A Wearable Computer Based American Sign Language Recognizer

Thad Starner, Joshua Weaver, and Alex Pentland
Room E15-383, The Media Laboratory
Massachusetts Institute of Technology
20 Ames Street, Cambridge MA 02139
thad,joshw,sandy@media.mit.edu

Abstract

Modern wearable computer designs package workstation level performance in systems small enough to be worn as clothing. These machines enable technology to be brought where it is needed the most for the handicapped: everyday mobile environments. This paper describes a research effort to make a wearable computer that can recognize (with the possible goal of translating) sentence level American Sign Language (ASL) using only a baseball cap mounted camera for input. Current accuracy exceeds 97% per word on a 40 word lexicon.

Keywords

sign language recognition, computer vision, wearable computing

1 Introduction

While there are many different types of gestures, the most structured sets belong to the sign languages. In sign language, where each gesture already has assigned meaning, strong rules of context and grammar may be applied to make recognition tractable.

To date, most work on sign language recognition has employed expensive "datagloves" which tether the user to a stationary machine [1] or computer vision systems limited to a calibrated area [2]. In addition, these systems have mostly concentrated on finger spelling, in which the user signs each word with finger and hand positions corresponding to the letters of the alphabet [3]. However, most signing does not involve finger spelling, but instead uses gestures which represent whole words, allowing signed conversations to proceed at or above the pace of spoken conversation.

In this paper, we describe an extensible system which uses one color camera pointed down from the brim of a baseball cap to track the wearer's hands in real time and interpret American Sign Language (ASL) using Hidden Markov Models (HMM's). The computation environment is being prototyped on a SGI Indy; however, the target platform is a self-contained 586 wearable computer with DSP coprocessor. The eventual goal is a system that can translate the wearer's sign language into spoken English. The hand tracking stage of the system does not attempt a fine description of hand shape; studies of human sign readers have shown that such detailed information is not necessary for humans to interpret sign language [4, 5]. Instead, the tracking process produces only a coarse description of hand shape, orientation, and trajectory. The hands are tracked by their color: in the first experiment via solidly colored gloves and in the second, via their natural skin tone. In both cases the resultant shape, orientation, and trajectory information is 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 [6]. 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 [7]. Consequently, sign language recognition seems an ideal machine vision application of HMM technology, offering the benefits of problem scalability, well defined meanings, a predetermined 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's grammar allows more flexibility in word order than English and sometimes uses redundancy for emphasis. Another variant, Signing Exact English (SEE), has more in common with spoken English but is not in widespread use in America. ASL uses approx-

imately 6000 gestures for common words and communicates obscure words or proper nouns through finger spelling.

Conversants in ASL may describe a person, place, or thing and then point to a place in space to store that object temporarily for later reference [5]. For the purposes of this experiment, this aspect of ASL will be ignored. Furthermore, in ASL the eyebrows are raised for a question, relaxed for a statement, and furrowed for a directive. While we have also built systems that track facial features [8], this source of information will not be used to aid recognition in the task addressed here.

While the scope of this work is not to create a user independent, full lexicon system for recognizing ASL, the system should be extensible toward this goal. Another goal is real-time recognition which allows easier experimentation, demonstrates the possibility of a commercial product in the future, and simplifies archiving of test data. "Continuous" sign language recognition of full sentences is necessary to demonstrate the feasibility of recognizing complicated series of gestures. Of course, a low error rate is also a high priority. For this recognition system, sentences of the

Table 1: ASL Test Lexicon

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

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 generated randomly 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 a total lexicon of forty words. The words were chosen by paging through Humphries *et al.* [9] and selecting those which would generate coherent sentences when chosen randomly for each part of speech.

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, immobile systems, and small training and test sets. Research in the area can be divided into image based systems and instrumented glove systems.

Tamura and Kawasaki demonstrate an early image processing system which recognizes 20 Japanese signs based on matching cheremes [10]. Charayaphan and Marble [11] demonstrate a feature set that distinguishes between the 31 isolated ASL signs in their training set (which also acts as the test set). More recently, Cui and Weng [12] have shown an image-based system with 96% accuracy on 28 isolated gestures.

Takahashi and Kishino [1] discuss a user dependent Dataglove-based system that recognizes 34 of the 46 Japanese kana alphabet gestures, isolated in time, using a joint angle and hand orientation coding technique. Murakami and Taguchi [13] describe a similar Dataglove system using recurrent neural networks. However, in this experiment a 42 static-pose finger alphabet is used, and the system achieves up to 98% recognition for trainers of the system and 77% for users not in the training set. This study also demonstrates a separate 10 word gesture lexicon with user dependent accuracies up to 96% in constrained situations. With minimal training, the glove system discussed by Lee and Xu [14] can recognize 14 isolated finger signs using a HMM representation. Messing *et. al.* [15] have shown a neural net based glove system that recognizes isolated fingerspelling with 96.5% accuracy after 30 training samples. Kadous [16] describes an inexpensive glove-based system using instance-based learning which can recognize 95 discrete Auslan (Australian Sign Language) signs with 80% accuracy. However, the most encouraging work with glove-based recognizers comes from Liang and Ouhyoung's recent treatment of Taiwanese Sign language [17]. This HHM-based system recognizes 51 postures, 8 orientations, and 8 motion primitives. When combined, these constituents form a lexicon of 250 words which can be continuously recognized in real-time with 90.5% accuracy.

3 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 accepted by the vision community. An early effort by Yamato *et al.* [18] uses discrete HMM's to recognize image sequences of six different tennis strokes among three subjects. This experiment is significant because it uses a 25x25 pixel quantized subsampled camera image as a feature vector. Even with such low-level information, the model can learn the set of motions and recognize them with respectable accuracy. Darrell and Pentland [19] use dynamic time warping, a technique similar to HMM's, to match the interpolated responses of several learned image templates. Schlenzig *et al.* [20] use hidden Markov models to recognize "hello," "good-bye," and "rotate." While Baum-Welch re-estimation was not implemented, this study shows the continuous gesture recognition capabilities of HMM's by recognizing gesture sequences. Closer to the task of this paper, Wilson and Bobick [21] explore incorporating multiple representations in HMM frameworks, and Campbell *et. al.* [22] use a HMM-based gesture system to recognize 18 T'ai Chi gestures with 98% accuracy.

4 Tracking Hands in Video

Previous systems have shown that, given some constraints, relatively detailed models of the hands can be recovered from video images [3, 23]. 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.

In this project we have tried two methods of hand tracking: one, using solidly-colored cloth gloves (thus simplifing the color segmentation problem), and two, tracking the hands directly without aid of gloves or markings. Figure 1 shows the cap camera mount, and Figure 2 shows the view from the camera's perspective in the no-gloves case.

In both cases color NTSC composite video is captured and analyzed at 320 by 243 pixel resolution on a Silicon Graphics 200MHz Indy workstation at 10 frames per second. When simulating the self-contained wearable computer under development, a wireless transmission system is used to send real-time video to the SGI for processing [24].

In the first method, the subject wears distinctly colored cloth gloves on each hand (a pink glove for the right hand and a blue glove for the left). 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 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 in the following section.

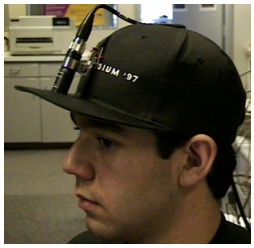

Figure 1: The baseball cap mounted recognition camera.

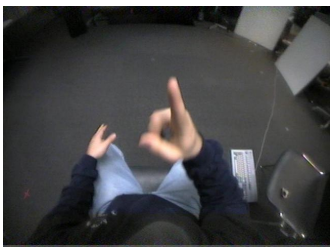

Figure 2: View from the tracking camera.

In the second method, the hands were tracked based on skin tone. We have found that all human hands have approximately the same hue and saturation, and vary primarily in their brightness. Using this information we can build an *a priori* model of skin color and use this model to track the hands much as was done in the gloved case. Since the hands have the same skin tone, "left" and "right" are simply assigned to whichever hand is currently leftmost and rightmost. Processing proceeds normally except for simple rules to handle hand and nose ambiguity described in the next section.

5 Feature Extraction and Hand Ambiguity

Psychophysical studies of human sign readers have shown that detailed information about hand shape is not necessary for humans to interpret sign language

[4, 5]. Consequently, we began by considering only very simple hand shape features, and evolved a more complete feature set as testing progressed [6].

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 seems 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 bounding 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 [25]. 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.

When tracking skin tones, the above analysis helps to model situations of hand ambiguity implicitly. When a hand occludes either the other hand or the nose, color tracking alone can not resolve the ambiguity. Since the nose remains in the same area of the frame, its position can be determined and discounted. However, the hands move rapidly and occlude each other often. When occlusion occurs, the hands appear to the above system as a single blob of larger than normal mass with significantly different moments than either of the two hands in the previous frame. In this implementation, each of the two hands is assigned the moment and position information of the single blob whenever occlusion occurs. While not as informative as tracking each hand separately, this method still retains a surprising amount of discriminating information. The occlusion event is implicitly modeled, and the combined position and moment information are retained. This method, combined with the time context provided by hidden Markov models, is sufficient to distinguish between many different signs where hand occlusion occurs.

6 Training an HMM network

While a substantial body of literature exists on HMM technology [26, 7, 27, 28], this section briefly outlines a traditional discussion of the algorithms. After outlining the fundamental theory in training and testing a discrete HMM, the result is then generalized to the continuous density case used in the experiments and contextual issues are discussed. For broader discussion of the topic, [7, 29] 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 one skip transition was determined to be sufficient for this task (Figure 3).

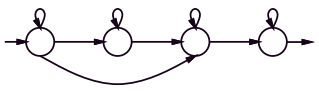

Figure 3: The four state HMM used for recognition.

There are three key problems in HMM use: evaluation, estimation, and decoding. 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:

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

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 [26, 7, 28].

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 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, [27] shows that this is often sufficient.

So far the discussion has assumed some method 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 [30]. 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 [28]. 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.

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. Initial training of the models might rely on manual segmentation. In this case, manual segmentation was avoided by evenly dividing the evidence among the models. Viterbi alignment then refines this approximation by automaticaly comparing signs in the training data to each other and readjusting boundaries until a mimimum variance is reached. Embedded training goes on step further and trains the models *in situ* allowing model boundaries to shift through a probabilistic entry into the initial states of each model [28]. Again, the process is automated. In this manner, a more realistic model can be made of the onset and offset of a particular sign in a natural context.

Generally, a sign can be affected by both the sign in front of it and the sign behind it. For phonemes 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. Such groupings of 2 or 3 units together for recognition has been shown to halve error rates in speech and handwriting recognition [6].

A final use of context is on the inter-word (when recognizing single character signs) or phrase level (when recognizing word signs). Statistical grammars relating the probability of the co-occurrence of two or more words can be used to weight the recognition process. In handwriting, where the units are letters, words, and sentences, a statistical grammar can quarter error rates [6]. In the absence of enough data to form a statistical grammar, rule-based grammars can effectively reduce error rates.

7 Experimentation

Since we could not exactly recreate the signing conditions between the first and second experiments, direct comparison of the gloved and no-glove experiments is impossible. However, a sense of the increase in error due to removal of the gloves can be obtained since the same vocabulary and sentences were used in both experiments.

7.1 Experiment 1: Gloved-hand tracking

The glove-based handtracking system described earlier worked well. In general, a 10 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. This hand tracking process produced an 16 element feature vector (each hand's x and y position, delta change in x and y, area, angle of axis of least inertia - or first eigenvector, length of this eigenvector, and eccentricity of bounding ellipse) that was used for subsequent modeling and recognition. Initial estimates for the means and variances

Table 2: Word accuracy of glove-based system

experiment	*training set*	*independent test set*
grammar	99.4% (99.4%)	97.6% (98%)
no grammar	96.7% (98%) (D=2, S=39, I=42, N=2500)	94.6% (97%) (D=1, S=14, I=12, N=500)

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 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. Contexts are not used, since a similar effect could be achieved with the strong grammar given this data set. Recognition occurs five times faster than real time.

Word recognition accuracy results are shown in Table 2; the percentage of words correctly recognized is shown in parentheses next to the accuracy rates. When testing on training, all 500 sentences were used for both the test and train sets. For the fair test, the sentences were divided into a set of 400 training sentences and a set of 100 independent test sentences. The 100 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 vocabulary substitutions when the grammar is used (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 correspond to signs with repetitive motion.

7.2 Analysis

The 2.4% error rate of the independent test set shows that the HMM topologies are sound and that the models generalize well. The 5.4% error rate (based on accuracy) of the "no grammar" experiment better indicates where problems may occur when extending the system. Without the grammar, signs with repetitive or long gestures were often inserted twice for each actual occurrence. In fact, insertions caused almost as many errors as substitutions. Thus, the sign "shoes" might be recognized as "shoes shoes," which is a viable hypothesis without a language model. However, a practical solution to this problem is the use of context training and a statistical grammar instead of the rule-based grammar.

Using context modeling as described above may significantly improve recognition accuracy in a more general implementation. While a rule-based grammar explicitly constrains the word order, statistical context modeling would have a similar effect while generalizing to allow different sentence structures. In the speech community, such modeling occurs at the "triphone" level, where groups of three phonemes are recognized as one unit. The equivalent in ASL would be to recognize "trisines" (groups of three signs) corresponding to three words, or three letters in the case of finger spelling. Unfortunately, such context models require significant additional training.

In speech recognition, statistics are gathered on word co-occurence to create "bigram" and "trigram" grammars which can be used to weight the liklihood of a word. In ASL, this might be applied on the phrase level. For example, the random sentence construction used in the experiments allowed "they like pill yellow they," which would probably not occur in natural, everyday conversation. As such, context modeling would tend to suppress this sentence in recognition, perhaps preferring "they like food yellow they," except when the evidence is particularly strong for the previous hypothesis.

Unlike our previous study [2] with desk mounted camera, there was little confusion between the signs "pack," "car," and "gray." These signs have very similar motions and are generally distinguished by finger position. The cap-mounted camera seems to have reduced the ambiguity of these signs.

7.3 Experiment 2: Natural skin tracking

The natural hand color tracking method also maintained a 10 frame per second rate at 320x240 pixel resolution on a 200MHz SGI Indy. The word accuracy results are summarized in Table 3; the percentage of words correctly recognized is shown in parentheses next to the accuracy rates.

7.4 Analysis

A higher error rate was expected for the gloveless system, and indeed, this was the case for less

Table 3: Word accuracy of natural skin system

experiment	*training set*	*independent test set*
grammar	99.3% (99%)	97.8% (98%)
no grammar	93.1% (99%) (D=5, S=30, I=138, N=2500)	91.2% (98%) (D=1, S=8, I=35, N=500)

constrained "no grammar" runs. However, the error rates for the strong grammar cases are almost identical. This result was unexpected since, in previous experiments with desktop mounted camera systems [2], gloveless experiments had significantly lower accuracies. The reason for this difference may be in the amount of ambiguity caused by the user's face in the previous experiments' hand tracking whereas, with the cap mounted system, this was not an issue.

The high accuracy rates and types of errors (repeated words) indicate that more complex versions of the experiment can now be addressed. From previous experience, context modeling or statistical grammars could significantly reduce the remaining error in the gloveless no grammar case.

8 Discussion and Conclusion

We have shown an unencumbered, vision-based method of recognizing American Sign Language (ASL). 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.

However, the cap camera mount is probably inappropriate for natural sign. Facial gestures and head motions are common in conversational sign and would cause confounding motion to the hand tracking. Instead a necklace may provide a better mount for determining motion relative to the body. Another possiblity is to place reference points on the body in view of the cap camera. By watching the motion of these reference points, compensation for head motion might be performed on the hand tracking data and the head motion itself might be used as another feature.

Another challenge is porting the recognition software to the self-contained wearable computer platform. The Adjeco ANDI-FG PC/104 digitizer board with 56001 DSP was chosen to perform hand tracking as a parallel process to the main CPU. The tracking information is then to be passed to a Jump 133Mhz 586 CPU module running HTK in Linux. While this CPU appears to be fast enough to perform recognition in real time, it might not be fast enough to synthesize spoken English in parallel (BT's "Laureate" will be used for synthesizing speech). If this proves to be a problem, newly developed 166Mhz Pentium PC/104 boards will replace the current CPU module in the system. The size of the current prototype computer is 5.5" x 5.5" x 2.75" and is carried with its 2 "D" sized lithium batteries in a shoulder satchel. In order to further reduce the obtrusiveness of the system, the project is switching to cameras with a cross-sectional area of 7mm. These cameras are almost unnoticeable when integrated into the cap. The control unit for the camera is the size of a small purse but fits easily in the shoulder satchel. More recently, as part of the Oct. 15, 1997 wearables fashion show at the MIT Media Laboratory, students from Bunka in Tokyo and Domus in Milan demonstrated potential designs for the system that may be socially acceptable to consumers.

With a larger training set and context modeling, lower error rates are expected and generalization to a freer, user independent ASL recognition system should be attainable. To progress toward this goal, the following improvements seem most important:

- Measure hand position relative to a fixed point on the body.
- Add finger and palm tracking information. This may be as simple as counting 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 both on the trisine level as well as the grammar/phrase level.
- Integrate explicit head tracking and facial gestures into the feature set.
- Collect experimental databases of native sign using the apparatus.
- Estimate 3D information based on the motion and aspect of the hands relative to the body.

These improvements do not address the user 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 initially sufficient, 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. Trisine context occurs at the sub-word level while grammar modeling occurs at the word level. However, this is at odds with context across word signs. 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 to creating an unconstrained sign language recognition system.

Acknowledgements

The authors would like to thank Tavenner Hall for her help editing and proofing early copies of this document.

References

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

[2] Starner T. and Pentland A. Real-time American Sign Language recognition from video using hidden markov models. MIT Media Laboratory, Perceptual Computing Group TR#375, Presented at ISCV'95.

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

[4] Poizner H., Bellugi U., and Lutes-Driscoll V. Perception of American Sign Language in dynamic point-light displays. J. Exp. Pyschol.: Human Perform., 7:430–440, 1981.

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

[6] Starner T., Makhoul J., Schwartz R., and Chou G. On-line cursive handwriting recognition using speech recognition methods. ICASSP, V–125, 1994.

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

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

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

[10] Tamura S. and Kawasaki S. Recognition of sign language motion images. Pattern Recognition, 21:343–353, 1988.

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

[12] Cui Y. and Weng J. Learning-based hand sign recognition. Intl. Work. Auto. Face Gest. Recog. (IWAFGR) '95 Proceedings, p. 201–206, 1995

[13] Murakami K. and Taguchi H. Gesture recognition using recurrent neural networks. CHI '91 Conference Proceedings, p. 237–241, 1991.

[14] Lee C. and Xu Y., Online, interactive learning of gestures for human/robot interfaces. IEEE Int. Conf. on Robotics and Automation, pp 2982-2987, 1996.

[15] Messing L., Erenshteyn R., Foulds R., Galuska S., and Stern G. American Sign Language computer recognition: its present and its promise. Conf. the Intl. Society for Augmentative and Alternative Communication, 1994, pp. 289-291.

[16] Kadous W. Recognition of Australian Sign Language using instrumented gloves. Bachelor's thesis, University of New South Wales, October 1995.

[17] Liang R. and Ouhyoung M., A real-time continuous gesture interface for Taiwanese Sign Language. Submitted to UIST, 1997.

[18] Yamato J., Ohya J., and Ishii K. Recognizing human action in time-sequential images using hidden Markov models. Proc. 1992 ICCV, p. 379–385. IEEE Press, 1992.

[19] Darrell T. and Pentland A. Space-time gestures. CVPR, p. 335–340, 1993.

[20] Schlenzig J., Hunter E., and Jain R. Recursive identification of gesture inputers using hidden Markov models. Proc. Second Ann. Conf. on Appl. of Comp. Vision, p. 187–194, 1994.

[21] Wilson A. and Bobick A. Learning visual behavior for gesture analysis. Proc. IEEE Int'l. Symp. on Comp. Vis., Nov. 1995.

[22] Campbell L., Becker D., Azarbayejani A., Bobick A., and Pentland A. Invariant features for 3-D gesture recognition. Intl. Conf. on Face and Gesture Recogn., pp. 157-162, 1996

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

[24] Mann S. Mediated reality. MIT Media Lab, Perceptual Computing Group TR# 260, 1995.

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

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

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

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

[29] Starner T. Visual recognition of American Sign Language using hidden Markov models. Master's thesis, MIT Media Laboratory, Feb. 1995.

[30] Juang B. Maximum likelihood estimation for mixture multivariate observations of Markov chains. AT&T Tech. J., 64:1235–1249, 1985.