Computer Vision and Image Understanding **81,** 358–384 (2001)
doi:10.1006/cviu.2000.0895, available online at http://www.idealibrary.com on IDEAL®

# A Framework for Recognizing the Simultaneous Aspects of American Sign Language

Christian Vogler and Dimitris Metaxas

*Vision, Analysis, and Simulation Technologies Laboratory, Department of Computer and Information Science, University of Pennsylvania, 200 S. 33rd Street, Philadelphia, Pennsylvania 19104-6389*
E-mail: cvogler@gradient.cis.upenn.edu, dnm@central.cis.upenn.edu

Received December 16, 1999; accepted September 27, 2000

The major challenge that faces American Sign Language (ASL) recognition now is developing methods that will scale well with increasing vocabulary size. Unlike in spoken languages, phonemes can occur simultaneously in ASL. The number of possible combinations of phonemes is approximately $1.5 \times 10^9$, which cannot be tackled by conventional hidden Markov model-based methods. Gesture recognition, which is less constrained than ASL recognition, suffers from the same problem. In this paper we present a novel framework to ASL recognition that aspires to being a solution to the scalability problems. It is based on breaking down the signs into their phonemes and modeling them with parallel hidden Markov models. These model the simultaneous aspects of ASL independently. Thus, they can be trained independently, and do not require consideration of the different combinations at training time. We show in experiments with a 22-sign-vocabulary how to apply this framework in practice. We also show that parallel hidden Markov models outperform conventional hidden Markov models. 

*Key Words:* sign language recognition; gesture recognition; human motion modeling; hidden Markov models.

## 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 10 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 [20]. A working speech-and-gesture interface is likely to entail a major paradigm shift away from point-and-click user interfaces toward a natural language dialogue-and-spoken command-based interface.



1077-3142/01 $35.00

Sign language recognition enters the picture in three ways. First, such a paradigm shift would leave those deaf people who depend on sign language as their primary mode of communication behind. There is a sense of urgency, because of the improvements in speech recognition. Unless we get sign language recognition to the same level of performance as speech recognition, accessibility of computers will become a major issue for the deaf.

Second, gesture recognition in itself is a difficult problem, because gestures are unconstrained, but gestures take place in the same visual medium as sign languages, and the latter possess a high degree of structure. This structure makes it easier to solve problems in sign language recognition first, before applying them to gesture recognition.

Third, a working sign language recognition system would make deaf–hearing interaction easier. Particularly public functions, such as the courtroom, conventions, and meetings, would become much more accessible to the deaf.

The main challenge in 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 highly inflected, which means that each sign can appear in many different forms, depending on subject, object, and numeric agreement. Thus, it is futile to model each form separately—there are simply too many of them. Instead, sign language recognizers must capture the commonalities among all signs. In speech recognition this problem is solved by modeling the language in terms of its constituent phonemes. In principle, the same solution applies to sign language recognition.

However, modeling the phonology of sign languages is much more difficult than modeling the phonology of spoken languages. In speech, the phonemes appear sequentially. In signed languages the phonemes can appear both in sequences 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 the possible different combinations of phonemes statically—for example, by training a hidden Markov model (HMM) for each combination—would be futile for anything but the smallest vocabularies.

In this paper we present a novel framework for modeling and recognizing American Sign Language (ASL). It consists of breaking the simultaneous aspects of ASL down into its constituent phonemes and modeling them with parallel hidden Markov models (PaHMMs). This is a new extension to hidden Markov models (HMMs).

In previous work, researchers have proposed other extensions to HMMs to model the interaction of several interacting processes in parallel, such as factorial hidden Markov models (FHMMs) [7] or coupled hidden Markov models (CHMMs) [3]. These extensions require modeling the interactions of the processes during the training phase, and thus require training examples of every conceivable combination of actions that can occur in parallel. Thus, it is doubtful that FHMMs and CHMMs will scale well in ASL recognition.

PaHMMs avoid these scalability problems by assuming that the processes are independent of one another (“independent channels”). As a consequence, the channels can be trained completely independently, before they are combined at recognition time. Thus, it is not necessary to provide training examples of all possible combinations of phonemes. There is linguistic evidence that ASL can be modeled at least partially as independent channels [18]. Hence, PaHMMs stand a much better chance than FHMMs and CHMMs of being scalable. Because gesture recognition is even less constrained than ASL recognition, PaHMMs are highly significant to gesture recognition research, as well.

We use 3D data as the input to our recognition framework. These data can be collected either with 3D computer vision methods, such as physics-based modeling [14–16], or with a magnetic tracking system, such as the Ascension Technologies MotionStar system. We use these 3D data to recognize continuous sentences over a 22-sign vocabulary, where the individual signs are modeled in terms of their constituent phonemes.

The remainder of this paper is organized as follows: First, we give an overview on related work. Then we describe the fundamentals of modeling ASL, as they apply to our recognition framework. We show how and why breaking down signs into their constituent phonemes is beneficial. We then describe the necessary extensions to existing phonological models of ASL to adapt them to ASL recognition. We also develop the phonological basis for modeling ASL in terms of independent channels.

Then we give a brief introduction to HMMs and describe the token-passing algorithm as the main recognition method. We briefly discuss FHMMs and CHMMs and why they cause problems for large-scale ASL recognition. We then develop the mathematics and algorithms behind PaHMMs, in order to overcome the scalability problems of FHMMs and CHMMs. We briefly discuss implementation issues that arise during the adaptation of HMMs for ASL recognition and provide experimental results that compare PaHMMs with conventional HMMs.

## 2. RELATED WORK

In the discussion of related work, we focus on previous work in sign language recognition. For coverage of gesture recognition, the survey in [24] is an excellent starting point. Other, more recent work is reviewed in [35].

Much previous work has focused on isolated sign language recognition with clear pauses after each sign, although the research focus is slowly shifting to continuous recognition. These pauses make it a much easier problem than continuous recognition without pauses between the individual signs, because explicit segmentation of a continuous input stream into the individual signs is very difficult. For this reason, and because of coarticulation effects, work on isolated recognition often does not generalize easily to continuous recognition.

Erensthteyn and colleagues used neural networks to recognize fingerspelling [6]. Waldron and Kim also used neural networks, but they attempted to recognize a small set of isolated signs [34] instead of fingerspelling. They used Stokoe's transcription system [29] to separate the handshape, orientation, and movement aspects of the signs.

Kadous used Power Gloves to recognize a set of 95 isolated Auslan signs with 80% accuracy, with an emphasis on computationally inexpensive methods [13]. Grobel and 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 [9].

Braffort described ARGo, an architecture for recognizing French Sign Language. It attempted to integrate the normally disparate fields of sign language recognition and understanding [2]. Toward this goal, Gibet and colleagues also described a corpus of 3D gestural and sign language movement primitives [8]. This work focused on the syntactic and semantic aspects of sign languages, rather than phonology.

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. It thus bypasses the difficult problem of segmentation entirely.

Starner and 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 [27]. They assumed that the smallest unit in sign language is the whole sign. This assumption leads to scalability problems, as vocabularies become larger. In [28] they applied their methods to wearable computing by mounting a camera on a hat.

Hienz and colleagues used HMMs to recognize a corpus of German Sign Language [12]. Their work was an extension of the work by Grobel and Assam in [9]; 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 our results in [31].

Nam and Wohn [23, 22] 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 superficial similarities to the phoneme-based approach in [33] and in this paper.

Liang and Ouhyoung used HMMs for continuous recognition of Taiwanese Sign Language with a vocabulary between 71 and 250 signs [17]. They worked with Stokoe's model [29] to detect the handshape, position, orientation, and movement 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 at a level higher than that of the HMMs.

We used HMMs and 3D computer vision methods to model phonological aspects of ASL [31, 33] with an unconstrained sentence structure. We used the Movement–Hold phonological model by Liddell and Johnson [18] extensively, so as to develop a scalable framework. In [32] we extended the conventional HMM framework to capture the parallel aspects of ASL, which ordinarily would make the recognition task too complex.

## 3. MODELING ASL

In this section we first give an overview on the relevant aspects of ASL linguistics, particularly ASL phonology. We describe the movement–hold phonological model in detail, as it forms the basis of our work. We then discuss its shortcomings and extend this model to make it suitable for ASL recognition.

ASL is the primary mode of communication for many deaf people in the USA. It is a highly inflected language; that is, many signs can be modified to indicate subject, object, and numeric agreement. They can also be modified to indicate manner (fast, slow, etc.), repetition, and duration [30, 29, 19]. Like all other languages, ASL has structure, which sets it clearly apart from gesturing. It allows us to test ideas in a constrained framework first, before attempting to generalize the results to gesture recognition problems.

In particular, managing the complexity of large data sets in gesture recognition is an area where ASL recognition work can yield valuable insights. As we shall explain in Section 3.3.2 and Section 4.2, managing complexity is already difficult in the relatively constrained field of ASL recognition, because signs can appear in many different forms. Gestures are much less constrained than ASL, so this problem will only be exacerbated. It is, therefore, important to develop methods that make the complexity of ASL and gesture recognition manageable.

**FIG. 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.

The large body of research on ASL linguistics, particularly ASL phonology, helps us to develop exactly these methods. Although there is no phonology of gestures, the ideas behind ASL phonology—that signs can be broken down into smaller parts—nevertheless applies to gesture recognition research [23, 22].

At this point we need to provide two essential definitions. The *strong hand* is 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 peoples, the strong hand is typically the person's right hand, and the weak hand is the person's left hand.

We now give an introduction to ASL phonology and discuss how it can be applied to ASL recognition. This overview is by no means exhaustive. For more information on ASL phonology, see for example [26, 4, 5, 18].

### *3.1. ASL Phonology*

A *phoneme* is defined to be the smallest contrastive unit in a language [30]; that is, a unit that distinguishes a word from another. In English, the sounds */c/*, */a/*, and */t/* (and their equivalents in regional dialects) are examples of phonemes. In ASL, 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 (Fig. 1) are examples of phonemes.

Modeling phonology helps to keep both speech and ASL recognition tractable [25, 33], because there is *only a small, limited number* of phonemes, as opposed to the unlimited number of words and signs that can be built with them. In English, there are approximately 40 distinct phonemes, whereas in ASL, there are approximately 150–200 distinct phonemes.[1]

For this reason, using phonemes is essential for building large-scale systems. It is practical to provide sufficient training data for a small set of phoneme models that can be used to construct every conceivable word in the language. On the other hand, it is not practical to provide sufficient training data for a very large set of word or whole-sign models, so as to achieve the same vocabulary size as with the set of phonemes.

There is still considerable controversy whether such units in ASL can justifiably be called "phonemes." Some linguists prefer to call them "cheremes" [29], because the roots

[1] This number applies to the Movement–Hold phonological model [18] described in Section 3.2. The numbers for other models vary slightly.

of "phoneme" can be traced back to the concept of speaking. Other linguists have argued that the subunits in ASL, such as the movements and locations described in the previous paragraph, do not function in the same way as phonemes in spoken languages. One of the reasons they give is that many of these subunits are redundant [5].

In this paper we do not attempt to argue for or against using the term "phoneme" for ASL. Whenever we use the term "phoneme," we mean the smallest identifiable subunits in ASL. In general, we choose to follow the terminology of spoken language linguistics, because many concepts have direct equivalents in ASL linguistics. In addition, the subunits in ASL function in the same way as phonemes in our recognition framework.

### *3.2. The Movement–Hold Model*

We are primarily interested in modeling signs as sequences of phonemes, because hidden Markov models are sequential by nature. Phonological models that emphasize sequential contrast are called *segmental models*. Such models split signs into multiple segments, during which the parameters of a sign can vary (see Fig. 2 for an example). Thus, they emphasize sequential contrast over simultaneous contrast.

Liddell and Johnson's Movement–Hold model [18] is one of the oldest segmental models. It consists of two major classes of segments, which are *movements* and *holds*. Movements are those segments, during which some aspect of the signer's configuration changes, such as a change in handshape, or a hand movement from one location to another. Holds, in contrast, are those segments, during which the hands remain translationally stationary.

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,"

**FIG. 2.** The signs for "interpreter" (top) and "teacher" (bottom) illustrate sequential contrast. They differ only in the first part of their movement sequence (left). The movements that make up this sign are examples of phonemes.

**FIG. 3.** *HMH* pattern. The sign for "good" consists of a hold at the chin (left), followed by a movement down and away from the body (left), followed by a hold contacting the weak hand (right).

Fig. 3), *MH* (a movement followed by a hold, such as "sit," Fig. 4), and *MMMH* (three movements followed by a hold, such as "father," Fig. 5). Attached to each segment is a *bundle of articulatory features* that describe the hand configuration, orientation, location, and nontranslational hand movements (e.g., wrist rotation, wriggling of fingers). In addition, movement segments have features that describe the type of movement (straight, round, sharply angled), as well as the plane and intensity of movement. See Fig. 6 for a schematic example.

In this paper we use only the aspects of the Movement–Hold model that describe hand movements and locations, because these are the easiest to capture with our 3D tracking system. Nevertheless, in the following sections we describe how a recognition framework could use all aspects of the Movement–Hold model, even though we currently do not take advantage of them. Future work should also incorporate the hand configuration parameters into the framework, but doing so requires a solution to the difficult problem of tracking fingers accurately. Table 1 and Fig. 7 give an overview of the transcriptions for the movements and locations that we use in our framework.

Furthermore, the locations can be modified with the distance from the body, and with the vertical and horizontal distance from the basic location. 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

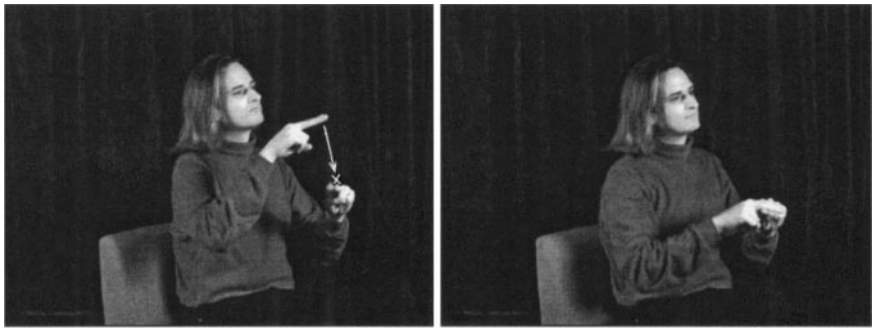

**FIG. 4.** *MH* pattern. The sign for "sit" consists of a downward movement onto the weak hand (left), followed by a hold contacting the weak hand (right).

**TABLE 1**
**Partial List of Movements**

| 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 | $rot_{Away}$, $rot_{Toward}$, $rot_{Up}$, $rot_{Down}$ |

*Note.* The description of the movements deviates from the approach used by the Movement–Hold model.

**FIG. 5.** *MMMH* pattern. The sign for "father" consists of three movements: tap on forehead, away from forehead, tap on forehead (left), followed by a hold contacting the forehead (right).

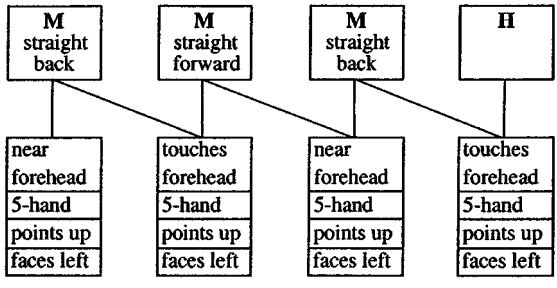


**FIG. 6.** Schematic description of the sign for "Father" in the Movement–Hold model. It consists of three movements, followed by a hold (compare with Fig. 5).

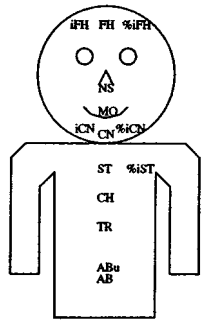


**FIG. 7.** Partial list of body locations used in the Movement–Hold model.

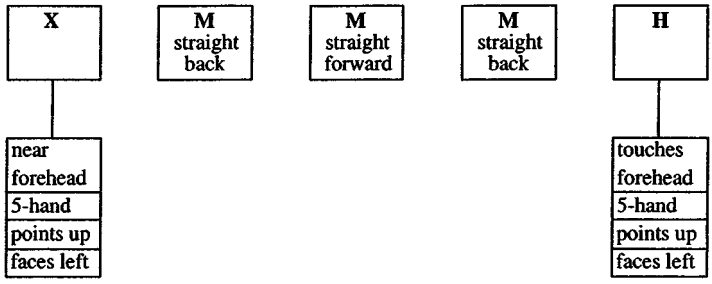


**FIG. 8.** Description of the sign for "father" with the help of X segments. Articulatory features are now attached only to holds and X segments. Compare with Fig. 6.

shoulder, the distance marker is suffixed with a 2. For example, *d-1-TR* means a location of a comfortable arm's length away from the right side of the trunk (torso). Further markers, such as "%" and "i" describe the vertical offset relative the basic location, and whether the location is on the same side or opposite side of the body as the hand. These are described in detail in [18].

The Movement–Hold model does not address nonmanual features, such as facial expressions. Because facial expressions constitute a large part of the grammar of signed languages, future work needs to address this shortcoming. Yet, the model has demonstrated convincingly that sequential aspects of ASL are important. Other recent phonological models all differ in details from the Movement–Hold model, but they all emphasize sequential aspects of ASL [26, 5, 4].

### *3.3. Extensions to the Movement–Hold Model*

There are some problems with the Movement–Hold model that prevent it from being applied to ASL recognition directly. We now discuss solutions to these problems.

*3.3.1. Articulatory features attached to movements.* One problem is that in the original description of the Movement–Hold model the articulatory features can be attached to both movements and holds. From a linguistic point of view attaching the articulatory features to movement segments is implausible, because these segments describe how the configuration is *changing*. The articulatory features, however, describe *static* aspects of the configuration.

From a technical point of view, there is no good way to attach the articulatory features to movement segments, because we would like 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 there is no hand movement. In movement segments we are interested in the type of movement and do not care about location. How do we model the location at the beginning of a sign that starts with a movement, then?

From these two points of view it becomes clear that the Movement–Hold model must be modified before it can be applied to recognition. To this end, we add a new type of segment called "X."[2] They are conceptually very similar to holds. The only difference is that, unlike holds, the hand need not be translationally stationary for any amount of time. The sole purpose of these segments is to provide an anchor for the articulatory features. Figure 8 shows how the X segments affect the sign for "father."

[2] We came up with this idea independently of Liddell and Johnson. Yet, the role of our 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.

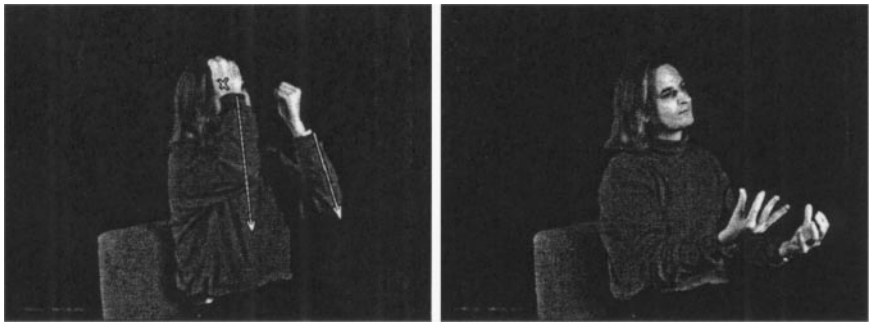

**FIG. 9.** The sign for "inform" demonstrates how several features in ASL change simultaneously. Both hands move, starting at different body locations. The handshape is symmetrical, but changes from a closed fist to a half-open hand during the sign.

*3.3.2. Sequential versus simultaneous aspects of ASL.* Adding X segments, as described in the previous section, is sufficient for recognizing ASL using only the strong hand [31], but 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 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 Fig. 9.

If we consider both hands in the Movement–Hold model, and assuming that there are 30 basic handshapes, 8 hand orientations, 8 wrist orientations, and 20 major body locations for each hand [29, 18], the number of different combinations of X and hold segments with attached articulatory features is $(30 \times 8 \times 8 \times 20)^2 \approx 1.5 \times 10^9$. Even if we take into account that the weak hand is constrained either to mirror the strong hand, or to use one of six basic handshapes [30], the number of combinations would still be approximately $2.9 \times 10^8$.

Modeling all such combinations a priori is not practical, because it would be impossible to obtain that many training examples from a signer. Foregoing ASL phonology and looking at ASL from the whole-sign level does not help either. Even though the cataloged vocabulary of ASL consists of only approximately 6000 signs, many signs can be highly inflected. Verbs like "give" can be modified in the starting location, ending location, handshape, and type of movement, so as to indicate subject, recipient, object, and manner of action. Thus, the number of possible cases to consider on the whole-sign level would be several orders of magnitude larger than 6000.

Therefore, in order to model ASL in a recognition framework, we need to make a major modification to the Movement–Hold model. Instead of attaching bundles of articulatory features to the X and hold segments, we break up the features into *channels* that can be used independently from one another. The most important channel consists of movements and hold segments that describe the type of movement and the body locations. Other channels consist of the handshape, the hand orientation, and the wrist orientation. Yet other channels describe the actions of the weak hand in the same way as for the strong hand.

Figure 10 shows how the sign for "father" is represented with this modification. Note that this figure only shows the channels for the strong hand, because "father" is a one-handed

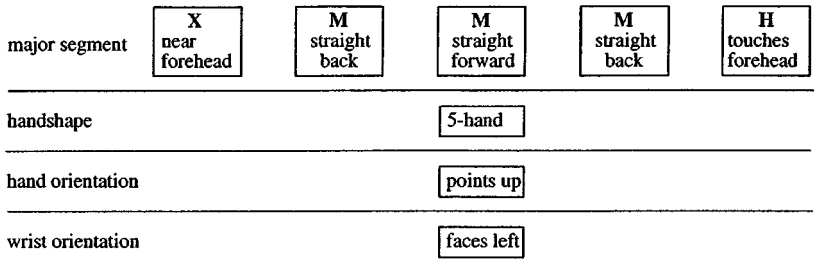


**FIG. 10.** The sign for "father," where the different features are modeled in separate channels. The handshape and orientations stay the same during the entire sign, so only one phoneme appears in each of these channels. Compare with Fig. 8.

sign. For two-handed signs, we model the channels for the strong and the weak hands independently from one another, as well. For example, the movements and holds of the strong and the weak hands are in two channels independent of each other.

By splitting the feature bundles into independent channels, we immediately gain 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. The independence of the channels guarantees that we can model them separately and put together new phoneme combinations during the recognition process on the fly.

### *3.4. Phonological Processes*

An application of ASL phonology to ASL recognition cannot be complete without taking phonological processes into account. A phonological process changes the appearance of an utterance through well-defined rules in phonology, but does not change the meaning of the utterance. Because the meaning is unchanged, it is best for a recognizer to handle the changes in appearance at the phonological level.

The most basic, and at the same time also most important, phonological process is called *movement epenthesis* [18]. It consists of the insertion of extra movements between two adjacent signs, and it is caused by the physical characteristics of sign languages. For example, in the sequence "father read," the sign for "father" is performed at the forehead, and the sign for "read" is performed in front of the trunk. Thus, an extra movement from the forehead to the trunk is inserted that does not exist in either of the two signs' lexical forms (Fig. 11).

**FIG. 11.** Movement epenthesis. The arrow in the middle picture indicates an extra movement between the signs for "father" and "read" that is not present in their lexical forms.

Movement epenthesis poses a problem for ASL recognizers, because the extra movement depends on which two signs appear in sequence. Within our extended Movement–Hold model, we handle such movements just like regular movements within a sign. We do not yet model any other phonological processes in ASL, such as hold deletion and metathesis (which allows for swapping of the order of segments in certain circumstances).

## 4. HIDDEN MARKOV MODELS

One of the main challenges in ASL recognition is to capture the variations in the signing of even a single human. HMMs are a type of statistical model embedded in a Bayesian framework and thus well suited for capturing these variations. In addition, their state-based nature enables them to describe how a signal changes over time.

We now briefly describe the properties of HMMs relevant to ASL recognition. We then describe possible extensions to the HMM framework and conclude with a description of parallel HMMs, our approach toward solving the problems associated with regular HMMs.

An HMM $\lambda$ consists of a set of $N$ states $S_1, S_2, \ldots, S_N$. At regularly spaced discrete time intervals, the system transitions from state $S_i$ to state $S_j$ with probability $a_{ij}$. The probability of the system initially starting in state $S_i$ is $\pi_i$. Each state $S_i$ generates output $O \in \Omega$, which is distributed according to a probability distribution function $b_i(O) = P\{\text{Output is } O \mid \text{System is in } S_i\}$. In most recognition applications $b_i(O)$ is a mixture of Gaussian densities.

### *4.1. The HMM Recognition Algorithm*

We now describe the main algorithm used for continuous recognition. For a discussion of how to estimate (i.e., train) the parameters of an HMM and how to compute the probability that an HMM generated an output sequence, see [25].

In many continuous recognition applications, the HMMs corresponding to individual signs are chained together into a network of HMMs. Then the recognition problem is reduced to finding the most likely state sequence through the network. That is, we would like to find a state sequence $Q = Q_1, \ldots, Q_T$ over an output sequence $O = O_1, \ldots, O_T$ of $T$ frames, such that $P(Q, O \mid \lambda)$ is maximized. Using

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

and by induction

$$\delta_{t+1}(i) = b_i(O_{t+1}) \cdot \max_{1 \le j \le N} \{\delta_t(j) a_{ji}\}, \tag{2}$$

$$P(Q, O \mid \lambda) = \max_{1 \le i \le N} \{\delta_T(i)\}, \tag{3}$$

the Viterbi algorithm computes this state sequence in $O(N^2 T)$ time, where $N$ is the number of states in the HMM network. Note that the Viterbi algorithm implicitly segments the observation into parts as it computes the path through the network of chained HMMs.

In this paper, we adapt a different formulation of the recognition algorithm, called the *token-passing algorithm* [37], for ASL recognition. It works as follows:

- Each state $S_i$ contains at time $t$ a *token* denoting $\delta_t(i)$ from Eq. (1).
- At time $t + 1$, for each $S_i$, pass tokens $\text{tok}_{ij}(t + 1) = \delta_i(t) a_{ij}$ to all states $S_j$ connected to $S_i$.

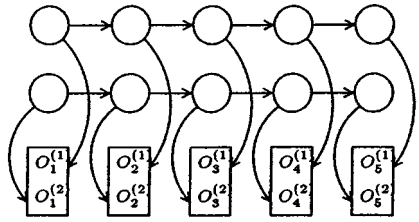


**FIG. 12.** FHMMs: The output is combined. $O_i^{(n)}$ denotes the output of the $n$th channel at the $i$th frame.

• Finally, for each state $S_j$, pick $\max_i\{\text{tok}_{ij}(t+1)\}$, and update this token to denote $\delta_j(t+1) = \text{tok}_{ij}(t+1)b_j(O_{t+1})$.

The token-passing algorithm is 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. Thus, 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 4.3.2 we explain why carrying such additional information can be useful. This functionality would be difficult to replicate with the Viterbi algorithm.

### *4.2. Extensions to HMMs*

Regular 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 require that the different channels described in Section 3.3.2 evolve in lockstep, passing through the same state at the same time. This lockstep property of regular HMMs is unsuitable for many applications. Sign language consists of parallel, possibly interacting, channels as described in Section 3.3.2. 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. If the channels evolved in lockstep, the movement of the weak hand would be impossible to capture.

Second, as discussed in Section 3.3.2, the number of possible combinations of phonemes occurring simultaneously is overwhelming. It is computationally infeasible to use on the order of $10^8$ HMMs, let alone to collect enough training data. For these two reasons, it is necessary to extend the HMM framework for ASL recognition.

In past research, two fundamentally different methods of extending HMMs have been described. The first method models the $C$ channels[3] in $C$ separate HMMs, effectively creating a metastate in an $C$-dimensional state space. It combines the output of the $C$ HMMs in a single output signal, such that the output probabilities depend on the $C$-dimensional metastate (Fig. 12). Such models are called factorial hidden Markov models (FHMMs). Because the output probabilities depend on the metastate, an optimal training method based on expectation maximization would take time exponential in $C$. Ghahramani and Jordan describe approximate polynomial-time training methods based on mean-field theory [7].

[3] Note that in the following we use the term, "channel" exclusively to clarify the relationship between different HMM extensions and ASL phonology (cf. Section 3.3.2). This does not mean that the algorithms we 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 we make for the channels in ASL.

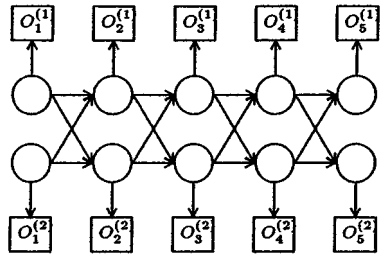

**FIG. 13.** CHMMs: The output is separate, but the states influence one another.

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 from state $S_t^{(i)}$ to $S_{t+1}^{(i)}$ in the HMM for channel $i$ does not only depend on the state $S_t^{(i)}$, but on the $S_t^{(j)}$ states in all channel, where $1 \leq j \leq C$ (Fig. 13). Such HMMs are called coupled hidden Markov models (CHMMs). Brand *et al.* describe polynomial-time training methods and demonstrate the advantages of CHMMs over regular HMMs in [3].

Unfortunately, FHMMs or CHMMs only provide a solution to the problem that regular HMMs force the channels to evolve in lockstep. They do not help with making the sheer number of possible phoneme combinations computationally tractable, because the training methods still require a priori modeling of all combinations. Thus, we need a new approach to modeling ASL with HMMs. We now describe parallel HMMs as a solution to the aforementioned two problems.

### *4.3. A New Approach: Parallel HMMs*

Parallel HMMs model the $C$ channels with $C$ independent HMMs with separate output (Fig. 14). Unlike CHMMs, the state probabilities influence one another only within the same channel. That is, PaHMMs are essentially regular HMMs that are used in parallel.

Hermansky *et al.*, as well as Bourlard and Dupont, first suggested the use of PaHMMs in the speech recognition field [10, 1]. They broke down the speech signal into subbands, which they modeled independently, so as to be able to exclude noisy or corrupted subbands, and merged the subbands during recognition with multilayered perceptrons. They demonstrated that subband modeling can improve recognition rates. Note that the goal of subband modeling differs from our goal of making ASL recognition methods scale. Subband modeling is concerned with eliminating unreliable parts of the speech signal, whereas we would like to develop a computationally tractable method of modeling all aspects of ASL.

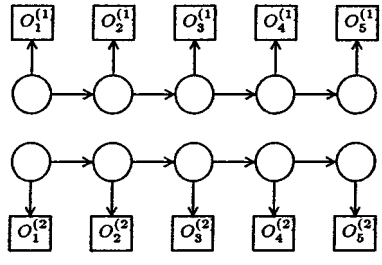

**FIG. 14.** PaHMMs: The output is separate, and the states of separate channels are independent. $O_i^{(n)}$ denotes the output of the $n$th channel at the $i$th frame.

PaHMMs are based on the assumption that the separate channels evolve independently from one another with independent output. The justification for using this independence assumption is that there is linguistic evidence that the different channels of ASL can be viewed as acting with a high degree of independence on the phoneme level [18]. As our experiments in Section 6 show an improvement in recognition rates, this assumption is at least partially valid.

As a consequence, the HMMs for the separate channels can be trained completely independently. Thus, the problem of modeling all possible combinations of phonemes disappears. Now it is necessary to consider only an order of $(30 + 8 + 8 + 20 + 40) \times 2$ HMMs instead of an order of $10^8$ HMMs (see Section 3.3.2 for an explanation of the numbers).

*4.3.1. Combination of the channels.* At some stage during recognition, it is necessary to merge the information from the HMMs representing the $C$ different channels. We would like to find (in log probability form)

$$\max_{Q^{(1)},\ldots,Q^{(C)}} \left\{\log P\left(Q^{(1)},\ldots,Q^{(C)},O^{(1)},\ldots,O^{(C)} \mid \lambda_1,\ldots,\lambda_C\right)\right\}, \tag{4}$$

where $Q^{(i)}$ is the state sequence of channels $i$ with output sequence $O^{(i)}$ through the HMM network $\lambda_i$. Furthermore, the $Q^{(i)}$ are subject to the constraint that they all follow the same sequence of signs. Because we assume the channels to be independent, the merged information consists of the product of the probabilities of the individual channels, so we can rewrite (4) as

$$\begin{aligned}&\max_{Q^{(1)},\ldots,Q^{(C)}} \left\{\log P\left(Q^{(1)},\ldots,Q^{(C)},O^{(1)},\ldots,O^{(C)} \mid \lambda_1,\ldots,\lambda_c\right)\right\}\\ &\quad= \max_{Q^{(1)},\ldots,Q^{(C)}} \left\{\sum_{i=1}^{C} \log P\left(Q^{(i)},O^{(i)} \mid \lambda_i\right)\right\}.\end{aligned} \tag{5}$$

Because HMMs assume that successive outputs are independent, we rewrite (5) as

$$\max_{Q^{(1)},\ldots,Q^{(C)}} \left\{\sum_{i=1}^{C} \log P\left(Q^{(i)},O^{(i)} \mid \lambda_i\right)\right\} = \max_{Q^{(1)},\ldots,Q^{(C)}} \left\{\sum_{j=1}^{W}\sum_{i=1}^{C} \log P\left(Q_{(j)}^{(i)},O_{(j)}^{(i)} \mid \lambda_i\right)\right\}, \tag{6}$$

where we split the output sequences into $W$ segments, and $Q_{(j)}^{(i)}$ and $O_{(j)}^{(i)}$ are the respective state and observation sequences in channel $i$ corresponding to segment $j$. Intuitively, this equation tells us that we can combine the probabilities as many times as desired at any stage of the recognition process, including the whole-sign level or the phoneme level.

It is desirable to weight the channel on a per-word basis, because in some two-handed signs the weak hand does not move. Such signs could be easily confused with one-handed signs where the weak hand happens to be in a position similar to that required by the two-handed sign. In these situations, the strong hand should carry more weight than the weak hand. If we let $\omega_{(j)}^{(i)}$ be the weight of word $j$ in channel $i$, the desired quantity to maximize becomes (from Eq. (6))

$$\max_{Q^{(1)},\ldots,Q^{(C)}} \left\{\sum_{j=1}^{W}\sum_{i=1}^{C} \omega_j^{(i)} \log P\left(Q_{(j)}^{(i)},O_{(j)}^{(i)} \mid \lambda_i\right)\right\}, \tag{7}$$

where $\sum_i \omega_j^{(i)} = C$ for fixed $j$.

Before we describe how the token-passing algorithm described in Section 4.1 needs to be modified for PaHMMs, we need to consider a subtle point. Consider using two channels to model the movements of the strong and the weak hands in ASL. What does the weak hand do in a one-handed sign? From a recognition point of view, we do not care, and thus we should assign a probability of one to anything that the weak hand does during the course of a one-handed sign.

Unfortunately, doing so would bias recognition toward one-handed signs, because the average log probabilities for one-handed signs would then be twice as large as the average log probabilities for two hands. Instead, we define the probability of the weak hand to be the same as the probability of the strong hand for one-handed signs.

*4.3.2. The recognition algorithm.* In principle, adapting the token-passing algorithm to PaHMMs consists of applying the regular token-passing algorithm to the HMMs in the separate channels, and combining the probabilities of the channels at word or phoneme ends according to (7). See Fig. 15 for an example with two channels (e.g., left and right hands).

In practice, the recognition algorithm is more complicated, because it must enforce the constraint that the paths $Q^{(i)}$ all touch exactly the same sequence of words. It does not make sense to combine the probabilities of tokens from different paths. 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 starting 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 hit a starting node of a sign, the recognizer looks 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.

At each word end the recognizer combines the probabilities of only those tokens that have the same path identifier. Here the advantage of the token-passing algorithm over the Viterbi algorithm becomes clear, because this information can be directly attached to the tokens.

In addition, a path in channel $k$ that contributes to maximizing (7) does not necessarily maximize the marginal probability $\sum_{j=1}^{W} \log P(Q_{(j)}^{(k)}, O_{(j)}^{(k)} \mid \lambda_k)$. To overcome the potential discrepancy between maximizing the joint and marginal probabilities, each state needs to keep track of a set of the first few best tokens, each with a unique path identifier. That is, instead of working with only one hypothesis per channel, the algorithm works with a

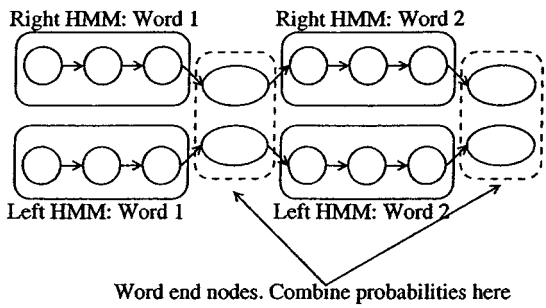


**FIG. 15.** The tokens are passed independently in the HMMs for the left and the right hands, and combined in the word end nodes.

maximum of $M$ hypotheses per channel, where $M$ is the cardinality of the token set. The actual number of hypotheses kept at any time depends on how much the paths in the different channels overlap.

To ensure that the algorithm assigns the probabilities of the strong hand to the weak hand when it encounters a one-handed sign (see the previous section for why this is necessary), we define two operations:

• **Join**(*node*) takes the tokens of the weak hand in word end node *node* and attaches them to the tokens of the strong hand in the same word end node. The attached token must have the same path identifier as the token that it is attached to.

• **Split**(*node*) detaches the weak hand tokens from the strong hand tokens in word start node *node*. 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 probabilities of the detached tokens with the probabilities of the strong hand for the last sign. Then it merges the tokens with the existing tokens of the weak hand in the same word start node.

If we denote the number of output frames with $T$, the modified token-passing algorithm is . given in the following algorithm.

ALGORITHM 1 (TOKEN PASSING ALGORITHM FOR PAHMMS).

```
1.  Initialize the tokens in the start nodes of the HMM network with log p = 0.
2.  for t = 1 to T
3.    for c = 1 to C
4.      for each state in all HMM states
5.        Pass the tokens in state to the adjacent states and merge them with the tokens
 in the adjacent states.
6.      end for
7.    end for
8.    for c = 1 to C
9.      for each node that is a word end node
10.       Combine the token probabilities.
11.       if node is a two-handed sign
12.         Join(node).
13.       end if
14.       for each node' adjacent to node
15.         Pass the tokens in node to node'
16.         if node' is a two-handed sign
17.           Split(node').
18.         end if
19.       end for
20.     end for
21.   end for
22. end for
```

Assuming that the token sets in each state have cardinality $M$ and are stored as lists sorted by log likelihood, passing the token set from one single state to another takes $O(M)$ time. Hence, step 5 takes $O(NM)$ 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.

The combined token probabilities in step 10 need to be computed only once per word end node for all channels, because they are the same across all channels. Thus, they can be cached for subsequent iterations over $C$ in the loop starting at step 8. The algorithm for combining the probabilities iterates over all token sets and stores them in a hash table with the path identifier as the key. With this hash table, the algorithm keeps track of the combined token probabilities, and whether a token occurs in all token sets. The latter is a necessary condition for a token to be in the combined set. Because hash tables have expected lookup times of $O(1)$ and there are at most $CM$ tokens looked up, the combination step runs in $O(CM)$ expected time over all channels.

Using hash tables with the path identifier as the key, **Join** in step 12 takes $O(M)$ expected time. Step 15 takes $O(M)$ time per call, by the same argument as for step 5. **Split** in step 17 takes $O(M)$ time per call, because it uses a token set merge internally. The loop in step 4 iterates $N$ times. The loops in steps 9 and 14 iterate $N$ times in the worst case, but are executed much less often in the average case, because there are fewer words than HMM states.

From all these individual times, it follows that the entire algorithm runs in

$$\begin{aligned} &O(T(CN \times NM + NCM + CN(M + N(M + M)))) \\ &\quad = O(T(CN^2M + NCM + CN^2M)) \\ &\quad = O(TCN^2M) \end{aligned} \tag{8}$$

expected time. That is, it takes time linear in the number of channels and in the number of tokens per state.

## 5. HMMs IN ASL RECOGNITION

As mentioned in Section 4.1, the basic idea behind HMM-based recognition is to chain the HMMs together into a network. The Viterbi algorithm finds the most likely path through this network, 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 work in the same way as that for speech recognition. However, there are some peculiarities in the network design and training process that are caused specifically by the properties of sign languages. We now describe what they are and how to manage them.

### *5.1. 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. However, in speech recognition, the composite models for the signs are then chained together into the recognition network. 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.

It is convenient to connect each HMM node that ends a sign 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 HMM node that starts a

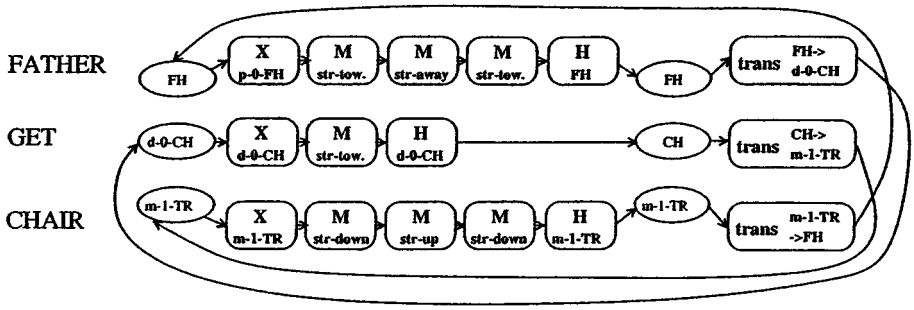


**FIG. 16.** Network that models the signs for "father," "get," and "chair" in terms of their constituent phonemes. Epenthesis is modeled explicitly with HMMs (labeled with "trans"). The oval nodes in this figure are the body locations at the beginning and the end of each sign.

sign to a node corresponding to its starting body location. These nodes are nonemitting; that is, they do not consume any input frames. The token-passing algorithm described in Section 4.1 works without modifications on such nodes. This trick reduces the number of arcs and thus the complexity of the HMM network.

Figure 16 shows how to chain together the phoneme and epenthesis HMMs, and the nonemitting body location nodes for the three signs for "father," "get," and "chair."

We have not provided any descriptions of the epenthesis movements yet. Ideally, they should be expressed in terms of the basic movements in the movement–hold model. Unfortunately, the exact appearance of these movements is poorly understood, and there exists almost no literature on them. For this reason, we choose a different approach to modeling these. It is based on the observation that an epenthesis movement is uniquely specified by the ending location of the preceding sign and the starting location of the following sign. Since there are 20 major body locations in ASL, this approach yields at most $20^2 = 400$ HMMs. 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 are modeled by the same HMM.

### *5.2. Training PaHMMs*

With PaHMMs, we need such a network for every channel. The word end nodes of each sign in each channel are associated with one another, as schematically shown in Fig. 15 in Section 4.3.2. These associations allow the recognition algorithm to combine the probabilities of each channel.

In principle, the HMMs can be trained independently for each channel with standard methods, such as Viterbi alignment [25] and embedded Baum–Welch reestimation [36]. 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 (hand movements and locations, handshape, orientation) 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 4.3.2, 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 input signal are covered by HMMs.

**FIG. 17.** These images show the 3D tracking of the sign for "father."

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 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 the states of the HMMs are assigned the global mean and covariance of the training data, is not the best initialization method. Instead, each channel is best initialized with a set of hand-labeled data. Our experiments showed that training the movements and holds of the weak hand in this fashion yields reasonable results.

We now provide experiments with phoneme modeling and PaHMMs to validate our approach.

## 6. EXPERIMENTS

We ran several continuous recognition experiments with 3D data to test the feasibility of modeling the movements of the left and the right hands with PaHMMs. Our database consisted of 400 training sentences and 99 test sentences over a vocabulary of 22 signs. The full transcriptions of these signs are listed in Appendix 1. The sentence structure was constrained only by what is grammatical in ASL. We performed all training and

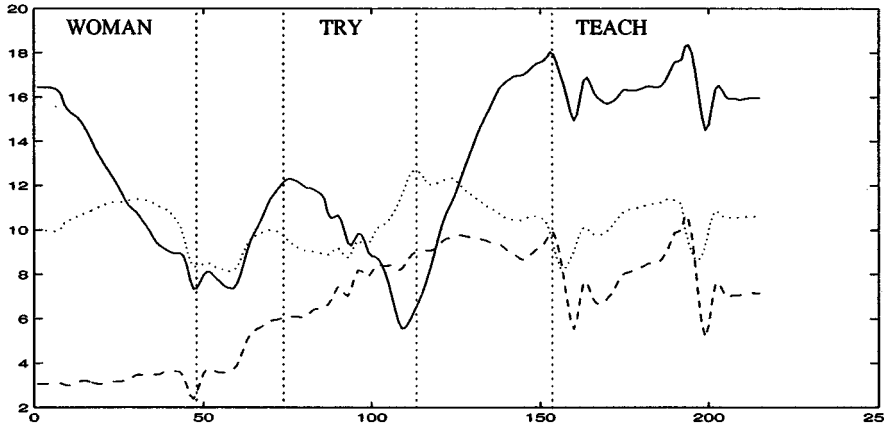


**FIG. 18.** Example of the 3D position signal for the sentence "woman try teach." The solid line is from the $x$ coordinate, the dashed line is from the $y$ coordinate, and the dotted line is from the $z$ coordinate. The unlabeled parts of the signal are epenthesis movements.

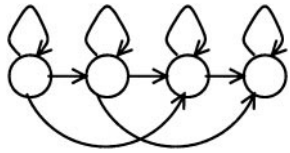

**FIG. 19.** Example of a 4-state HMM with the Bakis topology. This topology seems best to be able to absorb variations in speed.

testing with a heavily modified version of Entropic's Hidden Markov Model Toolkit (HTK).

We collect the sentences with an Ascension Technologies MotionStar 3D tracking system, and with our vision-based tracking system at 60 frames per second. The latter uses physics-based modeling to track the arms and the hands of the signer, as depicted in Fig. 17. The models are estimated from the images from a subset of three orthogonal cameras. These are selected on a per-frame basis depending on the occluding contour of the signer's limbs [14–16, 21].

The total number of unique segments was 89 for the right hand and 51 for the left hand, so we trained a total of 140 HMMs. In a testament to the clear advantage of phoneme-based modeling over whole-sign-based modeling, many HMMs had more than 30 training examples available.

We used an 8-dimensional feature vector for each hand. Six features consisted of 3D positions and velocities relative to the base of the signer's spine. For the remaining two features, we computed the largest two eigenvalues of the positions' covariance matrices over a window of 15 frames centered on the current frame. In normalized form, these two eigenvalues provide a useful characterization of the global properties of the signal [33]. Note that our goal is to evaluate a novel recognition algorithm, not the merits of different features.

Figure 18 shows an example of what the 3D position signal typically looks like, after the collection with the MotionStar system or with the computer vision system. This particular example is from the sentence "woman try teach." The coordinate system was right-handed, with the positive $x$ axis facing up, with inches as the unit of measurement. Note how the length of an epenthesis movement can vary greatly depending on the type of movement, justifying our choice to model them explicitly.

**TABLE 2**
**Regular HMMs: Results of the Recognition Experiments**

| Level | Accuracy | Details |
|---|---|---|
| sentence | 80.81% | $H = 80^a$, $S = 19^b$, $N = 99^c$ |
| sign | 93.27% | $H = 294$, $D = 3^d$, $S = 15$, $I = 3^e$, $N = 312$ |

*Note.* 80.81% of the sentences were recognized correctly, and 93.27% of the signs were recognized correctly.

[a] $H$ denotes the number of correctly recognized sentences or signs.

[b] $S$ denotes the number of substitution errors.

[c] $N$ denotes the total number of signs or sentences in the test set.

[d] $D$ denotes the number of deletion errors.

[e] $I$ denotes the number of insertion errors.

**TABLE 3**
**PaHMMs: Results of the Recognition Experiments, with Merging of the Token Probabilities at the Phoneme Level**

| Level | Accuracy | Details |
|---|---|---|
| sentence | 84.85% | $H = 84, S = 15, N = 99$ |
| sign | 94.23% | $H = 297, D = 3, S = 12, I = 3, N = 312$ |

*Note.* See Table 2 for an explanation of the terminology.

We used a Bakis topology [25] for all HMMs. In this topology, all states are connected to be themselves, the next state, and the state after that one (Fig. 19). This topology seems best to be able to cope with varying signing speeds and phoneme lengths. This observation also agrees with those made in Hienz *et al*. [11, 12]. Not counting nonemitting states, we used 7-state HMMs to model movements, 5-state HMMs to model holds, 1-state HMMs to model X segments, and 4-state HMMs to model epenthesis movements. The optimal number of states depends primarily on the frame rate and the feature vector used—with global features, fewer states are necessary. We fine-tuned this topology and the numbers of states for each type of model experimentally.

### *6.1. Comparison of PaHMMs and Regular HMMs*

The purpose of the experiments was to determine by how much does using PaHMMs with two channels improve recognition rates over regular HMMs with just one channel. Note that regular HMMs are equivalent to 1-channel PaHMMs. In the PaHMM experiments, one channel consisted of the movement and hold segments (describing the hand movements and locations) of the strong hand, and the other channel consisted of the corresponding segments of the weak hand. Thus, the difference between these two experiments lies in the addition of a channel with information from the weak hand.

To establish the baseline with regular HMMs, we first ran an experiment using only the 8-dimensional features (3D position, 3D velocities, and eigenvalues of the positions' covariance matrices) of the right hand. The results are given in Table 2. We did not test FHMMs, CHMMs, or regular HMMs with both hands, because even for the small 22-sign vocabulary the number of occurring phoneme combinations was far too large for the 400-sentence training set. The goal of these experiments was to demonstrate whether PaHMMs can outperform regular HMMs while preserving scalability, not to investigate whether PaHMMs perform better or worse than FHMMs and CHMMs.

**TABLE 4**
**Effect of Token Set on Recognition Rates, with Merging of the Token Probabilities at the Phoneme Level**

| Cardinality | Sentence accuracy (%) | Sign accuracy (%) |
|---|---|---|
| 2 | 82.83 | 92.95 |
| 3 | 84.85 | 94.23 |
| 5 | 84.85 | 94.23 |
| 8 | 84.85 | 94.23 |

**TABLE 5**
**Effect of the Level of Token Probability Merging on Recognition Rates[a]**

| Merge level | Sent. accuracy (%) | Sign accuracy (%) |
|---|---|---|
| Sign level | 84.85 | 94.23 |
| Phoneme level | 84.85 | 94.55 |

[a] In both cases, the token set had a cardinality of 3.

An analysis revealed that there were only seven sentences with incorrectly recognized two-handed signs. Each of these seven sentences involved a single substitution error. Thus, the maximum recognition rate that we could expect from this experiment, using PaHMMs to model both hands, was 87.88% on the sentence level and 96.47% on the sign level. Table 3 shows the actual recognition rates with PaHMMs, with merging of the token probabilities at the phoneme level.

Of the seven sentences with two-handed signs that the regular HMMs failed to recognize, the PaHMMs recognized four correctly. One of the other three sentences now contained an additional substitution error in a one-handed sign. All other sentences were not affected. That is, the PaHMMs recognized every single sentence correctly that was already recognized correctly by the regular HMMs.

We view this result as evidence that PaHMMs can improve recognition rates over regular HMMs, with no significant tradeoffs in recognition accuracy. This result also contributes evidence toward validating the assumption that the parallel channels in ASL can be modeled independently.

### *6.2. Factors Influencing PaHMM Accuracy*

There are two factors that can potentially influence the recognition accuracy of PaHMMs. The first factor is the required cardinality $M$ of the token set in each state. Recall from Section 4.3.2 that $M$ determines how many hypotheses are kept at most for each channel. Because the time complexity of the recognition algorithm is linear in $M$, the cardinality should be as small as possible. The second factor is the level of merging the token probabilities. Is it better to perform the merging at the phoneme level or at the whole-sign level?

Table 4 shows the results for token set cardinalities of 2, 3, 5, and 8. Recognition accuracy does not seem to be affected by cardinalities beyond 3. The log probabilities of the tokens are not significantly affected either. We expect that using more than two channels will not have a significant effect on the required cardinality of the token sets, provided that the HMMs in each channel have been well trained.

Table 5 shows the effect of merging the token probabilities at the whole-sign level. The level of merging has a small effect on recognition rates, but it is not significant.

## 7. CONCLUSIONS

We demonstrated that PaHMMs can improve the robustness of ASL recognition even on a small scale. Together with breaking down the signs into phonemes, they provide a powerful

and potentially scalable framework for modeling ASL. Because PaHMMs are potentially more scalable than other extensions to HMMs, they are an interesting research topic for gesture and sign language recognition.

Future research should establish how PaHMMs behave with larger vocabularies, and particularly with highly inflected signs that can exhibit a large number of phoneme combinations within one single sign. Future research should also add hand configuration and orientation as new channels to the PaHMM framework.

Once the viability of PaHMMs has been established for more channels and for larger vocabularies, the major outstanding challenges in modeling ASL recognition will be the integration of facial expressions, the use of space. Facial expressions are important, because they constitute 80% of the grammar of ASL. Space is important, because almost all subject–object relations are expressed in terms of locations in front of the body. Both facial expressions and the use of space will be essential in building a complete grammatical representation of the recognized ASL.

Semantic representation of ASL will also be important, particularly for deaf–hearing interaction. Because the structure of ASL is so different from spoken languages, it is necessary to do more research into parsing the recognized ASL constructs and converting them into a semantic representation.

## APPENDIX: PHONETIC TRANSCRIPTIONS

The following table gives the phonetic transcriptions of the 22-sign vocabulary for the strong hand. The phonemes beginning with *M* denote movements, the phonemes beginning with *H* denote holds, and the phonemes beginning with *X* denote the *X* segments.

| 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-$TR$} |
| don't mind | $H$-{$NS$}$M$-{$str_{DownRightAway}$}$H$-{$m$-1-$TR$} |
| good | $H$-{$MO$}$M$-{$str_{DownAway}$}$H$-{$m$-0-$CH$} |

| Sign | Transcription |
|---|---|
| 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-%$FH$} |

The following table gives the phonetic transcriptions of the 22-sign vocabulary for the weak hand. The Symbols' meanings are the same as in the previous table. In addition, $\emptyset$ indicates that the sign is one-handed; in these cases the weak hand does nothing.

| Sign | Transcription |
|---|---|
| I | $\emptyset$ |
| man | $\emptyset$ |
| woman | $\emptyset$ |
| father | $\emptyset$ |
| mother | $\emptyset$ |
| 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 | $\emptyset$ |
| relate | $X$-{$m$-1-%$TR$}$M$-{$str_{Right}$}$H$-{$m$-0-$TR$} |
| don't mind | $\emptyset$ |
| good | $H$-{$m$-0-$CH$} |
| gross | $\emptyset$ |
| sorry | $\emptyset$ |
| stupid | $\emptyset$ |
| beautiful | $\emptyset$ |

## ACKNOWLEDGMENTS

This work was supported in part by NSF Career Award NSF-9624604, ONR Young Investigator Proposal, NSF IRI-97-01803, AFOSR F49620-98-1-0434, and NSF EIA-98-09209.

## REFERENCES

1. H. Bourlard and S. Dupont, Subband-based speech recognition, in *Proceedings of the ICASSP*, 1997.
2. A. Braffort, ARGo: An architecture for sign language recognition and interpretation, in *Progress in Gestural Interaction. Proceedings of Gesture Workshop '96* (A. D. N. Edwards and P. A. Harling, Eds.), pp. 17–30, 1997. Springer-Verlag, Berlin.

3. 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.
4. D. Brentari, Sign language phonology: ASL, in *The Handbook of Phonological Theory* (J. A. Goldsmith, Ed.), Blackwell Handbooks in Linguistics, pp. 615–639, Blackwell, Oxford, 1995.
5. G. R. Coulter (Ed.), *Current Issues in ASL Phonology, Vol. 3, Phonetics and Phonology*, Academic Press, San Diego, CA, 1993.
6. 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*, 1996.
7. Z. Ghahramani and M. I. Jordan, Factorial hidden Markov models, *Machine Learning* **29**, 1997, 245–275.
8. S. Gibet, J. Richardson, T. Lebourque, and A. Braffort, Corpus of 3d natural movements and sign language primitives of movement, in *Gesture and Sign Language in Human–Computer Interaction. Proceedings of Gesture Workshop '97* (I. Wachsmuth and M. Fröhlich, Eds.), Springer-Verlag, Berlin, 1998.
9. 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, Orlando, FL, 1997*, pp. 162–167.
10. H. Hermansky, S. Tibrewala, and M. Pavel, Towards ASR on partically corrupted speech, in *Proceedings of the ICSLP*, pp. 462–465, 1996.
11. H. Hienz, B. Bauer, and K.-F. Kreiss, HMM-based continuous sign language recognition using stochastic grammars, in *Gesture-Based Communication in Human-Computer Interaction* (A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, Eds.), Vol. 1739, Lecture Notes in Artificial Intelligence, pp. 185–196, Springer-Verlag, Berlin, 1999.
12. H. Hienz, K.-F. Kraiss, and B. Bauer, Continuous sign language recognition using hidden Markov models, in *ICMI'99* (Y. Tang, Ed.), pp. IV10–IV15, Hong Kong, 1999.
13. 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, Wilmington, DE, 1996*, pp. 165–174.
14. I. Kakadiaris and D. Metaxas, 3d human body model acquisition from multiple views, in *Proceedings of the ICCV*, pp. 618–623, 1995.
15. 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*, pp. 980–984, 1994.
16. I. Kakadiaris, D. Metaxas, Model based estimation of 3d human motion with occlusion based on active multi-viewpoint selection, in *Proceedings of the CVPR*, pp. 81–87, 1996.
17. 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, Nara, Japan, 1998*, pp. 558–565.
18. S. K. Liddell and R. E. Johnson, American Sign Language: The phonological base, *Sign Lang. Stud.* **64**, 1989, 195–277.
19. C. Lucas (Ed.), *Sign Language Research: Theoretical Issues*, Gallaudet Univ. Press, Washington, DC, 1990.
20. D. McNeill, *Hand and Mind: what gestures reveal about thought*, Univ. of Chicago Press, Chicago, 1992.
21. D. Metaxas, *Physics-based Deformable Models: Applications to Computer Vision, Graphics and Medical Imaging*, Kluwer Academic, Dordrecht, 1996.
22. Y. Nam and K. Y. Wohn, Recognition and modeling of hand gestures using colored petri nets. *IEEE Trans. Syst. Man Cybernet. A*, 1999, in press.
23. Y. Nam and K. Y. Wohn, Recognition of space-time hand-gestures using hidden Markov model, in *ACM Symposium on Virtual Reality Software and Technology, 1996.*
24. V. Pavlovic, R. Sharma, and T. S. Huang, Visual interpretation of hand gestures for human–computer interaction: A review, *IEEE Trans. Pattern Anal. Mach. Intell.* **19**, 1997, 677–695.
25. L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. IEEE* **77**, 1989, 257–286.
26. W. Sandler, *Phonological Representation of the Sign: Linearity and Nonlinearity in American Sign Language*, Publications in Language Sciences, 32, Foris, Dordrecht, 1989.

27. T. Starner and A. Pentland, Visual recognition of American Sign Language using hidden Markov models, in *International Workshop on Automatic Face and Gesture Recognition, Zürich, Switzerland, 1995*, pp. 189–194.
28. T. Starner, J. Weaver, and A. Pentland, Real-time American Sign Language recognition using desk and wearable computer based video, *IEEE Trans. Pattern Anal. Mach. Intell.* **20**, 1998, 1371–1375.
29. 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]
30. C. Valli and C. Lucas, *Linguistics of American Sign Language: An Introduction*, Gallaudet Univ. Press, Washington, DC, 1995.
31. 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, Orlando, FL. 1997*, pp. 156–161.
32. C. Vogler and D. Metaxas, Parallel hidden Markov models for American Sign Language recognition, in *Proceedings of the IEEE International Conference on Computer Vision, Kerkyra, Greece, 1999*, pp. 116–122.
33. C. Vogler and D. Metaxas, Toward scalability in ASL recognition: Breaking down signs into phonemes. in *Gesture-Based Communication in Human-Computer Interaction* (A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, Eds.), Vol. 1739, Lecture Notes in Artificial Intelligence, pp. 211–224, Springer-Verlag, Berlin, 1999.
34. M. B. Waldron and S. Kim, Isolated ASL sign recognition system for deaf persons, *IEEE Trans. Rehabilitation Eng.* **3**, 1995, 261–271.
35. Y. Wu and T. Huang, Vision-based gesture recognition: A review, in *Gesture-Based Communication in Human-Computer Interaction* (A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, Eds.), Vol. 1739, Lecture Notes in Artificial Intelligence, pp. 103–115, Springer-Verlag, Berlin, 1999.
36. S. Young, J. Jansen, J. Odell, D. Ollason, and P. Woodland, *The HTK Book (for HTK 2.0).* Cambridge Univ. Press, Cambridge, UK, 1995.
37. 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.