Visual Recognition of American Sign Language Using Hidden Markov Models

Thad Starner and Alex Pentland
Perceptual Computing Section, The Media Laboratory,
Massachusetts Institute of Technology
Room E15-383, 20 Ames Street, Cambridge MA 02139, USA
E-mail: thad@media.mit.edu, sandy@media.mit.edu

Abstract

Hidden Markov models (HMM's) have been used prominently and successfully in speech recognition and, more recently, in handwriting recognition. Consequently, they seem ideal for visual recognition of complex, structured hand gestures such as are found in sign language. We describe an HMM-based system for recognizing sentence level American Sign Language (ASL) which attains a word accuracy of 99.2% without explicitly modeling the fingers.

1 Introduction

There has been a resurging interest in recognizing human hand gestures. While there are many interesting domains, one of the most structured sets of gestures are those belonging to any of the several sign languages. In sign language, each gesture already has assigned meaning, and strong rules of context and grammar may be applied to make recognition tractable.

To date, most work on sign language recognition has employed expensive wired "datagloves" which the user must wear [19]. In addition, these systems have mostly concentrated on finger signing, where the user spells each word with hand signs corresponding to the letters of the alphabet [3]. However, most signing does not involve finger spelling but, instead, gestures which represent whole words. This allows signed conversations to proceed at about the pace of spoken conversation.

In this paper we describe an estensible system which uses a single color camera to track hands in real time and interprets American Sign Language (ASL) using Hidden Markov Models (HMM). The hand tracking stage of the system does not attempt to produce a fine-grain description of hand shape; studies have shown that such detailed information may not be necessary for humans to interpret sign language [12, 16]. Instead, the tracking process produces only a coarse description of hand shape, orientation, and trajectory. Currently we require that the user wear inexpensive colored gloves to facilitate the hand tracking frame rate and stability. This shape, orientation, and trajectory information is then input to a HMM for recognition of the signed words.

Hidden Markov models have intrinsic properties which make them very attractive for sign language recognition. Explicit segmentation on the word level is not necessary for either training or recognition [18]. Language and context models can be applied on several different levels, and much related development of this technology has already been done by the speech recognition community. Consequently, sign language recognition seems an ideal machine vision application of HMM technology, offering the benefits of problem scalability, well defined meanings, a pre-determined language model, a large base of users, and immediate applications for a recognizer.

American Sign Language (ASL) is the language of choice for most deaf people in the United States. ASL uses its own grammar instead of borrowing from English. This grammar allows more flexibility in word placement and sometimes uses redundancy for emphasis. Another variant, English Sign Language has more in common with spoken English but is not in widespread use in America. ASL consists of approximately 6000 gestures of common words with finger spelling used to communicate obscure words or proper nouns.

Conversants in ASL may describe a person, place, or thing and then point to a place in space to temporarily store that object for later reference [16]. For the purposes of this experiment, this aspect of ASL will be ignored. Furthermore, in ASL the eyebrows are raised for a question, held normal for a statement, and furrowed for a directive. While there has been work in recognizing facial gestures [4], facial features will not be used to aid recognition in the task addressed.

While the scope of this work is not to create a person independent, full lexicon system for recognizing ASL, a desired attribute of the system is extensibility towards this goal. Another goal is to allow the creation of a real-time system by guaranteeing each separate component (tracking, analysis, and recognition) runs in real-time. This demonstrates the possibility of a commercial product in the future, allows easier experimentation, and simplifies archiving of test data. "Continuous" sign language recognition of full sentences is desired to demonstrate the feasibility of recognizing complicated series of gestures. Of course, a low error rate is also a high priority.

Table 1: ASL Vocabulary Used

part of speech	*vocabulary*
pronoun	I you he we you(pl) they
verb	want like lose dontwant dontlike love pack hit loan
noun	box car book table paper pants bicycle bottle can wristwatch umbrella coat pencil shoes food magazine fish mouse pill bowl
adjective	red brown black gray yellow

In this recognition system, sentences of the form "personal pronoun, verb, noun, adjective, (the same) personal pronoun" are to be recognized. This sentence structure emphasizes the need for a distinct grammar for ASL recognition and allows a large variety of meaningful sentences to be randomly generated using words from each class. Table 1 shows the words chosen for each class. Six personal pronouns, nine verbs, twenty nouns, and five adjectives are included making the total lexicon number forty words. The words were chosen by paging through Humphries *et al.* [8] and selecting those which would provide coherent sentences when generating random sentences. At first a naive eye was used to avoid ambiguities in the selected signs, but this was shortly subsumed by the coherency constraint.

2 Machine Sign Language Recognition

Attempts at machine sign language recognition have begun to appear in the literature over the past five years. However, these systems have generally concentrated on isolated signs and small training and test sets. Tamura and Kawasaki demonstrated an early image processing system which could recognize 20 Japanese signs based on matching cheremes [20]. Charayaphan and Marble [2] demonstrated a feature set that could distinguish between the 31 isolated ASL signs in their training set (which also acted as the test set). Takahashi and Kishino in [19] discuss a Dataglove-based system that could recognize 34 of the 46 Japanese kana alphabet gestures (user dependent) using a joint angle and hand orientation coding technique. The test user made each of the 46 gestures 10 times to provide data for principle component and cluster analysis. A separate test set was created from five iterations of the alphabet by the user, with each gesture well separated in time.

3 Previous Use of Hidden Markov Models in Gesture Recognition

While the continuous speech recognition community adopted HMM's many years ago, these techniques are just now entering the vision community. Most early work was limited to handwriting recognition [10, 11]. More recently, He and Kundu [5] report using continuous density HMM's to classify planar shapes. Another early effort by Yamato *et al.* [21] uses discrete HMM's to successfully recognize image sequences of six different tennis strokes among three subjects. This experiment is significant because it used a 25x25 pixel quantized subsampled camera image as a feature vector. Even with such low-level information, the model could learn the set of motions to perform respectable recognition rates. Schlenzig *et al* [15] also use hidden Markov models for visual gesture recognition. The gestures are limited to "hello," "good-bye," and "rotate." The authors report "intuitively" defining the HMM associated with each gesture and imply that the normal Baum-Welch re-estimation method was not implemented. However, this study shows the continuous gesture recognition capabilities of HMM's by recognizing gesture sequences.

4 Hidden Markov Modeling

While a substantial body of literature exists on HMM technology [1, 7, 13, 22], this section briefly outlines a traditional discussion of the algorithms. After outlining the fundamental theory in training and testing of a discrete HMM, this result is then generalized to the continuous density case used in the experiments. For broader discussion of the topic, [7, 17] are recommended.

A time domain process demonstrates a Markov property if the conditional probability density of the current event, given all present and past events, depends only on the jth most recent events. If the current event depends solely on the most recent past event, then the process is a first order Markov process. While the order of words in American Sign Language is not truly a first order Markov process, it is a useful assumption when considering the positions and orientations of the hands of the signer through time.

The initial topology for an HMM can be determined by estimating how many different states are involved in specifying a sign. Fine tuning this topology can be performed empirically. While different topologies can be specified for each sign, a four state HMM with skip transitions was determined to be sufficient for this task (Figure 1).

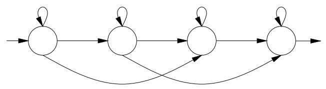

Figure 1: The four state HMM used for recognition.

There are three key problems in HMM use. These are the evaluation, estimation, and the decoding problems. The evaluation problem is that given an observation sequence and a model, what is the probability that the observed sequence was generated by the model ($Pr(\mathbf{O}|\lambda)$) (notational style from [7])? If this can be evaluated for all competing models for an observation sequence, then the model with the highest probability can be chosen for recognition.

$Pr(\mathbf{O}|\lambda)$ can be calculated several ways. The naive way is to sum the probability over all the possible state sequences in a model for the observation sequence:

$$Pr(\mathbf{O}|\lambda) = \sum_{all S} \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(O_t)$$

However, this method is exponential in time, so the more efficient forward-backward algorithm is used in practice. The following algorithm defines the forward variable α and uses it to generate $Pr(\mathbf{O}|\lambda)$ (π are the initial state probabilities, a are the state transition probabilities, and b are the output probabilites).

- $\alpha_1(i) = \pi_i b_i(O_1)$, for all states i (if $i \epsilon S_I, \pi_i = \frac{1}{n_I}$;otherwise $\pi_i = 0$)
- Calculating $\alpha()$ along the time axis, for $t = 2, ..., T$, and all states j, compute
$$\alpha_t(j) = [\sum_i \alpha_{t-1}(i) a_{ij}] b_j(O_t)$$

- Final probability is given by

$$Pr(\mathbf{O}|\lambda) = \sum_{i \epsilon S_F} \alpha_T(i)$$

The first step initializes the forward variable with the initial probability for all states, while the second step inductively steps the forward variable through time. The final step gives the desired result $Pr(\mathbf{O}|\lambda)$, and it can be shown by constructing a lattice of states and transitions through time that the computation is only order $O(N^2T)$. The backward algorithm, using a process similar to the above, can also be used to compute $Pr(\mathbf{O}|\lambda)$ and defines the convenience variable β.

The estimation problem concerns how to adjust λ to maximize $Pr(\mathbf{O}|\lambda)$ given an observation sequence $\mathbf{O}$. Given an initial model, which can have flat probabilities, the forward-backward algorithm allows us to evaluate this probability. All that remains is to find a method to improve the initial model. Unfortunately, an analytical solution is not known, but an iterative technique can be employed.

Using the actual evidence from the training data, a new estimate for the respective output probability can be assigned:

$$\bar{b}_j(k) = \frac{\sum_{t \epsilon O_t = v_k} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

where $\gamma_t(i)$ is defined as the posterior probability of being in state i at time t given the observation sequence and the model. Similarly, the evidence can be used to develop a new estimate of the probability of a state transition ($\bar{a}_{ij}$) and initial state probabilities ($\bar{\pi}_i$).

Thus all the components of model (λ) can be re-estimated. Since either the forward or backward algorithm can be used to evaluate $Pr(\mathbf{O}|\bar{\lambda})$ versus the previous estimation, the above technique can be used iteratively to converge the model to some limit. While the technique described only handles a single observation sequence, it is easy to extend to a set of observation sequences. A more formal discussion can be found in [1, 7, 22].

While the estimation and evaluation processes described above are sufficient for the development of an HMM system, the Viterbi algorithm provides a quick means of evaluating a set of HMM's in practice as well as providing a solution for the decoding problem. In decoding, the goal is to recover the state sequence given an observation sequence. The Viterbi algorithm can be viewed as a special form of the forward-backward algorithm where only the maximum path at each time step is taken instead of all paths. This optimization reduces computational load and additionally allows the recovery of the most likely state sequence. The steps to the Viterbi are

- Initialization. For all states i, $\delta_1(i) = \pi_i b_i(O_1)$; $\psi_i(i) = 0$
- Recursion. From $t = 2$ to T and for all states j, $\delta_t(j) = Max_i[\delta_{t-1}(i)a_{ij}]b_j(O_t)$; $\psi_t(j) = argmax_i[\delta_{t-1}(i)a_{ij}]$
- Termination. $P = Max_{s \epsilon S_F}[\delta_T(s)]$; $s_T = argmax_{s \epsilon S_F}[\delta_T(s)]$
- Recovering the state sequence. From $t = T-1$ to 1, $s_t = \psi_{t+1}(s_{t+1})$

In many HMM system implementations, the Viterbi algorithm is used for evaluation at recognition time. Note that since Viterbi only guarantees the maximum of $Pr(\mathbf{O}, S|\lambda)$ over all state sequences S (as a result of the first order Markov assumption) instead of the *sum* over all possible state sequences, the resultant scores are only an approximation. However, [13] shows that this is often sufficient.

So far the discussion has assumed some sort of quantization of feature vectors into classes. However, instead of using vector quantization, the actual probability densities for the features may be used. Baum-Welch, Viterbi, and the forward-backward algorithms can be modified to handle a variety of characteristic densities [9]. In this context, however, the densities will be assumed to be Gaussian. Specifically,

$$b_j(O_t) = \frac{1}{\sqrt{(2\pi)^n|\sigma_j|}} e^{\frac{1}{2}(O_t-\mu_j)'\sigma_j^{-1}(O_t-\mu_j)}$$

Initial estimations of μ and σ may be calculated by dividing the evidence evenly among the states of the model and calculating the mean and variance in the normal way. Whereas flat densities were used for the initialization step before, the evidence is used here. Now all that is needed is a way to provide new estimates for the output probability. We wish to weight the influence of a particular observation for each state based on the likelihood of that observation occurring in that state. Adapting the solution from the discrete case yields

$$\bar{\mu_j} = \frac{\sum_{t=1}^{T} \gamma_t(j) O_t}{\sum_{t=1}^{T} \gamma_t(j)}$$

and

$$\bar{\sigma_j} = \frac{\sum_{t=1}^{T} \gamma_t(j)(O_t - \bar{\mu_j})(O_t - \bar{\mu_j})^t}{\sum_{t=1}^{T} \gamma_t(j)}$$

For convenience, μ_j is used to calculate $\bar{\sigma_j}$ instead of the re-estimated $\bar{\mu_j}$. While this is not strictly proper, the values are approximately equal in contiguous iterations [7] and seem not to make an empirical difference [22]. Since only one stream of data is being used and only one mixture (Gaussian density) is being assumed, the algorithms above can proceed normally, incorporating these changes for the continuous density case.

5 Recovering Hands from Video

Previous systems have shown that, given some constraints, relatively detailed models of the hand can be recovered from video images [3, 14]. However, many of these constraints conflict with recognizing ASL in a natural context, either by requiring simple, unchanging backgrounds (unlike clothing), not allowing occlusion, requiring carefully labelled gloves, or being difficult to run in real time.

Since real-time recognition is a goal in this project, several compromises were made. The subject wears distinctly colored gloves on each hand (a yellow glove for the right hand and an orange glove for the left) and sits in a chair before the camera. Figure 2 shows the view from the camera's perspective and gives an impression of the quality of video that is used. Color NTSC composite video is captured and analyzed at a constant 5 frames per second at 320 by 243 pixel resolution on a Silicon Graphics Indigo 2 with Galileo video board. To find each hand initially, the algorithm scans the image until it finds a pixel of the appropriate color. Given this pixel as a seed, the region is grown by checking the eight nearest neighbors for the appropriate color. Each pixel checked is considered to be part of the hand. This, in effect, performs a simple morphological dilation upon the resultant image that helps to prevent edge and lighting aberrations. The centroid is calculated as a by-product of the growing step and is stored as the seed for the next frame. Given the resultant bitmap and centroid, second moment analysis is performed as described earlier.

Figure 2: View from the tracking camera.

6 Feature Extraction

Previous experience has shown that it is often best to start simple and evolve a feature set [18]. Since finger spelling is not allowed and there are few ambiguities in the test vocabulary based on individual finger motion, a relatively coarse tracking system may be used. Based on previous work, it was assumed that a system could be designed to separate the hands from the rest of the scene. Traditional vision algorithms could then be applied to the binarized result. Aside from the position of the hands, some concept of the shape of the hand and the angle of the hand relative to horizontal seemed necessary. Thus, an eight element feature vector consisting of each hand's x and y position, angle of axis of least inertia, and eccentricity of bounding ellipse was chosen. The eccentricity of the bounded ellipse was found by determining the ratio of the square roots of the eigenvalues that correspond to the matrix

$$\begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix}$$

where a, b, and c are defined as

$$a = \int\int_{I'} (x')^2 dx'dy'$$

$$b = \int\int_{I'} x'y'dx'dy'$$

$$c = \int\int_{I'} (y')^2 dx'dy'$$

(x' and y' are the x and y coordinates normalized to the centroid)

The axis of least inertia is then determined by the major axis of the bounding ellipse, which corresponds to the primary eigenvector of the matrix [6]. Note that this leaves a 180 degree ambiguity in the angle of the ellipses. To address this problem, the angles were only allowed to range from -90 to +90 degrees.

7 Training an HMM network

When using HMM's to recognize strings of data, such as continuous speech, cursive handwriting, or ASL sentences, several methods can be used to bring context to bear in training and recognition. A simple context modeling method is embedded training. While initial training of the models might rely on manual segmentation or, in this case, evenly dividing the evidence amoung the models, embedded training trains the models *in situ* and allows boundaries to shift through a probabilistic entry into the initial states of each model [22].

Generally, a sign can be affected by both the sign in front of it and the sign behind it. In speech, this is called "co-articulation." While this can confuse systems trying to recognize isolated signs, the context information can be used to aid recognition. For example, if two signs are often seen together, recognizing the two signs as one group may be beneficial.

A final use of context is on the word level. Statistical grammars relating the probability of the co-occurrence of two or more words can be used to weight the recognition process. Grammars that associate two words are called bigrams, whereas grammars that associate three words are called trigrams. Rule-based grammars can also be used to aid recognition.

8 Experimentation

The handtracking system as described earlier worked well. Occasionally tracking would be lost (generating error values of 0) due to lighting effects, but recovery was fast enough (within a frame) so that this was not a problem. A 5 frame/sec rate was maintained within a tolerance of a few milliseconds. However, frames were deleted where tracking of one or both hands was lost. Thus, a constant data rate was not guaranteed.

Of the 500 sentences collected, six had to be thrown out due to subject error or outlier signs. In general, each sign ranged from approximately 1 to 3 seconds in length. No intentional pauses were placed between signs within a sentence, but the sentences themselves were distinct.

Initial estimates for the means and variances of the output probabilities were provided by iteratively using Viterbi alignment on the training data (after initially dividing the evidence equally amoung the words in the sentence) and then recomputing the means and variances by pooling the vectors in each segment. Entropic's Hidden Markov Model ToolKit (HTK) is used as a basis for this step and all other HMM modeling and training tasks. The results from

Table 2: Word accuracy

	on training	*on indep. test set*
grammar	99.5%	99.2%
no gram.	92.0% (97% corr.) (D=9, S=67, I=121, N=2470)	91.3% (97% corr.) (D=1, S=16, I=26, N=495)

the initial alignment program are fed into a Baum-Welch re-estimator, whose estimates are, in turn, refined in embedded training which ignores any initial segmentation. For recognition, HTK's Viterbi recognizer is used both with and without a strong grammar based on the known form of the sentences. The actual recognition step runs at a rate five times faster than real time. Contexts are not used, since a similar effect could be achieved with the strong grammar given this data set.

Word recognition results are shown in Table 2. When testing on training, all 494 sentences were used for both the test and train sets. For the fair test, the sentences were divided into a set of 395 training sentences and a set of 99 independent test sentences. The 99 test sentences were not used for any portion of the training. Given the strong grammar (pronoun, verb, noun, adjective, pronoun), insertion and deletion errors were not possible since the number and class of words allowed is known. Thus, all errors are substitutions when the grammar is used (and accuracy is equivalent to percent correct). However, without the grammar, the recognizer is allowed to match the observation vectors with any number of the 40 vocabulary words in any order. Thus, deletion (D), insertion (I), and substitution (S) errors are possible. The absolute number of errors of each type are listed in Table 2. The accuracy measure is calculated by subtracting the number of insertion errors from the number of correct labels and dividing by the total number of signs. Note that, since all errors are accounted against the accuracy rate, it is possible to get large negative accuracies (and corresponding error rates of over 100%). Most insertion errors occurred at signs with repetitive motion.

9 Analysis and Discussion

While these results are far from being sufficient to claim a "working system" for ASL recognition, they do show that this approach is promising. The high recognition rate on the training data indicates that the HMM topologies are sound and that the models are converging. Even so, the remaining 6.5% error rate on the "no grammar" case (error rates will be based on accuracy measures) indicates that some fine tuning on the feature set and model is in order. The 0.8% error rate on the independent test set shows that the models are generalizing well. However, a close look at the text produced by the recognition process shows some of the limitations of the feature set. Since the raw positions of the hands were used, the system was trained to expect certain gestures in certain locations. When this varied due to subject seating position or arm placement, the system could become confused. A simple fix to this problem would be to use position deltas in the feature vector instead.

Examining the errors made when no grammar was used shows the importance of finger position information. Signs like "pack," "car," and "gray" have very similar motions. In fact, the main difference between "pack" and "car" is that the fingers are pointed down for the former and clenched in the latter. Since this information was not available in the model, confusion could occur. While recovering general and/or specific finger position may be difficult in real time in the current testing area, simple palm orientation could be used for discrimination. In the current system, a simple implementation would be to paint the back of the gloves a different color than the palm.

A more interesting problem with the no grammar results was that signs with repetitive or long gestures were often inserted twice for each actual occurrence. In fact, insertions caused more errors than substitutions. Thus, a sign "shoes" might be recognized as "shoes shoes," which is a viable hypothesis without a language model. However, both problems can be addressed using context training or a statistical or rule-based grammar.

Using context modeling as described before may improve recognition accuracy. While the rule-based grammar explicitly constrained the word order, statistical context modeling would have a similar effect while leaving open the possibility of different sentence structures. In addition, bisine (two sign) and trisine (three sign) contexts would help fine-tune the training on the phrase level. However, trisine modeling would not support the tying of the beginning pronoun to the ending pronoun as does the grammar. If task oriented or domain centered sentences were used instead of randomly generated sentences, context modeling and a statistical grammar would improve performance considerably. For example, the random sentence construction allowed "they like pill gray they" which would have a low probability of occurrence in everyday conversation. As such, context modeling would tend to suppress this sentence in recognition unless strong evidence was given for it.

While extending this recognition system to the full 6000 word ASL lexicon would present many problems, some basic improvements could be made in order to begin adapting the system to the task:

- Use deltas instead of absolute positions. An alternative is to determine some feature on the subject from which the positions can be measured (for example, the centroid of the subject).
- Add finger and palm tracking information. This may be as simple as how many fingers are visible along the contour of the hand and whether the palm is facing up or down.
- Collect appropriate domain or task oriented data and perform context modeling.

These improvements do not address the subject independence issue. Just as in speech, making a system which can understand different subjects with their own variations of the language involves collecting data from many subjects. Until such a system is tried, it is hard to estimate the number of subjects and the amount of data that would comprise a suitable training database. Independent recognition often places new requirements on the feature set as well. While the modifications mentioned above may be sufficient initially, the development process is highly empirical.

So far, finger spelling has been ignored. However, incorporating finger spelling into the recognition system is a very interesting problem. Of course, changing the feature vector to address finger information is vital to the problem, but adjusting the context modeling is also of importance. With finger spelling, a closer parallel can be made to speech recognition. Here, trisine context is at a lower level than grammar modeling and will have more of an effect. A point of inquiry would be switching between the different modes of communication. Can trisine context be used across finger spelling and signing? Is it beneficial to switch to a separate mode for finger spelling recognition? Can natural language techniques be applied, and if so, can they also be used to address the spatial positioning issues in ASL? The answers to these questions may be key in creating an unconstrained sign language recognition system.

10 Conclusion

We have shown an unencumbered way of recognizing American Sign Language (ASL) through the use of a video camera. Through use of hidden Markov models low error rates were achieved on both the training set and an independent test set without invoking complex models of the hands. With a larger training set and context modeling, lower error rates are expected and generalization to a freer, person-independent ASL recognition system should be obtainable.

References

[1] L. Baum. An inequality and associated maximization technique in statistical estimation of probabilistic functions of markov processes. *Inequalities*, 3:1–8, 1972.

[2] C. Charayaphan and A. Marble. Image processing system for interpreting motion in American Sign Language. *Journal of Biomedical Engineering*, 14:419–425, Sept. 1992.

[3] B. Dorner. Hand shape identification and tracking for sign language interpretation. In *IJCAI Workshop on Looking at People*, 1993.

[4] I. Essa, T. Darrell, and A. Pentland. Tracking facial motion. IEEE Workshop on Nonrigid and articulated Motion, Austin TX, Nov 94.

[5] Y. He and A. Kundu. Planar shape classification using hidden markov models. In *Proc. 1991 IEEE Conf. on Comp. Vision and Pat. Rec.*, p. 10–15. IEEE Press, 1991.

[6] B. Horn. *Robot Vision*. MIT Press, NY, 1986.

[7] X. Huang, Y. Ariki, and M. Jack. *Hidden Markov Models for Speech Recognition*. Edinburgh Univ. Press, Edinburgh, 1990.

[8] T. Humphries, C. Padden, and T. O'Rourke. *A Basic Course in American Sign Language*. T. J. Publ., Inc., Silver Spring, MD, 1980.

[9] B. Juang. Maximum likelihood estimation for mixture multivariate observations of markov chains. *AT&T Technical Journal*, 64:1235–1249, 1985.

[10] A. Kundu, Y. He, and P. Bahl. Handwritten word recognition: a hidden markov model based approach. 22: 283–297, 1989.

[11] R. Nag, K. Wong, and F. Fallside. Script recognition using hidden markov models. In *ICASSP 86*, 1986.

[12] H. Poizner, U. Bellugi, and V. Lutes-Driscoll. Perception of american sign language in dynamic point-light displays. 7: 430–440, 1981.

[13] L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, p. 4–16, Jan. 1996.

[14] J. Rehg and T. Kanade. DigitEyes: vision-based human hand tracking. School of Computer Science Technical Report CMU-CS-93-220, Carnegie Mellon Univ., Dec. 1993.

[15] J. Schlenzig, E. Hunter, and R. Jain. Recursive identification of gesture inputers using hidden markov models. In *Proc. of the Second Annual Conf.on Appl. of Comp. Vision*, p. 187–194, 1994.

[16] G. Sperling, M. Landy, Y. Cohen, and M. Pavel. Intelligible encoding of ASL image sequences at extremely low information rates. *Comp. Vision, Graphics, and Image Proc.*, 31:335–391, 1985.

[17] T. Starner. Visual Recognition of American Sign Language Using Hidden Markov Models. Master's thesis, MIT Media Laboratory, Feb. 1995.

[18] T. Starner, J. Makhoul, R. Schwartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *ICASSP*, 1994.

[19] T. Takahashi and F. Kishino. Hand gesture coding based on experiments using a hand gesture interface device. *SIGCHI Bulletin*, 23(2):67–73, 1991.

[20] S. Tamura and S. Kawasaki. Recognition of sign language motion images. volume 21, p. 343–353, 1988.

[21] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequetial images using hidden markov model. In *Proc. 1992 IEEE Conf. on Comp. Vision and Pat. Rec.*, p. 379–385. IEEE Press, 1992.

[22] S. Young. *HTK: Hidden Markov Model Toolkit V1.5*. Cambridge Univ. Eng. Dept. Speech Group and Entropic Research Lab. Inc., Washington DC, Dec. 1993.