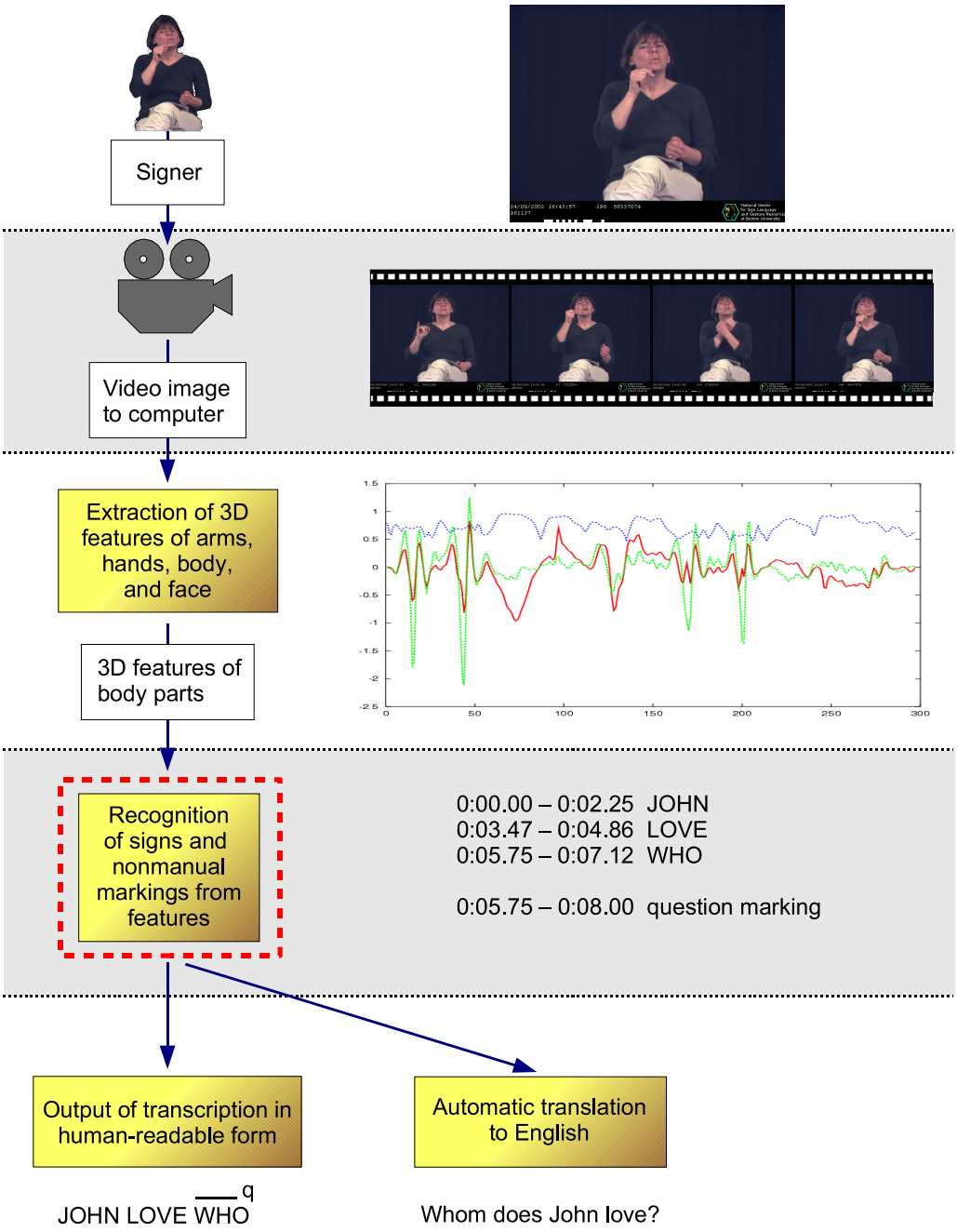


Figure 1.1: High-level overview of a complete sign language recognition system. This thesis addresses the second last stage (recognition of signs from features). Source of the sign language snapshots: [1].

a set of time intervals along with information on which sign took place in each of these intervals.

The fifth and last stage poses the third major task: rendering the annotation as a human-readable transcription for linguistic research, or translating it into English or some other language. Because signed languages are very different from their respective spoken language counterparts (ASL is different from English, German Sign Language is different from German, etc.), automatic translation of this annotation into English poses challenges similar to the ones posed by automatic translation between two spoken languages.

Each of these three major tasks constitutes a large research area of its own. This thesis discusses only the second task (the fourth stage): recognition of the signs from the data signal. To reiterate, the input is a series of 3D feature vectors representing the signs, and the output is a description of what sign took place when. This task is also the one most closely related to what speech-to-text systems do. For extracting the data signal from video see [37, 35, 36, 20, 19, 47, 27, 28]. For translation between English and ASL, see [85, 84].

1.2 Parallels to Speech Recognition

It is no coincidence that I frequently mention speech recognition (or speech-to-text systems) in this thesis. There are many parallels between sign language recognition and speech recognition. Drawing on the decades of research in in the latter field allows the sign language recognition field to hit the ground running, instead of having to recreate every last detail from scratch and having to solve every problem anew. Yet, there should be no illusions about the current state of the art in sign language recognition: it is decades behind speech recognition, and will probably remain so for the foreseeable future. For all these reasons, and to put the contributions of this thesis into perspective, I now make a brief excursion into the speech recognition field and its history.

Fundamentally, the input and output of speech recognition and sign language recognition systems are similar. In speech recognition the input signal is a sequence of feature vectors representing the acoustic properties of an utterance. The system then attempts to recognize the utterances from the feature vectors, and spits out an annotation of what word occurred when.

A raw audio waveform, however, is at best a confusing jumble of information, from which it is very hard to recognize anything useful. Therefore, the first thing that the system does is a transformation of the audio signal into a representation that is more suitable for recognition. There is a large body of research on these transformations; two examples of such transformations are filter bank methods based on the Fast Fourier Transform [15] and linear predictive coding, which reduces the size of the encoding by predicting the values based on the history of the signal [60, 59]. This aspect of speech recognition is not particularly helpful for sign language recognition, because audio and video data are so different.

The next aspect, in contrast, helps a lot. Once the data have been suitably encoded as a series of feature vectors, it is time for the actual recognition of the utterances. The biggest challenge in recognizing the feature vectors is their variability. Even when the same person says the same thing twice in a row, the acoustic signal will still not come out exactly the same twice in a row.

Initially, template matching was a popular method to account for variability. With this technique the feature vectors are compared against a predetermined stored set of templates, where the closest match determines the recognition result. Nowadays, statistical models, most notably hidden Markov models (HMMs) [58], dominate the field; see also Chapter 3. In this approach speech is represented by a set of probabilistic models, one HMM per word, and recognition consists of finding the set of models that could have generated the input signal with the highest probability; that is, the most likely match between a particular model and the input. Using probabilities instead of templates enables the system to deal

with variations in a robust and straightforward manner.

HMMs also have the advantage that they can be trained automatically, so there is no need to adjust their parameters by hand. Basically, the training process functions as follows: The system is given a representative set of labeled utterances, called the **training data (set)**, such that it knows what each utterance means. Then it adjusts the parameters of the HMMs using well-defined mathematical formulae, such that their probabilities are maximized for the training data. The idea is that later, when the recognition system is live, it will encounter only utterances that are similar enough to the trained ones that the HMM probabilities are a good match.

The counterpart to the training data set is the **test data (set).** It is useful when evaluating a recognition system for recognition accuracy. It, too, is a representative collection of utterances that is fed to the recognition system, so as to check how good its recognition rates are. To make sure that the evaluation results are representative of the accuracy of a live recognition session, it is imperative to keep the training and test data separate at all times, and not to use any material from the test data for training. Only this strict separation ensures that feeding a recognition system with the test data set is comparable to running a recognizer with live user input.

These concepts apply directly to sign language recognition. Because we already know from speech recognition that HMMs work well and can be trained easily, it makes sense to use HMMs in sign language recognition, as well. Moreover, all the algorithms for training and recognition are already well-known (see also Section 3.1.2), so sign language recognition gets a considerable head start over early speech recognition systems here.

There are two basic types of speech recognition: isolated recognition, and continuous recognition. The former means that each word is spoken by itself, out of context, with clearly marked pauses between each word. The latter means that the words are spoken as sentences, just as in natural speech, with no pauses in between. From a theoretical standpoint, continuous speech recognition is much more interesting than isolated speech

recognition, because it has to deal with a variety of problems and phenomena that isolated recognition does not.

Two examples shall suffice here: First, if there are no clearly marked boundaries between words, how does the recognition system know where one word ends and the next one begins? The surprising answer is that an HMM-based system does not have to know at all, because of the properties of the HMM recognition algorithm, which I describe in Section 3.1.2. This property is very desirable, because explicit temporal segmentation of an utterance — both in speech and sign language recognition — is a very difficult problem. Second, because continuous recognition deals with entire sentences, it has to handle **inflection** of words. This process changes a word according to a grammatical function; for example, the word "play" gains the suffix "-ed" in the past tense.

Inflection causes lot of complications in recognition systems, because now a single word can generate many different forms; in some languages there are more than in others. It is impractical to use one HMM per form, because there are simply too many of them. It would be very difficult to get enough examples of all of them to train the HMMs in a reliable manner. Speech recognition research solved this problem by breaking down each word into its fundamental building blocks, **phonemes,** an approach inspired by the linguistics of spoken languages. The idea is that the number of phonemes in a language is a finite; moreover, it is actually very small, yet every word can be built out of phonemes. Then, in principle, there is one HMM per phoneme. A phoneme breakdown was one of the key ideas that made large-scale speech recognition systems feasible [40].

Continuous sign language recognition can draw on the same ideas. The goal of this thesis is to lay the groundwork for a systematic approach to breaking down the signs into their constituent phonemes, so as to make large-scale recognition systems feasible in the future. However, the parallels to speech recognition only go so far, and it turns out that accomplishing this breakdown is much harder in the sign language recognition field than in the speech recognition field. It is for this reason that sign language recognition is currently

so far behind the state of the art in speech recognition.

In a sense, this thesis mirrors the early work on speech recognition. To simplify the problem, such early work considered only small vocabularies with a restrictive set of assumptions about the utterances, and tested out the various approaches that eventually led to large-scale systems. I impose similar kinds of restrictions and assumptions, which I address throughout the remainder of this document. I now discuss the goals of this thesis in more detail and why these are so difficult to attain.

1.3 Goals

Continuous sign language recognition systems are — analogous to continuous speech recognition — theoretically far more interesting than isolated sign language recognition systems, because the latter are less complex to model. A lot of the complexity of continuous recognition systems arises out of putting the signs together in a sentence, and not out of the signs themselves [68, 72, 73]. For example, the problem of handling transitions between two signs (which I discuss in Section 2.4.1), arises only when multiple signs appear in sequence. For the remainder of this thesis, unless otherwise noted, I talk about continuous sign language recognition.

As mentioned in the previous section, the main challenge in continuous sign language recognition is to find a modeling paradigm that is powerful enough to capture the language, yet scales to large vocabularies. Signed languages are especially highly inflected, which means that each sign can appear in many different forms, depending on subject and object agreement [71, 45]. Thus, it is futile to model each form separately — there are simply too many of them. Instead, sign language recognizers must break down the signs into their constituent phonemes to capture the commonalities among all signs.

This idea sounds great in principle, but is actually devilishly difficult to implement. Modeling the phonology of sign languages is much more difficult than modeling the

phonology of spoken languages. A major cause of this difficulty can be expressed in just two words: *simultaneous events*. In speech, the phonemes appear sequentially, whereas in signed languages the phonemes can appear both in sequence and simultaneously. For example, a sign can consist of two hand movements in sequence, but the handshape and hand orientation can change at the same time. As a consequence, there is a large number of possible combinations of phonemes that can occur in parallel. Attempting to capture all of them statically — for example, by training an HMM for each combination — would be futile for anything but the smallest vocabularies. In addition, the larger the number of models, the longer it will take the system to recognize sign language sequences. Thus, it is important to develop sign language recognition systems that are capable of managing the modeling complexity, and the computational complexity that is induced by the former.

The modeling complexity of sign languages, and how to reduce it, is the central theme of this thesis. Tackling it requires a two-pronged approach with contributions both from sign language linguistics and HMM theory. On the linguistics side the phonological models that exist for sign languages are often incomplete, and, worse, unlike for spoken languages there is no consensus on what exactly constitutes a phoneme [17]. As a result, it is not possible to apply research from linguistics directly; rather, the phonological models require some modification before they are suitable for recognition systems. On the HMM theory side, conventional HMMs are unsuitable for modeling the simultaneous aspects of sign languages, because these would require modeling of all possible combinations of phonemes. Some extensions to HMMs have been proposed in the past for modeling simultaneous processes [9, 25, 6, 30], but most of these fall short for the same reason. Hence, it is necessary to make additions to the HMM algorithms in a manner compatible with the requirements for sign languages.

In this thesis I present a framework for continuous sign language recognition. It addresses both topics by example of American Sign Language (ASL) recognition — how to extend phonological models of ASL in a manner suitable for recognition systems, and how

to extend HMM frameworks to allow the modeling of the simultaneous aspects of ASL. On the linguistics side, I discuss the Movement-Hold phonological model [44], which is ideal as a starting point for HMM-based recognition, because it emphasizes the sequential aspects of ASL. Nevertheless, it falls short when it comes to modeling the simultaneous aspects for a recognition system without getting bogged down in the complexity of the modeling task. I show how breaking up the simultaneous aspects of ASL into independent channels helps overcome this shortcoming.

On the HMM theory side, I introduce parallel HMMs (PaHMMs) as the counterpart to modeling the simultaneous aspects of ASL in independent channels. PaHMMs have the property of being capable of modeling the channels independently from one another. This property contributes the single greatest step toward reducing the modeling complexity of ASL, but it is not without cost; for the independence assumption is unlikely to hold fully in all situations. Part of the goal of this thesis is to see whether making this assumption is practical; that is, whether the experimental results justify it.

The main goal of this thesis is to develop an ASL recognition system that has the potential to scale to large vocabularies, and to do a preliminary validation of this framework with some example data sets. Because a large corpus of ASL signs suitable for recognition experiments is not yet publicly available, and because of the amount of work involved in collecting and annotating a truly large data set, validating the framework with larger data sets is beyond the scope of this thesis.

It is important to realize that the experiments described in this thesis are just a pilot study aimed at showing that the framework has potential. I make a restrictive set of assumptions about the data sets; in particular, I do not use inflection, and the data sets were not collected from native signers. However, I believe that the framework is sufficiently powerful to allow for relaxing most of these assumptions, one by one, in future work and validation.

As I mentioned in Section 1.1, this thesis is concerned only with the actual recognition

stage of a complete sign-to-text system. For this reason, instead of working with cameras and computer vision applications, I work with 3D data that I collected with an Ascension Technologies MotionStar™ system and a Virtual Technologies Cyberglove™. There is no reason why these could not be provided by a computer vision tracking system in a future application, but discussing them is beyond the scope of this thesis. The output of the recognition system is a representation of what sign took place when, which then could be further translated into English (a very difficult problem in its own right, which is addressed in [85]), or transformed into a human-readable transcription suitable for further processing. This thesis does not pursue these two applications further.

The scope of this thesis is restricted to the lexical aspects of ASL, so I do not discuss ways to recognize facial expressions, other nonmanual features of ASL, and the gestural elements of signs. Just modeling the lexical aspects in a manner that scales well to large vocabularies is challenging enough in itself. It is my hope, however, that eventually this framework will be extended to include the gestural aspects, as well.

The reasons for choosing ASL have both a simple and a practical side. The simple reason is that at the time of writing this thesis, I have been using ASL for more than 7 years, and thus was able to provide the data sets myself. In addition, it allowed me to ensure that the data set was compatible with the assumptions and requirements that I make during the the preliminary validation of the framework. Nevertheless, it needs to be stressed again that future validation has to be done with native signers.

The practical reason is that ASL is the most-researched sign language on the planet, with the largest body of literature. In particular, the major phonological models that have been proposed for sign languages are based on ASL. I expect nevertheless that the methods that I describe in this thesis easily apply to other sign languages beside ASL. The vocabulary and the grammatical constructs are different across sign languages, but the underlying mechanisms that make sign language recognition such a fascinating and challenging research topic are not.

The remainder of this thesis is organized as follows: I discuss related work. Then I address the phonological modeling side of the recognition framework. In the chapter after that I address HMM theory and discuss extensions to PaHMMs to fit in with the modeling approach from the previous chapter. Then I provide experimental results that back up the arguments presented in the modeling and HMM theory chapters. Finally, I discuss how the recognition framework would need to be adapted to native signers, present my conclusions, and touch on possible avenues for future work.

1.4 Related Work

There are many different approaches to gesture recognition, such as HMM-based approaches [40], hierarchical approaches [18], maximum-likelihood approaches [64], and representation space approaches [5, 14]. For coverage of gesture recognition, the survey in [53] is an excellent starting point. Other, more recent work is reviewed in [81]. In addition, the gesture recognition work by Y. Nam and K. Y. Wohn [49, 48] used three-dimensional data as input to HMMs for continuous recognition of gestures. They introduced the concept of movement primes, which make up sequences of more complex movements. The movement prime approach bears some similarities to the phoneme-based approaches to modeling ASL in [76, 75, 77] and in this thesis. The work on iconic gestures in [65] is also of particular relevance, because it tackles the very difficult problem of associating a gesture with the class of objects that it stands for. Work on classifier signs is likely to encounter similar problems, especially with respect to the movement component.

Early work on sign language recognition focused on isolated recognition with clear pauses after each sign, but the research focus has now shifted to continuous recognition. In the area of isolated sign language recognition, R. Erensthteyn and colleagues used neural networks to recognize fingerspelling [23]. M. B. Waldron and S. Kim also used neural networks to recognize a small set of isolated signs [78]. They used Stokoe's transcription

system [70] to separate the handshape, orientation, and movement aspects of the signs.

M. W. Kadous used Power Gloves to recognize a set of 95 isolated Auslan signs with 80% accuracy, with an emphasis on computationally inexpensive methods [33]. He later extended his work to a general temporal classifier in [34]. K. Grobel and M. Assam used HMMs to recognize isolated signs with 91.3% accuracy out of a 262-sign vocabulary. They extracted 2D features from video recordings of signers wearing colored gloves [29].

A. Braffort described ARGo, an architecture for recognizing French Sign Language. It attempted to integrate the normally disparate fields of sign language recognition and understanding [8]. Toward this goal, S. Gibet and colleagues also described a corpus of 3D gestural and sign language movement primitives [26]. This work focused on the syntactic and semantic aspects of sign languages, rather than phonology.

C. Wang, W. Gao, and J. Ma described a large-scale HMM-based isolated recognition system for Chinese Sign Language with a very impressive vocabulary size of more than 5000 signs [79]. They used some tricks from speech recognition, such as clustering Gaussian probabilities, and fast matching, to achieve real-time recognition and recognition rates of 95%. These results show that some of the fast speech recognition algorithms directly carry over to sign language recognition. They collected the data with magnetic tracking systems and Cybergloves.

Most work on continuous sign language recognition is based on HMMs, which offer the advantage of being able to segment a data stream into its constituent signs implicitly. Such work thus bypasses the difficult problem of temporal segmentation of the data signal entirely.

T. Starner and A. Pentland used a view-based approach with a single camera to extract two-dimensional features as input to HMMs with a 40-word vocabulary and a strongly constrained sentence structure [67]. They assumed that the smallest unit in sign language is the whole sign. This assumption leads to scalability problems, as vocabularies become larger. In [69] they applied their methods to wearable computing by mounting a camera

on a hat.

G. Fang and colleagues proposed an approach to signer-independent continuous recognition based on an integration of simple recurrent networks (SRNs) and HMMs [24]. They used the SRNs to segment the continuous sentences into individual signs. The recognition rates were 92% over a test data set of 100 sentences with a 208-sign vocabulary.

H. Hienz and colleagues used HMMs to recognize a corpus of German Sign Language [32]. Their work was an extension of the work by K. Grobel and M. Assam in [29]; that is, it used colored gloves, and it was 2D-based. They also experimented with stochastic bigram language models to improve recognition performance. The results of using stochastic grammars largely agreed with the results in [72] and in this thesis. B. Bauer and K.-F. Kraiss from the same group later extended the framework to break down the signs into smaller units. These units were unlike phonemes, however, because they were determined computationally via clustering, instead of being determined linguistically. For this breakdown they achieved an accuracy of 92.5% in isolated sign language recognition experiments [3, 4].

The work most closely related to this thesis was done by R. H. Liang and M. Ouhyoung. They used HMMs for continuous recognition of Taiwanese Sign Language with a vocabulary between 71 and 250 signs [43, 42], which they extracted from a Cyberglove in conjunction with a magnetic 3D tracker. They worked with Stokoe's system [70] to detect the handshape, position, and orientation aspects of the running signs. Unlike other work in this area, they did not use the HMMs to segment the input stream implicitly. Instead, they segmented the data stream explicitly based on discontinuities in the movements. They integrated the handshape, position, orientation, and movement aspects through stochastic parsing and a dynamic programming algorithm at a higher level than the HMMs. The main assumption of their work is that the sign can be represented as a series of postures. Although this assumption is valid in many cases, there are some signs where the movement cannot be captured solely in terms of postures, such as the distinction between the signs

for SIT and CHAIR. See Figure 2.3 on page 32 for pictures of these two signs.

In this assumption lies the main difference between their work and this thesis. The modeling approach in this thesis views movements as an important component in its own right; in fact, it considers movements to constitute the primary component. Another difference is that this thesis considers the modeling of the simultaneous aspects of signed languages a cornerstone, and one of the main challenges standing in the way of building large-scale systems. In contrast, the work by R. H. Liang and M. Ouhyoung simply considers all necessary combinations of simultaneous aspects, which is not likely to scale well.

I now discuss the properties of ASL and how to model them, including the simultaneous aspects.

Chapter 2

American Sign Language Modeling

Before going into detail, it is appropriate to review what exactly this thesis aims to accomplish by modeling ASL, because these considerations heavily influence my modeling approach. The first and foremost consideration is making the modeling and computational complexity as small as possible, which means that there should be as few models as possible, and it should be easy to use them to capture combinations of simultaneous events.

Another consideration is to have dedicated models, such as hidden Markov models (HMMs), for every single point in time on the data signal, even if from a linguistic standpoint nothing interesting may be happening at that particular point in time, or even if some of these HMMs may seem redundant. The reason is that if the data signal can be expressed in terms of a continuous series of HMMs without gaps in between, the recognition problem is greatly simplified, because it is not necessary to segment the data signal into individual parts; see Section 3.1.2 on the Viterbi recognition algorithm for a more in-depth explanation of why this is the case. This consideration also means that for the purposes of implementing a recognition system, the boundary between phonemic modeling and phonetic modeling becomes fuzzy.

ASL **phonology** provides an abstraction of ASL within the scope of a recognition system. This abstraction leads to many key ideas for reducing the modeling complexity.

It cannot be overstated, however, that in the end, it can only provide broad guidance for implementing a recognition system. It often becomes necessary to break the abstraction, and to draw on what is happening on the **phonetic** level. This level describes how a sign is physically pronounced, including the movements that are not explicitly mentioned in a phonemic description of a sign, because they can be inferred. In the following sections I will point out whenever the phonemic abstraction is broken, and what the implications are.

It is also important to note that this thesis constitutes a *pilot project* for a more systematic approach to modeling ASL than previous work, which necessarily limits the scope of what aspects of ASL recognition it already implements. A lot of the discussion on inflection in the following sections is there to make the point that for a fully scalable recognition system it is necessary to model ASL phonology, and that it is necessary to handle simultaneous events in a systematic manner. This thesis is all about establishing such a modeling approach and verifying that is feasible in principle; that is, that it is possible to achieve reasonable recognition rates with phonology-based modeling on a limited data set that does not yet contain inflected signs. Future work needs to do a full verification, by using this approach to capture, among other things, all the different ways in which inflection actually affects the appearance of a sign.

In the rest of this chapter I first give an overview of ASL and the relevant aspects of ASL linguistics, particularly ASL phonology. I then discuss the Movement-Hold phonological model, particularly the aspects that make it a suitable choice for ASL recognition. Next, I discuss extensions to the model to address some of its shortcomings, and a major reformulation of the modeling approach that allows it to capture the simultaneous aspects of ASL in a computationally tractable manner. Finally, I address the handshape.

2.1 Basics

ASL is the primary mode of communication for many deaf people in the USA. It is primarily expressed through the hands, but facial expressions and full-body movements also play an important role, particularly for grammatical functions. In early work by Stokoe, it was proposed that the configuration that describes an ASL sign at any moment consists of five aspects, which are handshape, hand orientation, location, movement, and facial expressions [70]. The majority of the signs are performed with one hand, but there are many two-handed signs, as well. It is often important to distinguish between the two hands of a signer. In the following I use the term **strong hand** to denote the hand that performs the one-handed signs and the major component of two-handed signs. The **weak hand** is the opposite of the strong hand. In the case of right-handed persons, the strong hand is typically the person's right hand, and the weak hand is the person's left hand.

ASL is a highly inflected language; that is, many signs can be modified according to some grammatical function, such as number, subject-verb agreement, and verb-object agreement. They can also be modified to indicate aspect (e.g., fast, slow), repetition, and duration [38]. For the remainder of this thesis it is crucial to note that there is a fundamental difference between inflection in spoken languages and inflection in signed languages. In spoken languages, inflection can be viewed as a sequential process, where the ending of a word is modified, or where one segment of a word is replaced by another. In contrast, in signed languages inflection takes place through both sequential and simultaneous processes. In a single of these simultaneous processes, only one aspect of the sign's configuration is affected, while the other aspects remain the same as in the sign's citation form. For example, in the sign for GIVE, the movement can change to indicate the manner in which the object is being given to another person.

There is another process in ASL, called *classifier incorporation*, that greatly complicates matters. It can occur in a variety of different signs, but is particularly prevalent in

Form of sign	Type of process	Meaning
I GIVE	basic form	I give (basic form)
#1-GIVE *(sign* starts *at location #1 associated with subject)*	sequential	she *(referent associated with location #1)* gives
I GIVE-**#2** *(sign* ends *at location #2 associated with indirect object)*	sequential	I give him *(referent associated with location #2)*
I GIVE-**large-object** *(use both hands with "open" handshape supporting a large, bulky object, instead of single flat hand)*	simultaneous	I give a large *(referent of a class for which handshape is appropriate)* object
I GIVE-**slow** *(move the hand slowly, reluctantly)*	simultaneous	I give something, but do not really want to *(indicated by slow, reluctant manner of movement)*
#1-GIVE-**slow-large-object-#2**	sequential, simultaneous	she reluctantly gives him a large object

Table 2.1: Examples of processes that can change the appearance of a sign and contribute to the enormous complexity of modeling ASL for the purposes of a recognition system. This set is not exhaustive.

the sign for GIVE. In this sign the handshape can change to indicate the type of object being given; that is, the handshape representing the type of object is incorporated into the sign. It is another example of a simultaneous process in ASL, and although distinct from inflection, poses the same kind of problems to a recognition system. Table 2.1 gives an overview of some of the types of sequential and simultaneous processes that can occur in ASL based on the single verb GIVE.

The existence of simultaneous processes in ASL fundamentally alters the nature of the recognition problem, and makes ASL recognition much harder than speech recognition. The reason is that, given a signal representing an utterance, simultaneous processes are

much more difficult to model in terms of their individual contributions than sequential processes, without also triggering a combinatorial explosion of all these contributions. I explore this problem further in Sections 2.5 and 3.2.

Many approaches to sign language recognition in the past have used a "whole-sign" approach that modeled each sign as an unbreakable entity [68, 69, 31, 32, 33, 73, 74, 78]. From this point of view, a single basic form of a sign — such as the sign for GIVE — can generate a myriad of different appearances, each expressing something different about its subject, object, and aspect. As a result, such approaches would be forced to have a separate, distinct model for *each distinct appearance of a sign.* A simple argument shows that such an approach will not scale with the vocabulary size of a recognition system at all:

A look through the general-purpose and specialized ASL dictionaries turns up approximately 6,000 signs[1]. Some signs allow an extremely high number of variations through inflection and incorporation, such as the sign for GIVE — which represents an extreme case — whereas other signs are barely inflected, or not inflected at all, such as the sign for FATHER. For this reason, it is difficult to estimate the actual number of distinct appearances that a sign can take through inflection and classifier incorporation. Moreover, it is difficult to estimate among how many of the distinct appearances a recognition system must actually distinguish. For example, in principle, the sign for GIVE could take an infinite number of starting and end points, but in the context of a conversation, only the starting and ending points that correspond to referents are of interest.

Nevertheless, it is clear that this number is substantially larger than the approximately 6,000 signs in the lexicon. Modeling such a large number of different appearances explicitly is prohibitively expensive, especially when using one of the many classical approaches to pattern recognition (such as HMMs) that, in order to be effective, need to be trained with

[1]The actual number used in everyday conversation is smaller, but the point of this calculation is a worst-case estimate.

multiple examples from each distinct class of appearances.

It follows that developing modeling approaches that can succinctly capture all these different appearances is of prime importance. At the same time it is also one of the major challenges in ASL recognition. Particularly the simultaneous processes make the task of modeling ASL far more complex than the task of modeling spoken languages, but the sequential processes also pose challenges. As the next few sections will explore, it is a nontrivial task to determine how exactly to break up the signs into smaller, simpler parts, with a view toward reducing the modeling complexity. Overcoming these challenges and developing a viable modeling approach is a cornerstone of this thesis.

At this point it is useful to draw a parallel to speech recognition. If speech recognition treated words as unbreakable wholes, as early work on sign language recognition did, it would suffer from the same problem of having to model too many different appearances of words. Speech recognition solves this problem by turning toward the phonology of spoken languages. The key idea is that each word is made up of a *very small, limited number of phonemes.* By expressing words in terms of their constituent phonemes, speech recognition systems need only model this limited number of phonemes, as opposed to every conceivable inflected form that a word can take up in a language. In principle, applying the same idea to ASL recognition — modeling signs in terms of their constituent phonemes — should help reduce the modeling complexity. In practice, however, there are three stumbling blocks along the way that complicate adopting phonemic modeling for ASL recognition systems:

1. Despite the large body of research, ASL phonology is still in its infancy. There is much less consensus for sign language phonology than for spoken language phonology about the basic phonological analysis of the languages.

2. None of the phonological models has been seriously tested with an ASL recognition system in practice. Even some of the sequential aspects of each model are difficult

to apply unchanged to a recognition system.

3. The manner in which phonological models handle simultaneous processes is difficult to adapt to recognition systems. Here the parallels to speech recognition break down almost completely, because speech recognition systems, generally speaking, do not have to contend with simultaneity similar to what is shown in Table 2.1. There are simultaneous events in speech, such as coarticulation effects[2], but in speech recognition it is possible to capture most of them with a sequential abstraction. In general terms, this abstraction works via putting together all possible combinations of simultaneous events and stringing them together, which is currently not practical in the sign language recognition field (cf. Section 2.5).

In the following sections I give an overview on ASL phonology, and address these three stumbling blocks.

2.2 ASL Phonology

This overview on ASL phonology is by no means exhaustive. The goal of this section is to discuss the aspects of phonology that have been applied to sign language recognition in the past, and the aspects that are relevant to sign language recognition. For more information on ASL phonology, see for example [61, 10, 11, 17, 44, 62, 54, 55].

A **phoneme** is defined to be the smallest contrastive unit in a language [71]; that is, the smallest unit that can distinguish morphemes (units of meaning) from another. In English, the sounds *[k]*, *[æ]*, and *[t]* (which may have different realizations in regional dialects) are examples of phonemes, as can easily be seen by comparing the minimal pairs “cat” - “hat,” “bat” - “bet,” and “bet” - “bed.” In ASL, the equivalents of phonemes in spoken languages are the various handshapes, locations, orientations and movements.

[2]Note that there are counterparts to coarticulation effects in signed languages, as well, particularly with respect to the handshape.

As an example, consider the movement of the hand toward the chin in the sign for MOTHER, or the location of the hand in front of the chin at the beginning of this sign (Figure 2.1). The arguments that these two units are examples of ASL phonemes are analogous to the arguments for English: As Figure 2.2 on the following page shows, the signs for MOTHER - FATHER differ only in location (front of the chin vs. front of the forehead), and the signs for MOTHER - GRANDMOTHER differ only in movement (tapping movement vs. arced movement away from chin).

Figure 2.1: The sign for MOTHER. The first picture shows the starting configuration of this sign; the second one shows the ending configuration. The white X indicates contact between the thumb and the chin after each tap. The location of the hand at the chin and the tapping movements are examples of phonemes.

Note that the early work by W. Stokoe preferred the term "cheremes" [70] over "phonemes," but the term "phoneme" has long since become established in sign language linguistics [17, 12, 10, 62, 61]. Just as in spoken languages, the *number of phonemes in ASL is limited* and small compared to the number of signs. The exact number is still a matter of debate and depends greatly on the phonological model used. Stokoe's system [70], for instance, identifies 55 units, whereas Liddell and Johnson's Movement-Hold model identifies more than 100 [44].

(a) MOTHER (b) FATHER

(c) MOTHER (d) GRANDMOTHER

Figure 2.2: Contrast between signs that identify location and movement as phonemes in ASL. (a) and (b) differ only in location; (c) and (d) differ only in movement.

2.3 Stokoe's System

W. Stokoe was one of the first people to shatter the then commonly held belief that signs are unanalyzable entities. He realized that signs can indeed be broken down into smaller parts [70], and used this observation for devising a transcription system. This transcription system assumes that signs can be broken down into three parameters, which consist of the location of the sign (*tabula* or *tab*), the handshape (*designator* or *dez*), and the movement (*signation* or *sig*). The two phonemes mentioned in Figure 2.2 on the page before are examples of the tab and sig in a sign, respectively. Stokoe's system has long since been superseded by more sophisticated models. Nevertheless, it needs to be mentioned for the simple reason that a lot of work in computational sign language linguistics — if it takes advantage of sign language research at all — limits itself to the scope of Stokoe's work, such as [66, 43].

This model has a fundamental weakness in that it assumes that the tab, dez, and sig contrast only simultaneously. That is, it ignores the sequential processes in ASL, such as the ones described in Table 2.1 on page 25. Variations in the sequence of these three parameters within a sign are considered not to be significant, which means that, for example, Stokoe's model cannot distinguish between morphemes with a single movement, and a morpheme in which this movement is repeated, with all other characteristics being equal. As [44] points out, comparing the signs for SIT and CHAIR in Figure 2.3 on the next page reveals that, contrary to the assumptions of Stokoe's model, this difference is significant. Similarly, this model fails to capture reduplication of a movement, such as the movement in the sign for GIVE, which can express a variety of aspectual distinctions, such as repeated actions, habitual actions, and so on [38, 52].

A look at the early sign language recognition system described in [66] reveals that the sequential aspects of sign language have an important effect on the accuracy of the system. This work specifically mentions that the recognizer could get confused about

(a) SIT (b) CHAIR

Figure 2.3: The signs for SIT and CHAIR. They demonstrate that the difference between a single movement and a repeated movement is significant, thus exposing a weakness of Stokoe's model (Section 2.3).

whether a movement had occurred only once, twice, or more, with the result that a lot of words would mistakenly be recognized multiple times in a row. It gives as a possible solution that the recognizer could check whether a word has been repeated; however, this solution still would not be able to handle the contrast between SIT and CHAIR — and other signs involving reduplication — well, because for such signs the number of movements is the sole distinguishing feature. Therefore, sequentiality in ASL really needs to be handled at the modeling level, instead of the recognition level. The work on sign language animation in [84] and machine translation in [85] demonstrates that computational sign language linguistics can greatly profit from looking beyond Stokoe's model and considering sequentiality.

The next section describes the Movement-Hold model, which incorporates the sequential aspects of ASL.

2.4 Movement-Hold Model

S. Liddell and R. Johnson argued convincingly against Stokoe's assumption that there was no sequential contrast in ASL. They went even further and made sequential contrast the basis of ASL phonology [44]; that is, instead of emphasizing the simultaneous occurrence of phonemes in ASL, they emphasized sequences of phoneme segments. Such models are called *segmental models.*

S. Liddell and R. Johnson describe two major classes of segments in their Movement-Hold model in [44], which they call **movements** and **holds.** Movements are defined as those segments during which some aspect of the sign's configuration changes, such as a change in handshape, a hand movement, or a change in hand orientation. Holds are defined as those segments during which all aspects of the sign's configuration remain unchanged; that is, the hands remain stationary for a brief period of time.

Signs are made up of sequences of movements and holds. Some common sequences are *HMH* (a hold followed by a movement followed by another hold, such as GOOD, Figure 2.4 on the following page), *MH* (a movement followed by a hold, such as SIT, Figure 2.3 on the page before, (a)), and *MMMH* (three movements followed by a hold, such as MOTHER, Figure 2.5 on the following page). Movement segments have features that describe the type of movement (straight, round, sharply angled), as well as the plane and intensity of movement. In addition, attached to each segment is a **bundle of articulatory features** that describe the hand configuration, orientation, location, and local movements (I have more to say about the latter in Section 2.5.3). See Figure 2.5 on the next page for a schematic example.

From a linguistic point of view, the Movement-Hold model has several serious shortcomings, particularly the absence of nonmanual features and facial expressions, and the presence of redundancy — the feature bundles, such as the ones in Figure 2.5, are very repetitive. The lack of nonmanual features means that eventually a recognition system

(a) GOOD, beginning *(b) GOOD, end*

Figure 2.4: *HMH* pattern. The sign for GOOD consists of a hold at the chin, followed by a movement down and away from the body (left picture), followed by a hold contacting the weak hand (right picture).

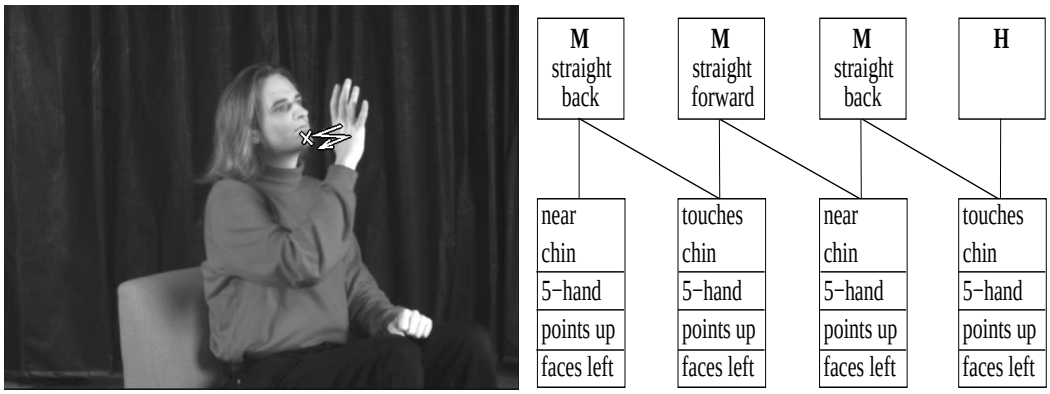


Figure 2.5: Schematic description of the sign for MOTHER in the Movement-Hold model. It consists of the *MMMH* pattern; that is, three movements, followed by a hold. A movement segment always has two feature bundles attached, which describe the configuration before and after that movement. A hold always has only one feature bundle attached, which describes the configuration during the hold.

will have to make additions to the Movement-Hold model to capture the full range of expressiveness of ASL; these are beyond the scope of this thesis.

Its redundancy, however, raises interesting issues for recognition systems. As a concrete example, consider the sign for MOTHER in Figure 2.5 on the preceding page again. This sign consists of a tapping movement toward the chin in the first segment, which is reduplicated in the third segment. In between, in the second segment, is a movement away from the chin ("M straight forward" in the figure). D. Perlmutter argues in [54] that this movement is, in fact, an intra-sign transitional movement that exists only for the purposes of moving the hand back into position, so that it can perform the reduplication. From this point of view, this movement does not belong into the phonemic representation of MOTHER at all, because it is totally predictable. Hence, the phonemic description of this sign should be the MMH pattern, rather than the MMMH pattern.

Under these circumstances the MMMH pattern given in Figure 2.5 is a phonetic representation of the sign, rather than a phonemic representation. As mentioned at the beginning of this chapter, it is essential to have a continuous, unbroken string of models representing the physical data signal of the sign. With the MMH representation, there would be a gap encompassing the movement of the hand away from the forehead. Thus, for modeling ASL within the scope of a recognition system, the phonetic MMMH representation is clearly superior, because it does not contain this gap. It follows that at this moment the recognition system already breaks the abstraction barrier between phonetics and phonology.

This argument brings to mind the question whether it would not be better just to drop ASL phonology entirely and to go straight to ASL phonetics. The answer is no — looking at ASL phonology is still very helpful. It provides valuable insights into how exactly to model ASL (see Section 2.5.2 for such an insight), and it can help constrain the recognition problem[3].

[3]As another specific example of how research into phonology may help, consider the case of intra-sign transitions. On the phonetic level, there is no distinction between intra-sign transitions that have no representation on the phonemic level, and movements that are represented by segments on the phonemic

There are other segmental models, such as W. Sandler's Hand-Tier model [61] and D. Brentari's syllable-based model [10], and further work based on the Movement-Hold model [55]. These models differ primarily in what exactly constitutes a segment, but all share a common set of assumptions about the segmental structure of signs. Describing them in detail is beyond the scope of this thesis. Whether the Movement-Hold model is truly the best choice among the segmental models for ASL recognition systems is an open question that should be resolved in future research.

The Movement-Hold model — or another segmental model, with the caveat that in the end the recognition system needs a phonetic representation of signs — looks like a natural fit for hidden Markov model-based recognition, because there can be a one-to-one correspondence between the segments of a sign and individual hidden Markov models. More specifically, an HMM for a hold segment can naturally capture the body location at that hold, and an HMM for a movement segment can naturally capture the type and direction of hand movement (straight, round) during that segment. Section 3.4 describes this mapping in more detail. However, it is less clear how to fit in the other aspects of the articulatory features, such as the handshape, and the role of the weak hand, because these involve simultaneous aspects of ASL (which I address in Section 2.5).

---

level. Thus, at a first glance, the recognition system would not need to make any distinction between them either.

The work by D. Perlmutter, however, shows that such intra-sign transitional movements do not have the properties that segments normally have [54]. A direct consequence of this work is that for such movements, we do not need to care about the other features, such as the handshape, location, and orientation. Focusing on only one feature, while ignoring the others, greatly reduces the modeling complexity. Thus, in the end it may be advantageous for a recognition system to distinguish between intra-sign transitions and and phonemically represented movements, after all. However, pursuing this point is beyond the scope of this thesis. Yet, I do pursue transitions *between* signs in Section 2.4.1. The results from that section are encouraging, and make Perlmutter's point of view a promising avenue to investigate in future work.

(a) MOTHER (b) movement epenthesis (c) BUY

Figure 2.6: Movement epenthesis. The arrow in the middle picture indicates an extra movement between the signs for MOTHER and BUY , which is not present in their citation forms. Source of images: [1].

2.4.1 Movement Epenthesis

Liddell and Johnson also discuss the transitions between any two signs. They view them as a phonological process in ASL [44]. A phonological process changes the appearance of an utterance through well-defined rules in phonology, but does not change the meaning of the utterance. They call this particular process **movement epenthesis**[4]. It consists of the insertion of extra movements between two adjacent signs, and is caused by the physical characteristics of sign languages. For example, in the sequence MOTHER BUY the sign for MOTHER is performed at the chin, and the sign for BUY is performed in neutral space in front of the trunk. Thus, an extra movement from the chin to the trunk is inserted that does not exist in either of the two signs' citation forms (Figure 2.6). Note that according to the analysis of Perlmutter [54], briefly discussed in the previous section, the same type of movement also exists within signs, but I do not pursue this point any further in this thesis.

[4]Whether the transitions between signs indeed should be viewed as a phonological process, and thus really should be called epenthesis, is open to debate. In spoken language, an example of epenthesis is the possessive case of "Chris," which is "Chris's" and is pronounced with an extra *[ə]* sound between the two "s." For building a recognition system, epenthesis in spoken language poses similar problems to the transitions between signs, and therefore I choose to use the term "movement epenthesis" throughout this thesis.

Movement epenthesis poses a problem to ASL recognizers, because the movements are too long and too visible to be ignored, and thus have a significant effect on recognition accuracy [72]. The appearance of the movement depends on the context of two signs appearing in sequence, so it is not possible to incorporate these movements into the lexical representation. In this respect, movement epenthesis poses problems that are very similar to the problems that coarticulation effects pose to speech recognition systems: In speech, the actual acoustic waveform of a particular phoneme is affected by the context of the preceding and succeeding phonemes. A very popular method in speech recognition to deal with these effects is to use *context-dependent models*, especially triphone models [41, 21]. These model a phoneme in the context of the preceding and the succeeding one, so there is one HMM per valid combination of three phonemes.

Previous work in sign language recognition has taken a variety of ad-hoc approaches to handling movement epenthesis, which range from pretending that these movements do not exist [66, 67, 68, 69], always modeling sequences of two signs in context [73] (this approach was inspired by triphone models in speech recognition, mentioned in the previous paragraph), to using bigram language models, which express the probabilities of which signs can immediately follow another. The additional constraints introduced by bigram language models compensate for the uncertainty introduced by movement epenthesis [31, 32].

The experimental results in Section 4.4.1 show that pretending that the movements do not exist is by far the worst solution. Modeling sequences of two signs in context fares better, but has two significant disadvantages that make it unsuitable for further exploration: First, modeling whole signs in context defeats the purpose of using ASL phonology to reduce the complexity of the modeling task. Second, the modeling complexity of this approach is $O(V^2)$, where V is the vocabulary size, because a separate HMM is needed for each sequence of two signs. With the 6,000 signs in the ASL vocabulary, up to 36 million HMMs would be needed overall, which is even worse than the number of HMMs induced

by inflection in the whole-sign modeling approach (see page 26). Bigram language models, on the other hand, are a viable approach, because even if there are not enough data available to estimate the probabilities of some sequences of signs, it is reasonable to approximate the unknown probabilities with a uniform distribution [82]. As a result, bigram modeling does not suffer from the same complexity problems as modeling sequences of signs in context.

However, using bigram models to tackle movement epenthesis is also a case of applying a good solution to the wrong problem. For all purposes, bigram models provide a computationally inexpensive way to introduce grammatical constraints into a recognition system. Their effect is to reduce the size of the search space. Epenthesis, on the other hand, has nothing to do with the size of the search space, because it is a phenomenon that appears naturally between any two signs. It is just a side-effect of the reduction of the search space that the system also becomes less vulnerable to being thrown off track by the transitions between signs.

In a phonology-based modeling approach to ASL there is no reason why movement epenthesis cannot be handled on the modeling level, instead. The best solution is to model these movements explicitly. In addition, explicit modeling is completely orthogonal to using bigram language models, so both of these approaches can be combined for best recognition performance (see also Section 4.4.1 for experimental results backing up these claims).

Within the Movement-Hold model framework, epenthesis movements simply constitute another movement between two signs, as shown in Figure 2.7 on the following page. Note that according to Perlmutter's analysis in [54], epenthesis would again be something that happens on the phonetic level, and thus modeling it explicitly in a recognition system breaks the abstraction barrier between phonology and phonetics.

In this thesis I choose to model the inter-sign epenthesis movements separately from the normal movement segments, because the former exhibit a key difference compared to

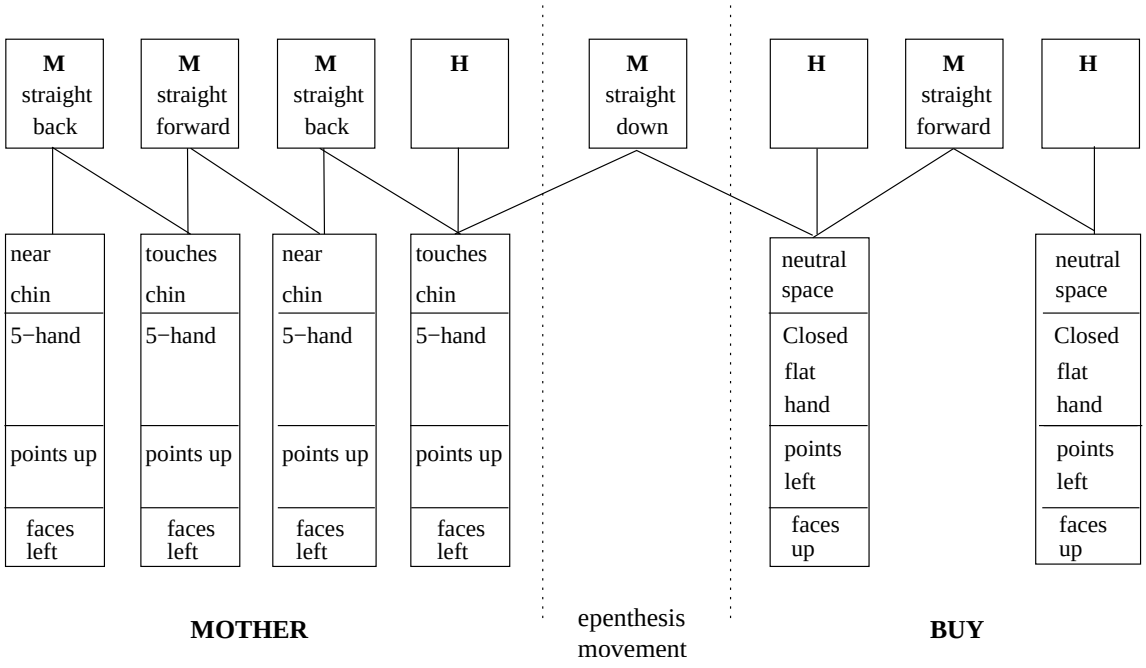


Figure 2.7: Movement epenthesis in the Movement-Hold model is described by just another movement. This diagram denotes the strong hand only. Compare with Figure 2.6 on page 37.

the latter: The epenthesis movement path is completely determined by its beginning and ending locations. In contrast, movement segments need to specify whether the path between two locations is straight, curved, or angled. In addition, it does not matter what type of epenthesis movement the recognition system detects, as long as it does not adversely affect the recognition of the surrounding signs. In other words, there is more leeway for variation in the HMMs that capture epenthesis.

Choosing this approach does not change the fact that the inter-sign epenthesis movements are limited in number. Because they are completely determined by their starting and ending locations, in the worst case there are L^2 different movements to consider, where L is the number of different major locations in ASL. This number is relatively small; for

example, the Movement-Hold model identifies 20 major body locations for ASL. In addition, because of the increased leeway for variation, it is possible to exploit similarities between epenthesis movements to reduce the number of epenthesis HMMs. For example, for practical purposes, there is no difference between the movement from the forehead to the chest, and from the chin to the chest, so they can be captured by the same model.

The most difficult part of following this approach to handling movement epenthesis is deciding which one of the L^2 epenthesis HMMs to use between two signs. In principle, the ending location of the preceding sign and the starting location of the succeeding sign should determine the starting and ending locations of the epenthesis movements, based upon which the choice would be easy. In this thesis I make this assumption and follow through with the experiments accordingly.

Unfortunately, in practice, especially with native signers, things are not so simple, because the locations of successive signs actually influence one another, which directly affects which epenthesis HMMs need to be used. I discuss this topic in more depth in Chapter 5 when I talk about how the recognition framework needs to be changed to accommodate native signers.

2.4.2 Extensions to the Movement-Hold Model

One problem with the original description of the Movement-Hold model is that the articulatory features can be attached to movements, without also being attached to holds. From a linguistic point of view attaching the articulatory features to movement segments without anchoring them with a hold, as well, is implausible, because the former describe how the configuration is *changing*. The articulatory features, in contrast, describe *static* aspects of the configuration.

From a recognition point of view, there is no good way to attach the articulatory features to movement segments, because the system has to estimate fundamentally different

parameters in the segment types: In hold segments, we are interested in the location of the hands relative to the body and require that the hands are held stationary. In movement segments we are interested in the type of movement and do not care about location. As a result, in the *MH*, *MMH*, and *MMMH* patterns the estimation of the body location would take place only at the end of the sign, even though it is relevant both at the beginning and at the end of a sign. The MMMH pattern, as exemplified by the signs for FATHER and MOTHER (Figure 2.5 on page 34), would be particularly troublesome for a recognizer that adopted the Movement-Hold model unchanged. It would go through no less than three HMMs for movements, each with an indeterminate location, before finally encountering the HMM for a hold, which would fix the location. The potential for recognition errors during these three movements would be very high.

From these two points of view it becomes clear that it is not possible to apply even only the sequential aspects of the Movement-Hold model to a recognition system without modifications. There must be some type of segment that allows the modeling of the body location at the beginning of a sign, even if it does not start with a hold. To this end, I add a new type of segment called **"X"**[5]. This segment is conceptually very similar to a hold. The only difference is that, unlike holds, the hand need not be stationary for any amount of time. The sole purpose of this segment is to provide an anchor for the articulatory features. Intuitively, it provides for taking a "snapshot" of the sign's configuration at a particular point in time. Figure 2.8 on the following page shows how the X segments affect the sign for MOTHER. In fact, it should be possible to insert the X segments between every two movements in a sign (Figure 2.9 on the next page), so as to get an even better estimate of the locations in a sign, but this thesis does not pursue this point further.

There is a possible alternative view of X segments, which this thesis does not pursue

[5]I came up with this idea independently of Liddell and Johnson. Yet, the role of the X "segments" seems to be very similar to the the X segments in the latest, as of yet unpublished, version of the Movement-Hold model (personal communication, 1999). It is curious to see how in this case engineering and linguistic concerns coincided.

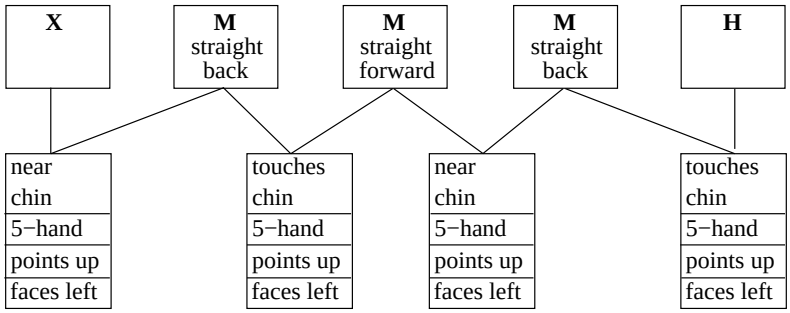


Figure 2.8: The sign for MOTHER with the X segment type added. Compare with Figure 2.5 on page 34. Also see Figure 2.9 for a possibly better model of the sign with multiple X segments.

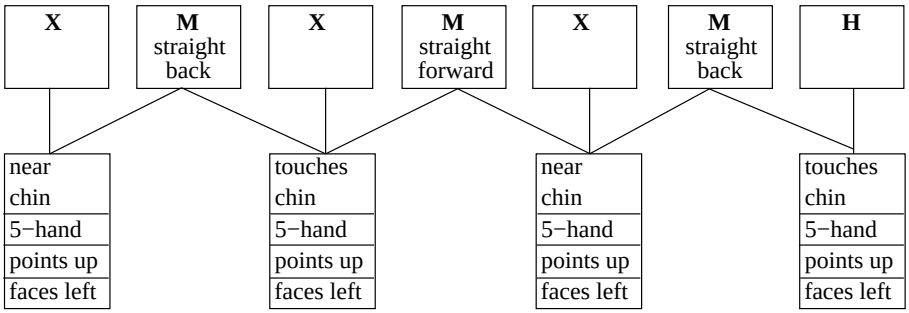


Figure 2.9: An alternative view of the sign for MOTHER with multiple X segments added. This view is perhaps better than the one of Figure 2.8, because it allows for modeling the location before *every* movement in a sign.

any further, either. Instead of simply providing a way to take a snapshot of a location, they could also serve as a representation of a variable body location. The justification for this view comes from the nature of signs that do not start with a hold: their beginning body location is not fully predictable, and strongly depends on the context of the preceding signs. For following up on this view, it would first be necessary to obtain a full computational model of how the locations of signs affect one another, as mentioned in the previous section. For the remainder of this thesis, I assume that the X segments refer to a fixed location, but this assumption could be relaxed in future work.

This section concludes the discussion of the purely sequential aspects of ASL. It does not seem possible to incorporate the other aspects of the articulatory features, such as the handshape, without first tackling the general problem of simultaneous aspects of articulation. I discuss how to model these next.

2.5 Simultaneity

Adding X segments, as described in the previous section, is sufficient for recognizing ASL using only the strong hand movements and locations [72]. Yet, even with this addition, the Movement-Hold model breaks down completely for modeling both hands and their associated handshapes and orientations, which are contained in the articulatory features.

The problem is the sheer number of possible combinations of simultaneous features. Unlike speech, where phonemes occur only in sequence, in ASL phonemes occur both in sequence and in parallel. For example, some signs are two-handed, so both hands must be modeled. In addition, several features can change at the same time, as depicted in the sign for INFORM in Figure 2.10 on the following page.

To get an idea of the magnitude of the problem that recognition systems face here, it is illuminating to consider the completely naïve approach first: what about simply tossing all possible combinations of features together, without regard for linguistic constraints and

Figure 2.10: The sign for INFORM demonstrates how several features in ASL change simultaneously. Both hands move, starting at different body locations. Simultaneously, the handshapes change from a flat, closed hand to an open, partially cupped hand during the sign. Source of images: [1].

interdependencies between features? Then the recognition system would have to look at all possible combinations of handshapes, hand orientations, wrist orientations, and locations and movement types of the strong and weak hands, respectively. This approach leads to a combinatorial explosion, as can easily be seen by multiplying all the numbers of possible respective features together. If, for example, we assume that there are 40 distinct handshapes, 8 distinct hand orientations, 8 wrist orientations, and 20 major body locations, then the number of possible feature combinations, for both the strong hand and the weak hand, would be $(40 \times 8 \times 8 \times 20)^2$, which is more than one billion.

Of course, it would be ludicrous to assume that ASL really exhibits that many combinations, as is evident from the many linguistic constraints, such as handshape constraints [2], and dependencies between handshape, hand orientation, and location (e.g., signs that touch a part of the signer's body with the thumb or a specific finger have only a few orientations that are even physically possible). Yet, many past approaches to sign language recognition have implicitly worked under the assumption that the features can be combined freely without constraints; especially the whole-sign-based approaches have done so, such as [69, 73, 31].

2.5.1 Independent Channels

The combinatorial explosion stemming from the naïve approach highlights the enormous complexity of modeling the simultaneous aspects of ASL. Modeling all these combinations *a priori* is infeasible both from a modeling and a computational point of view. From the former point of view, it would be impossible to collect enough data for all these combinations in a reasonable time frame. From the latter point of view, attempting to identify the correct models and to match them against the sequence of signs to be recognized would take far too long to be of practical use.

To get rid of the combinatorial explosion, there are two possible solutions:

1. Find a computational model for the dependencies and interrelationships among the features.

2. Find a way to decouple the simultaneous events from one another, so that it is possible to look at only one event at a time.

The first solution is currently not practical. Although there is much work on various aspects of ASL phonology, this work is nowhere near being ready for synthesis into a comprehensive computational model. Alternatively, a full statistical analysis of a large annotated corpus of sign language data could help determine these dependencies, but work in this direction is still emerging [1]. This area, however, holds great promise and provides many interesting opportunities for future work.

As a result, for the purposes of this thesis, only the second solution is left. To make modeling the simultaneous aspects of ASL tractable, it is necessary to decouple the simultaneous events from one another. To this end, I make a major break from the Movement-Hold model. Instead of attaching bundles of articulatory features to the X and hold segments, I break up the features into **channels** that can be viewed as being **independent** from one another. From the different aspects of the articulatory features in the Movement-Hold model a channel arises for each of the following, for each hand:

movement channel the channel consisting of the body locations and the movements between the body locations

handshape channel the channel consisting of the handshape

orientation channel the channel for the hand orientation (e.g., pointing up, forward, etc.)

rotation channel the channel for the wrist rotation

The splitting of the features into independent channels constitutes one of the computer-science-related novelties and contributions in this thesis. The major difference to the approach taken in [42] is that I still attempt to model and maintain a segmental structure of the sign in each of these channels (which I discuss in detail in the next section). The idea behind this approach comes from from the decades of linguistic research into ASL, most of which focused on a particular respective aspect of the features, such as the handshape or location.

Note that the list of channels is still very preliminary, as this thesis constitutes just a pilot project to study the feasibility of this approach. Most likely, it will be possible to merge two or more channels because of interdependencies, especially the ones for hand orientation and wrist rotation. Such a merging should be explored in future work. In the experiments in this thesis, I concentrate on the movement and handshape channels, as these contain the bulk of the phonemes that have been described for ASL [70, 2, 54, 56, 44, 61, 12].

Splitting the feature bundles up into independent channels immediately yields a major reduction in the complexity of the modeling task. It is no longer necessary to consider all possible combinations of phonemes, and how they can interact. Instead, it is enough to model the phonemes in a single channel at one time, and just to look at the phonemic and phonetic phenomena in each channel separately. In each channel, the phonemes can be

represented by only a small number of different HMMs, which all belong to the same aspect of the sign's configuration, such as the handshape, or hand movement. Combinations of phonemes from different channels are easy to put together on the fly at recognition time, particularly in conjunction with the parallel hidden Markov model recognition framework that I describe in the next chapter.

The downside of using independent channels is that they entail making a major assumption about the structure of the simultaneous processes in ASL, which in all likelihood is not valid. In essence, these independent channels are a case of an engineering tradeoff, so as to make the recognition problem tractable, versus theoretically correct modeling of ASL. In the case of the independence or dependence between the strong and the weak hand, S. Liddell and R. Johnson argue that the phonological processes of both hands are independent from each other [44], although they note the constraints on the weak hand from [2]. Overall, however, they provide very little data to back up their argument. In addition, even if it were true that the phonological processes are independent, this does not necessarily imply that the physical movements that the hands perform are also fully independent. Nevertheless, the experiments in Section 4.4.3 show that modeling ASL in terms of independent channels yields tangible benefits to sign language recognizers.

I now discuss the structure of the channels in more detail.

2.5.2 Channel Structure

Each channel still consists of movement, hold, and X segments. Conceptually, nothing changes about these three segment types. A hold in a channel means that the aspect of the configuration does not change for a certain amount of time; a movement means that the aspect of the configuration changes to another; and an X segment means that a "snapshot" is taken of the aspect. Figure 2.11 on the next page shows how the sign for MOTHER is represented with this modification. Note that this figure only shows the channels for the

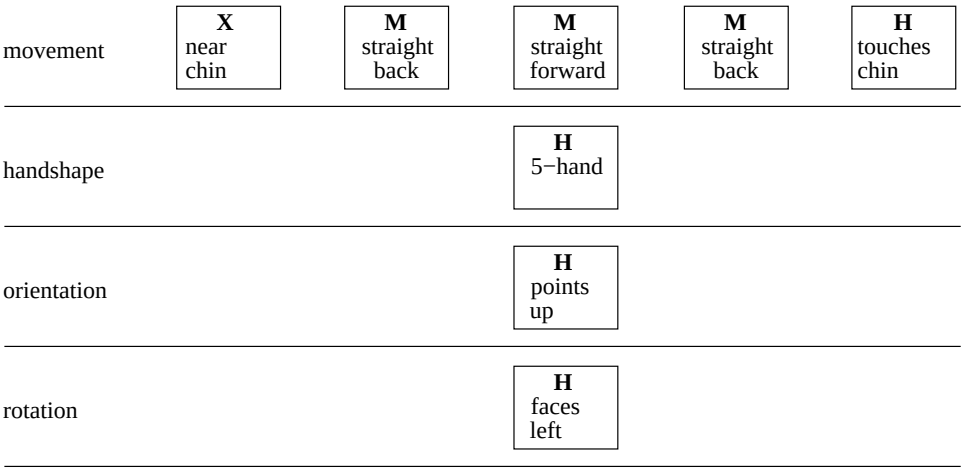


Figure 2.11: The sign for MOTHER, where the different features are modeled in separate channels. The handshape, orientation, and rotation stay the same during the entire sign, so only one hold appears in each of these channels. Compare with Figure 2.8 on page 43.

strong hand, because MOTHER is a one-handed sign. Figure 2.12 on the following page shows the modeling of the sign for INFORM with independent channels, as an example of a two-handed sign where multiple channels change simultaneously. Note that INFORM is an inflected sign, but as mentioned earlier, the work described in this thesis does not yet handle the different appearances that arise out of inflection, so here the ending location of this sign is fixed.

Representing the handshape channel in terms of movements and holds may look innocuous, perhaps even like a logical consequence of my earlier exposition of the Movement-Hold model. However, in fact, it implies a major step away from the phonemic abstraction down to the phonetic level. As a consequence, the diagrams shown in Figures 2.11 and 2.12 are almost wholly phonetic, instead of phonemic, representations. Nevertheless, they have been inspired by the assumptions common to the Movement-Hold and other segmental phonological models, so looking at ASL phonology did provide some necessary guidance toward devising the recognition framework in this thesis.

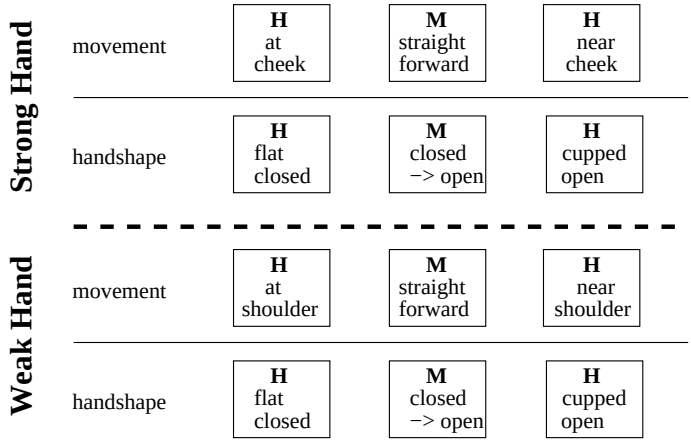


Figure 2.12: The sign for INFORM, where the different features are modeled in separate channels. Note how several channels change simultaneously. For the purposes of this thesis, the ending location of this sign is fixed, even though in reality it is an inflected sign. See Figure 2.10 on page 45 for a picture of this sign.

In the handshape channel, "movement" segments capture a transition from one handshape to another, and "hold" segments imply that the handshape does not change. Although breaking down the handshapes into these two segment types is consistent with the Movement-Hold modeling approach, from a phonemic point of view there is no good reason for this breakdown. Unlike hand movements (such as arcs, straight lines), the handshape transition is normally determined by the handshapes at the beginning and end of this transition; that is, the transition does not add any information to what is already determined by the static handshapes. It is for this reason that this representation needs to be viewed as a phonetic one, as opposed to a phonemic one.

The rationale for choosing this type of representation lies in what I stated at the beginning of this chapter: it is essential to represent the data signal with a continuous series of models, even if parts of the signal do not carry any linguistically relevant information. Modeling every channel phonetically in terms of movements and holds accomplishes exactly this.

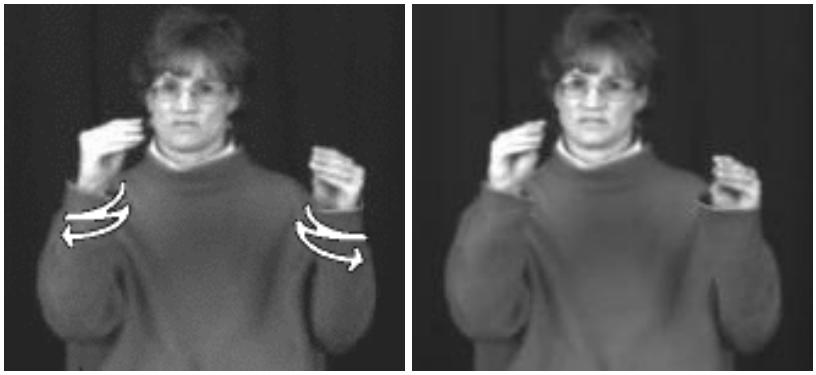


Figure 2.13: Example of a local movement. The sign for SELL consists of wrist rotations. Local movements are modeled phonetically within the recognition framework. Source of images: [1].

2.5.3 Local Movements

At this point it is finally possible to integrate the local (or "secondary") movements described by the Movement-Hold model into the recognition framework. I previously mentioned these on page 33, but until now have ignored them.

Local movements are types of repetitive movements — especially repeated wrist rotations and wiggling of the fingers — that can be superimposed on another movement, or even a hold (in the original sense of the Movement-Hold model, not the phonetic representation from the previous section). Unlike regular movement segments, they do not induce a change in the sign's configuration per se, and the exact number of such movements can be indeterminate. Consequently, the original Movement-Hold model does not represent them in segments of their own [44]. Two examples of local movements are exhibited by the signs for SELL shown in Figure 2.13, and for WHO shown in Figure 2.14 on the following page.

With the representation described in the previous section, modeling such types of signs is straightforward. The wrist rotations in the sign for SELL are modeled as a series of

Figure 2.14: The sign for WHO. The index finger wiggles back and forth, but the hand itself is held. In the Movement-Hold model, this sign consists of a single hold with a local movement in the articulatory features. Source of image: [1].

"movement" segments that represent wrist movements back and forth, respectively (Figure 2.15 on the next page). The handshape wiggles are modeled as a series of movements that describe the handshape changes that take place throughout the wiggle. The indeterminate number of repetitions during the wiggle can easily be handled by allowing the recognition system to loop back through this movement an arbitrary number of times.

In the next section I describe modeling the handshape in more detail.

2.6 Handshape

The handshape is one of the most important parts of ASL, not only to distinguish signs from one another in the ASL lexicon, but also in the role of *classifiers*. These are signs that represent a class of objects collectively, while they trace a path of such an object through space. For example, the classifier that represents a person is the outstretched index finger — typically in an upright position. Likewise, the classifier that represents a vehicle, such as a car, is the 3-handshape, with thumb, index, and middle fingers extended, and the

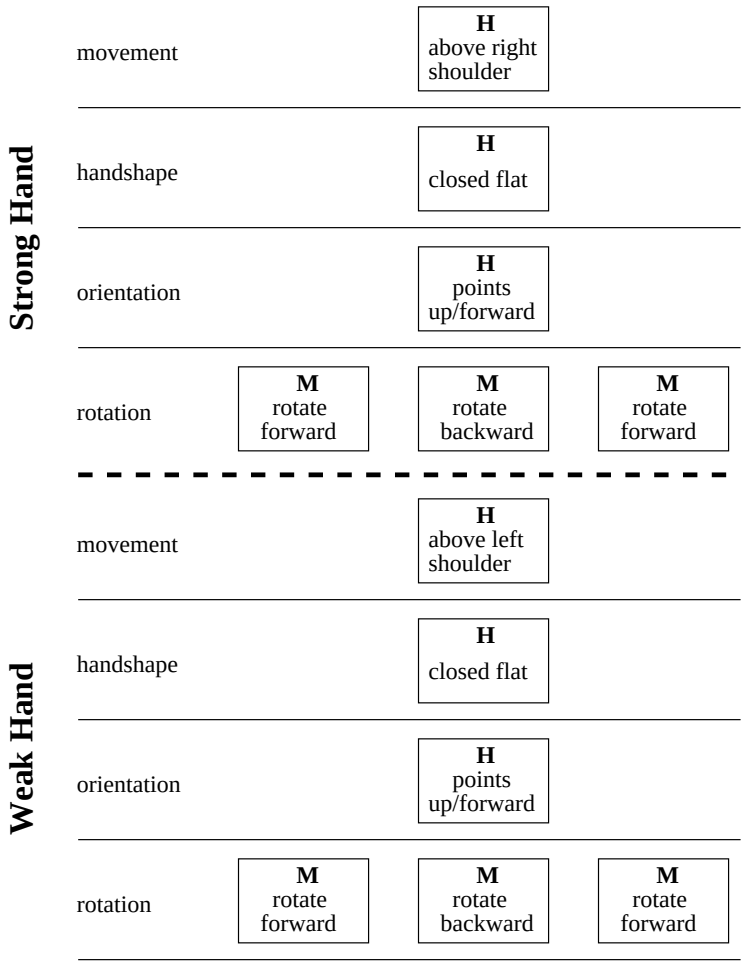


Figure 2.15: Representation of the sign for SELL. It shows an example of how local movements can be modeled in terms of segments. Compare with Figure 2.13 on page 51.

thumb pointing straight up, while the other two fingers point forward. The handshape in classifiers is so important because it forms the distinctive element of the sign, and in many situations it is the only unvarying aspect of the sign's configuration[6]. All other aspects, such as orientation, position, and movement, are free-form and change according to what the signer expresses. Hence, classifiers are highly gestural in nature. They closely mimic what the referent of the classifier actually does, such as walking from one point to another in a zigzag path, falling down, and so on, instead of following some prescribed set of movements from the ASL lexicon.

Thus, recognizing the handshape is of critical importance in a complete recognition system; not only to make the recognition of lexical ASL signs more robust, but also to take the first step toward building a bridge to gesture recognition systems. Yet, comparatively little research exists about the handshape in the sign language recognition literature. The reason is probably that handshape data are difficult to capture, especially the exact constellations of the fingers. In addition, the experimental results in [72, 73, 69, 68] demonstrate that it is possible to recognize many ASL signs without knowing the exact handshape. These results are consistent with experiments on human perception, such as the one in [57].

Most continuous recognition algorithms have worked with 2D data, such as [69, 68, 31, 32], which makes it more difficult to capture the handshape from the outset. In addition, even 3D computer vision algorithms have been unable to capture the handshape with the degree of accuracy required for sign language recognition, unless they used some kinds of markers on the fingers, such as in [22, 31]. Other work has used DataGloves^{TM} or Cybergloves^{TM}, such as the work by Liang and Ouhyoung, and Erenshteyn and colleagues [42, 43, 23]. The work by Liang represents the handshape by quantizing the joint angles into a discrete set of variables, and representing a finite set of handshapes, according

[6]The handshape can change when something is being done to the referent of the classifier. For example, if a car were squished in a crash, the fingers of the classifier would bend to reflect this. Nevertheless, the handshape is what allows identification of the classifier's referent.

to Stokoe's model [70], modified for Taiwanese Sign Language. The work by Erenshteyn and colleagues uses a hierarchy of neural networks to classify the handshape.

Most phonological models for ASL, particularly the Movement-Hold model and Stokoe's model, treat the handshape as belonging to a fixed set, without any further structure or hierarchy. In contrast, the work by W. Sandler [62, 63] and D. Corina [16] treats them as entities that can be broken down further. The advantage of this latter treatment is that it allows for setting constraints and making predictions as to which two handshapes can occur in sequence. One such constraint given by W. Sandler is that during a handshape change within a sign, the same fingers must be *selected* in both handshapes; that is, the fingers that were closed and extended, respectively, in the first handshape cannot switch roles and become extended and closed, respectively, in the second handshape.

These constraints may be useful in restricting the number of models needed to capture the handshape in a recognition system. To see why this restriction could be important, recall that it is advantageous to model the transitions between two handshapes explicitly, because it ensures that any point on the data signal is accounted for with an HMM (cf. the beginning of Chapter 2 and Section 2.5.2). I take this approach in my modeling framework, as exemplified by the representation of the sign for INFORM in Figure 2.12 on page 50: The handshape undergoes a transition from a closed flat handshape to an open, partially cupped handshape, which is explicitly captured in a movement segment. The experimental results in Section 4.3.4 justify this approach.

This approach, however, has a potential problem. In the case of the epenthesis models, it is possible to merge multiple models, and thus to reduce the number of models, by taking advantage of the similarities among some epenthesis movements (such as a movement from the chin to the chest, and a movement from the forehead to the chest). Unfortunately, the same approach is not feasible for reducing the number of handshape change models, because the handshapes are mostly very dissimilar, and because the joint angles of the hand possess many more degrees of freedom than the 3D coordinates of the location.

As a result, restricting the number of handshape transitions can be supremely useful. It does not matter with small data sets, such as the ones that I collected for this thesis, but future work needs to address this issue, and then the constraints can provide a useful starting point.

2.7 Summary

In this chapter I have given an overview of ASL phonology. I have explained the Movement-Hold model in more depth and how it can serve as a useful basis for a recognition system. I also have discussed movement epenthesis — the transitions between signs —, and how it is useful to model them explicitly, in contrast with other work on sign language recognition.

The original Movement-Hold model has two shortcomings, as far as ASL recognition is concerned. The first one pertains to failing to capture the location of signs that start with a movement segment. The second one pertains to the sheer number of phoneme combinations in the feature bundles, which results in enormous computational and modeling complexity. I have shown how to address the first shortcoming by adding the X segment. I also have addressed the second shortcoming by splitting up the features into independent channels, which describe movement, handshape, orientation, and rotation. These independent channels contribute to a large reduction of the modeling complexity. I also have shown that it can be useful to represent signs on the phonetic instead of the phonemic level, and that the phonetic representation is guided by the existing phonological models of ASL.

There are other aspects of ASL, such as facial expressions, the relationship between gesture and sign language, and the use of space to denote referents, but these are beyond the scope of this thesis. In the next chapter I discuss the HMM recognition framework and how to apply the material from this chapter to it.

Chapter 3

Hidden Markov Model Recognition Framework

In this chapter I discuss the recognition framework, particularly what kind of approach to use to perform the actual recognition. It is important to use some kind of approach that can capture the variations that invariably arise in human movements. To this end I describe hidden Markov models, and extensions to them, which make it possible to use them to recognize both the sequential and simultaneous aspects of sign language. I then conclude with a brief description of how the phoneme modeling ties into the recognition framework.

3.1 Hidden Markov Models

One of the main challenges in ASL recognition is to capture the variations in the signing of even a single human. In general, humans never perform exactly the same movement twice, even if they intend to. There are always slight variations from one movement to the next one, so a recognition framework must be able to account for them. The most common approach toward handling such variations is to use some kind of statistical model.

Hidden Markov models (HMMs) are a type of statistical model embedded in a Bayesian framework and thus well suited for capturing variations. In addition, their state-based nature enables them to describe how a signal changes over time, which is ideal for activity recognition.

I now describe the properties of HMMs that are relevant to ASL recognition. I then describe possible extensions to the HMM framework that allow modeling of multiple, simultaneous processes, and go on with a description of Parallel HMMs, which are this thesis's main contribution toward overcoming the problems associated with regular HMMs.

3.1.1 Definition of HMMs

An HMM λ consists of a set with N states $S = \{S_1, S_2, \ldots, S_N\}$, together with transitions between states. The system is in one of the HMM's states at any given time. At regularly spaced discrete time intervals, the system takes an outgoing transition from its current state to a new state.

Each transition from S_i to S_j has an associated probability a_{ij} of being taken. Each state S_i also has an initial probability π_i of the system starting in S_i. Clearly, $\sum_i a_{ij} = 1$, and $\sum_i \pi_i = 1$. In addition, each state S_i generates output $k \in \Omega$ — where Ω is the sample space —, which is distributed according to a probability distribution function $b_i(k) = P\{\text{Output is } k | \text{System is in } S_i\}$. In activity recognition applications, frequently, $\Omega = \mathbb{R}^n$, and b_i is actually defined to be a mixture of continuous probability density functions of the form

$$b_i(O) = \sum_{m=1}^{M} c_{im} G_m(O, \mu_{im}, U_{im}), \tag{3.1}$$

where $O \in \Omega = \mathbb{R}^n$, M is the number of mixtures in state i, c_{im} is the weight of mixture m in state i, and G_m is a Gaussian probability density function with mean $\mu_{im} \in \mathbb{R}^n$ and covariance matrix U_{im}. In such cases the sample space is continuous instead of discrete, but this change has no effect on the equations and proofs related to HMMs, apart from a

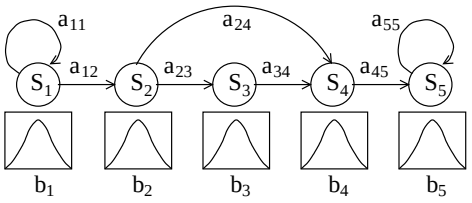


Figure 3.1: Example left-right HMM with its transition and output probabilities. "Left-right" means that transitions occur only from left to right, and never backward. Only transitions with nonzero probabilities are shown.

constant scaling factor[1]. The advantage of using such Gaussian density mixtures is that given a large enough M, they can approximate any other probability density function. The larger M, however, the more training data are required, so in practice most applications either use a single Gaussian, or use only a few mixtures.

A schematic example of an HMM is given in Figure 3.1. The model depicted there is also an example of a model with a *left-right* topology; that is, $a_{ij} > 0$ implies $j \geq i$. In other words, transitions only flow forward from lower states to the same state or higher states, but never backward. This type of topology is the most commonly used one for modeling processes over time.

O is often called the **observation** or **output.** Intuitively, in the case of ASL recognition, O corresponds to the configuration of a sign at a certain point in time. If Ω is multidimensional, O is actually a vector, and is often called the **feature vector,** as first mentioned in Section 1.1. The feature vector can contain such information as positions and velocities of the hands, handshape information, or measurements of the power spectrum in speech recognition. I have more to say about the feature vector in the discussion of the experiments in Section 4.3.

If the sign is sampled at regular intervals then the sign can be represented by a sequence $\mathbf{O} = O_1 O_2 \ldots O_T$, where T is the number of samples taken. This sequence is commonly

[1]The continuous sample space, however, does have an effect on the parallel HMM extension that I describe later. I address this effect in Section 3.3.5.

called the **observation sequence.**

I use $\mathbf{Q} = Q_1 Q_2 \ldots Q_T$, where $Q_i \in S$, to denote a sequence of states associated with the observation sequence $\mathbf{O} = O_1 O_2 \ldots O_T$. The probability that an HMM λ generated this observation sequence via this state sequence is

$$P(\mathbf{O}, \mathbf{Q}|\lambda) = \pi_{Q_1} b_{Q_1}(O_1) \prod_{t=2}^{T} a_{Q_{t-1}Q_t} b_{Q_t}(O_t), \tag{3.2}$$

where π_{Q_1} is the probability of the system starting in state Q_1, $b_{Q_t}(O_t)$ is the probability of state Q_t generating output O_t, and $a_{Q_{t-1}Q_t}$ is the transition probability from state Q_{t-1} to state Q_t. This equation implies two important properties of HMM theory: First, it makes the assumption that successive observations O_t, O_{t+1} are independent. Second, it makes the so-called *Markov assumption* that the transition to the next state depends only on the current state, and not on the entire state history. In practice, time-varying signals and processes are unlikely to satisfy these assumptions fully, but the computational simplicity and elegance of HMMs often makes this trade-off worthwhile. I do not pursue this point any further in this thesis; for more information see [59, 58].

Next I describe the three classic problems from HMM theory, and how they relate to sign language recognition.

3.1.2 The Three Fundamental HMM Problems

The literature defines three fundamental problems in HMM theory [58]:

1. Given an HMM λ, and an observation sequence $\mathbf{O}$, what is the probability $P(\mathbf{O}|\lambda)$ that the HMM λ generated this sequence? *(recognition of isolated signs)*

2. Given an HMM λ, and an observation sequence $\mathbf{O} = O_1 O_2 \ldots O_T$, where T is the number of samples in $\mathbf{O}$, what is the state sequence $\mathbf{Q} = Q_1 Q_2 \ldots Q_T$, $Q_i \in S$, that maximizes the probability $P(\mathbf{Q}, \mathbf{O}|\lambda)$? *(recognition of continuous sign sequences)*

3. Given an HMM λ, and an observation sequence $\mathbf{O}$, how should the parameters π_i, a_{ij}, and $b_i(O)$ of λ be adjusted to maximize the probability $P(\mathbf{O}|\lambda)$ that the HMM λ generates this sequence? *(training of HMMs to recognize specific signs)*

The first problem corresponds to maximum likelihood recognition of an unknown data sequence with a set of HMMs, each of which corresponds to a sign. For each HMM, the probability $P(\mathbf{O}|\lambda)$ is computed that it generated the unknown sequence, and then the HMM with the highest probability is selected as the recognized sign. For computing $P(\mathbf{O}|\lambda)$, let $\mathbf{Q} = Q_1, Q_2, \ldots, Q_T$ be a state sequence in λ. Define the *forward variable* $\alpha_t(i)$ as follows:

$$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, Q_t = S_i|\lambda), \quad 1 \leq i \leq N, \tag{3.3}$$

$$\alpha_1(i) = \pi_i b_i(O_1), \tag{3.4}$$

$$\alpha_{t+1}(i) = b_i(O_{t+1}) \sum_{j=1}^{N} \alpha_t(j) a_{ji}, \quad 1 \leq t \leq T-1. \tag{3.5}$$

Then

$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{3.6}$$

Like Equation 3.2 on the preceding page, these equations assume that the O_i are independent, and they make the Markov assumption that a transition depends only on the current state. This method is called the *forward-backward algorithm* and computes $P(\mathbf{O}|\lambda)$ in $O(N^2T)$ time, where N is the number of states in the HMM. Unfortunately, this algorithm is useful only for recognizing isolated signs, because in order to match each HMM to a sign with this algorithm, it is necessary to know at which points in time the sign begins and ends. Temporal segmentation of a signal, however, is typically a very difficult problem. The solution to the next problem is far more useful for continuous recognition.

The second problem corresponds to finding the most likely path $\mathbf{Q}$ through an HMM λ, given an observation sequence $\mathbf{O}$, and is equivalent to maximizing $P(\mathbf{Q}, \mathbf{O}|\lambda)$ from Equation 3.2 on page 60. Let

$$\delta_t(i) = \max_{Q_1,\ldots,Q_{t-1}} P(Q_1 Q_2 \ldots Q_t = S_i, \mathbf{O}|\lambda), \tag{3.7}$$

$$\delta_{t+1}(i) = b_i(O_{t+1}) \cdot \max_{1\leq j\leq N} \{\delta_t(j)\, a_{ji}\}, \tag{3.8}$$

$$\max_Q P(\mathbf{Q}, \mathbf{O}|\lambda) = \max_{1\leq i\leq N} \{\delta_T(i)\}. \tag{3.9}$$

$\delta_t(i)$ corresponds to the maximum probability of all state sequences that end up in S_i at time t. Equations 3.8 and 3.9 follow from Equation 3.7 by induction on t. The **Viterbi algorithm** is a dynamic programming algorithm that, using Equation 3.9, computes both the maximum probability $P(\mathbf{Q}, \mathbf{O}|\lambda)$ and the state sequence $\mathbf{Q}$ in $O(N^2T)$ time.

The recovery of the state sequence makes the Viterbi algorithm invaluable for continuous recognition, because it entirely bypasses the difficult problem of temporal segmentation. Instead, a sequence of HMMs corresponding to individual signs is concatenated into a network, as schematically depicted in Figure 3.3 on page 67, and explained in more detail in Section 3.1.3. This network can be viewed as a single, large HMM. Thus, the most likely state sequence through this HMM network recovers the sequence of signs.

The Viterbi algorithm also has the desirable property that it can be optimized with the beam-searching heuristic [51, 40]. While updating $\delta_{t+1}(i)$, this optimization considers only those states S_j in the HMM network for which $\delta_t(j)$ is above a threshold value. The assumption is that if the probability of a partial path through the network becomes too low, it cannot contribute to the most likely path. Beam-searching is essential for making large-scale applications tractable.

There is an alternative formulation of the Viterbi algorithm, which is called the **token passing algorithm** [83]. It is given in Algorithm 1 on the next page.

Algorithm 1 Token passing algorithm [83]. After each iteration over t, $\text{tok}_t(i) = \delta_t(i)$ (see Equation 3.8 on the preceding page for a definition of δ), so the algorithm is equivalent to the Viterbi algorithm.

```
1: Initialize each state S_i in λ with a token tok_1(i) set to δ_1(i).
2: for t = 2 to T do
3:    for each state S_i do
4:       tok_t(i) ← 0
5:    end for
6:    for each state S_i do
7:       for each state S_j connected to S_i do
8:          tok_t(j) ← max{tok_t(j), tok_{t-1}(i) a_ij b_j(O_t)} (* Pass token from state i to
            state j *)
9:       end for
10:   end for
11: end for
```

It is easy to verify that after each iteration over t, $\forall i$: $\text{tok}_t(i) = \delta_t(i)$. Thus, the token passing algorithm is functionally equivalent to the Viterbi algorithm. The main difference between the two algorithms is that the former updates the probabilities via the outgoing transitions of a state, whereas the latter updates the probabilities via the ingoing transitions of a state. Hence, only the order in which the probabilities are updated, is different. The advantage of token passing is that each token can easily be tagged with additional information, such as the path through the network, or word-by-word probabilities. In Section 3.3.3 I explain why carrying additional information can be useful. This functionality would be difficult to replicate with the Viterbi algorithm.

The third problem corresponds to training the HMMs with data, such that they are able to recognize previously unseen data correctly after the training phase. There exists no analytical solution for maximizing $P(\mathbf{O}|\lambda)$ for given observation sequences, but an iterative procedure, called the *Baum-Welch procedure*, maximizes $P(\mathbf{O}|\lambda)$ locally via expectation maximization. In the case of continuous density output probabilities, as in Equation 3.1 on page 58, the reestimation process works as follows:

First, it initializes the states with a mean and covariance matrix, which can be either taken from hand-labeled data, or simply be the global mean and covariance of all data. Likewise, it initializes the transition probabilities to some values. The reestimation procedure is sensitive to the initial means and covariances, but the initial transitions do not have much effect on the final, trained HMMs.

Recall the definition of the forward variable α from Equation 3.3 on page 61. Define the *backward variable* β as

$$\beta_t(i) = P(O_{t+1}O_{t+2}, \ldots, O_T | Q_t = S_i, \lambda), \tag{3.10}$$

$$\beta_T(i) = 1, \tag{3.11}$$

$$\begin{aligned} \beta_t(i) &= \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \\ & 1 \leq i \leq N,\ 1 \leq t \leq T-1. \end{aligned} \tag{3.12}$$

Furthermore, define ξ and γ as

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(\mathbf{O}|\lambda)}, \tag{3.13}$$

$$\gamma_t(i) = \sum_{i=1}^{N} \xi_t(i,j). \tag{3.14}$$

$\sum_t \xi_t(i,j)$ can be interpreted as the expected number of transitions from S_i to S_j; likewise $\sum_t \gamma_t(i)$ can be interpreted as the expected number of transitions taken from S_i to any state S_j. With these interpretations, the reestimation formulae for the transitions and output probabilities are

$$\bar{\pi}_i = \gamma_1(i), \tag{3.15}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{3.16}$$

$$\bar{c}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)}, \tag{3.17}$$

$$\bar{\mu}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m) O_t}{\sum_{t=1}^{T} \gamma_t(j,m)}, \tag{3.18}$$

$$\bar{U}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m)(O_t - \mu_{jm})(O_t - \mu_{jm})^\top}{\sum_{t=1}^{T} \gamma_t(j,m)}. \tag{3.19}$$

Each application of the Baum-Welch procedure takes $O(N^2T)$ time, as it computes the forward and backward variables. Repeated use of this procedure causes the parameters to converge to a maximum probability [58], typically after 5–15 iterations. In practice, it is important to reestimate the parameters of HMMs with more than one training sequence, so as to prevent the variances $\bar{U}_{jm}$ from tending toward zero. This case is called overtraining, and essentially relegates HMMs to fancy templates, critically degrading recognition performance. With multiple training examples, the training procedure accumulates the parameters $\bar{\pi}, \bar{a}_{ij}, \bar{c}_{jm}, \bar{\mu}_{jm},$ and $\bar{U}_{jm}$ individually for each example, and then assigns the average of each of these parameters as the final estimate.

I now describe in more detail how the token passing algorithm interacts with HMM networks, so as to recover the sequence of signs.

3.1.3 Networks of HMMs

As I mentioned briefly in the previous section, the Viterbi and token passing algorithm recover the state sequence through an HMM, without the need for explicit temporal segmentation of the observation sequence. Both continuous speech and sign language recognition systems take advantage of this property by concatenating the individual word or phoneme

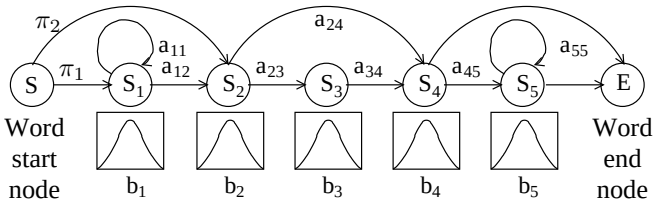


Figure 3.2: HMM with extra word start and word end nodes. These two nodes are non-emitting states; that is, they do not generate observation frames. Only transitions with nonzero probabilities are shown. Compare with Figure 3.1 on page 59.

HMMs into a network. Then the recognition algorithm automatically recovers a continuous sequence of words or signs from this network. Normally, it would not be clear how exactly to assign the transition probabilities between the states of one HMM and the states of another, but in conjunction with the token passing algorithm a very elegant solution exists, which is described in more detail in [82, 83].

This solution consists of adding two extra states to each HMM, which are called the **word start** and **word end nodes**[2]. Unlike the other states, they are non-emitting; that is, they have no output probability associated with them. As a result, whenever a token passes through them, no observation frame is consumed. The word start node has transitions to all other states S_i in the HMM, whose probabilities correspond to π_i. Similarly, each state S_i has a transition to the word end node, which is trained with the Baum-Welch procedure in the normal way. See Figure 3.2 for a schematic example. This solution works with a slight modification to the token passing algorithm, which pushes the tokens through such non-emitting states after all other steps have been completed in Algorithm 1 on page 63[3].

The addition of such nodes makes it very easy to concatenate HMMs into a network. To do so, simply connect the word end node of one HMM to the word start node of another HMM, with transition probability 1. Figure 3.3 on the next page shows a schematic

[2]Note that these nodes do not necessarily have to start and end a *word*, respectively. They could as well start and end a phoneme, respectively.

[3]If we view HMMs as probabilistic finite automata, ϵ-transitions in finite automata theory are a close conceptual counterpart to non-emitting HMM states.

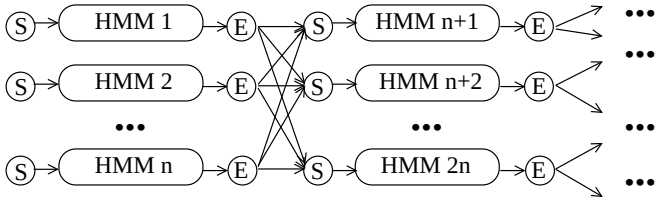


Figure 3.3: Concatenation of HMMs into a network. S and E denote word start and word end nodes, respectively.

example of an HMM network. These nodes also make combining the probabilities easier in the parallel HMM recognition algorithm, which I describe in the following sections as an extension to conventional HMMs.

3.2 Extensions to HMMs

Conventional HMMs are a poor choice for modeling sign language for two reasons: First, they are capable of modeling only one single process that evolves over time. Thus, they would require that the different channels described in Section 2.5 are merged into a single process, forcing them to evolve in a tightly coupled manner and to pass through the same state at the same time. This kind of tight coupling is unsuitable for many applications that require modeling of simultaneous processes. For example, if a one-handed sign precedes a two-handed sign, the weak hand often moves to the location required by the two-handed sign before the strong hand starts to perform it[4], at a somewhat indeterminate point in time. If the channels were too tightly coupled, this movement of the weak hand would be impossible to capture.

Second, as discussed in Section 2.5, the number of possible combinations of phonemes occurring simultaneously is overwhelming. It is computationally infeasible to use on the

[4]This process is called *anticipation* in the linguistics literature.

order of one billion HMMs, let alone to collect enough training data. For these two reasons, it is necessary to extend the HMM framework for ASL recognition.

Past research has addressed two fundamentally different methods of extending HMMs. The first method models the C channels[5] with C separate HMMs, effectively creating a meta-state in a C-dimensional state space. Intuitively, this meta-state is contained in the Cartesian product of the state spaces of the individual channels' HMMs. In an individual channel c, where $1 \leq c \leq C$, the state transition parameters $a_{ij}^{(c)}$ and output probabilities $b_i^{(c)}(O)$ of an HMM in that channel do not depend on any information from any of the other channels $d, d \neq c$. The output probabilities of the HMMs in the C channels are then merged into a single output probability for all channels, in a manner dependent on the C-dimensional meta-state. How exactly these probabilities are combined is application-specific. The method of combination could be, among others, the sum or the product of the output of the individual HMMs in the C channels.

Such models are called factorial hidden Markov models (FHMMs, see Figure 3.5 and Figure 3.4 on page 72 for a comparison with conventional HMMs). Because the output probabilities depend on the meta-state, a training method based on expectation maximization, which computes the optimal HMM parameters, would take time exponential in C. Z. Ghahramani and M. Jordan describe approximate polynomial-time training methods based on mean-field theory [25].

The second method consists of modeling the C channels in C HMMs, whose state probabilities influence one another, and whose outputs are separate signals. That is, the transition probability $a_{Q_t Q_{t+1}}^{(c)}$ from state $Q_t^{(c)}$ to $Q_{t+1}^{(c)}$ in the HMM for channel c does not only depend on the state $Q_t^{(c)}$, but on the $Q_t^{(d)}$ states in all channels d, where $1 \leq d \leq C$.

[5]Note that in the following I use the term "channel" exclusively to clarify the relationship between different HMM extensions and ASL phonology (cf. Section 2.5). This does not mean that the algorithms that I describe in the following sections are restricted to modeling channels in ASL. They can model other processes that take place in parallel, as long as they satisfy the same assumptions as I make for the channels in ASL.

Such HMMs are called coupled hidden Markov models (CHMMs, see Figure 3.6). Essentially, CHMMs are the exact opposite of FHMMs. In FHMMs, the transition probabilities are left alone, and the output probabilities are combined into a single whole. Conversely, in CHMMs, the transition probabilities depend on the states in other channels, and the output probabilities are left alone. M. Brand, N. Oliver, and A. Pentland describe polynomial-time training methods and demonstrate the advantages of CHMMs over conventional HMMs in [9].

Unfortunately, neither of these two extensions help solve the fundamental modeling complexity problem of ASL. Both of these approaches require training the interactions between these different channels, either through the combined output probabilities, or through the interactions between states across channels. As a result, there would need to be enough training examples available for all possible interactions between ASL phonemes across the channels. As I established in Section 2.5, providing enough training examples would require consideration of more than one billion combinations of phonemes. From the modeling and tractability points of view there is no difference between training one billion HMMs — one for each phoneme combination —, or providing enough data to train only one FHMM or CHMM per phoneme, but with the billion ways of interaction between them.

At the heart of this quandary is that the approaches that I have described so far all require *a-priori* modeling of all the possible combinations of phonemes. It is therefore necessary to devise an extension to HMMs that allows decoupling of the combinations of phonemes, and modeling them at recognition time, instead of training time. To this end, I now describe parallel HMMs as a possible solution.

3.3 A New Approach: Parallel Hidden Markov Models

Parallel HMMs (PaHMMs) model the C channels with C independent HMMs with separate output (Figure 3.7 on page 73). Unlike CHMMs, the state probabilities influence one another only within the same channel; that is, PaHMMs are essentially conventional HMMs that are used in parallel.

H. Hermansky, S. Tibrewala, and M. Pavel, as well as H. Bourlard and S. Dupont, first suggested the use of PaHMMs in the speech recognition field [30, 6], with the goal of excluding noisy subbands of the speech signal. They broke down the speech signal into subbands, which they model independently, so as to be able to exclude noisy or corrupted subbands, and merged them during recognition with multi-layered perceptrons. They demonstrated that subband modeling can improve recognition rates. The goal of excluding noisy (i.e., unreliable) subbands, however, is very different from this thesis's goal of solving the modeling complexity problems in the ASL recognition field. Hence, although PaHMMs themselves are not new, their application to modeling simultaneous processes in ASL is.

PaHMMs are based on the assumption that the separate channels evolve independently from one another with independent output. This assumption is the counterpart to assuming that the channels can be modeled independently from one another in Section 2.5. As the experiments in Section 4.4.3 show an improvement in recognition rates, there is some evidence that this independence assumption is at least partially valid.

As a consequence, it is now possible to train the HMMs for the separate channels completely independently from the other channels, and to put the channels together *at recognition time, on the fly.* The implications of taking this step cannot be understated: The modeling complexity of ASL is *reduced by many orders of magnitude,* because it is now no longer necessary to view all possible combinations of phonemes. Now it is just necessary to consider each phoneme by itself, so the total number of HMMs required is

now only the sum of all phonemes, instead of the product. With the same assumptions as in Section 2.5 on page 45, the sum is only $(40 + 8 + 8 + 20 + 40) \times 2$ HMMs, which is many orders of magnitude less than their product.

There are some special algorithmic considerations to take into account when a system uses PaHMMs to recognize sign language. The first consideration is when and how to combine the information in the channels. The next consideration is what to do if a channel does not contain any interesting information, such as all channels for the weak hand, when the data signal represents a one-handed sign. The final consideration is how to constrain the recognition algorithm to ensure that the recognition results from each channel are consistent; for example, it typically would not make sense for the strong hand and weak hand models to recognize two different signs. In the next sections I address these considerations, before developing the PaHMM recognition algorithm.

3.3.1 Channel Combination

At some stage during recognition it is necessary to merge the information from the HMMs representing the C different channels. For convenience of notation and computation, from now on I write all probabilities in logarithmic form[6]. In an extension of the Viterbi algorithm described in Section 3.1.2 the recognition system should find the maximum joint probability of all channels, that is

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \log P(\mathbf{Q}^{(1)}, \ldots, \mathbf{Q}^{(C)}, \mathbf{O}^{(1)}, \ldots, \mathbf{O}^{(C)} | \lambda^{(1)}, \ldots, \lambda^{(C)}) \right\}, \tag{3.20}$$

[6]The actual implementation of HMM recognition algorithms virtually always works with log probabilities, because they provide a numerically far more stable way to represent the extreme range of probabilities between 0.0 and 1.0 [58] — sometimes the first dozens or hundreds of binary digits of the probabilities are all zero, with a lot of variation in the order of magnitude across different probabilities. Besides, modern floating point units have typically more and faster addition units than multiplication units, so it makes sense to add log probabilities, instead of multiplying raw probabilities.

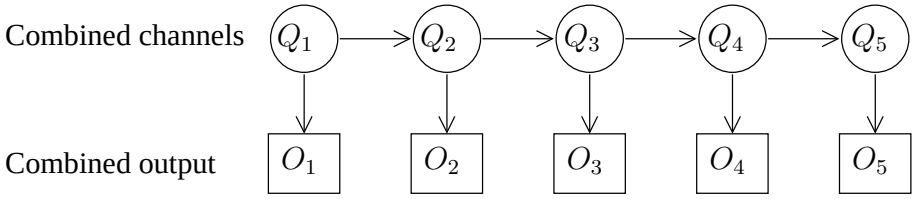


Figure 3.4: Example state sequence of conventional HMMs: Both the output and the states are combined, so the channels are tightly coupled. Q_t denotes the state of all channels at the tth frame. O_t denotes the output of all channels at the tth frame.

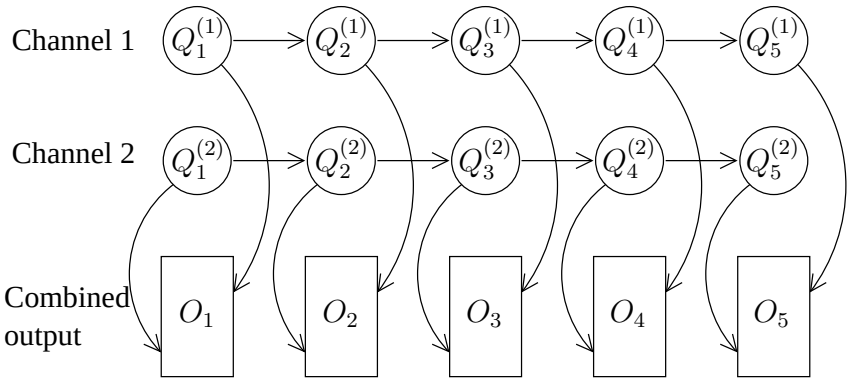


Figure 3.5: Example state sequence of FHMMs: The output is combined, but the transitions from one state to another within a channel are independent from other channels. $Q_t^{(c)}$ denotes the state of channel c at the tth frame. O_t denotes the combined output of all channels at the tth frame.

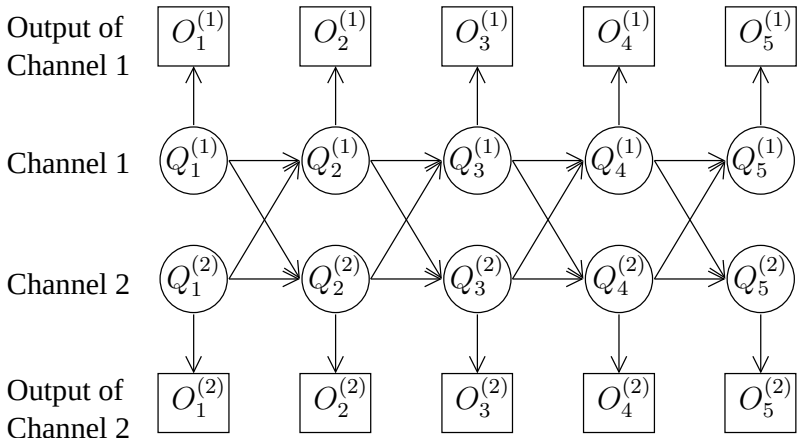


Figure 3.6: Example state sequence of CHMMs: The output of each channel is separate, but the transitions from one state to another also depend on the states of the other channels. $Q_t^{(c)}$ denotes the state of channel c at the tth frame. $O_t^{(c)}$ denotes the output of channel c at the tth frame.

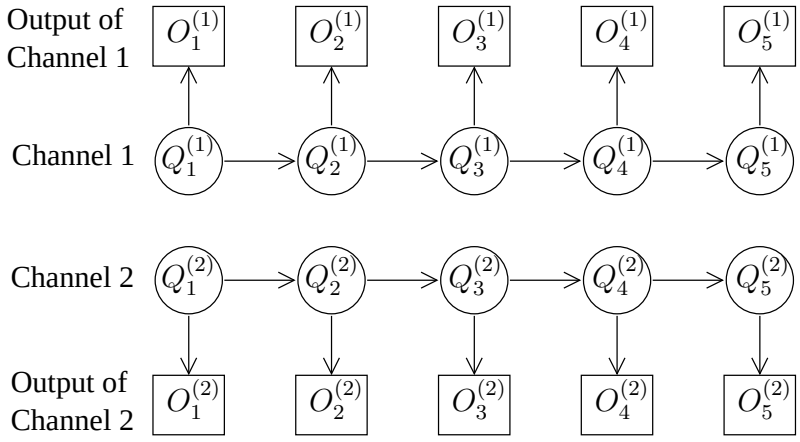


Figure 3.7: Example state sequence of PaHMMs: Both the output and the transitions from one state to another within a channel are independent from other channels. $Q_t^{(c)}$ denotes the state of channel c at the tth frame. $O_i^{(c)}$ denotes the output of channel c at the tth frame. Compare with Figures 3.6, 3.5, and 3.4 on the page before.

where $\mathbf{Q}^{(c)}$ is the state sequence of channel c, $1 \leq c \leq C$, with observation sequence $\mathbf{O}^{(c)}$ through the HMM network $\lambda^{(c)}$. Recall that this observation sequence corresponds to some unknown sign sequence in channel c, which is to be recognized. Because in PaHMMs the channels are independent, the merged information consists of the product of the probabilities of the individual channels — or the sum of the log probabilities —, so Equation 3.20 becomes

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{\log P(\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)},\mathbf{O}^{(1)},\ldots,\mathbf{O}^{(C)}|\lambda^{(1)},\ldots,\lambda^{(C)})\right\} = \max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{\sum_{c=1}^{C} \log P(\mathbf{Q}^{(c)},\mathbf{O}^{(c)}|\lambda^{(c)})\right\}. \tag{3.21}$$

Working from this equation, we can make a strong statement about combining the probabilities in the recognition algorithm:

Theorem The Viterbi recognition algorithm for PaHMMs can combine the partial probabilities from the individual channels as many times as desired at any stage of the recognition process, including the whole-sign level or the phoneme level.

Proof Given some $\mathbf{O}^{(c)}$ for an HMM in channel c, split it up into W adjoining subsequences. The idea is to split up the observation sequence into its constituent signs or phoneme segments. Let $\mathbf{O}_w^{(c)}$ be the wth subsequence of $\mathbf{O}^{(c)}$. Similarly, let $\mathbf{Q}_w^{(c)}$ be the state subsequence of $\mathbf{Q}^{(c)}$ that generates $\mathbf{O}_w^{(c)}$. In other words, $\mathbf{Q}_w^{(c)}$ is the state sequence that generates a single sign or phoneme segment contained in $\mathbf{O}^{(c)}$. Let $P(\mathbf{Q}_w^{(c)},\mathbf{O}_w^{(c)}|\lambda^{(c)})$ be the probability that this state sequence $\mathbf{Q}_w^{(c)}$ generated this observation subsequence $\mathbf{O}_w^{(c)}$.

With these definitions, it is possible to derive the following from Equation 3.21:

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{c=1}^{C} \log P(\mathbf{Q}^{(c)}, \mathbf{O}^{(c)} | \lambda^{(c)}) \right\} = \max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{w=1}^{W} \sum_{c=1}^{C} \log P(\mathbf{Q}_w^{(c)}, \mathbf{O}_w^{(c)} | \lambda^{(c)}) \right\}. \tag{3.22}$$

The detailed derivation is given in Appendix A.

Intuitively, (3.22) simply proves that

- it is possible to compute the path probabilities for each subsequence separately.
- it is possible to combine the probabilities for the individual subsequences first, before coalescing them into the whole sequence.

Because the number and exact boundaries of the W subsequences are arbitrary, an immediate corollary is that the recognition algorithm can combine the partial probabilities as many times as desired at any stage of the recognition process, including the whole-sign level or the phoneme level. □

Note that it is still not necessary to know the sign or phoneme boundaries beforehand, because the token passing algorithm implicitly segments the observation sequence. The freedom to combine the probabilities at any moment becomes important as soon as the recognition algorithm enforces the channel consistency constraints discussed in Section 3.3.3.

Equation 3.22 also states that the basic idea underlying the PaHMM recognition algorithm is to apply the conventional token passing algorithm to the HMMs in the separate channels, and to combine the probabilities of the tokens across the channels at word end nodes. See Figure 3.8 on the next page for an example with two channels (e.g., the strong and weak hands' movement channels). This basic idea, however, needs to be modified

to account for channels that contain little or no information, and for channel consistency constraints.

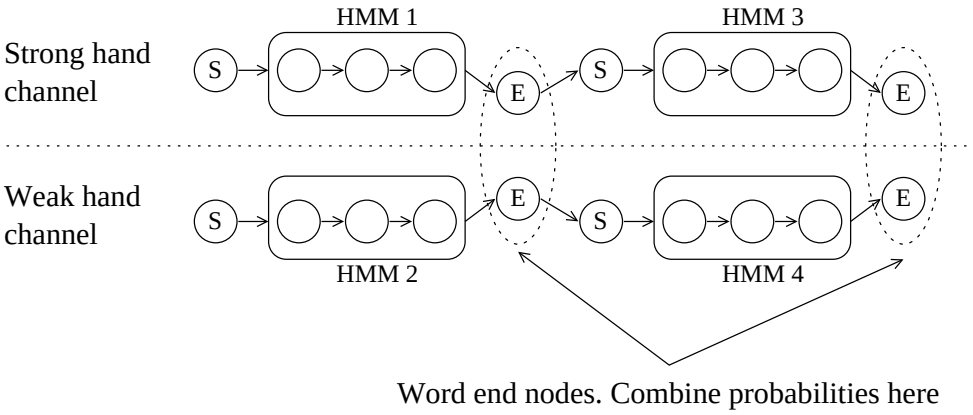


Figure 3.8: Example of token probability combination. The tokens are passed independently in the HMMs in the strong and weak hands' movement channels, and combined in the word end nodes. S and E denote word start and end nodes, respectively.

3.3.2 Channels with Little or No Information

There are many situations in ASL where a channel does not contain much information, or none at all. The most obvious example arises out of the difference between one-handed and two-handed signs. In one-handed signs, the weak hand plays no role; in fact, it often returns to a neutral position during those signs. Thus, during one-handed signs the channels associated with the weak hand do not contain any useful information at all[7].

Another situation that frequently arises is the situation of two-handed signs where the weak hand does not move, such as the sign for SIT (see Figure 2.3 on page 32). During one-handed signs the position of the weak hand is not significant, but if it coincidentally

[7]In reality, this statement makes a simplifying assumption. The video data from the National Center for Sign Language and Gesture Resources [1] show clearly that, for instance, the weak hand may hold the position of a two-handed sign during one-handed signs. So, in future work, it will be interesting to look more closely at what exactly the weak hand does, but this topic is beyond the scope of this thesis, and also beyond the current state of the art in sign language recognition.

moves into a position required by the sign for SIT, and if the strong and weak hand channels carry equal weight, the result from the weak hand channel could bias the recognition process so much that such a one-handed sign is incorrectly recognized as SIT. The main distinguishing feature of such two-handed signs is the activity of the strong hand, not the placement of the weak hand. Hence, for such two-handed signs the strong hand should carry more weight than the weak hand.

Consequently, it is desirable to weight the channels on a per-sign basis, depending on how much information each channel provides. Let $\omega_w^{(c)}$ be the weight of channel c for the sign (or some other subsequence according to the proof discussed in the previous section). Then the desired quantity to maximize becomes, from Equation 3.22 on page 75:

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{w=1}^{W} \sum_{c=1}^{C} \omega_w^{(c)} \log P(\mathbf{Q}_w^{(c)}, \mathbf{O}_w^{(c)} | \lambda^{(c)}) \right\}, \tag{3.23}$$

where

$$\sum_{c=1}^{C} \omega_w^{(c)} = C$$

for fixed w.

From a theoretical point of view, one-handed signs simply correspond to the case where the weights $\omega_w^{(c)}$ are all zero for the channels representing data from the weak hand. Nevertheless, it is advantageous to treat this case specially when enforcing the channel consistency constraints, for reasons that I discuss in the next section.

3.3.3 Channel Consistency Constraints

The final missing piece necessary for adapting the token passing algorithm to PaHMMs is how to ensure that the recognition results from the channels are consistent. If, for example, the movement channel for the strong hand yields the sequence FATHER READ, and the movement channel for the weak hand yields the sequence MOTHER SIT, there is no way

that these two results can be reconciled in a consistent manner, even though they may constitute the maximum given by Equation 3.23. A consistent recognition result must satisfy the additional constraint that the paths $\mathbf{Q}^{(c)}$ all can generate the same sequence of signs[8].

The easiest way to enforce this constraint is to assign unique path identifiers to the tokens as follows:

- Every time a token with a particular path identifier hits the word start node of a sign for the first time, it is assigned a new unique path identifier. The recognizer stores the new path identifier of this token in a lookup table with the old path identifier and the name of the sign as the keys.

- If a subsequent token hits the starting node of a sign, the recognizer uses this table to look up the new path identifier, based on the token's path identifier and the name of the sign. It then assigns this new path identifier to the token.

With this method, tokens have the same path identifier if, and only if, they have touched the same sequence of signs. At each word end node the recognizer combines the probabilities of only those tokens that have the same path identifier and discards the rest. Here the advantage of the token passing algorithm over the Viterbi algorithm becomes clear, because with the former algorithm the information can be directly attached to the tokens, whereas with the latter algorithm keeping track of this information would be quite complicated.

Imposing this constraint, however, means that a path in channel c that contributes to maximizing the joint probability in Equation 3.23 on the preceding page in a consistent

[8]In reality — like the assumptions behind channel weights in the previous section —, this constraint constitutes a simplifying assumption. There are situations when the two hands can sign different phrases, but dealing with these is beyond the scope of this thesis, and probably will remain beyond the scope of recognition systems for some time to come.

manner, may no longer maximize the marginal probability

$$\sum_{w=1}^{W} \omega_w^{(c)} \log P(\mathbf{Q}_w^{(c)}, \mathbf{O}_w^{(c)} | \lambda^{(c)}).$$

The less-than-desirable implication of the discrepancy between the maximum joint and marginal probabilities is that now finding an exact solution to the maximization problem would take time exponential in the size of the HMM network. A similar problem exists with CHMMs, although for a different reason [9]. The CHMM algorithms use an approximation instead, which uses multiple path heads.

The PaHMM recognition algorithm can use a similar idea, which is based on multiple hypotheses. Instead of containing only a single token at any point in time, each HMM state contains a set of tokens representing the M best hypotheses (i.e., the tokens with the M highest probabilities). Each of these has a unique path identifier. This approximation is based on the assumption that even if the correct path does not maximize the marginal probability in a channel, its probability is high enough to end up among the best few. Making this assumption seems reasonable if the HMMs are well-trained.

There is one last point to consider. Enforcing the channel consistency constraints via path identifiers and multiple hypotheses has a special effect on the case where $\omega_w^{(c)} = 0$; that is, where a channel carries no information. As an example, consider the movement channels of the strong and weak hands. During one-handed signs, the position of the weak hand needs to be allowed to be arbitrary. Hence, the semantics of this case should be that, essentially, the channel can take arbitrary observation sequences, without affecting the recognition result.

Yet, without further refinement, enforcing path constraints with multiple hypotheses does not have these semantics. The reason is that, even if the channel contains no information, the tokens still have to pass through *some* model in that channel. On a first glance, this model should be a "noise model," which is designed to capture arbitrary observation

sequences from a channel. Unfortunately, it is impossible to train it such that the probabilities of these arbitrary sequences are high enough everywhere. It is very likely that, as a result, the probabilities of those tokens will be so low that some other non-noise model yields tokens with a higher probability. If this happens, and if the number M of hypotheses is small, these tokens will displace the ones from the noise model in the token set. As a consequence, the correct path identifiers in that channel will be lost, which implies that the entire correct path will be discarded, no matter what the token probabilities from the information-containing channels are. To return to the example of the strong and weak hands' movement channels, what would happen is that one-handed signs would consistently be recognized as some other two-handed sign, even though the probabilities of the strong hand channels are much higher for the correct one-handed sign than the probabilities for two-handed signs.

The naïve solution to this problem would be to increase the number of hypotheses, but it would increase the running time of the recognition algorithm significantly, and unnecessarily. A much better solution is to suspend token passing in those channels whenever the algorithm encounters a segment with weight 0. Then, once the tokens in the other channels have passed this segment, token passing resumes at the point immediately past it. For convenience and ease of understanding, in the following I describe the details of this solution specifically for one-handed and two-handed signs, because one-handed signs are the only instance in this thesis where a channel has weight 0. It would not be too difficult to generalize this method to arbitrary channels and movements.

First, let each token carry the following variables:

id the path identifier of the token.

combined the combined probability of all channels up to the last time the marginal probabilities were combined. In other words, it contains

$$\sum_{w=1}^{V}\sum_{c=1}^{C}\omega_w^{(c)}\log P(\mathbf{Q}_w^{(c)},\mathbf{O}_w^{(c)}|\lambda^{(c)}),$$

where V is the number of segments from Equation 3.23 on page 77 up to this point.

marginal the marginal probability of the token in this channel since they were last combined. In other words, it contains a part of the probability $\log P(\mathbf{Q}_w^{(c)},\mathbf{O}_w^{(c)}|\lambda^{(c)})$, where w is the segment in which the token currently is.

Define the following procedures:

MERGE(T, T'): This operation works on two token sets T and T'. It replaces T' with the set of M tokens from T and T' with the M highest probabilities, subject to the constraint that all M tokens have different path identifiers. In other words, it merges the token set T into T'.

JOIN($S_i^{(c)}$): $S_i^{(c)}$ must be an HMM state that is a word end node, and c is the weak hand channel. This procedure takes the tokens of the weak hand in state $S_i^{(c)}$ and attaches them to the tokens of the strong hand in state $S_j^{(d)}$, where d is the corresponding strong hand channel, and where $S_j^{(d)}$ corresponds to the word end node of the same sign as $S_i^{(c)}$. The attached token must have the same path identifier as the token that it is attached to. Because at word end nodes the recognition algorithm previously combined the probabilities, the tokens in $S_i^{(c)}$ and $S_j^{(d)}$ share a set of common path identifiers, so the attachment always succeeds.

SPLIT($S_i^{(d)}$): $S_i^{(d)}$ must be an HMM state in the strong hand channel d that is a word start node. This procedure detaches the weak hand tokens that have previously

been attached with JOIN. More specifically, it detaches them from the strong hand tokens in $S_i^{(d)}$into a temporary token set $\hat{S}_i^{(d)}$. It checks for each detached token, whether the last sign in the path was one-handed or two-handed. If it was one-handed, SPLIT updates the probability of the each detached token with the corresponding probability of the strong hand as follows: Let ts be the token from the strong hand, and let tw be the attached token from the weak hand. Let $tw.combined \leftarrow ts.combined$[9].

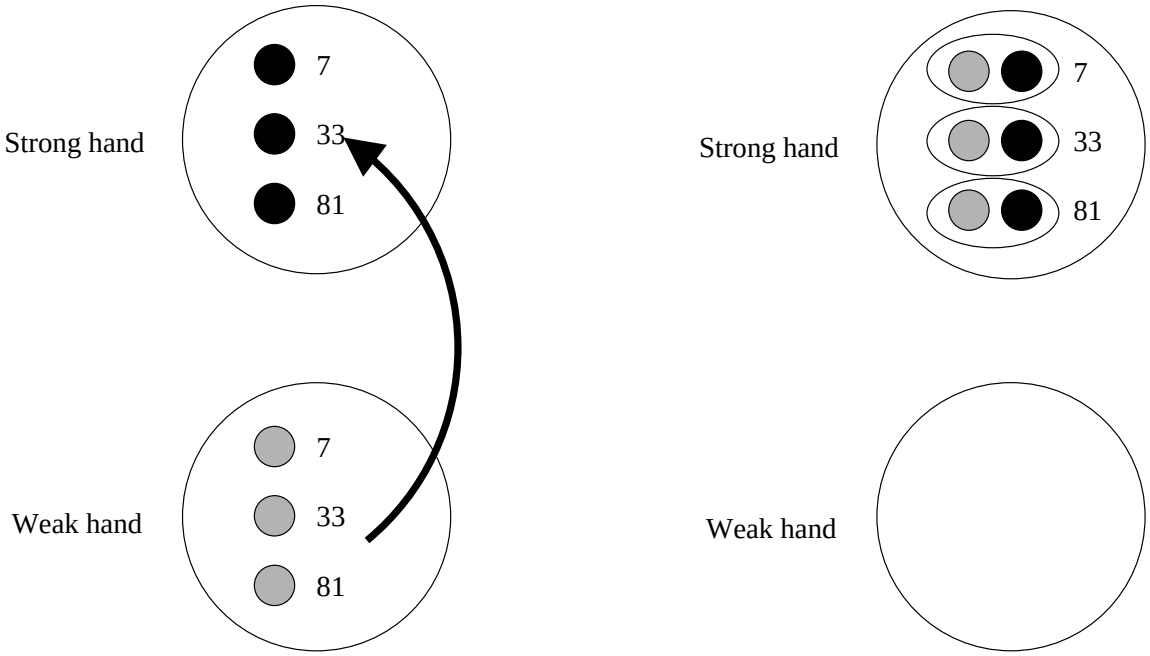
Then SPLIT finds the existing tokens of the weak hand in state $S_i^{(c)}$, where c is the corresponding weak hand channel, and where $S_i^{(c)}$ corresponds to the word start node of the same sign as $S_i^{(d)}$. Finally, it calls MERGE($\hat{S}_i^{(d)}, S_i^{(c)}$), so the detached tokens from the temporary set are merged into the existing weak hand token set.

Figure 3.9 on the following page illustrates the concept of JOIN. It corresponds to the notion of suspending token passing whenever a channel with weight 0 is encountered. Likewise, SPLIT corresponds to the notion of resuming token passing. The net result is that during those times when a weak hand token is attached to a strong hand token, the weak hand channel does not contribute, so computationally it is equivalent to having weight 0. With these two operations, it is finally possible to devise the modified token passing algorithm for PaHMMs.

3.3.4 The PaHMM Recognition Algorithm

Define $\text{tok}_t^{(c)}(i)$ to be the token set in state $S_i^{(c)}$ at frame t. Define the operation of multiplying $\text{tok}_t^{(c)}(i).marginal$ with a scalar to be the token set, where each element's *marginal*

[9] $tw.marginal$ and $ts.marginal$ are 0 at this point, because JOIN and SPLIT take place at a moment when the channels were just combined in a word end node. Hence, the algorithm can ignore them in these two procedures.



(a) Before JOIN

(b) After JOIN

Figure 3.9: The JOIN operation on tokens with path identifiers 7, 33, and 81. The weak hand tokens are attached to the strong hand tokens with the same path identifier, respectively, and carried through the recognition network. SPLIT reverses this operation.

has been multiplied with this scalar. Recall that T is the number of frames in the observation sequences $\mathbf{O}^{(c)}$, that N is the number of states per channel in the HMM network, and that C is the number of channels. With these definitions the recognition algorithm is given in Algorithm 2.

Algorithm 2 PaHMM token passing algorithm. See Algorithm 3 on the next page for COMBINE_CHANNELS(t). Also compare with the conventional token passing algorithm (Algorithm 1 on page 63).

1: Initialize the token sets in the start nodes of the HMM network with frame 0, path identifier 0, and $\log p = 0$
2: **for** $t = 1$ to T **do**
3: **for** $c = 1$ to C **do**
4: **for** each state $S_i^{(c)}$ **do**
5: $\text{tok}_t^{(c)}(i) \leftarrow \varnothing$
6: **end for**
7: **for** each state $S_i^{(c)}$ **do**
8: **for** each state $S_j^{(c)}$ connected to $S_i^{(c)}$ **do**
9: MERGE $\left(\text{tok}_{t-1}^{(c)}(i).marginal \cdot a_{ij}^{(c)} b_j^{(c)}(O_t^{(c)}),\ \text{tok}_t^{(c)}(j)\right)$. (* Pass token set from state i to state j in channel c *)
10: **end for**
11: **end for**
12: **end for**
13: COMBINE_CHANNELS(t) (* see Algorithm 3 on the following page *)
14: **end for**

Analysis of the PaHMM Recognition Algorithm

The following analysis pertains to Algorithm 2. Assuming that the token sets in each state have cardinality M (i.e., the M best hypotheses are retained), step 5 takes $O(M)$ time, hence steps 4–6 take $O(NM)$ time. If the token sets are stored as lists sorted by log likelihood, MERGE takes $O(M)$ time. Hence, steps 7–11 take $O(N^2M)$ time per frame, where N is the number of states in the HMM network. This bound describes the worst case when every state is adjacent to every other one. As a result, step 3–12 take

Algorithm 3 Algorithm for combining the probabilities of the channels.

```
1: COMBINE_CHANNELS(t)    (* t is the frame number *)
2: for c = 1 to C do
3:    for each state S_i^(c) that is a word end node do
4:       COMBINE_NODE(S_i^(c)) (* see Algorithm 4 on the next page *)
5:       if S_i^(c) belongs to a two-handed sign and c is a weak-hand channel then
6:          JOIN(S_i^(c))
7:       end if
8:       for each word start node S_j^(c) adjacent to S_i^(c) do
9:          MERGE the tokens tok_t^(c)(i) into tok_t^(c)(j),
            updating the path identifier of the merged tokens.
10:      end for
11:   end for
12:   for each state S_i^(c) that is a word start node do
13:      if S_i^(c) belongs to a two-handed sign and c is a strong-hand channel then
14:         SPLIT(S_i^(c))
15:      end if
16:   end for
17: end for
18: for c = 1 to C do
19:   for each state S_i^(c) that is a word end node do
20:      tok_t^(c)(i).marginal ← 1 (* The probabilities have been combined, so clear out
         accumulated marginal probabilities *)
21:   end for
22: end for
```

Algorithm 4 Algorithm for combining probabilities of a word end node.

COMBINE_NODE($S_i^{(c)}$) (* $S_i^{(c)}$ is a word end node *)
target token set $\leftarrow \varnothing$
$w \leftarrow$ the sign that $S_i^{(c)}$ belongs to
for $k = 1$ to M **do**
  tok $\leftarrow$ the mth token in tok$^{(c)}(i)$
  $count \leftarrow 0$
  for $d = 1$ to C **do**
    $S^{(d)} \leftarrow$ state in channel d that belongs to the the same word end node as $S_i^{(c)}$
    tok$^{(d)} \leftarrow$ the token set of $S^{(d)}$
    Find tok_other $\in$ tok$^{(d)}$, such that tok_other.id = tok.id
    if tok_other **exists then**
      $count \leftarrow count + 1$ (* For channel consistency constraint check *)
      tok.$combined \leftarrow$ tok.$combined \cdot \omega_w^{(d)}$tok_other.$marginal$
    end if
  end for
  if $count = C$ **then**
    Put tok in target token set (* Channel consistency constraint: path occurred in all channels *)
  end if
end for
Sort target token set by combined probability

$O(C \times (NM + N^2M)) = O(CN^2M)$ time altogether. Step 1 takes $O(N)$ time.

Now the analysis of step 13 — COMBINE_CHANNELS in Algorithm 3 on page 85 — follows. First of all, note that the combination of the probabilities in a word end node in step 4 needs to be done only once for all channels, because the combined probabilities are the same for a word end node across all channels. Thus, they can be cached for subsequent iterations over C in the loop starting at step 2. The algorithm for combining the probabilities can iterate over all token sets and store them in a table associated with the word end node across all channels. As a result, the complexity $\mathcal{X}$ for this step can be taken out of the *for* loop in step 2.

Using hash tables with the path identifier as the key, JOIN in step 6 takes $O(M)$ expected time. MERGE in step 9 takes $O(M)$ time per call, by the same argument as above. SPLIT in step 14 takes $O(M)$ time per call, because it, too, calls MERGE. The loops in steps 3–11 and 12–16 iterate N times in the worst case, although they are are executed much less often in the average case, because there are fewer words than HMM states. Hence, steps 3–11, excluding step 4, take $O(N \times (M + M + M))$ expected time, and steps 12–16 take $O(N \times M)$ expected time. Overall, steps 2–17, again excluding step 4, take $O(N \times (M + M + M) + N \times M) = O(NCM)$ time.

As explained before, step 4 can be taken out of the loop on C, so it is only executed N times overall in the worst case, resulting in a complexity of $O(\mathcal{X} \times N)$, where $\mathcal{X}$ is the complexity of COMBINE_NODE in Algorithm 4 on the page before. Hence, the total running time of steps 2–17, including step 4, is $O(NCM + \mathcal{X}N)$. Step 20 is executed N times in the worst case, by the same argument as for steps 3–11, and 12–16. Thus, steps 19–21 take $O(CM)$ time altogether, and as a result, COMBINE_CHANNELS takes $O(NCM + \mathcal{X}N + CM)$ time.

Now the analysis of COMBINE_NODE in Algorithm 4 on the preceding page remains,

so as to determine its complexity $\mathcal{X}$. The critical operation takes place in step 10 consists of finding tokens across other channels, if they exist, that share the same path identifier. This operation could be done via a simple linear search in $O(M)$ time, which is the easiest solution for the common case when M is small. A more elaborate solution would consist of recasting the algorithm, such that the lookup and counting are done with hash tables. In this case, the expected lookup time would be $O(1)$, at the expense of having to allocate a sophisticated data structure. With hash tables and $O(1)$ expected lookup time, steps 3–19 take $O(CM)$ expected time overall. Step 2 takes $O(M)$ time, and step 20 takes $O(M \log M)$ time. Hence, overall COMBINE_NODE takes $\mathcal{X} = O(M + CM + M \log M) = O(CM + M \log M)$ time.

It follows that COMBINE_CHANNELS takes

$$O(NCM + \mathcal{X}N + CM) = O(NCM + N \times (CM + M \log M) + CM)$$
$$= O(NM \times (C + \log M))$$

expected time. The loop in step 2 in Algorithm 2 on page 84 is executed T times. From putting all these individual run times together, it follows that the entire PaHMM recognition algorithm runs in

$$O(N + T \times (CN^2M + NM \times (C + \log M)))$$

expected time, but because the number of hypotheses is far smaller than the number of states in the HMM network; that is, typically $M \ll N$:

$$NM \times (C + \log M) \quad = \quad O(N^2CM),$$

and thus the complexity can be simplified to

$$O(TN^2CM) \tag{3.24}$$

expected time, compared to the conventional HMM recognition algorithm, which runs in time

$$O(TN^2). \tag{3.25}$$

That is, the algorithm takes time linear in the number C of channels and in the number M of hypotheses. Furthermore, it becomes clear that the bulk of the time is spent in the part analogous to the conventional token passing algorithm, particularly the token set merge in step 9 in Algorithm 2 on page 84. The time spent on the COMBINE_CHANNELS procedure is asymptotically negligible compared to the rest of the algorithm.

This section concludes the theoretical description of the PaHMM recognition framework. Before I discuss how to tie it in with the phoneme-based modeling from Chapter 2, I need to address a practical concern that arises out of using Gaussian densities as output probability distributions.

3.3.5 Practical Considerations

In Section 3.1.1, I mentioned that frequently in activity recognition applications the sample space Ω is continuous, and that consequently the state output probabilities are Gaussian mixtures, as described in Equation 3.1 on page 58. This recognition framework is no exception — I use a continuous sample space throughout.

The problem is that Gaussians are not probability distributions; they are *probability densities*. Yet, in HMMs they are used as probability distributions, with the effect that the probabilities of a random variable "distributed" along a particular density do not add up to 1. Normally, this peculiarity is of no further concern in HMM theory, because qualitatively it does not change the HMM equations in Section 3.1.1 [58]. They need to be adjusted only for constant scaling factors, which depend on the constant factors in an

n-dimensional Gaussian density of the following form, taken from Equation 3.1:

$$\begin{aligned} G(O,\mu,U) &= Ae^{-\frac{1}{2}(O-\mu)^\top U(O-\mu)}, \\ A &= \frac{1}{\sqrt{(2\pi)^n|U|^{-1}}}. \end{aligned}$$

Recall that evaluating the output probability of a particular state with a particular observation O consists of actually computing $G(O,\mu,U)$ for this observation, and taking it as the probability. The equations for the Viterbi algorithm in Section 3.1.2, and thus by extension for the token passing algorithm, tacitly assume that the factors A are of comparable magnitude. This assumption is valid, as long as the HMMs all have been trained on the same channel with the same data, but it does not hold for HMMs that come from different channels. It is not uncommon to see the factors A from one channel being many orders of magnitude larger or smaller than the ones from another channel. As a result, their respective output probabilities also differ by many orders of magnitude.

Such large differences in orders of magnitude introduce biases into the PaHMM recognition process. For the sake of the discussion, assume that the factors A in the movement channel are many orders of magnitude larger than the ones in the handshape channel. Suppose that there are some tokens $tok_{combined}$ in the recognition network that already have gone through a word end node, so the partial probabilities from both channels have already been combined once. Also, suppose that there are some tokens $tok_{uncombined}$ that have not gone through any word end nodes yet, so their probabilities have not been combined at all yet.

If both channels are weighted equally, $tok^{hand}_{combined}$ in the handshape channel will be many orders of magnitude larger than the respective $tok^{hand}_{uncombined}$, because of the effect of the larger probabilities from the movement channel. This difference results in a bias toward recognizing many words, because the $tok^{hand}_{combined}$ crowd out the $tok^{hand}_{uncombined}$, and many words mean more token probability combinations. Conversely, the $tok^{move}_{combined}$

in the movement channel will be many orders of magnitude smaller than the respective $tok^{move}_{uncombined}$, because of the effect of the smaller probabilities from the handshape channel. Here, this difference results in a bias toward few words during recognition, by the same argument as before. The net effect is an imbalance between the channels — one channel favors sentences of many words, whereas the other one favors sentences of few words, with a predictably negative effect on recognition accuracy.

The solution to this problem is simple, and drastic: Before starting recognition on the trained HMMs, compute the median factor A_{median} for each channel, and normalize the output probabilities in each channel by the respective median factor. This solution does not affect the relative probabilities within a single channel, but makes it possible to combine probabilities across channels. Even so, after normalization, a residual difference between the probability magnitudes of different channels may remain, necessitating a larger number of hypotheses. The experiments in Section 4.4.4 show that combining the probabilities after each phoneme, instead of each word, counteracts this residual difference.

Now nearly all the pieces for the ASL recognition framework are in place. Next I discuss the one missing piece — how to apply the modeling from Chapter 2 to the PaHMM recognition framework.

3.4 Application of the Movement-Hold Model to HMMs

As I mentioned in Section 3.1.3, the basic idea behind HMM-based recognition is to chain the HMMs together into a set of networks. The PaHMM token passing algorithm finds the most likely path through these networks, and thus recovers the sequence of signs. For the most part, chaining the HMMs corresponding to phonemes in ASL together into a network and training the HMMs works in the same way as for speech recognition.

To see how the chaining works, consider the strong hand movement channel of the sign for MOTHER, given in Figure 3.10, as a concrete example. The start and end nodes of

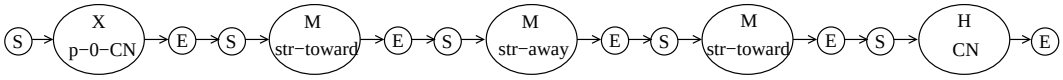


Figure 3.10: Composite HMM for MOTHER in the strong hand movement channel. It is chained together from the individual phoneme HMMs. S and E denote the word start and end nodes of each phoneme HMM, respectively. Compare with the phonetic Movement-Hold structure in Figure 2.8 on page 43. See Appendix B for an explanation of the symbols used.

the HMMs corresponding to the phonemes in the sign are simply connected by transitions, which all have an associated probability of 1. However, there are some peculiarities in the network design and training process that are caused specifically by the properties of sign languages. I now describe what they are and how to manage them.

3.4.1 Incorporating Movement Epenthesis

In speech recognition, the individual words are expanded into their constituent phonemes, and the phoneme HMMs are then chained together in the order in which they appear in the words. Up to this point, chaining together the HMMS in ASL recognition works in exactly the same way. In speech recognition, the composite models for the words are then chained together into the recognition network. However, we cannot do the same in ASL recognition, because it would ignore movement epenthesis. Instead, we need to provide the epenthesis models and chain them into the recognition network, as well.

Because in this thesis I assume that the epenthesis HMMs depend on the ending and starting body locations of signs, as described in Section 2.4.1, it is convenient to connect each word end node to a node corresponding to its ending body location in the HMM network, instead of connecting it to the epenthesis HMMs directly. Similarly, it is convenient to connect each word start node to a node corresponding to its starting body location. Just like the word start and word end nodes described in Section 3.1.3, these nodes are non-emitting. This trick reduces the number of state transitions and thus the complexity of the

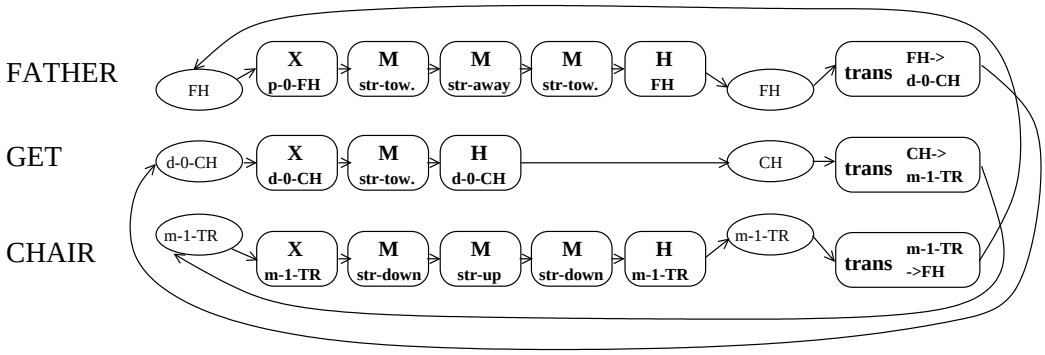


Figure 3.11: Network that models the strong hand movement channel of the signs for FATHER, GET, and CHAIR in terms of their constituent phonemes. Epenthesis is modeled explicitly with the HMMs labeled with "trans." The ovals in this figure depict the nodes for the body locations at the beginning and the end of each sign. For lack of space, the word start and end nodes are not shown explicitly in this figure, but any transition from any HMM depicted in this figure must go through the model's corresponding word start and end nodes. See Appendix B for an explanation of the symbols used.

HMM network.

Figure 3.11 shows how to chain together the phoneme and epenthesis HMMs, and the body location nodes for the three signs for FATHER, GET, and CHAIR.

3.4.2 Training the PaHMMs

In principle, the HMMs can be trained independently for each channel with standard methods, such as Viterbi alignment [58] and embedded Baum-Welch training [82]. The difference between the normal Baum-Welch training procedure described in Section 3.1.2 and embedded training is that the latter works on chains of HMMs that have been strung together according to the training sentence. The former method is suitable for isolated recognition, and the latter method is suitable for continuous recognition.

Yet again, the nature of ASL causes complications, because the weak hand does not do anything meaningful during one-handed signs. Therefore, training the channels for the weak hand is more complicated than training the channels for the strong hand. During

recognition, this problem is handled by the JOIN and SPLIT functions, as described in Section 3.3.3, so there are no HMMs for one-handed signs in weak hand channels in the HMM network. Embedded Baum-Welch reestimation, however, requires that all parts of the observation sequence are covered by HMMs.

One possible solution to this problem is to use a "noise" model for the weak hand in one-handed signs during the training phase. This noise model is shared across all one-handed signs and initialized with the global mean and covariance of the training data. It is not used at all during the recognition phase.

The introduction of the noise model, however, makes the training process of the weak hand channels more sensitive than usual to initial state distributions, and the initial mean and covariance estimations. For this reason, the popular and normally sufficient flat start scheme, where all states of the HMMs are assigned the global mean and covariance of the training data, is not the best choice. Instead, each channel is best initialized with a set of presegmented data. There are two ways to obtain the segmentation: either use a set of hand-labeled data, or train the models from the strong hand channels first, and use the token passing algorithm on the training data to find the state alignment. More specifically, in the latter approach, for each training utterance, the corresponding phoneme HMMs are chained together into a long network in the order in which they appear in the utterance. The token passing algorithm then finds the optimal state alignment, because the only possible path through the network touches the phonemes in the correct order. The alignment of the respective word start and end nodes yields the frame numbers at which each respective sign starts and ends.

Hand-labeling sign language data is a notoriously tedious task, and it is easy to make mistakes in the segmentation between the signs, because the exact boundaries are not at all clear, even on slow-motion video. For this reason, in this thesis, I choose to use the alignment from the token passing algorithm.

3.5 Summary

In this chapter I have described the recognition framework, in which the phoneme modeling from the previous chapter takes place. I have introduced HMMs and described how the token passing algorithm is suitable for continuous recognition without explicit temporal segmentation. Conventional HMMs, however, are unsuitable for recognition of simultaneous channels in ASL, because of the enormous complexity of modeling all combinations of all channels. To this end, I have contributed a new approach in the form of PaHMMs, which assumes that the channels are independent from one another. With this assumption, it is possible to model channel combinations on the fly at recognition time, instead of training time, thus eliminating the modeling complexity problems. I have developed a specialized token passing algorithm for PaHMMs to make them suitable for American Sign Language recognition. Finally I have shown how to apply the phoneme modeling from the previous chapter to the PaHMM framework, by chaining the individual phoneme and epenthesis HMMs together.

In the next chapter I discuss some experimental results that back up the theory and claims made in this and the previous chapters.

Chapter 4

Experiments

In this chapter I discuss a series of experiments that compare various aspects of the recognition system against one another. Their purpose is to provide justification for the modeling approach that I take in Chapter 2. In the following, I discuss the evaluation criteria, the data sets that I collected for the experiments, then briefly discuss the feature vectors suitable for the HMM-based sign language recognition, and then provide the experimental results.

4.1 Evaluation Criteria

To evaluate the recognition experiments, it is helpful to distinguish among three different types of recognition errors. Assuming that the correct sentence is “FATHER READ BOOK,“ these are:

Substitution error One word is confused with another. Example: “**MOTHER** READ BOOK” instead of **“FATHER** READ BOOK.”

Deletion error One word is incorrectly dropped. Example: “FATHER BOOK” instead of “FATHER **READ** BOOK.”

Insertion error One word is incorrectly inserted. Example: "FATHER READ **GIVE** BOOK" instead of "FATHER READ BOOK."

The standard method to compute these errors consists of aligning the recognized sentences versus the ground truth via a weighted string matching algorithm. In all these experiments, substitutions were assigned weights of 10, and deletions and insertions were each assigned weights of 7 [82]. With these error measures it becomes possible to compute the *word accuracy*. Intuitively, it constitutes the percentage of words that the recognizer handles correctly.

In the following I denote the number of words in the test set with "N," the number of substitution errors with "S," the number of deletion errors with "D," and the number of insertion errors with "I."

The word accuracy can then be computed in two steps. First, let

$$H = N - S - D.$$

This number, H, is the number of correctly spotted words in the test set, without regard for insertion errors. The word accuracy is

$$Acc = H - I.$$

While optimizing for the accuracy of a recognizer, it is useful to make a distinction between the number of correctly spotted words (*H*) and the word accuracy (*Acc*). The reason is that the quality of the trained models and the quality of the feature vector mainly affect the number of substitution and deletion errors. In contrast, the number of insertion errors depends less on the quality of the trained models and more on the weights of the transitions from one model to the next one. Hence, it makes sense to optimize for the number of spotted words first.

Nevertheless, the final criterion must encompass the number of insertion errors, as well. For this reason the word accuracy is the evaluation criterion for all continuous recognition experiments. For isolated recognition experiments, in contrast, the evaluation criterion is simply the number of correctly recognized signs out of the total test set.

4.2 Data Collection

I collected two different data sets, which I signed myself. Both of these consisted of continuous sentences — with no pauses between the signs —, and had an unconstrained word order. The sentences were constrained only by what is grammatical in ASL. Both data sets were confined to the lexical aspects of ASL, so they did not incorporate any gestural elements or inflection via the use of space.

The first, and earlier, data set consisted of 486 sentences, between 2 and 12 signs long, and a total of 2345 signs from a 53-sign vocabulary. The vocabulary is given in Table 4.1 on page 100. I collected these data with an Ascension Technologies Flock of Birds^{TM} system, which recorded the data at 25 frames per seconds. It consists of a magnet and six sensors that detect their rotation, $\vec{q_\theta}$, and translation, $\vec{q_c}$ within the magnetic field. The coordinate system was right-handed, with the origin at the base of the signer's spine and the x axis facing up. This first data set was not generated with phoneme modeling in mind, and did not contain any handshape information. In addition, 25 frames per second is a rather low sampling rate for recording American Sign Language, where many movements happen very quickly.

The second, and later, data set consisted of 499 sentences, between 2 and 7 signs long, and a total of 1604 signs from a 22-sign vocabulary. This vocabulary is given in Table 4.2 on page 100, and their detailed phonetic transcriptions for the movement channel are given in Appendix B. I collected these data with an Ascension Technologies MotionStar^{TM} system, which is a more advanced version of the Flock of Birds system. It records the

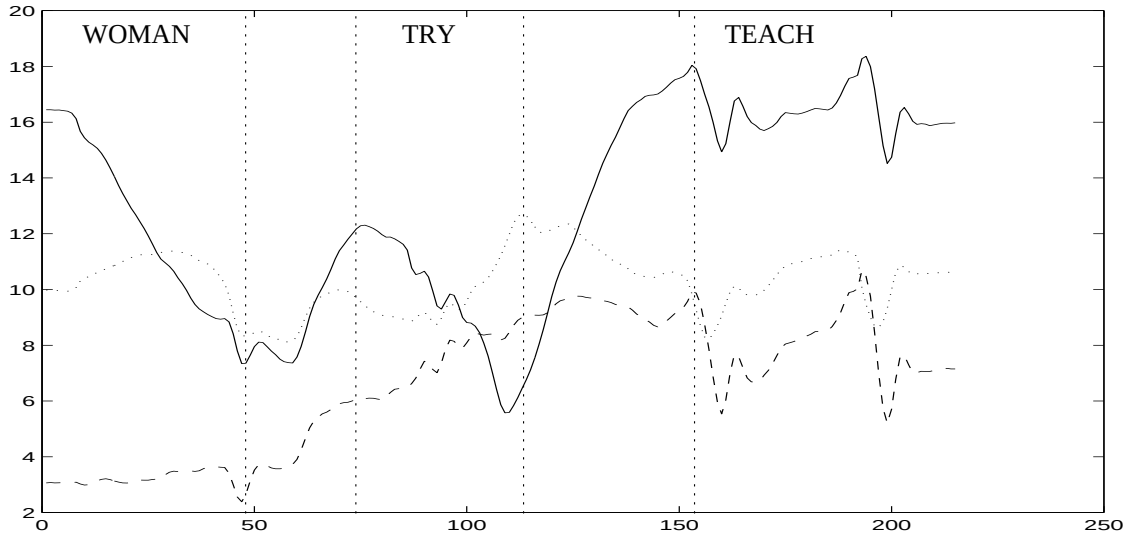


Figure 4.1: Example of the 3D position signal for the sentence WOMAN TRY TEACH. The solid line is from the x coordinate; the dashed line is from the y coordinate; and the dotted line is from the z coordinate. The unlabeled parts of the signal are epenthesis movements.

same type of data as the other system, but at a rate of 60 frames per second. The coordinate system was identical to the one used with the Flock of Birds. In addition, I collected data from the strong hand with a Virtual Technologies Cyberglove^{TM}, which records wrist yaw, pitch, and the joint and abduction angles of the fingers, also at 60 frames per second. The cyberglove needs to be calibrated carefully to a particular human, so as to deliver accurate joint angles. Unlike the first data set, this one was well suited to phoneme modeling.

Note that both vocabularies contain inflected signs, such as the signs for INFORM, and GIVE. However, in these two data sets, for the purposes of the pilot study that this thesis constitutes, I did not consider inflection. I just signed such signs in their citation forms.

Figure 4.1 shows an example of what the data from the MotionStar system look like, by way of the sentence WOMAN TRY TEACH.

The raw data from the Flock of Birds and MotionStar systems are not the best choice for the feature vectors for HMM-based recognition, because among others, they do not

Category	Signs used
Nouns	AMERICA, CHRISTIAN, CHRISTMAS, BOOK, BROTHER, CHAIR, COLLEGE, FAMILY, FATHER, FRIEND, INTERPRETER, LANGUAGE, MAIL, MOTHER, NAME, PAPER, PRESIDENT, SCHOOL, SIGN, SISTER, TEACHER
Pronouns	I, MY, YOU, YOUR
Verbs	ACT, CAN, GIVE, HAVE, INTERPRET, LIKE, MAKE, READ, SIT, TEACH, TRY, VISIT, WANT, WILL, WIN
Adjectives	DEAF, GOOD, HAPPY, RELIEVED, SAD
Other	IF, FROM, FOR, HI, HOW, WHAT, WHERE, WHY

Table 4.1: The early 53-sign vocabulary.

Category	Signs used
Nouns	CHAIR, FATHER, MAN, MOTHER, INTERPRETER, TEACHER, WOMAN
Pronouns	I
Verbs	DON'T-MIND, GET, INFORM, INTERPRET, LIE, RELATE, SIT, TEACH, TRY
Adjectives	BEAUTIFUL, GOOD, GROSS, STUPID
Other	SORRY

Table 4.2: The later 22-sign vocabulary. The phonetic transcriptions are given in Appendix B.

contain information about the direction of movements. I now discuss ways to extract more information from these data and how to add them to the feature vector.

4.3 Feature Vector

Recall that the feature vector is a vector of numeric values that represents the signs at every frame, and constitutes the input to the HMM recognition system. Unlike speech recognition, which has decades of research behind it, there is no consensus on what constitutes good features for HMM-based gesture and sign language recognition. Each approach uses different features, such as the 2D positions of the hands and the orientation of an oval approximating the hand orientation [67], the areas of different parts of a colored glove [29, 31], vector quantization of joint angles [43], projection of the 3D positions of the hands onto the principal-component planes [49, 48], and a mix of 3D positions and the polar coordinates of these 3D positions projected on the frontal and horizontal planes [73]. The work in [13] attempted to resolve some of the questions around which features are good for human activity recognition, by exploring different feature vectors for recognition of Tai'Chi sequences. Unfortunately, the data sets used in that work are too small to provide conclusive answers.

4.3.1 Feature Vector Comparison Experiments

This thesis does not attempt to provide a conclusive answer to the question of which feature vectors are optimal for sign language recognition. Nevertheless, it is possible to take a few considerations into account. First, we would expect feature vectors based on 3D data to do better than feature vectors based on 2D data. Second, there seems to be a general tendency for complex feature vectors to do better than simple ones, although the more complex the feature vector, the higher is the danger of overtraining the HMMs. Third, velocities provide

important information about the direction of movement. They can easily be extracted from the 3D positions with one of the common discrete derivative filters.

To test these assumptions, I ran a set of isolated recognition experiments. For this particular type of experiment, isolated recognition is much more convenient to work with than continuous recognition, because it allows for easy selection of random training and test sets. In contrast, if the training and test sets for continuous recognition are selected at random, manual intervention is necessary to merge movement epenthesis models for which not enough training data are available.

The set of features tested in these experiments included wrist position coordinates of both hands (denoted by x, y, z), wrist position expressed in polar coordinates in the x-y plane (denoted by r_{xy}, θ_{xy}), polar coordinates in the x-z plane (denoted by r_{xz}, θ_{xz}), wrist position expressed in spherical coordinates (denoted by r, θ, ϕ), and wrist orientation angle (denoted by δ), as well as derivatives of these (denoted by a dot).

Overall, I ran more than $10,000$ experiments on the 53-sign vocabulary with different feature vectors and different training and test data sets, all of which were selected at random on a per-experiment basis. Three quarters of the examples for each sign were in the training set, and the rest were in the test set. Each selection yielded 178 test examples per experiment. A selection of the results is given in Table 4.3 on the following page.

Although the results of the isolated recognition experiments are, for most part, very close, some trends emerge: First, more complex feature vectors consistently exhibit lower standard deviations, which indicates that they indeed are more robust than simple feature vectors. Second, as desirable as velocities by themselves would be as feature vectors for the movement components of signs, they perform poorly compared to other feature vectors. As a result, coming up with feature vectors that are invariant with respect to location is difficult. Third, 3D feature vectors are significantly more robust than 2D feature vectors, so there is some justification in working with 3D data, even though it is much harder to capture than 2D data.

Features	μ	σ	B	W	N
x, y, z	98.42%	0.99%	100.0%	93.8%	463
x, y	97.75%	1.20%	100.0%	94.9%	118
r_{xy}, θ_{xy},z	98.72%	0.79%	100.0%	95.5%	494
r_{xy}, θ_{xy}	98.06%	1.26%	100.0%	94.9%	118
r_{xy}, r_{xz},θ_{xy}, θ_{xz}, x, y, z	98.78%	0.78%	100.0%	94.9%	882
r, θ, ϕ	96.48%	1.31%	100.0%	93.3%	210
$\dot{x}, \dot{y}, \dot{z}$	96.87%	1.21%	100.0%	93.3%	167
x, y, z, δ	98.25%	0.92%	100.0%	95.5%	167
$\dot{r}_{xy}, \dot{\theta}_{xy}$, $\dot{z}$	96.28%	1.04%	98.9%	93.8%	120
$\dot{r}, \dot{\theta}, \dot{\phi}$	95.89%	1.29%	98.9%	92.1%	150

Table 4.3: Results of isolated sign recognition with two-and three-dimensional features. μ, σ, B, W, and N correspond to the average percentage of correctly recognized signs, standard deviation, best case, worst case, and number of experiments, respectively. All experiments used a test set of 178 signs.

Polar coordinates slightly outperform Cartesian coordinates, but the difference is not large enough to justify using them in place of Cartesian coordinates in conjunction with 3D data, especially because polar coordinates are an unnatural choice for 3D features. The reason is that it is necessary to choose a projection plane, whose depth must be expressed in Cartesian coordinates. This choice is essentially arbitrary, and thus there would always be a component of the feature vector that is treated differently from the rest in an arbitrary manner.

A look at the best and worst results for each feature vector, respectively, is also very revealing. It highlights the dangers of using just one single experiment to compare feature vectors, such as in [13]. For a *single* experiment to yield statistically significant results, either the number of examples in the test set must be sufficiently large — which is not the case for the mere 178 signs in the isolated recognition test set —, or the difference in recognition accuracy must be sufficiently large. Unfortunately, with the amount of work involved in collecting a sufficiently large corpus suitable for continuous sign language recognition, this point will remain a potential stumbling block for some time to come.

4.3.2 Global Features

There is another point about feature vectors that deserves to be expanded on. Most features that have been used for recognition are extremely localized in the sense that they do not provide any information on the data signal even a few hundred milliseconds from the sampling point. Both the position of the hands in the signing space and the velocities of the hands are examples of local features. They do not reveal anything about the long-term appearance of the sign.

In contrast, in the Movement-Hold model, the phonemes in the movement channel describe geometric properties of the signal on a more global level, such as movements along a straight line, or along an arc. Thus, it is desirable to have a quantitative measure of some of these global properties. An example of such a measure is how well the signal fits a line or a plane within a specific fixed-size time window.

This measure can be easily computed through a principal component analysis of the 3D positions; that is, by estimating the covariance matrix over the points in the window and taking its eigenvalues. If the largest eigenvalue is significantly larger than the other two eigenvalues, the signal fits a line well. If the two largest eigenvalues are nearly equally large, and significantly larger than the smallest eigenvalue, the signal fits a plane well. To quantify these relationships with two numbers, the recognition system can take the square roots of the two largest eigenvalues, and normalize them such that the sum of the square roots of all three eigenvalues is 1.

4.3.3 Handshape Features

Most approaches in the past used joint and abduction angles as features for the hand, whenever these had been available, such as [43, 42, 23]. These are very low-level features, however; essentially they are the counterpart to the local features described in the previous section. The experimental results shown below indicate that these are not the best choice

for recognizing the handshape (see also Section 3.1.1).

Therefore, it makes sense to use a more global description of the handshape. For sign languages, a representation of the degree of openness of a finger [7] seems particularly useful, which also is used by Sandler's phonological model of the handshape [63]. For obtaining such a representation, consider the relationship among the fingertip, the metacarpophalangeal joint (MPJ, the finger joint closest to the palm), and the degree of openness. If a finger is fully extended — that is, fully open —, the distance between the fingertip and the MPJ is maximized. Likewise, if the finger is fully bent — that is, fully closed —, the distance between these two points is minimized. For obtaining yet another measure of the openness, consider connecting the fingertip and the three finger joint angles into a quadrilateral, as shown in Figure 4.2 on the next page, where the base is the line between the fingertip and the MPJ. The height of the quadrilateral, expressed as the distance between the proximal interphalangeal joint (PIJ, the joint closest to the MPJ) and the base line, is maximized when the finger is fully closed. Likewise, the height is minimized when the finger is fully open.

Therefore, these two measures — the width and the height of the quadrilateral described by a finger — are a direct expression of a finger's openness. Note that the MPJ angle only rotates this quadrilateral, but does not affect its dimensions. Thus, whether a finger is open or closed is largely independent of the MPJ angle. Hence, the MPJ angle should be part of the feature vector in addition to the width and height measurements. Together with the abduction angles (the angles measuring the spread between adjacent fingers), these features all together constitute a somewhat higher-level representation of the hand than the raw joint angles.

This discussion so far has focused only on individual fingers. Another possibly helpful global measure may be the degree of openness of the hand as a whole, even though the feature vector experiments in this thesis do not conclusively establish whether the measure helps or hurts recognition accuracy. To this end, consider the sites of all finger joints, with

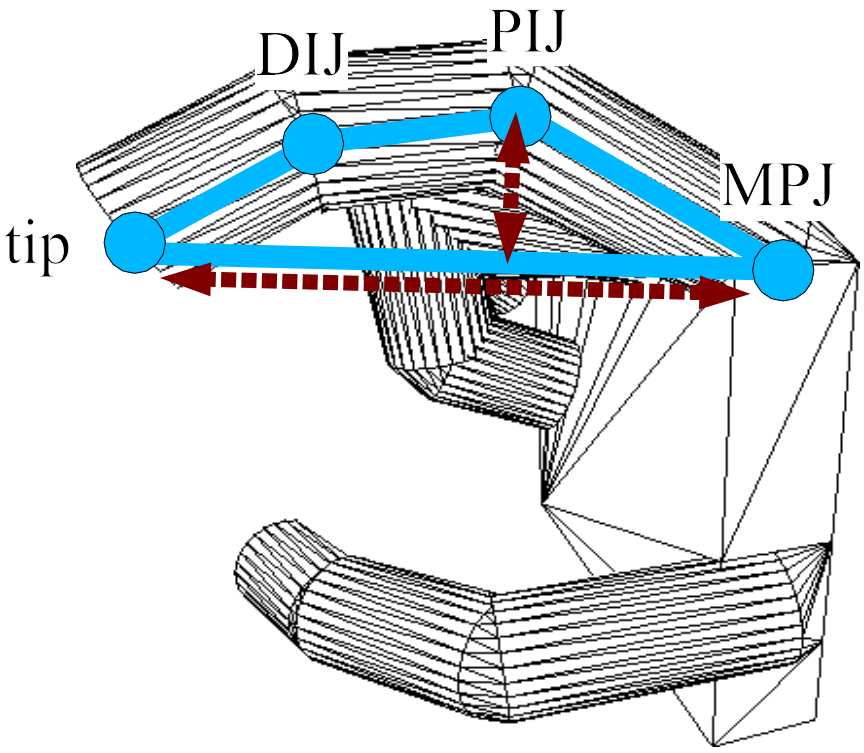


Figure 4.2: Measure of the openness of a finger. It depends on the width and height of the quadrilateral described by the sites on the three finger joints and the fingertip.

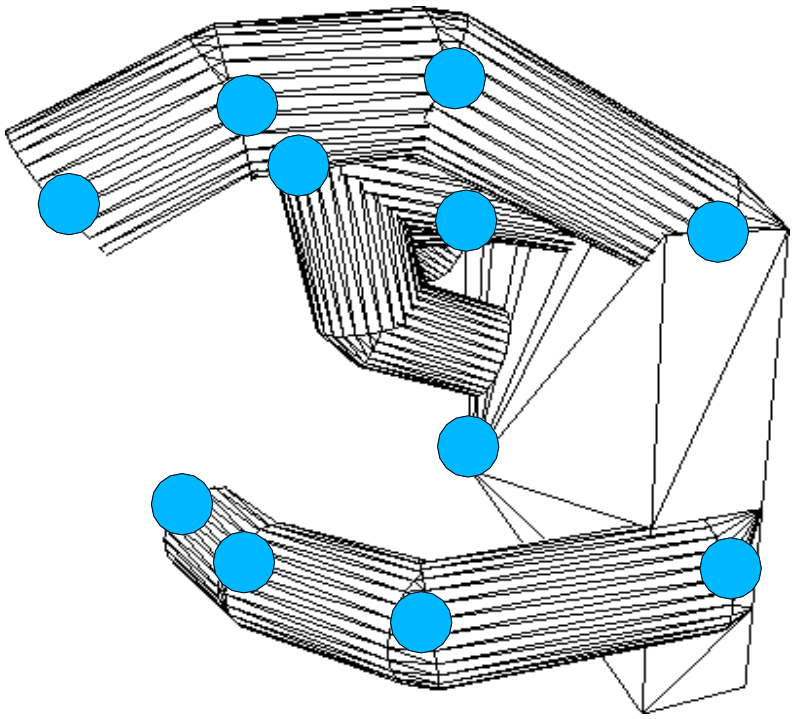


Figure 4.3: Measure of the overall degree of hand openness. The volume of the convex hull (depicted by the marked points) of the joint sites provides a measure of how much space the handshape occupies.

a joint on the pinky side added on the same height as the thumb MPJ for symmetry. The convex hull of all these sites gives an indication of the overall space that the handshape occupies (Figure 4.3), where an open hand occupies more space than a closed one.

The volume of this convex hull constitutes the desired measure. It can be computed with the classical formula

$$\vec{A}_i = v_{i,|v_i|} \times v_{i,1} + \sum_{j=1}^{|v_i|-1} v_{i,j} \times v_{i,j+1}, \tag{4.1}$$

$$V = \sum_{i=0}^{F} v_{i,1} \cdot \vec{A}_i, \tag{4.2}$$

where V is the volume, F is the number of faces in the convex hull, $v_{i,j}$ is vertex j in face i, and $|v_i|$ is the number of vertices in face i.

4.3.4 Handshape Feature Comparison Experiments

To determine the relative merits of using joint angles, measures of finger openness, and measures of hand openness, I ran continuous recognition experiments on the 22-sign data set with the same training and test sets as the ones used in the main experiments in Section 4.4. The only channel used in these experiments was the handshape channel. There were three different types of experiments overall, which tested the following feature vectors, respectively:

- all finger joint and abduction angles.
- MPJ angles, abduction angles, quadrilateral width and height.
- MPJ angles, abduction angles, quadrilateral width and height, convex hull volume.

For each feature vector, the experiments varied the number of HMM states and parameters; for example, the number of outgoing transitions from each state. In all cases, the transitions between two handshapes were captured explicitly with HMMs of their own, as described in Section 2.6. Because many signs share the same handshape, it is impossible to identify a sign uniquely from the handshape alone. For this reason, and in contrast to all the other experiments discussed in this thesis, the evaluation criterion was not the percentage of correctly recognized signs, but rather the percentage of correctly recognized handshapes. In addition, whenever the same handshape occurred multiple times in a row, I contracted these occurrences into a single handshape. In every other respect, the evaluation criterion encompassed deletion, substitution, and insertion errors among the handshapes exactly analogous to the word accuracy measure described in Section 4.1. The results are given in Table 4.4 on the next page.

Feature Vector	μ	σ	Median	Best	N
joint angles	83.15%	15.44%	82.66%	98.68%	1912
quadrilateral	95.21%	5.37%	96.83%	99.47%	1956
quadrilateral + volume	95.01%	5.22%	96.57%	99.21%	1956

Table 4.4: Results of continuous handshape feature vector comparisons. μ, σ, Median, Best, and N correspond to the average handshape accuracy, standard deviation, best case, and number of experiments, respectively.

The number of experiments for the joint angles in this table is slightly lower than the number of experiments for the other two types of feature vectors, because in some instances the Viterbi beam search (see Section 3.1.2) caused all tokens in the network to expire prematurely. The results show clearly that global descriptions of the handshape are far more robust than the raw joint angles. The results for and against using the convex hull volume are inconclusive. They are slightly better for the case without the convex hull volume, but this difference is not significant. To resolve this question, experiments with larger data sets and more handshapes are needed.

These results also highlight that for this particular data set, modeling the transitions between handshapes explicitly with HMMs of their own works well. However, the number of handshapes involved in this data set is too small to provide a conclusive answer to the question whether this kind of modeling will scale well to larger vocabularies and data sets.

4.3.5 Feature Vector Summary

Based on the points discussed in the previous sections, I chose among the following mixes of feature vectors for the continuous recognition experiments:

- $x, y,$ and z positions, $\dot{x}, \dot{y},$ and $\dot{z}$ velocities, polar angles θ_{xy}, θ_{xz} of the positions' projection into the x-y and x-z planes, and wrist rotation angles δ extracted from the orientation angles of the sensors,

- $x, y,$ and z positions, $\dot{x}, \dot{y},$ and $\dot{z}$ velocities,
- $x, y,$ and z positions, $\dot{x}, \dot{y},$ and $\dot{z}$ velocities, and normalized two largest eigenvalues e_1, e_2 over a 15-sample window,
- metacarpophalangeal joint angles, abduction angles, and quadrilateral width-height-based measure of finger bending; with and without convex hull volume.

I now discuss the continuous recognition results, which constitute the main results of this thesis.

4.4 Continuous Recognition Results

In the continuous sign language recognition experiments there were no pauses between the individual signs in a sentence. The sentences themselves were clearly marked apart. The experiments on the early 53-sign data set were designed to test the different methods of handling movement epenthesis (see Section 2.4.1 for details). For these experiments I split the set of 486 sentences randomly into a training set with 389 examples containing 1886 signs, and a test set with 97 examples containing 456 signs. Each sign in the vocabulary occurred at least once in the test set. The training and test sets were the same throughout all these experiments, and no portion of the test set was used for training in any way.

The experiments on the later 22-sign data set were designed to compare whole-sign modeling with epenthesis, breaking down the signs into phonemes, feature vectors with and without global features, and modeling a single channel versus multiple channels with PaHMMs. I split the data set for these experiments randomly, too, into 400 training examples containing 1292 signs, and 99 test examples containing 312 signs. Just like in the 53-sign experiments, the training and test sets were the same throughout all experiments, and no part of the test set was used for training in any way.

Type of experiment	Word accuracy	Details
context-independent	87.71%	H=416, D=8, S=32, I=16, N=456
context-dependent	89.91%	H=424, D=6, S=26, I=14, N=456
bigram, context-dependent	91.67%	H=426, D=7, S=23, I=8, N=456
epenthesis modeling	92.11%	H=426, D=7, S=23, I=6, N=456
bigram, epenthesis modeling	95.83%	H=438, D=11, S=7, I=1, N=456

Table 4.5: Comparison of approaches to handling movement epenthesis. The feature vector consisted of $(x, y, z, \dot{x}, \dot{y}, \dot{z}, \theta_{xy}, \theta_{xz}, \delta)$ of both hands; see the end of Section 4.3 for an explanation. H denotes the number of correct signs, D the number of deletion errors, S the number of substitution errors, I the number of insertion errors, and N the total number of signs in the test set.

4.4.1 Methods to Handle Movement Epenthesis

I ran five experiments on the 53-sign set to compare the relative merits of different approaches to handling movement epenthesis with conventional HMMs. The first experiment used simple modeling of one HMM per sign, and ignored the transitions between signs altogether. The second experiment used context-dependent HMMs, which means one HMM per sequence of two signs. The third experiment used context-dependent HMMs and added bigram language models, where a probability is assigned to each sequence of two signs. These probabilities weight the transition between the word end and word start nodes of the HMMs. To obtain rough estimates of these probabilities I ran statistics on the training corpus. The fourth experiment used one HMM per sign and modeled movement epenthesis explicitly with separate HMMs. The fifth experiment was similar to the fourth, but added bigram language models, like the third experiment. All experiments modeled both hands with conventional HMMs, so they were captured in a highly coupled manner.

The results are given in Table 4.5. Two observations stand out: First, modeling movement epenthesis explicitly yields large benefits. Second, modeling movement epenthesis and bigram language models complement each other well, so if a recognition system uses bigram models there is no reason not to use epenthesis models, as well.

Type of experiment	Word accuracy	Details
whole-sign	92.95%	H=293, D=6, S=13, I=3, N=312
phoneme-level, local features	90.06%	H=286, D=8, S=18, I=5, N=312
phoneme-level, global features	93.27%	H=294, D=3, S=15, I=3, N=312

Table 4.6: Comparison of whole-sign and phoneme modeling. The feature vector of the first two experiments consisted of $(x, y, z, \dot{x}, \dot{y}, \dot{z})$ of the strong hand (i.e., the movement channel). The feature vector of the third experiment consisted of $(x, y, z, \dot{x}, \dot{y}, \dot{z}, e_1, e_2)$; see the end of Section 4.3 for an explanation. H denotes the number of correct signs, D the number of deletion errors, S the number of substitution errors, I the number of insertion errors, and N the total number of signs in the test set.

4.4.2 Phoneme Modeling and Global Features

I ran three experiments on the 22-sign set, of which the first one was a control experiment, which used one HMM per sign and modeled movement epenthesis explicitly, similar to the fourth experiment in the previous section. The goal of this experiment was to provide a baseline for whole-sign modeling. The second experiment compared this baseline to phoneme modeling of the strong hand with the same feature vector as in the control experiment. The third experiment also tested phoneme modeling, but added the two largest, normalized eigenvalues of a 15-frame window to the feature vector as global features. All experiments used conventional HMMs and captured only the strong hand's movement channel. In the phoneme modeling experiments there were a total of 43 unique HMMs for the X, movement, and hold segments, and 46 HMMs for the epenthesis models — compared to the whole-sign experiment with 22 HMMs for the signs, and the 46 HMMs for the epenthesis models. If the number of signs in the vocabulary were larger, phoneme modeling would require fewer HMMs than whole-sign modeling.

The results are given in Table 4.6. Phoneme modeling sacrifices some recognition accuracy, but the benefits of the reduced modeling complexity outweigh this loss. Using global features makes a big difference and allows phoneme modeling to recoup the loss in recognition accuracy.

4.4.3 PaHMMs with Multiple Channels

I ran four experiments on the 22-sign set to compare the recognition accuracy of modeling just the strong hand's movement channel with conventional HMMs, versus the accuracy of using PaHMMs. The first experiment was the same one as the global features experiment from the previous section. The second experiment used PaHMMs to capture the movement channels of both hands. In this experiment there were a total of 89 unique HMMs for the movement channel of the strong hand, as in the previous section, and a total of 51 unique HMMs for the movement channel of the weak hand, of which 29 were epenthesis models. The third experiment used PaHMMs to capture the movement and handshape channels of the strong hand. There were a total of 71 unique HMMs in the handshape channel, of which 58 were models describing the transition between two handshapes. The fourth, and final experiment used all three channels: movements of the strong and weak hand, respectively, and the handshape of the strong hand. The number of hypotheses used for the PaHMM token passing algorithm was 3 in the second, 4 in the third, and 10 in the fourth experiment, respectively. The probabilities were combined at the phoneme level; that is, after each phoneme. The results are given in Table 4.7 on page 115.

An analysis revealed that the single-channel recognition experiments with conventional HMMs left only seven sentences with incorrectly recognized two-handed signs. Each of these seven sentences involved a single substitution error. Thus, the maximum theoretical recognition rate that PaHMMs could have achieved through modeling the movement channels of both hands was 87.88% on the sentence level and 96.47% on the word accuracy level. Of the seven sentences with two-handed signs that the conventional HMMs failed to recognize, the PaHMMs recognized four correctly in the second experiment. One of the other three sentences now contained an additional substitution error in a one-handed sign. All other sentences were not affected.

In the third experiment the PaHMMs corrected many recognition errors from the single-channel recognition experiment, but in three cases introduced errors into sentences that were recognized correctly in the single-channel experiment. In the fourth experiment using all three channels performed slightly, but not significantly, worse than using only the two channels from the strong hand. The explanation for this result most likely stems from the nature of the data set that I used for the experiments: It does not contain any minimal pairs of signs with identical handshapes and hand movements, where the only significant difference lies in the number of hands used. Hence, adding the information about the movements of the weak hand does not increase the discriminative power of the PaHMM networks any further.

Clearly, in future work a larger data set with appropriate minimal pairs is needed to determine conclusively how much the PaHMM recognition framework can benefit from adding even more channels. Yet, overall even these results seem to validate the assumption that in practice channels can be modeled independently from one another.

4.4.4 Effect of PaHMM Parameter Adjustments

To investigate how various changes in the parameters of the recognition experiments affect PaHMM recognition accuracy, I ran several experiments that varied these parameters. These experiments tested using different feature vectors for the handshape, varying the number of hypotheses, and combining the probabilities only after each word, instead of each phoneme.

In the first set of experiments I used PaHMMs to test the different feature vectors for the handshape that I had previously compared in Section 4.3.4. These experiments were aimed at resolving the question of how much PaHMMs can compensate for known inferior feature vectors (in this case, the raw finger joint angles). The results, given in Table 4.8 on page 116, are surprising, to say the least. As a part of a PaHMM recognition network,

Type of experiment	Sentence accuracy	Word accuracy	Details
movement channel strong hand, HMM	80.81%	93.27%	H=294, D=3, S=15, I=3, N=312
movement channel both hands, PaHMM	84.85%	94.55%	H=297, D=3, S=12, I=2, N=312
movement channel strong hand, hand-shape strong hand, PaHMM	88.89%	96.15%	H=302, D=2, S=8, I=2, N=312
all three channels, PaHMM	87.88%	95.51%	H=301, D=1, S=10, I=3, N=312

Table 4.7: Comparison of conventional HMMs and PaHMMs. The conventional HMMs modeled the strong hand's movement channel, whereas the PaHMMs modeled a combination of multiple channels. The number of hypotheses used was 3 in the second experiment was, 4 in the third experiment, and 10 in the fourth experiment. The probabilities were combined at the phoneme level. The movement channel feature vector consisted of $(x, y, z, \dot{x}, \dot{y}, \dot{z}, e_1, e_2)$, and the handshape channel feature vector consisted of MPJ angles, quadrilateral measurements, and abduction angles (see Section 4.3.5 for an explanation). H denotes the number of correct signs, D the number of deletion errors, S the number of substitution errors, I the number of insertion errors, and N the total number of signs in the test set.

Type of experiment	Sentence accuracy	Word accuracy	Details
quadrilateral	88.89%	96.15%	H=302, D=2, S=8, I=2, N=312
quadrilateral + volume	86.87%	95.51%	H=299, D=2, S=11, I=1, N=312
joint angles	89.90%	96.15%	H=302, D=1, S=9, I=2, N=312

Table 4.8: Comparison of different handshape feature vectors with PaHMMs. All experiments modeled the movement and handshape channels of the strong hand. The number of hypotheses was 4 in all experiments, and the probabilities were combined at the phoneme level. The movement channel feature vector consisted of $(x, y, z, \dot{x}, \dot{y}, \dot{z}, e_1, e_2)$, and the handshape channel feature vector consisted of the same features as the ones in Table 4.4 on page 109 (see Section 4.3.5 for an explanation). H denotes the number of correct signs, D the number of deletion errors, S the number of substitution errors, I the number of insertion errors, and N the total number of signs in the test set.

the raw joint angles held their own, and in fact, regarding sentence accuracy even slightly surpassed the more sophisticated description of the handshape with the MPJ angles, finger quadrilateral width and height, and abduction angles. Hence, at least to some extent, PaHMMs also make the recognition system more robust to bad data. This property is strikingly similar to the original intended use of PaHMMs in the speech recognition field to diminish the effect of noise in speech [30, 6]. The question of whether the convex hull volume of the hand should be included in the feature vector, still cannot be resolved with this data set.

In the second set of experiments I compared merging the PaHMM token probabilities after each phoneme and merging them only after each word. The results are given in Figure 4.4 on the following page. They indicate clearly that merging on the phoneme level is superior, because it requires only approximately half the number of hypotheses compared to merging on the word level, before full recognition accuracy is reached.

In the third set of experiments I investigated the effect of the number of hypotheses in more detail. I did these comparisons for merging on the word level, because there the different channels differ more starkly than for merging on the phoneme level. The results are

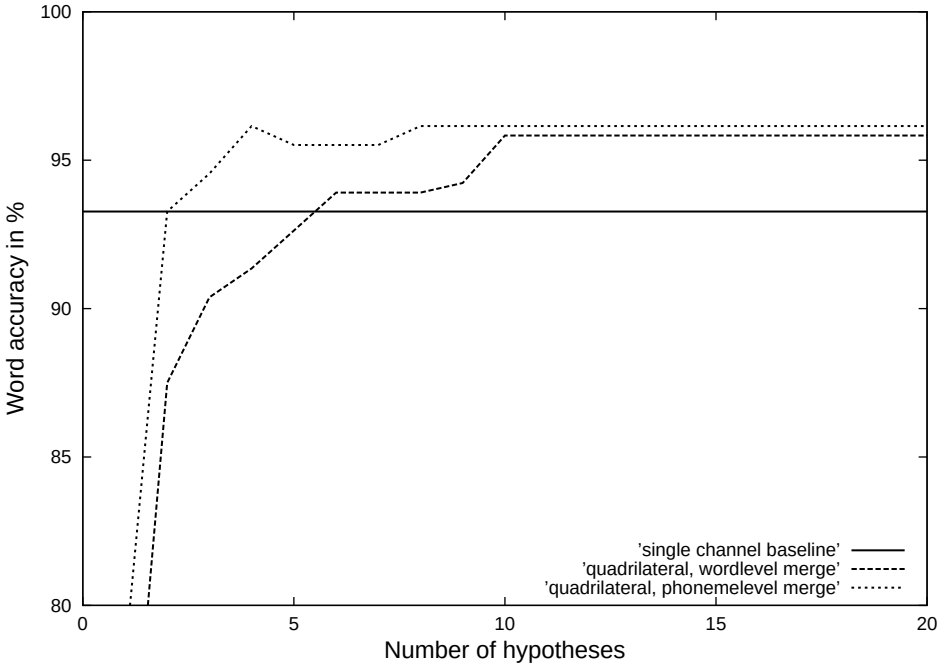


Figure 4.4: Comparison of merging the probabilities of the channels after each word and after each phoneme. Merging after each phoneme decreases the minimum required number of hypotheses. The channels were the strong hand movement and handshape. The feature vector is explained in more detail in Section 4.3.5.

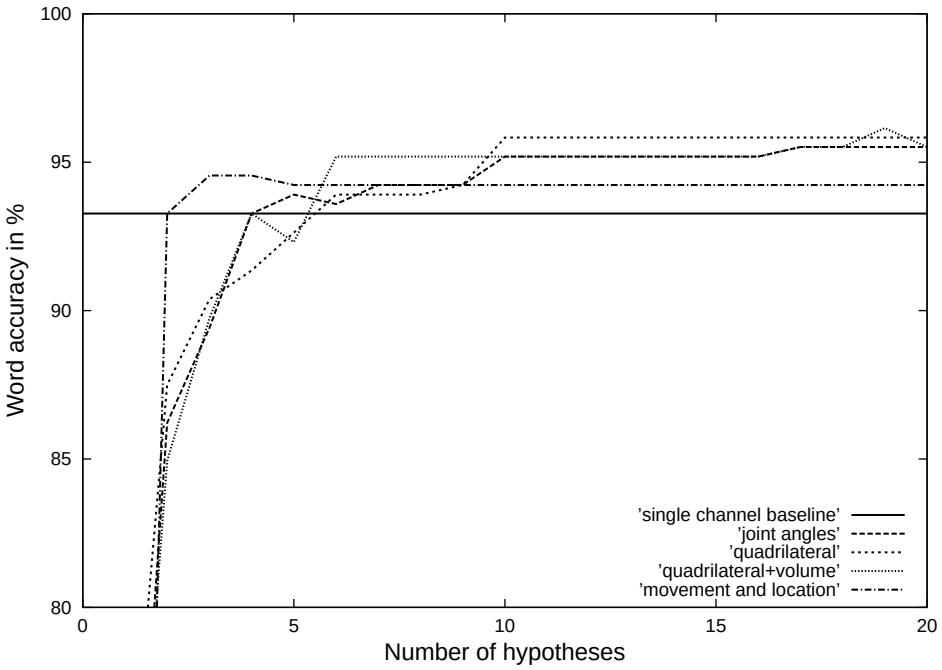


Figure 4.5: Effect of the number of hypotheses on recognition accuracy on the word level. These results are for the the combination of the strong and weak hand movement channels, and various combinations of the strong hand movement and handshape channels. The handshape feature vector varied in each experiment, and is explained in more detail in Section 4.3.5. The horizontal line denotes the baseline recognition accuracy with only one channel.

given in Figure 4.5. They show that for combining the movement channels fewer hypotheses are needed than for combining the movement channel with the handshape channel. The reason is most likely that, in HMM theory, the Gaussian densities, which describe the continuous data signals, are used like probability distribution functions, but they really are not. In the case of the movement channels, the feature vectors are identical, and hence the magnitudes of the Gaussian densities in the HMM states are very similar, as well. As a result, recognition is unbiased toward either channel.

In contrast, the feature vectors of the movement and handshape channels are radically

different, with different dimensions, so the magnitudes of the Gaussian densities are different, as well, even after normalizing them. This difference unbalances the probabilities in the different channels. Section 3.3.5 discusses this problem in more detail.

4.5 Summary

In this chapter I have presented a number of experiments on ASL recognition that aim at validating the assumptions made and the approach taken in this thesis. The main results from these experiments are:

- Complex feature vectors and global features help robustness.
- A higher-level representation of the handshape with a measure of how much a finger is bent is substantially more robust than raw joint angles.
- Modeling movement epenthesis explicitly with dedicated HMMs works better than cross-sign context-dependent HMMs.
- Phoneme modeling of ASL does not substantially hurt recognition accuracy over whole-sign modeling.
- PaHMMs yield better recognition rates, thus justifying that the independence assumption can be made in practice.
- Merging the PaHMM probabilities after each phoneme reduces the minimum required number of recognition hypotheses per channel.
- PaHMMs can to a certain extent compensate for nonoptimal feature vectors in a channel.

Overall, the results hold promise that the phoneme modeling and the modeling with independent channels via PaHMMs will allow the framework to scale to larger vocabularies in

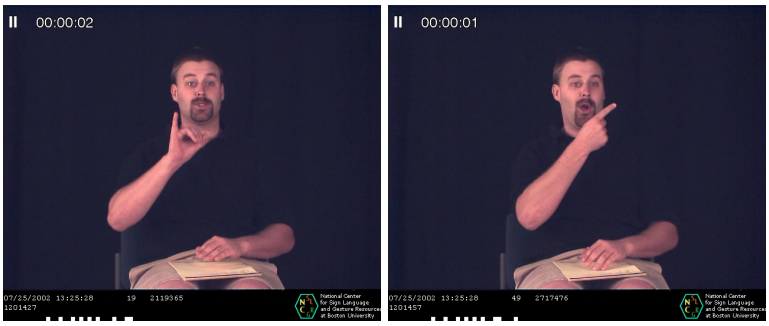
future work, and that the computational cost of larger vocabularies will not be prohibitive. There is an important caveat, however: I collected all the experimental data myself, but I am not a native ASL signer. Hence, these results are promising within the scope of a pilot study, but they need to be validated and reaffirmed in future work. I now discuss differences to native signers, and how the recognition framework would need to be adapted in future work.

Chapter 5

Adaptation to Native Signers

A comparison of my data set with the videos of native signers from the National Center for Sign Language and Gesture Resources (NCSLGR) at Boston University [1] highlights several important differences that need to be accounted for when applying the framework to native signers. Before I address these in detail, first note that this discussion only pertains to signs and sentences that are similar in structure to the ones that I used in my data sets. To this end, I deliberately exclude all discussion of inflection, and video samples that exhibit inflection, because I did not address inflection in my own data sets.

Indexical Signs I did not make use of indexical signs in my data set, even when they would have been appropriate. These signs, which are very commonplace, point to a particular location in space, and serve grammatical functions, among them determiners, pronouns, and adverbials [50]. In the role as a definite determiner ("this particular person or object"), they express that the signer is talking about a particular subject or object, as opposed to just someone or something. In their role as pronouns, which are identically articulated to definite determiners, they reference an entity that has been placed in the signing space before. In their role as adverbials they express precise information about a



(a) JOHN

(b) indexical reference (adverbial)

Figure 5.1: Example of an indexical sign, in this case an adverbial. The fingerspelling of the name “John” is followed by a indexical reference over the shoulder, which indicates that John is somewhere behind the signer. Source of images: [1].

location in space. In contrast to determiners, adverbials possess more freedom of articulation; for example, the indexing could be modified to a long movement to indicate that the referent is far away, or to a bending around a corner to indicate that the referent is physically around the corner [50].

Figure 5.1 shows an example of such an adverbial in the sequence JOHN IX-shoulder. The signer spells out the name of “John” and then points over the shoulder, indicating that John is present in the vicinity somewhere behind him.

The use of indexical signs is so prevalent among the video footage from the NCSLGR that it is one of the first things that should be accounted for in an extension of my framework to native signers. In their role as determiners and pronouns, the recognition framework must be able to recognize the location in space that they point to. In contrast, capturing their role as adverbials is much more difficult, because of the greater freedom of articulation. Computationally, this role poses yet another example of a simultaneous event: the pointing motion is simultaneously modified by the characteristics of the referent, such as its distance relative to the signer.

Weak hand behavior In the discussion about the need for channel weights in Section 3.3.2 I made the assumption that the weak hand channels do not contain any interesting information during one-handed signs. The video snapshots in Figure 5.2 on the next page show clearly that this assumption is false. They show a sequence of the sign WHAT-FOR and a manual expression of indefiniteness with both hands raised and the palm facing up. These signs can be performed separately, strictly one after the other, but in this particular sequence the weak hand is already in position for the following two-handed sign at the beginning of the sign for WHAT-FOR. On a first glance, it seems that that a recognition system can simply ignore this phenomenon, because the information is still contained in two adjacent signs.

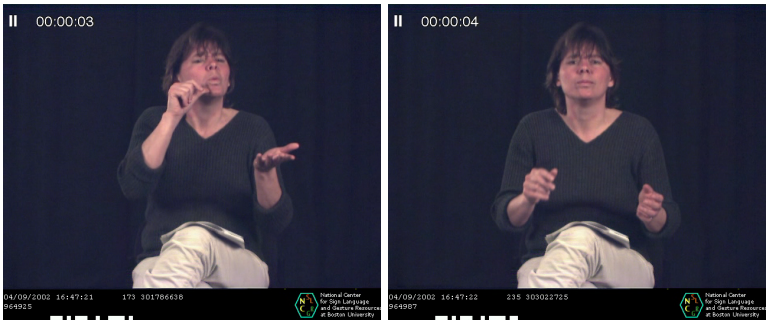
The next example, however, shatters this argument. Figure 5.3 on the following page shows a sequence that ends in the one-handed sign for WHO, with the weak hand raised in the same question position as before. Unlike in the previous example, however, the two-handed sign does not follow at all; the hands simply return to a neutral position indicating that the sequence is over. In other words, the weak hand expresses the particle of indefiniteness simultaneously with the articulation of WHO. Consequently, there do exist situations involving one-handed signs when the action of the weak hand is significant.

Clearly, a future extension of the recognition framework to native signers will have to contend with this phenomenon. The assumption that channels are independent from one another is very helpful here, but there is no obvious way to capture this phenomenon without further research. The crux of the problem is detecting somehow whether the weak hand does something significant during a one-handed sign, as opposed to just moving into position for a sign, or staying in a neutral pose.

Generalization of movement epenthesis The next difference highlights a potentially much thornier problem than the previous two. I made the assumption that the HMM to use for movement epenthesis is completely determined by the ending and starting locations of

(a) WHAT-FOR (b) manual marker of indefiniteness

Figure 5.2: Example of the weak hand holding a position during a one-handed sign. The sign for WHAT-FOR is performed with the index finger repeatedly moving away from the temple in a twisting movement. Meanwhile, the weak hand already holds the position of the following sign, which is a manual expression of indefiniteness. Source of images: [1].



(a) WHO and indefiniteness (b) Hands return to neutral position

Figure 5.3: Example of the weak hand performing the manual expression of a question simultaneously with the sign for WHO. Unlike in the sequence in Figure 5.2, the two-handed sign marking indefiniteness does *not follow.* Source of images: [1].

the surrounding two signs, but mentioned at the end of Section 2.4.1 that in practice with native signers things are not quite so simple.

Figures 5.4 on the next page and 5.5 on the following page provide a good counterexample, which demonstrates the problem with this assumption. By way of the sequence JOHN LIKE CHOCOLATE ("John likes chocolate.") these figures exhibit an interesting phenomenon with the sign for LIKE. In its citation form this sign starts with a contact hold at the chest followed by a movement straight away from the chest. Within my modeling framework, this sequence thus would be represented as the hold at the chest, followed by the movement straight away from the chest, followed by an epenthesis movement from the endpoint of LIKE to the starting point of the sign for CHOCOLATE on top of the weak hand. However, in this video sequence the movement straight away from the chest never happens. Instead, there is only the epenthesis movement from the *starting* location of the sign for LIKE to the starting location of the sign for CHOCOLATE.

This phenomenon poses two problems to the recognition framework at once: First, it is not possible to determine which epenthesis HMM to use solely from the ending and starting locations of the surrounding two signs. The locations of a sequence of signs influence one another, and there is currently no computational model of how exactly these locations are affected. Second, the movement in the citation form of LIKE is assimilated into the epenthesis movement to the sign for CHOCOLATE, and the question is how to capture the assimilation with an HMM. There is no easy answer to this question, because for the scalability reasons outlined throughout this thesis, it is infeasible to model all possible contexts of two signs explicitly, and to use a separate HMM for each of these.

For an answer to the first question, some help from linguistic research into ASL is needed. For an answer to the second question, one possible avenue of research is synthesizing a new HMM from the movement in the citation form of LIKE and the epenthesis movement to the starting location of CHOCOLATE. Possibly, parametric HMMs (PHMMs) [80] could be of help. The difference between conventional HMMs and PHMMs

(a) JOHN (b) LIKE (c) CHOCOLATE

(d) JOHN (e) LIKE (f) CHOCOLATE

Figure 5.4: Example of how context affects the movements in signs. In the sequence JOHN LIKE CHOCOLATE, the movement for LIKE is absorbed into the transition between the location for LIKE and the location for CHOCOLATE (see also Figure 5.5). This phenomenon is typical of native signers. Accounting for it will at the very least require extending the concept of movement epenthesis within the ASL recognition framework. Source of images: [1].

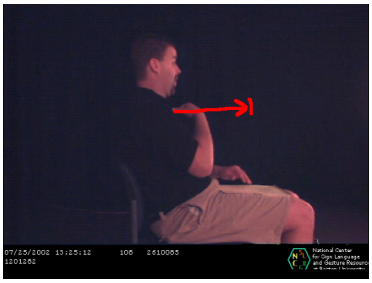


Figure 5.5: The citation form of the sign for LIKE. Source of image: [1].

is that the latter can leave a degree of freedom open via a parameter, and at recognition time simultaneously match the data signal and estimate this parameter. It may be possible somehow to parameterize the synthesis of the epenthesis and movement models, but more research on the feasibility of this idea is needed.

Physical characteristics of movements The last difference is the manner in which I perform movements, which sometimes is different from native signers. This difference is subtle, and people are hard-pressed to pinpoint it, but the consensus (from personal communication) seems to be that the rhythm and stress of my movements are different. These differences seem to affect how the data signal evolves temporally, frame by frame. In addition, other differences include the exact handshapes chosen, the transitions between signs (see Figure 5.4 on the page before for an example), and the amount of inflection in signs. These differences are very similar to the ones exhibited by nonnative speakers of spoken languages.

In the current recognition framework I do not consider the exact temporal characteristics of the data signal at all. In fact, the state-based nature of left-right HMMs allows them to absorb a certain amount of temporal variation. For this reason, I do not think that the differences in rhythm would currently have an appreciable effect on the framework. However, future work should certainly keep an eye out for it, especially research into feature vectors that highlight subtle temporal characteristics of the data signal. In contrast, the other differences are significant, and make it all the more imperative that future work uses data sets from native signers.

Chapter 6

Conclusions and Future Work

In this thesis I have described a framework for ASL recognition, which is based on phonological modeling of the language. I have presented the Movement-Hold model as the basis for the modeling approach, and presented various extensions to this model to adapt them to the requirements of the recognition framework, to cope with modeling complexity, and to make modeling the simultaneous aspects of ASL feasible.

I have presented PaHMMs, and a modification of the token passing algorithm geared specifically toward sign languages, as the basis for the recognition aspects of the framework. Within the scope of a pilot project, I have verified through experiments that this framework holds promise for further refinement and future applications. More specifically, this thesis makes the following contributions:

- Application of the Movement-Hold model — a segmental phonological model — to ASL recognition, which allows the breakdown of signs into segments of phonemes, thereby reducing sequential modeling complexity.

- Explicit modeling of movement epenthesis — the transitions between the signs —, which greatly increases the robustness of the system.

- Extension of the Movement-Hold model with a new type of segment that makes estimation of the location of signs more robust.

- Modeling of the simultaneous aspects of ASL with independent channels, thereby reducing the simultaneous modeling complexity by many orders of magnitude, and making the comprehensive modeling of the simultaneous aspects of ASL feasible in the first place.

- Development of the PaHMM recognition algorithm with multiple hypotheses, specifically geared toward the peculiarities of sign language. This recognition algorithm is the counterpart to the modeling of the simultaneous aspects of ASL in independent channels.

- Modeling of the handshape and incorporation into the PaHMM recognition framework as further validation of the independent channel modeling.

With these contributions, the recognition framework described in this thesis has the potential to form the basis for large-scale recognition, with much larger vocabularies than those that have been used in past and current research into sign language recognition. It is imperative that large-scale continuous recognition becomes feasible at the same time as speech recognition becomes mainstream, or deaf people risk losing all the gains that they have made toward accessibility of computers.

The areas for future work based on this thesis can be broadly classified into five categories. These are enhancements of the phonemic and phonetic modeling of ASL, algorithmic enhancements, advancing the field of gesture recognition, signal processing, and applying computer vision research to capture facial expressions and hand movements, so as to get rid of the cumbersome MotionStar system and the cyberglove. I now briefly discuss each category in turn.

6.1 Modeling Enhancements

In this thesis I have only scratched the surface of modeling ASL. There are many possible avenues for further research, but the most important of them all is to verify the modeling in this framework with native signers and larger vocabularies. Along this direction, one of the most fascinating and interesting problems is the question of how the locations of signs influence one another and the epenthesis movements between the signs, as discussed in the previous chapter. Similar problems occur in other channels, as well, so they are not restricted to only locations and movements; for example, coarticulation effects on the handshape are particularly interesting. Solving this problem is also essential to ensure that future large-scale recognition systems work well with native signers.

On the phonemic level it is worthwhile to investigate the effects of intra-sign epenthesis, such as the movement back from the forehead in the sign for FATHER (see also Section 2.4). It may be possible to take advantage of these effects to make the recognition of such signs more robust, and to reduce the complexity of modeling these signs. Also, the Movement-Hold model is by far not the last word on ASL phonology, so it looks worthwhile to incorporate the more recent work on phonology [54, 63, 61, 10, 11] into the framework.

The next step would be to extend the approach in this thesis to capture and recognize inflection in signs, and classifier incorporation (i.e., incorporation of a handshape to denote the type of object). Such an extension will need to address the question of how many phonemes really need to be modeled. For instance, how many different HMMs are needed for various movements and locations before it becomes possible to recognize referents in space and actions on these referents? Work in this area would go a long way toward establishing how difficult it really is to build scalable recognition systems.

Another interesting avenue of research on the phonological level is recognition of aspect; that is, whether a sign is performed fast or slowly, or in a tense manner, reduplication,

paths that are taken in reduplication, and so on. Aspect is important, because it takes on a lot of the roles of adverbs in spoken languages. The framework in this thesis does currently not make any provisions for handling aspect at all, so future work would need to find ways to add aspect to the phonemic modeling of ASL, and to recognize it with HMMs.This kind of work will probably require using feature vectors that highlight the temporal characteristics of the data signal, because normally HMMs tend to absorb temporal variations.

Leveraging the syntax of ASL, including nonmanual signals, would impose constraints on what signs can appear when in which form. The net result would be a reduction of the search space via a reduction of the number of possible branches from each HMM in the recognition network. Because recognition accuracy is closely related to the size of the search space, the recognition system could become more robust overall, especially with larger vocabularies.

Another worthwhile area of investigation is relaxing the assumption that the channels are independent from one another. It may be possible to run a statistical analysis on fully annotated sign language corpora, such as the one from [1], to determine the relative probabilities of clusters of simultaneous events. These probabilities may make it possible to bias recognition toward commonly occurring clusters, possibly increasing the overall robustness of the system.

6.2 Algorithmic Enhancements

In this thesis I have laid the modeling and algorithmic groundwork for a potentially scalable recognition system. It is to be expected, however, that further algorithmic improvements are necessary for building a truly large-scale real-world system, because the algorithmic complexity of an HMM framework is still substantial, even more so the complexity of the PaHMM extensions. As the vocabulary becomes large, so does the size of the HMM networks, especially when we would like to consider inflection. The Viterbi beam search,

mentioned in Section 3.1.2, helps exclude irrelevant parts of the networks, but by itself is not enough. Fortunately, it seems that other algorithmic tricks from speech recognition, such as fast matching to select a set of candidate HMMs, are equally applicable to sign language recognition [79]. Future work, therefore, should investigate how they interact with the PaHMM recognition framework.

Another worthwhile area for investigation is the topic of the channel weights. In this thesis I have ignored the problem of determining them. A rough guideline is to assign the weights based on a manual inspection of how much discriminative power each channel carries, but for large-scale recognition, this approach will clearly be inadequate, because it is unlikely that the optimal weights will be the same for every possible appearance of a particular inflected sign. It is equally clear that it is not possible to train the weights computationally for every possible simultaneous combination of features, because then the old problem of a combinatorial explosion would rear its ugly head again.

A possible improvement consists of using artificial neural networks to combine the channels, similar to the approach taken by the original work on PaHMMs in speech recognition [6, 30]. In this approach, the neural networks would be trained on a fixed, representative number of simultaneous combinations, with the hope that they will be able to make reasonable inferences about previously unseen combinations. In this case, two important questions need to be resolved: First, what is a good, representative selection of simultaneous events? Second, what precisely should be the input and output of the neural networks? Should they be channel weights, probabilities, or something else?

6.3 Gesture Recognition

The incorporation of the handshape into the framework opens the way into the realm of classifier signs and other signs with a high gestural component. Recognizing and interpreting the gestural components — which are much less constrained than the lexical parts

of ASL —, is a fascinating and important research problem. Any progress in this area will build a bridge toward the less constrained field of gesture and immediately benefit the research into gesture recognition, which in turn would benefit research into human-computer interaction and alternative user interfaces.

For gesture recognition, a good starting point from the work in this thesis would be research into how to distinguish signs with classifier handshapes and classifier incorporation from those signs where the handshape is simply part of the citation form. Once this problem has been solved, it will become possible to investigate recognition of the meaning of various classifier handshapes; for example, to recognize whether they represent a vehicle, a person, and so on. From there on, it may be only a small additional step to recognize and various types of gestures and to interpret what they stand for.

Another avenue worth investigating further is whether it makes sense for a gesture recognition system to break down gestures into smaller components to reduce the complexity of modeling them, similar to the approach taken in this thesis. The work by Y. Nam K. Wohn uses a similar concept, called movement primes, with some success [49, 48].

6.4 Signal Processing

As far as preparing and processing of the 3D data signal and the feature vector for optimum recognition accuracy are concerned, the current state of the art in sign language recognition is far behind speech recognition. The latter has decades of research behind it, during which a range of sophisticated methods evolved [59]. In contrast, in sign language recognition, and the related field of gesture recognition, there do not even exist any thorough and conclusive comparisons of the various features that have been used so far. In general — and this thesis is no exception — the choices of feature vectors in the field have an ad-hoc feeling to them. Among them are fast Fourier transform-based methods [39], 2D coordinate-based methods [69], vector quantization methods [42, 43], joint angles, and

other descriptions of the handshape (in this thesis). What is missing is a sound theoretical investigation of what feature vectors work best, and why.

A related area consists of how noise affects various features. Magnetic tracking systems, such as the MotionStar^{TM} system are very susceptible to the amount of metal in the immediate environment, and any change in the amount or positioning induces distortions and noise in the measurements of the 3D positions and orientations of the body parts. Likewise, inaccuracies and tracking errors from computer vision systems (see also the next section) contribute to noisy data signals. Rather than attempting to eliminate all noise, investigating what kind of features and signal processing techniques on the 3D data signal are the most robust to noise may be worthwhile.

6.5 Computer Vision

The previous three sections discussed enhancements aimed at improving the robustness and accuracy of a recognition system. Improvements in computer vision are different in that they do not necessarily contribute to a more robust or more accurate system. Nevertheless, work in this area is quite possibly more crucial than almost anything else to future mainstream acceptance of sign language and gesture recognition systems. Sign languages are inherently three-dimensional, and thus it stands to reason that 3D-based features do better than 2D-based features (see [74] for some evidence on this claim). The problem is how to get these 3D features. The MotionStar and Cyberglove are extremely intrusive and cumbersome to use, require expert technical support, and are not available at most locations where a sign language recognition system would be useful.

Ideally, the 3D data should come via computer vision from a camera, which is much less intrusive than the alternatives. Although 3D computer vision methods have made steady progress over the years, especially with deformable 3D models of the human body parts [19, 36, 47, 27, 28], they still require further improvements before they are suitable

for sign language recognition. Among the outstanding problems in the cited work are large displacements (which happen frequently with the arms and hands of a signer), and accurate tracking of very long video sequences. The contrast between different signs can be very subtle, especially for locations in the face area, so accurate tracking is imperative.

Currently, the most promising application of 3D computer vision methods to sign language recognition looks to be face tracking, for two reasons: First, displacements in the human face are much smaller than displacement of the arms and hands — compare, for example, raising an eyebrow to moving the hand from the forehead to the trunk. Second, facial expressions constitute a very important part of sign languages, especially the grammar [71], and facial movements are impractical to capture with anything but computer vision. A good starting point along these lines would be devising a 3D face model that is capable of exercising and capturing the sometimes subtle deformations that arise during facial expressions in sign languages.

Overall, 3D computer vision is an extremely difficult and often frustrating research area. For this reason, future work should also investigate how far it is possible to go with just 2D computer vision methods, and 2D features. A problem with such an approach is that movements that are closely related in 3D space (such as straight movements along different directions) can look very different, and completely unrelated, when they are projected into a 2D space. For this reason it is doubtful that the modeling approach taken in this thesis will work unchanged with 2D methods. However, it may be possible to use some aspects of ASL phonology to predict the appearance of signs and phonemes in 2D space, and to take advantage of these predictions.

Appendix A

Derivation of the Channel Combination Equation

This appendix gives the details of the proof in Section 3.3.1 that the Viterbi recognition algorithm for PaHMMs can combine the partial probabilities from the individual channels as many times as desired at any stage of the recognition process, including the whole-sign level or the phoneme level.

First, a few definitions are necessary. Recall that $O_i^{(c)}$ is the ith observation in an observation sequence $\mathbf{O}^{(c)}$, and that likewise $Q_i^{(c)}$ is the ith state in the state sequence $\mathbf{Q}^{(c)}$ that generates $\mathbf{O}^{(c)}$. Given some $\mathbf{O}^{(c)}$ for an HMM in channel u, split it up into the W adjoining subsequences. Let $\mathbf{O}_w^{(c)}$ be the wth subsequence of $\mathbf{O}^{(c)}$, and let len$(\mathbf{O}_w^{(c)})$ be the number of observations in this subsequence. Then let $O_{w,k}^{(c)}$ be the kth observation in this subsequence. Similarly, let $\mathbf{Q}_w^{(c)}$ be the state subsequence of $\mathbf{Q}^{(c)}$ that generates $\mathbf{O}_w^{(c)}$, and let $Q_{w,k}^{(c)}$ be the kth HMM state in this state subsequence. See Figure A.1 on page 139 for an example of how an observation could be split up into three subsequences, and for a visual example of what all these symbols mean.

Using Equation 3.2 on page 60, we can expand Equation 3.21 on page 74 into the

sequence of states:

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{c=1}^{C} \log P(\mathbf{Q}^{(c)}, \mathbf{O}^{(c)} | \lambda^{(c)}) \right\} = \tag{A.1}$$
$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{c=1}^{C} \left(\log \left(\pi_{Q_1^{(c)}} b_{Q_1^{(c)}}(O_1^{(c)}) \right) + \sum_{i=2}^{T} \log \left(a_{Q_{i-1}^{(c)} Q_i^{(c)}} \, b_{Q_i^{(c)}}(O_i^{(c)}) \right) \right) \right\},$$

where T is the number of frames in the observation sequence. Let

$$\log P(\mathbf{Q}_w^{(c)}, \mathbf{O}_w^{(c)} | \lambda^{(c)}) := \tag{A.2}$$
$$\log \left(\tau_w^{(c)} \, b_{Q_{w,1}^{(c)}}(O_{w,1}^{(c)}) \right) + \sum_{k=2}^{\text{len}(\mathbf{O}_w^{(c)})} \log \left(a_{Q_{w,k-1}^{(c)} Q_{w,k}^{(c)}} \, b_{Q_{w,k}^{(c)}}(O_{w,k}^{(c)}) \right),$$

where

$$\tau_w^{(c)} := \begin{cases} \pi_{Q_{1,1}^{(c)}} & \text{if } w = 1 \\ a_{Q_{w-1,\text{len}(\mathbf{O}_{w-1}^{(c)})}^{(c)} Q_{w,1}^{(c)}} & \text{otherwise.} \end{cases}$$

Equation A.2 describes the probability that the state subsequence $\mathbf{Q}_w^{(c)}$generated the observation subsequence $\mathbf{O}_w^{(c)}$ (e.g., a word or phoneme) of $\mathbf{O}^{(c)}$. $\tau_w^{(c)}$ is either the probability of the HMM starting in state $Q_{1,1}^{(c)}$ at the beginning of the first subsequence, or the transition probability from one subsequence to the next one. To visualize the meaning of the symbols used in these definitions, consider Figure A.1 on page 139. Using these definitions, and the associative and commutative laws of addition, we can condense the terms

in (A.1) into the probabilities of subsequences of $\mathbf{O}^{(c)}$:

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{c=1}^{C} \left(\log \left(\pi_{Q_1^{(c)}} b_{Q_1^{(c)}}(O_1^{(c)}) \right) + \sum_{i=2}^{T} \log \left(a_{Q_{i-1}^{(c)} Q_i^{(c)}} \, b_{Q_i^{(c)}}(O_i^{(c)}) \right) \right) \right\} =$$

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{c=1}^{C} \sum_{w=1}^{W} \log P(\mathbf{Q}_w^{(c)}, \mathbf{O}_w^{(c)} | \lambda^{(c)}) \right\} =$$

$$\max_{\mathbf{Q}^{(1)},\ldots,\mathbf{Q}^{(C)}} \left\{ \sum_{w=1}^{W} \sum_{c=1}^{C} \log P(\mathbf{Q}_w^{(c)}, \mathbf{O}_w^{(c)} | \lambda^{(c)}) \right\}. \tag{A.3}$$

The last line of Equation A.3 gives the desired derivation of Equation 3.22 on page 75.

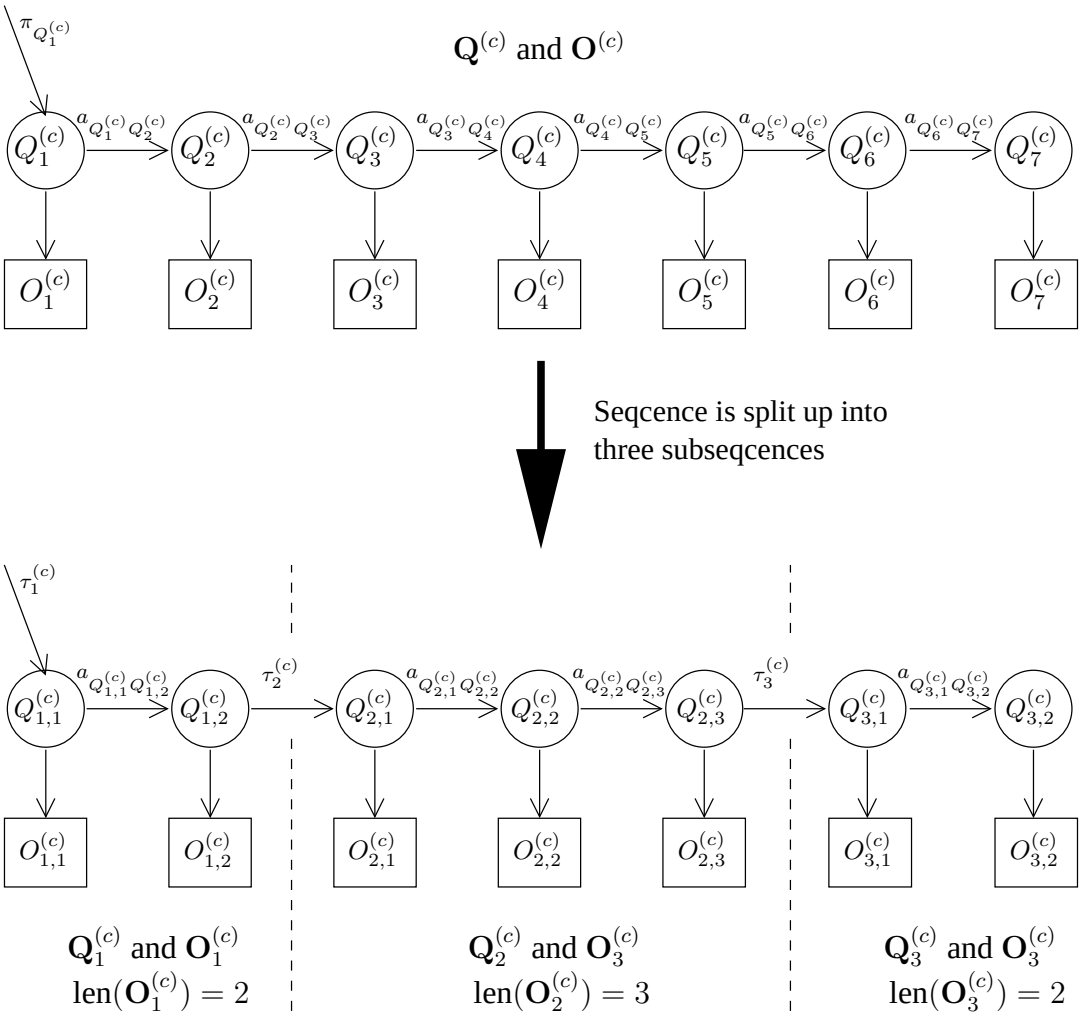


Figure A.1: Example of splitting an HMM state and observation sequence in channel c into three subsequences. See Equation A.2 on page 137, and the preceding text for a description of the notation used in this figure.

Appendix B

Phonetic transcriptions

In Table B.1 and Figure B.1 on the next page I give an overview of the different descriptions of movements and locations that I used in the 22-sign vocabulary. These are adapted from the Movement-Hold model. All body locations given in this figure can be modified with the distance from the body, and with the vertical and horizontal distance from the basic location, as follows: If a location does not touch the body, it can be prefixed with one of these distance markers: *p* (proximal), *m* (medial), *d* (distal), or *e* (extended), in order of distance to the body. If a location is centered in front of the body, the distance marker is suffixed with a 0. If the location is at the side of the chest, the distance marker is suffixed with a 1, and if the location is to the right (or left) of the shoulder, the distance marker is

Movement	Transcriptions used
straight	str_{Away}, str_{Toward}, str_{Down}, str_{Up}, str_{Left}, str_{Right}, $str_{DownAway}$, $str_{DownRightAway}$
short straight	$str_{ShortUp}$, $str_{ShortDown}$
circle in vertical plane	rnd_{VP}
wrist rotation	rot_{Away}, rot_{Toward}, rot_{Up}, rot_{Down}

Table B.1: Partial list of movements. Note that the description of the wrist movements deviates from the approach used by the Movement-Hold model.

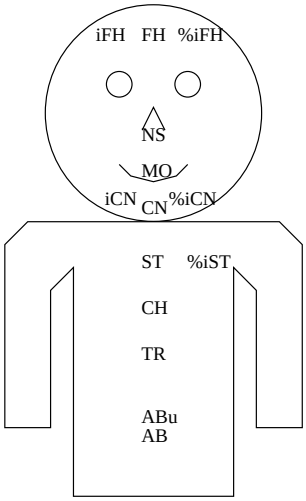


Figure B.1: Partial list of body locations used in the Movement-Hold Model.

suffixed with a 2. For example, *d-1-TR* means a location a comfortable arm's length away from the right side of the trunk (torso). Further markers describe the vertical offset to the basic location and whether the location is on the same side or opposite side of the body as the hand. These are not used in the experiments and are described in detail in [44].

The following two tables contain the phonetic transcriptions of the movement channels for the 22-sign vocabulary used throughout the thesis. The transcriptions largely follow the conventions of the Movement-Hold model described in Section 2.4. Table B.1 and Figure B.1 provide the key to the abbreviations used in these two tables. The phonemes beginning with "M" denote movements, the phonemes beginning with "H" denote holds, and the phonemes beginning with "X" denote the X segments described in Section 2.4.2. "$\varnothing$" indicates that the sign is one-handed; in these cases the weak hand does nothing.

Strong hand

Sign	Transcription
I	*X-{p-0-CH} M-{str_{Toward}} H-{CH}*
MAN	*H-{FH} M-{str_{Down}} M-{str_{Toward}} H-{CH}*
WOMAN	*H-{CN} M-{str_{Down}} M-{str_{Toward}} H-{CH}*
FATHER	*X-{p-0-FH} M-{str_{Toward}} M-{str_{Away}} M-{str_{Toward}} H-{FH}*
MOTHER	*X-{p-0-CN} M-{str_{Toward}} M-{str_{Away}} M-{str_{Toward}} H-{CN}*
INTERPRETER	*X-{m-1-CH} M-{rot_{Down}} M-{rot_{Up}} M-{rot_{Down}} X-{m-1-CH} M-{str_{Down}} H-{m-1-TR}*
TEACHER	*X-{m-1-CH} M-{rot_{Away}} M-{rot_{Toward}} M-{rot_{Away}} X-{m-1-CH} M-{str_{Down}} H-{m-1-TR}*
CHAIR	*X-{m-1-TR} M-{$str_{ShortDown}$} M-{$str_{ShortUp}$} M-{$str_{ShortDown}$} H-{m-1-TR}*
TRY	*X-{p-1-TR} M-{$str_{DownRightAway}$} H-{d-2-AB}*
INFORM	*H-{iFH} M-{$str_{DownRightAway}$} H-{d-2-TR}*
SIT	*X-{m-1-TR} M-{$str_{ShortDown}$} H-{m-1-TR}*
TEACH	*X-{m-1-CH} M-{rot_{Away}} M-{rot_{Toward}} M-{rot_{Away}} H-{m-1-CH}*
INTERPRET	*X-{m-1-CH} M-{rot_{Down}} M-{rot_{Up}} M-{rot_{Down}} H-{m-1-CH}*
GET	*X-{d-0-CH} M-{str_{Toward}} H-{p-0-CH}*
LIE	*X-{iCN} M-{str_{Left}} H-{%iCN}*
RELATE	*X-{m-1-TR} M-{str_{Left}} H-{m-0-TR}*
DONT-MIND	*H-{NS} M-{$str_{DownRightAway}$} H-{m-1-TR}*
GOOD	*H-{MO} M-{$str_{DownAway}$} H-{m-0-CH}*
GROSS	*X-{ABu} M-{rnd_{VP}} M-{rnd_{VP}} H-{ABu}*
SORRY	*X-{%iSTu} M-{rnd_{VP}} M-{rnd_{VP}} H-{%iSTu}*
STUPID	*X-{p-0-FH} M-{str_{Toward}} H-{FH}*

BEAUTIFUL	*X-{p-0-FH} M-{rnd_{VP}} H-{p-0-iFH}*

Weak hand

Sign	Transcription
I	∅
MAN	∅
WOMAN	∅
FATHER	∅
MOTHER	∅
INTERPRETER	*X-{m-1-%CH} M-{rot_{Up}} M-{rot_{Down}} M-{rot_{Up}} X-{m-1-%CH} M-{str_{Down}} H-{m-1-%TR}*
TEACHER	*X-{m-1-%CH} M-{rot_{Away}} M-{rot_{Toward}} M-{rot_{Away}} X-{m-1-%CH} M-{str_{Down}} H-{m-1-%TR}*
CHAIR	*H-{m-1-%TR}*
TRY	*X-{p-1-%TR} M-{$str_{DownLeftAway}$} H-{d-2-%AB}*
INFORM	*H-{%iNS} M-{$str_{DownLeftAway}$} H-{d-2-%TR}*
SIT	*H-{m-1-%TR}*
TEACH	*X-{m-1-%CH} M-{rot_{Away}} M-{rot_{Toward}} M-{rot_{Away}} H-{m-1-%CH}*
INTERPRET	*X-{m-1-%CH} M-{rot_{Up}} M-{rot_{Down}} M-{rot_{Up}} H-{m-1-%CH}*
GET	*X-{d-0-CH} M-{str_{Toward}} H-{p-0-CH}*
LIE	∅
RELATE	*X-{m-1-%TR} M-{str_{Right}} H-{m-0-TR}*
DONT-MIND	∅
GOOD	*H-{m-0-CH}*
GROSS	∅

SORRY	∅
STUPID	∅
BEAUTIFUL	∅

Handshape

Sign	Transcription
I	*1-Hand*
MAN	*5-Hand*
WOMAN	*5-Hand*
FATHER	*5-Hand*
MOTHER	*5-Hand*
INTERPRETER	*F-Hand F→Flat Flat-Hand*
TEACHER	*ClosedFlat-Hand ClosedFlat-Hand→Flat-Hand*
CHAIR	*H-Hand*
TRY	*T-Hand*
INFORM	*ClosedFlat-Hand→curl-Hand curlHand*
SIT	*H-Hand*
TEACH	*ClosedFlat-Hand*
INTERPRET	*F-Hand*
GET	*5-Hand 5-Hand→S-Hand S-Hand*
LIE	*ClosedFlat-Hand*
RELATE	*Open-Middle-Hand Open-Middle-Hand→ClosedMiddle-Hand ClosedMiddle-Hand*
DONT-MIND	*1-Hand*
GOOD	*Flat-Hand*
GROSS	*Curl-Hand*

SORRY	*A-Hand*
STUPID	*S-Hand*
BEAUTIFUL	*Flat-Hand Flat-Hand→ClosedFlat-Hand ClosedFlat-Hand*

Bibliography

[1] Data collected at the National Center for Sign Language and Gesture Resources, Boston University, under the supervision of C. Neidle and S. Sclaroff. Available online at `http://www.bu.edu/asllrp/ncslgr.html`, 2002.

[2] R. Battison. *Lexical borrowing in American Sign Language*, pages 19–58. Linstok Press, Silver Spring, MD, 1978. Reprinted as "Analyzing Signs" in Lucas, C. and Valli, C. Linguistics of American Sign Language, 1995.

[3] B. Bauer and K.-F. Kraiss. Towards an automatic sign language recognition system using subunits. In I. Wachsmuth and T. Sowa, editors, *Gesture and Sign Language in Human-Computer Interaction. International Gesture Workshop*, volume 2298 of *Lecture Notes in Artificial Intelligence*, pages 64–75. Springer, 2001.

[4] B. Bauer and K.-F. Kraiss. Video-based sign recognition using self-organizing subunits. In *International Conference on Pattern Recognition*, 2002.

[5] A. F. Bobick and R. C. Bolles. The representation space paradigm of concurrent evolving object descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):146–156, February 1992.

[6] H. Bourlard and S. Dupont. Subband-based speech recognition. In *Proceedings of the ICASSP*, 1997.

[7] P. Boyes-Braem. *Features of the handshape in American Sign Language*. PhD thesis, University of California, Berkeley, 1981.

[8] A. Braffort. ARGo: An architecture for sign language recognition and interpretation. In A. D. N. Edwards P. A. Harling, editor, *Progress in Gestural Interaction. Proceedings of Gesture Workshop '96*, pages 17–30, Berlin, New York, 1997. Springer.

[9] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[10] D. Brentari. Sign language phonology: ASL. In J. A. Goldsmith, editor, *The Handbook of Phonological Theory*, Blackwell Handbooks in Linguistics, pages 615–639. Blackwell, Oxford, 1995.

[11] D. Brentari. *A prosodic model of sign language phonology*. Language, speech, and communication. MIT Press, Cambridge, MA, 1998.

[12] D. Brentari and J. A. Goldsmith. Secondary licensing and the nondominant hand in ASL phonology. In Coulter [17], pages 19–41.

[13] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland. Invariant features for 3-D gesture recognition. In *2nd International Workshop on Face and Gesture Recognition*, Killington, VT, 1996.

[14] L. W. Campbell and A. F. Bobick. Recognition of human body motion using phase space constraints. In *Proceedings of ICCV*, pages 624–630, 1995.

[15] J. Cooley. How the FFT gained acceptance. *IEEE Speech Processing Magazine*, pages 10–13, 1992.

[16] D. P. Corina. To branch or not to branch: Underspecification in ASL handshape contours. In Coulter [17], pages 63–95.

[17] G. R. Coulter, editor. *Current Issues in ASL Phonology*, volume 3 of *Phonetics and Phonology*. Academic Press, Inc., San Diego, CA, 1993.

[18] Y. Cui, D. L. Swets, and J. J. Weng. Learning-based hand sign recognition using SHOSLIF-M. In *Proceedings of ICCV*, 1995.

[19] D. de Carlo and D. Metaxas. Optical flow constraints on deformable models with applications to face tracking. *IJCV*, 38(2):99–127, July 2000.

[20] Douglas DeCarlo, Dimitris Metaxas, and Matthew Stone. An anthropometric face model using variational techniques. In *Proc. of SIGGRAPH*, pages 67–74, 1998.

[21] L. Deng, M. Lennig, P. Seitz, and P. Mermelstein. Large vocabulary word recognition using context-dependent allophonic hidden Markov models. *Computer Speech and Language*, 4(4):345–357, 1990.

[22] B. Dorner. Chasing the colour glove: Visual hand tracking. Master's thesis, Simon Fraser University, 1994.

[23] R. Erenshteyn and P. Laskov. A multi-stage approach to fingerspelling and gesture recognition. In *Proceedings of the Workshop on the Integration of Gesture in Language and Speech*, Wilmington, DE, USA, 1996.

[24] G. Fang, W. Gao, X. Chen, C. Wang, and J. Ma. Signer-independent continuous sign language recognition based on SRN/HMM. In I. Wachsmuth and T. Sowa, editors, *Gesture and Sign Language in Human-Computer Interaction. International Gesture Workshop*, volume 2298 of *Lecture Notes in Artificial Intelligence*, pages 76–85. Springer, 2001.

[25] Z. Ghahramani and M. I. Jordan. Factorial Hidden Markov Models. *Machine Learning*, 29:245–275, 1997.

[26] S. Gibet, J. Richardson, T. Lebourque, and A. Braffort. Corpus of 3D natural movements and sign language primitives of movement. In I. Wachsmuth and M. Fröhlich, editors, *Gesture and Sign Language in Human-Computer Interaction. Proceedings of Gesture Workshop '97*, Berlin, New York, 1998. Springer.

[27] S. Goldenstein, C. Vogler, and D. Metaxas. Affine arithmetic based estimation of cue distributions in deformable model tracking. In *Proceedings of CVPR*, pages 1098–1105, 2001.

[28] S. Goldenstein, C. Vogler, and D. Metaxas. Directed acyclic graph representation of deformable models. In *IEEE workshop on motion and video computing (WMVC)*, 2002.

[29] K. Grobel and M. Assam. Isolated sign language recognition using hidden Markov models. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 162–167, Orlando, FL, 1997.

[30] H. Hermansky, S. Tibrewala, and M. Pavel. Towards ASR on partically corrupted speech. In *Proceedings of the ICSLP*, pages 462–465, 1996.

[31] H. Hienz, B. Bauer, and K.-F. Kreiss. HMM-based continuous sign language recognition using stochastic grammars. In A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, editors, *Gesture-Based Communication in Human-Computer Interaction*, volume 1739 of *Lecture Notes in Artificial Intelligence*, pages 185–196. Springer, 1999.

[32] H. Hienz, K.-F. Kraiss, and B. Bauer. Continuous sign language recognition using hidden Markov models. In Y. Tang, editor, *ICMI'99*, pages IV10–IV15, Hong Kong, 1999.

[33] M. W. Kadous. Machine recognition of Auslan signs using PowerGloves: Towards large-lexicon recognition of sign language. In *Proceedings of the Workshop on the Integration of Gesture in Language and Speech*, pages 165–174, Wilmington, DE, USA, 1996.

[34] M. W. Kadous. *Temporal classification: Extending the classification paradigm to multivariate time series*. PhD thesis, University of New South Wales, 2002.

[35] I. Kakadiaris and D. Metaxas. 3D human body model acquisition from multiple views. In *Proceedings of the ICCV*, pages 618–623, 1995.

[36] I. Kakadiaris and D. Metaxas. Model based estimation of 3D human motion with occlusion based on active multi-viewpoint selection. In *Proceedings of the CVPR*, pages 81–87, 1996.

[37] I. Kakadiaris, D. Metaxas, and R. Bajcsy. Active part-decomposition, shape and motion estimation of articulated objects: A physics-based approach. In *Proceedings of the CVPR*, pages 980–984, 1994.

[38] E. S. Klima and U. Bellugi. *The Signs of Language*. Harvard University Press, Cambridge, MA, 1979.

[39] C. Lee and Y. Xu. Online, interactive learning of gestures for human/robot interfaces. In *IEEE International Conference on Robotics and Automation*, 1996.

[40] C.-H. Lee, J.-L. Gauvain, R. Pieraccini, and L. R. Rabiner. Subword-based large-vocabulary speech recognition. *AT&T Technical Journal*, 72(5):25–35, 1993.

[41] K.-F. Lee, H. Hon, and R. Reddy. An overview of the SPHINX speech recognition system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(1):35–45, 1990.

[42] R.-H. Liang. *Continuous Gesture Recognition System for Taiwanese Sign Language*. PhD thesis, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, R.O.C, June 1997.

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

[44] S. K. Liddell and R. E. Johnson. American Sign Language: The phonological base. *Sign Language Studies*, 64:195–277, 1989.

[45] C. Lucas, editor. *Sign Language Research: Theoretical Issues*. Gallaudet University Press, Washington DC, 1990.

[46] D. McNeill. *Hand and mind: what gestures reveal about thought*. University of Chicago Press, Chicago, 1992.

[47] D. Metaxas. *Physics-based deformable models: applications to computer vision, graphics and medical imaging*. Kluwer Academic Publishers, 1996.

[48] Y. Nam and K. Y. Wohn. Recognition and modeling of hand gestures using colored petri nets. IEEE transactions on Systems, Man and Cybernetics (A), 1999.

[49] Y. Nam and K. Y. Wohn. Recognition of space-time hand-gestures using hidden Markov model. ACM Symposium on Virtual Reality Software and Technology, 1996.

[50] C. Neidle, J. Kegl, D. MacLaughlin, B. Bahan, and R. G. Lee. *The syntax of American Sign Language*. Language, Speech, and Communication. MIT Press, Cambridge, Massachusetts, 2000.

[51] H. Ney, R. Haeb-Umbach, B.-H. Tran, and M. Oerder. Improvements in beam search for 10000-word continuous speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 9–12, 1992.

[52] C. Padden. *Interaction of morphology and syntax in American Sign Language*. PhD thesis, University of California, San Diego, 1983.

[53] V. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

[54] D. Perlmutter. On the segmental representation of transitional and bidirectional movements in ASL phonology. In S. P. Fischer, editor, *Theoretical Issues in Sign Language Research*, volume 1, pages 67–80. University of Chicago Press, 1990.

[55] D. Perlmutter. Sonority and syllable structure in American Sign Language. *Linguistic Inquiry*, 23(3):407–442, 1992.

[56] D. M. Perlmutter. Sonority and syllable structure in American Sign Language. In Coulter [17], pages 227–261.

[57] H. Poizner, U. Bellugi, and V. Lutes-Driscoll. Perception of American Sign Language in dynamic point-light-displays. *Journal of Experimental Psychology: Human Perception and Performance*, 7(2):430–440, 1981.

[58] L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[59] L. R. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., Englewoods Cliffs, NJ, 1993.

[60] C. Rowden. Analysis. In C. Rowden, editor, *Speech Processing*, pages 35–96. McGraw-Hill, New York, NY, 1992.

[61] W. Sandler. *Phonological Representation of the Sign: Linearity and Nonlinearity in American Sign Language*. Number 32 in Publications in Language Sciences. Foris Publications, Dordrecht, 1989.

[62] W. Sandler. Linearization of phonological tiers in ASL. In Coulter [17], pages 103–129.

[63] W. Sandler. Representing Handshapes. In W. H. Edmondson and R. Wilbur, editors, *International Review of Sign Linguistics*, volume 1, chapter 5, pages 115–158. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, 1996.

[64] J. Siskind and Q. Morris. A maximum-likelihood approach to visual event classification. In *Proceedings of the Fourth European Conference on Computer Vision*. Cambridge, UK: Springer-Verlag, 1996.

[65] T. Sowa and I. Wachsmuth. Interpretation of shape-related iconic gestures in virtual environments. In I. Wachsmuth and T. Sowa, editors, *Gesture and Sign Language in Human-Computer Interaction*, volume 2298 of *Lecture Notes in Artificial Intelligence*. Springer, 2001.

[66] T. Starner. Visual recognition of American Sign Language using Hidden Markov Models. Master's thesis, MIT Media Laboratory, February 1995.

[67] T. Starner and A. Pentland. Visual recognition of American Sign Language using Hidden Markov Models. In *International Workshop on Automatic Face and Gesture Recognition*, pages 189–194, Zürich, Switzerland, 1995.

[68] T. Starner and A. Pentland. Real-time American Sign Language recognition from video using Hidden Markov Models. Technical Report 375, MIT Media Laboratory, 1996.

[69] T. Starner, J. Weaver, 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(12):1371–1375, 1998.

[70] 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, Silver Spring, MD, 1960. Revised 1978.

[71] C. Valli and C. Lucas. *Linguistics of American Sign Language: An Introduction*. Gallaudet University Press, Washington DC, 1995.

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

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

[74] C. Vogler and D. Metaxas. ASL recognition based on a coupling between HMMs and 3D motion analysis. Technical Report MS-CIS-98-21, MS-CIS-98-21, Department of Computer and Information Science, University of Pennsylvania, 1998.

[75] C. Vogler and D. Metaxas. Parallel hidden Markov models for American Sign Language recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 116–122, Kerkyra, Greece, 1999.

[76] C. Vogler and D. Metaxas. Toward scalability in ASL recognition: Breaking down signs into phonemes. In A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, editors, *Gesture-Based Communication in Human-Computer Interaction*, volume 1739 of *Lecture Notes in Artificial Intelligence*, pages 211–224. Springer, 1999.

[77] C. Vogler and D. Metaxas. A framework for recognizing the simultaneous aspects of American Sign Language. *Computer Vision and Image Understanding*, (81):358–384, 2001.

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

[79] C. Wang, W. Gao, and J. Ma. A real-time large vocabulary recognition system for Chinese Sign Language. In I. Wachsmuth and T. Sowa, editors, *Lecture Notes in Artificial Intelligence*, volume 2298, pages 86–95. Springer, 2002.

[80] A. D. Wilson and A. F. Bobick. Nonlinear PHMMs for the interpretation of parameterized gesture. In *Proceeedings of the CVPR*, pages 879–884, 1998.

[81] Y. Wu and T. Huang. Vision-based gesture recognition: A review. In A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, editors, *Gesture-Based Communication in Human-Computer Interaction*, volume 1739 of *Lecture Notes in Artificial Intelligence*, pages 103–115. Springer, 1999.

[82] S. Young, J. Jansen, J. Odell, D. Ollason, and P. Woodland. *The HTK book (for HTK 2.0)*. Cambridge University, 1995.

[83] S. Young, N. Russell, and J. Thornton. Token passing: a conceptual model for connected speech recognition systems. Technical report, F_INFENG/TR38 Cambridge University, 1989.

[84] L. Zhao, M. Costa, C. Vogler, W. Schuler, and N. Badler. Modifying movement manner using adverbs. In *Fourth International Workshop on Autonomous Agents: Communicative Agents in Intelligent Virtual Environments*, Barcelona, Spain, June 2000.

[85] L. Zhao, K. Kipper, W. Schuler, C. Vogler, N. Badler, and M. Palmer. A machine translation system from English to American Sign Language. In *Proceedings of the Association for Machine Translation in the Americas. Published in Lecture Notes in AI series of Springer-Verlag*, pages 54–67, 2000.