# Continuous Recognition of Motion Based Gestures in Sign Language

Daniel Kelly, John Mc Donald, Charles Markham
Computer Science Department
National University of Ireland, Maynooth
dankelly@cs.nuim.ie

## Abstract

*We present a novel and robust system for recognizing two handed motion based gestures performed within continuous sequences of sign language. While recognition of valid sign sequences is an important task in the overall goal of machine recognition of sign language, detection of movement epenthesis is important in the task of continuous recognition of natural sign language. We propose a framework for recognizing valid sign segments and identifying movement epenthesis. Our system utilizes a single HMM threshold model, per hand, to detect movement epenthesis. Further to this, we develop a novel technique to utilize the threshold model and dedicated gesture HMMs to recognize gestures within continuous sign language sentences. Experiments show that our system has a gesture detection ratio of* $0.956$ *and a reliability measure of* $0.932$ *when spotting 8 different signs from 240 video clips.*

## 1. Introduction

Sign language recognition systems are an ideal test environment for motion based gesture recognition algorithms for human computer interfaces (HCI). In practice HCI would involve composition of individual gestures just as sign sentences are compositions of individual signs. One of the main difficulties with recognizing motion based hand gestures is that the hand(s) must move from the end point of the previous gesture to the start point of the next gesture. These inter gesture transition periods are called movement epenthesis [9] and are not part of either of the gestures. While the co-articulation effects that arise between signs can hold useful information in a small number of signs, movement epenthesis occur very frequently between consecutive signs, thus movement epenthesis should be dealt with first [11]. An accurate gesture recognition system should therefore be able to distinguish between valid sign segments and movement epenthesis. This work describes a framework for the recognition of spatiotemporal gestures and identification of movement epenthesis.

### 1.1. Related Work

One proposed solution to movement epenthesis detection is an explicit segmentation model were subsets, of features from gesture data, are used as cues for valid gesture start and end point detection [13, 8]. The limitation of this explicit segmentation model arises from the difficulty in creating general rules for sign boundary detection that could be applied to all types of gestures [11].

An approach to dealing with continuous recognition without explicit segmentation is to use Hidden Markov Models (HMM) for implicit sentence segmentation. Starner et al. [14] and Bauer and Kraiss [2] model each word or subunit with a HMM and then train the HMMs with data collected from full sentences. A downside to this is that training on full sentence data may result in a loss in valid sign recognition accuracy due to the large variations in the appearance of all the possible movement epenthesis that could occur between two signs.

Wang et al. [19] also use HMMs to recognize continuous signs sequences with $92.8\%$ accuracy, although signs were assumed to end when no hand motion occurred. Assan et al. [1] model the HMMs such that all transitions go through a single state, while Gao et al. [4] create separate HMMs that model the transitions between each unique pair of signs that occur in sequence. Vogler at al. [18] also use an explicit epenthesis modeling system where one HMM is trained for every two valid combinations of signs.

While these works have had promising results in gesture recognition and movement epenthesis detection, the training of such systems involves a large amount of extra data collection, model training and recognition computation due to the extra number of HMMs required to detect movement epenthesis.

Few researchers have addressed the problem of movement epenthesis without explicitly modeling these movements. Yang et al [20] proposed an ASL recognition method based on an enhanced Level Building algorithm and a Trigram grammar model. Their method was based on a dynamic programming approach to spot signs without explicit movement epenthesis models. The recognition rate was



2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops
978-1-4244-4441-0/09/$25.00 ©2009 IEEE

83% with 39 signs, articulated in 25 different sentences. Their work is based on a two step approach for the recognition of continuous signs, where the first step recognizes the possible signs in the sentence and the second applies a grammar model to the possible signs. They report only the results obtained after the second step which applies a trigram grammar model to the signs. The sensitivity of the system to the grammar model was shown in the experiments where the recognition rate of the system decreased from 83% to 68% when a the trigram model was replaced by a bigram model. To extend a sign recognition system to a more general gesture recognition framework for HCI, the implementation of a grammar model may be unfeasible. In this work we show that we can effectively perform the first step of recognizing gestures from within sign sentences independent of any grammar rules.

We propose a HMM based gesture recognition framework which accurately spots and classifies motion based gestures, within a continuous sequence of sign language, as one of a number of pre trained gestures as well as calculating the probability that the given gesture sequence is or is not a movement epenthesis. The novelty of our work is that the movement epenthesis detection is carried out by a single parallel HMM, per hand, and requires no extra data collection, training or grammar modeling.

## 2. Feature Extraction

The focus of this work is to develop a continuous gesture spotter and classification technique, availing of features extracted from a video stream. For completeness, we briefly describe the feature tracking techniques used, though we do not consider it to be the novel part of our work. From the definition of a spatiotemporal gesture [15], we must track the position and movement of the hands in order to describe a gesture sequence. We expand on the work a of hand posture recognition system proposed Kelly et al [5] to build a feature extraction system for spatiotemporal gesture recognition. Tracking of the hands is performed by tracking colored gloves using the Mean Shift algorithm [3].

Face and eye positions are also used as gestures cues. Face and eye detection is carried out using a cascade of boosted classifiers working with haar-like features proposed by Viola and Jones [16]. A set of public domain classifiers [10], for the face, left eye and right eye, are used in conjunction with the OpenCV implementation of the haar cascade object detection algorithm. We define the raw features extracted from each image as follows; right hand position $(RH_x, RH_y)$, left hand position $(LH_x, LH_y)$, face position $(FC_x, FC_y)$, face width $(FW)$, left eye position $(LE_x, LE_y)$ and right eye position $(RE_x, RE_y)$.

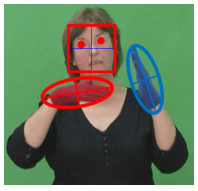


Figure 1. Extracted Features from Image

### 2.1. Feature Processing

A spatiotemporal gesture is defined by the hands' position and movement, where the position refers to the hands' location relative to the body and movement traces out a trajectory in space.

Using the raw features, extracted from the image using the method described in Section 2, the observation vector we use to model a gesture is comprised of a combination of features calculated from the raw features. We carry out performance evaluations on a number different feature combinations in order to find features which best classify spatiotemporal features and movement epenthesis. These evaluations will be discussed in Section 5.

We define $O_t$ as an observation vector made at time $t$, where $O_t = \{o_1, o_2, ..., o_M\}$ and $M$ is the dimension of the feature vector. A particular gesture sequence is then defined as $\Theta = \{O_1, O_2, ..., O_T\}$.

To calculate the probability of a specific observation $O_t$, a probability density function of an M-dimensional multivariate gaussian is implemented (see Equation 1). Where $\mu$ is the mean vector and $\Sigma$ is the covariance matrix.

$$\aleph(O_t|\mu, \Sigma) = (2\pi)^{-\frac{N}{2}} |\Sigma|^{-\frac{1}{2}} exp\left(-\frac{1}{2}(O_t-\mu)^T \Sigma^{-1}(O_t-\mu)\right) \quad (1)$$

## 3. Hidden Markov Models

Hidden Markov Models (HMMs) are a type of statistical model and can model spatiotemporal information in a natural way. HMMs have efficient algorithms for learning and recognition, such as the Baum-Welch algorithm and Viterbi search algorithm [12].

A HMM is a collection of states connected by transitions. Each transition (or time step) has a pair of probabilities: a transition probability (the probability of taking a particular transition to a particular state) and an output probability (the probability of emitting a particular output symbol from a given state).

We use the compact notation $\lambda = \{A, B, \pi\}$ to indicate the complete parameter set of the model where $A$ is a matrix storing transitions probabilities and $a_{ij}$ denotes the probability of making a transition between states $s_i$ and $s_j$. $B$ is a matrix storing output probabilities for each state and $\pi$ is a vector storing initial state probabilities.

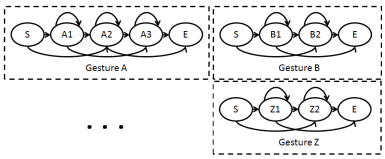
HMMs can use either a set of discrete observation symbols or they can be extended for continuous observations signals. In this work we use continuous multidimensional observation probabilities calculated from a multivariate probability density function.

### 3.1. HMM Threshold Model

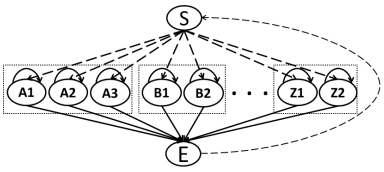
Lee and Kim [7] proposed a HMM threshold model to handle non-gesture patterns. The threshold model was implemented to calculate the likelihood threshold of an input pattern and provide a confirmation mechanism for provisionally matched gesture patterns. We build on the work carried out by Lee and Kim to create a framework for calculating a probability distribution of a two hand input sign using continuous multidimensional observations. The computed probability distribution will include probability estimates for each pre-trained sign as well as a probability estimate that the input sign is a movement epenthesis.

In general, a HMM recognition system will choose a model with the best likelihood as the recognized gesture if the likelihood is higher than a predefined threshold. However, this simple likelihood threshold often does not work, thus, Lee and Kim proposed a dynamic threshold model to define the threshold of a given gesture sequence.

A property of the left-right HMM model implies that a self transition of a state represents a particular segment of a target gesture and the outgoing state transition represents a sequential progression of the segments within a gesture sequence. With this property in mind, an ergodic model, with the states copied from all gesture models in the system, can be constructed as shown in Figure 2(a) and 2(b). Figure 2(b) shows the threshold model as a simplified version of the ergodic model where dotted lines denote null transitions (i.e. no observations occur between transitions).




(a)




(b)

Figure 2. (a) Dedicated Gesture Models (b) Threshold Model

States are copied such that output observation probabilities and self transition probabilities are kept the same, but all outgoing transition probabilities are equally assigned as $a_{ij} = \frac{1-a_{ii}}{N-1}\ \forall j, i \neq j$ where N is the number of states excluding the start and end states (The start and end states produce no observations).

As each state represents a subpattern of a pre-trained gesture, constructing the threshold model as an ergodic structure makes it match well with all patterns generated by combining any of the gesture sub-patterns in any order. The likelihood of the threshold model, given a valid gesture pattern, would be smaller than that of the dedicated gesture model because of the reduced outgoing transition probabilities. However, the likelihood of the threshold model, given an arbitrary combination of gesture sub-patterns, would be higher than that of any of the gesture models, thus the threshold model, denoted as $\overline{\lambda}$, can be used as a movement epenthesis likelihood measure.

## 4. System Overview

Our system initializes and trains a dedicated parallel HMM [17] for each gesture to be recognized. Each parallel HMM consists of two separate HMMs that model the right and left hand gesture respectively. A description of the models observations, training and recognition process is presented in the following sections.

### 4.1. Model Training

Each dedicated gesture model is trained on isolated signs performed by a fluent signer. Before training a HMM using the Baum-Welch algorithm, the model must first be initialized. Initialization includes the computation of an initial state transition matrix and calculation of each states' emission variables $\mu$ and $\Sigma$. In order to initialize these components of the HMM, an understanding of the gesture segmentation, or state transitions, must be built. One approach to achieving this would be to explicitly hand label different subunits or gesture phonemes [19]. Part of the goal of this work is to create a general data collection, training and recognition system. Data collection consists of a recording step and a labeling step. Labeling is an integral step in creating valid sign data, thus we envisage that all data will be labeled by fluent signers. Since movement and position of the hands are two of the four building blocks of sign language which Stokoe [15] identified, manually breaking these building blocks into smaller subunits would be an unintuitive and time consuming step for fluent signers to segment in a consistent manner. With this in mind, a training system was developed to initialize and train data with minimum human intervention where signs are labeled at a sign level and not at a phoneme level.

We implement an automated HMM initialization and training model in our system. We extend an iterative HMM training model proposed by Kim at al [6] to develop a HMM initialization and training model which includes an extra parameter selection layer. The parameter selection layer finds the best combination of $(S, R)$, where $S$ is the total number of states in the HMM and $R$ is the reach of a state (i.e. in a left-right model, the reach is the number of states that it is

possible to transition to from the current state).

For a particular sign, we collect data from a number of video sequences of a fluent signer performing that sign. This produces a set of observation sequences $\Delta_c = \{\Theta_c^1, \Theta_c^2, ..., \Theta_c^K\}$ where $c$ is the index of the sign being modeled and $K$ is the total number of training examples. To initialize $\lambda_c$, the HMM which will model the sign indexed by $c$, we first choose a random gesture sequence $\Theta_c^r$ from $\Delta_c$ and calculate $S-1$ indices of $\Theta_c^r$ which best segment the gesture into $S$ sub-gestures. The $S-1$ indices are calculated by performing principal component analysis on the gesture sequence, performing a k-means clustering technique on the principal components and finally finding the $S-1$ indices which best divide the data into their corresponding k-means clusters.

The gesture data is then broken into the $S$ subsets and the mean vector $\mu$ and the covariance matrix $\Sigma$ is calculated for each state. The Baum-Welch algorithm[12] is then applied to $\lambda_c$ using all training data $\Delta_c$. After training, the Viterbi algorithm[12] is run on $\Theta_c^r$ to produce most probable state sequence. The initial $S$ sub-gestures are then realigned to match the Viterbi path. This re-estimation and realignment process is continued until the likelihood, produced by the Baum-Welch algorithm, converges. The overall process is repeated for different combinations of $(S, R)$ to find the combination which produces the highest likelihood from the Baum-Welch re-estimation. Figure 4.1 gives an overview of the iterative training and parameter selection procedure.

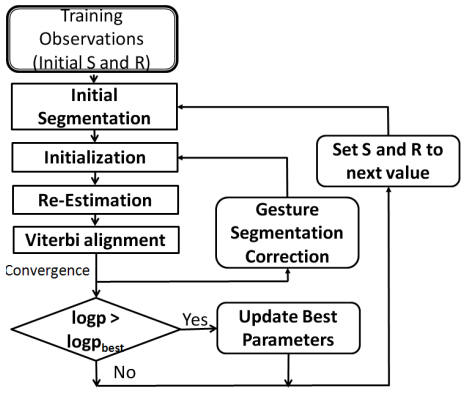



Figure 3. HMM Initialization and Training Procedure

It is desirable to weight $\lambda_{Lc}$ and $\lambda_{Rc}$, the left hand HMM and right hand HMM respectively, due to variations in information held in each of the hands for a particular sign. The weighting applied in our system is based on a variance measure of the observation sequences. Using data from all observation sequences $\Theta_{Lc}^k$ and $\Theta_{Rc}^k$, where $1 \leq k \leq K$, $K$ is the total number of training examples and $\Theta_{Lc}$ and $\Theta_{Rc}$ are the left and right hand observations respectively. The variance of the left and right hand observations are calculated by calculating the variance of each observation dimension $\sigma_{Lc}^2[i]$ and $\sigma_{Rc}^2[i]$, where $0 \leq i \leq D$ and $D$ is the dimension of the observation vectors. The left HMM weight, $\omega_{Lc}$, and right HMM weight, $\omega_{Rc}$, are then calculated as using Equations 2 and 3.

$$\omega_{Lc} = \sum_{i=0}^{D} \frac{\sigma_{Lc}^2[i]}{(\sigma_{Lc}^2[i] + \sigma_{Rc}^2[i]) \times D} \quad (2)$$

$$\omega_{Rc} = \sum_{i=0}^{D} \frac{\sigma_{Rc}^2[i]}{(\sigma_{Lc}^2[i] + \sigma_{Rc}^2[i]) \times D} \quad (3)$$

### 4.2. Sign Recognition

Given an unknown sequence of sign observations $\Theta_L$ and $\Theta_R$, the goal is to accurately classify the sign as either a movement epenthesis or as one of the $C$ trained signs. To classify the observations, the Viterbi algorithm is run on each model given the unknown observation sequences $\Theta_L$ and $\Theta_R$, calculating the most likely state paths through each model $c$. The likelihoods of each state path, which we denote as $P(\Theta|\lambda_{Lc})$ and $P(\Theta|\lambda_{Rc})$, are also calculated.

We calculate the overall likelihoods of a dedicated gesture and a movement epenthesis with the equations defined in Equations 4 and 5.

$$P(\Theta|\lambda_c) = P(\Theta_L|\lambda_{Lc})\omega_{Lc} + P(\Theta_R|\lambda_{Rc})\omega_{Rc} \quad (4)$$

$$\Psi_c = \frac{P(\Theta_L|\overline{\lambda_L})\Gamma_{Lc} + P(\Theta_R|\overline{\lambda_R})\Gamma_{Rc}}{2} \quad (5)$$

Where $\Gamma_{Lc}$ and $\Gamma_{Rc}$ are constant scalar values used to tune the sensitivity of the system to movement epenthesis.

The sequence of observations can then be classified as $c$ if $P(\Theta|\lambda_c) \geq \Psi_c$ evaluates to be true.

## 5. Isolated Experiments

We perform a set of experiments on isolated gestures to evaluate the best performing feature vector and evaluate the HMM threshold model system when compared to a standard HMM model system. To evaluate the performance of our recognition framework, a set of eight different signs, as performed by a fluent signer, were recorded and manually labeled. The set of eight test signs were not selected to be visually distinct but to represent a suitable cross section of the spatiotemporal signs that can occur in sign language. A visual example of a signer performing each of the eight signs is shown in Figure 4.

A set of observation sequences $\Delta_c$ were extracted from the video sequence (where $1 \leq c \leq C$) and divided into a training set, $\Delta_c^\tau$, and a test set, $\Delta_c^\zeta$. For the experiments

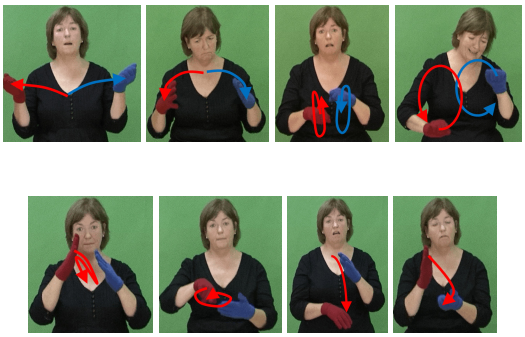


Figure 4. Example of the eight different signs the system was tested on (a) Newspaper, (b) A lot, (c) Bike, (d) Clean, (e) Paint, (f) Plate, (g) Lost, (h) Gone

we report in this paper, a set of 5 training signs and a set of 5 test signs were recorded for each sign. Each dedicated gesture model $\lambda_c$ was then trained on $\Delta_c^\tau$ using our training procedure described in Section 4.1. The threshold models were then created using the trained gesture models. An additional set of observations $\Delta_E$, which represent a collection of movement epenthesis, were also extracted from the video sequences to test the performance of the threshold model. For each valid sign, we recorded 10 movement epenthesis that occurred before and after the valid sign in different sign language sentences. An additional set of 20 random movement epenthesis were also recorded, resulting in a test set of 100 samples to evaluate the system on.

The classification of a gesture is based on a comparison of a weighted threshold model likelihood with the weight denoted as $\Gamma_c$. In our ROC analysis of the system, we vary the weight, $\Gamma_c$, over the range $0 \leq \Gamma_c \leq 1$ and then create a confusion matrix for each of the weights. This procedure is carried out for both the left hand weights, $\Gamma_{Lc}$, and the right hand weights, $\Gamma_{Rc}$. To evaluate the performance of different features, we performed a ROC analysis on the models generated from the different feature combinations and calculated the area under the curve (AUC) for each feature vector model as shown in Table 1.

It can be seen from the AUC measurements shown in Table 1 that the best performing feature, with an AUC of $0.949$, was the feature, $F_7 = \{RP_x, RP_y, V_x, V_y, D_H\}$, which describes the position of the hands relative to the eyes, the direction of the movement of the hand and the distance between the two hands. To evaluate the performance of the threshold model, when applied to multi dimensional sign language observations, we compare the performance of our system to a modified version of our system with no threshold model. The modified version of the system uses the same dedicated HMMs but the sequence of observations is classified as $c$ only if the gesture likelihood is greater than a predefined static threshold. A ROC analysis of the modified systems classifications showed that the best performing

Table 1. AUC Measurements for Different Feature Combinations

| **Features** | **ROC AUC** |
|---|---|
| $F_1$ - Hand Direction $(V_x, V_y)$ | 0.8614 |
| $F_2$ - Hand Direction $(V_x, V_y)$+ Distance Between Hands $(D_H)$ | 0.698 |
| $F_3$ - Hand Direction $(V_x, V_y)$+ Distance Between Eyes and Hand $(D_E)$ | 0.7391 |
| $F_4$ - Hand Positions Relative to Eyes $(RP_x, RP_y)$ | 0.789 |
| $F_5$ - Hand Positions Relative to Eyes $(RP_x, RP_y)$ + Distance Between hands $(D_H)$ | 0.936 |
| $F_6$ - Hand Positions Relative to Eyes $(RP_x, RP_y)$ + Hand Direction $(V_x, V_y)$ | 0.807 |
| **$F_7$ - Hand Positions Relative to Eyes $(RP_x, RP_y)$ + Hand Direction $(V_x, V_y)$ + Distance Between hands $(D_H)$** | **0.949** |

feature was also the feature $F_7$. The AUC of the ROC graph produced by this feature was 0.897. From the experiments we have carried out, the performance of the system with the threshold model was $5.2\%$ better than that of the system without the threshold model.

## 6. Continuous Recognition

Thus far we have described a framework for classifying a given observation sequence as one of a number of pre trained gestures or as a movement epenthesis. We perform experiments to show the robustness of this framework for recognizing isolated gestures with a ROC area under the curve measurement of $0.949$. We will now describe our system for spotting and classifying spatiotemporal gestures within continuous sequences of natural sign language.

The first step in our spotting algorithm is gesture end point detection. To detect a gesture end point in a continuous stream of gesture observations $\Theta = \{O_1, O_2, ..., O_T\}$, we calculate the model likelihoods of observation sequence $\theta = \{O_{T-L}, O_{T-L-1}, ..., O_T\}$ where $\theta$ is a subset of $\Theta$ and $L$ defines the length of the observation subset used. In the work we report we set $L$ to the average length of the observation sequences used to train the system.

A candidate gesture, $\kappa$, with end point, $\kappa_e = T$, is flagged when $\exists c : P(\Theta|\lambda_c) \geq \Psi_c$. Figure 5 illustrates the likelihood time evolution of the gesture model "Lost" when given an observation sequence where the signer performs the "Lost" sign. It can be seen from Figure 5 that a number of candidate end points occur between $T = 16$ and $T = 21$.

For each candidate end point we calculate a corresponding start point $\kappa_s$. Different candidate start points are eval-

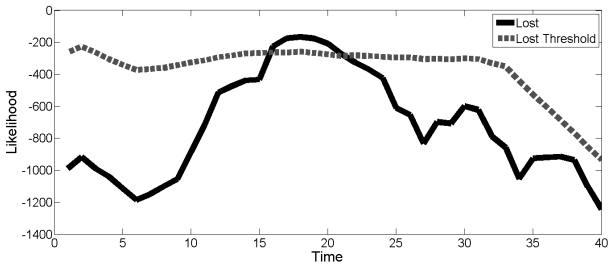


Figure 5. Likelihood evolution of "Lost" gesture model and associated threshold model

uated using the measurement shown in Equation 6 where $\Phi_c(\Theta)$ is normalized metric (between 0 and 1) which measures the strength of gesture $c$ given observations $\Theta$.

$$\Phi_c(\Theta) = \frac{P(\Theta|\lambda_c)}{P(\Theta|\lambda_c)+\Psi_c} \quad (6)$$

To find a candidate start point, the metric $\Phi_c(\Theta_{s\kappa_e})$ is calculated over different values of $s$, where $\Theta_{s\kappa_e} = \{O_s, O_{s+1}, ..., O_{\kappa_e}\}$ and $(\kappa_e - L^2) \leq s < \kappa_e$. The candidate gesture start point $\kappa_s$, is then found using Equation 7.

$$\kappa_s = \underset{s}{\text{argmax}} \Phi_c(\Theta_{s\kappa_e}) \quad (7)$$

The start and end point detection algorithm may flag candidate gestures which overlap and for this reason we expand on our continuous sign recognition algorithm with a candidate selection algorithm. The purpose of the candidate selection algorithm is to remove overlapping candidate gestures such that the single most likely gesture is the only remaining gesture for a particular time frame.

We will use a sample sign language sentence "I Lost Book" to illustrate our candidate selection algorithm in the context of our gesture and threshold likelihood evaluation, where the system was trained on the following 8 signs; "Paper", "Alot", "Bike", "Clean", "Paint", "Plate", "Lost" and "Gone". Figure 6 illustrates the difference between the gesture model likelihood $P(\Theta|\lambda_c)$ and its corresponding threshold $\Psi_c$, where positive values indicates $P(\Theta|\lambda_c) \geq \Psi_c$. We illustrate only 4 gesture model likelihoods as all other gesture model likelihoods never exceed their corresponding threshold.

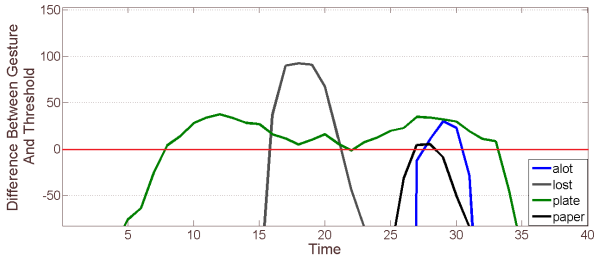


Figure 6. Gesture And Corresponding Threshold Model Likelihood Difference

The first step in the candidate selection algorithm is to cluster overlapping gestures, with the same gesture classification, together. Each of these candidate gestures, within the cluster, have an associated metric $\kappa_p = \Phi_c(\Theta_{\kappa_s\kappa_e})$. We remove all but one candidate gesture from this cluster leaving only the candidate gesture, $\kappa^B$, with the highest $\kappa_p$ value. We repeat this step for each cluster to produce a set of candidate gestures $\Upsilon = \{\kappa^{B1}, \kappa^{B2}, ..., \kappa^{BK}\}$, where $K$ is the total number of clusters created from grouping overlapping gestures, with the same gesture classification, together. Figure 7 shows the time segments and $\Phi$ metrics of each candidate gesture after the first candidate selection step.

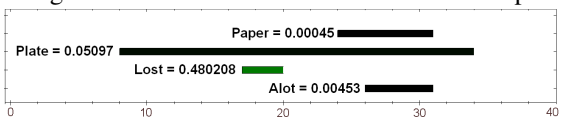


Figure 7. Candidate Gestures, $\Upsilon$, after first candidate selection step

The second step in the candidate selection algorithm is an iterative selection step to remove the least probable candidate gestures as shown in Algorithm 1.

**Input**: Set of Candidate Gestures $\Upsilon$
**Output**: Set of Recognized Gestures
Sort($\Upsilon$) by In Order of Increasing $\kappa_P^B$
**for** $i \leq K$ **do**
  **if** $\exists j \in J = \{i+1, i+2, ..., K\}$, *such that* $\Upsilon[j]$ *overlaps with* $\Upsilon[i]$ **then**
    Remove $\Upsilon[i]$ from $\Upsilon$;
  **end**
**end**

**Algorithm 1**: Second Step of Candidate Selection Algorithm

Figure 8 shows the time segments and $\Phi$ metrics of the recognized gestures after the second candidate selection step where the sign "Lost" is correctly recognized.

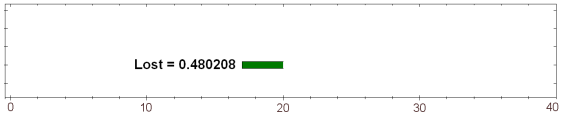


Figure 8. Recognized Gestures, $\Upsilon$, after final candidate selection step

## 7. Continuous Experiments

To evaluate the performance of our recognition framework, we use the same set of eight signs used in the isolated experiments in Section 5. A total of 240 video clips (30 videos per sign) of sign language sentences being performed by a fluent signer were recorded. Each video clip contained at least one of the eight chosen signs and was recorded at 25 frames per second with an average length of 7 seconds. In order to robustly evaluate the performance

of our system, each of the 240 different sign language sentences, used to test and train the system, was performed in a mixture of different styles. The variations in the style of signs performed are similar to the types of variations that can occur in sign language in real world situations and thus testing our system on these signs gives a good indication of how our system will perform in real world scenarios. Although we have not currently performed experiments on multiple signers, the types of style variations in our data does represent some of the variations which occurs between the styles of different signers. An example of the sign variations introduced to our data is as follows: the sign "A lot" was performed within the sentence "My children eat A LOT of sweets" and different variations of this sentence were performed in the following ways; different question styles (yes/no questions, different "wh" questions) and different emotional styles (a state of surprise, wonder, anger, shock and joy). The start and end points of the eight different gestures, within the full sign language sentences, were then labeled by a certified sign language interpreter. According to the labels in the full sentences, isolated observation sequences $\Delta_c$ were extracted from the video sequences (where $1 \leq c \leq C$) to form a training set, $\Delta_c^\tau$. For the experiments we report in this paper, a set of 10 training signs were used for each sign. Each dedicated gesture model $\lambda_c$ was then trained on $\Delta_c^\tau$ using the automated training procedure described in Section 4.1. The threshold models were then created using states from the trained gesture models. The remaining 160 video clips, that were not part of the training procedure, were used to evaluate the performance of our continuous sign spotting and classification techniques. Observation sequences were extracted from each video clip and our continuous recognition algorithm, described in Section 6, processed the observation sequences to spot and classify signs within the videos.

In the gesture spotting and classification task, there are three types of errors: The insertion error occurs when the spotter reports a nonexistent gesture, the deletion error occurs when the spotter fails to detect a gesture, and the substitution error occurs when the spotter falsely classifies a gesture. From these error measures we define two performance metrics there are two performance metrics used which we define in Equation 8, where $CS$ is the number of correctly spotted gestures, $IG$ is the number of input gestures and $IE$ is the number of insertion errors.

$$DetectionRatio = \frac{CS}{IG} \quad Reliability = \frac{CS}{IG + IE} \quad (8)$$

Table 2 shows the performance of our system when spotting and classifying signs within continuous sequences of video. The experiment shows an overall detection rate of 95.6% and an overall reliability of 93.2%. We perform a secondary experiment to evaluate the performance of the start and end point detection relative to a human sign language translator. Table 2 shows the average absolute difference between the spotters start and end points and the human interpreters start and end points for signs that were correctly spotted and classified. The overall start point error was only 7.8 frames and the overall end point error was only 8.15 frames. From this experiment we can conclude that our spotter is capable of detecting start points, within an average of 312 milliseconds of a human interpreter, and end points, within an average of 326 milliseconds of a human interpreter.

Table 2. Continuous Spotter and Classifier Performance

| Sign | #Correct | $\#D^\dagger$ | $\#I^\ddagger$ | $\#S^{\dagger\dagger}$ | $Det^*$ | $Rel'$ | Start Error | End Error |
|---|---|---|---|---|---|---|---|---|
| Gone | 20 | 0 | 0 | 0 | 1.0 | 1.0 | 2.5 | 8.4 |
| Alot | 20 | 0 | 0 | 0 | 1.0 | 1.0 | 1.5 | 1.6 |
| Lost | 20 | 0 | 0 | 0 | 1.0 | 1.0 | 1.5 | 3.5 |
| Plate | 19 | 0 | 1 | 0 | 0.95 | 0.90 | 8.1 | 12.2 |
| Bike | 20 | 0 | 0 | 0 | 1.0 | 1.0 | 12.1 | 12.0 |
| Paint | 20 | 0 | 0 | 0 | 1.0 | 1.0 | 26.1 | 20.7 |
| Paper | 16 | 0 | 1 | 3 | 0.8 | 0.76 | 5.9 | 1.6 |
| Clean | 18 | 0 | 1 | 1 | 0.9 | 0.85 | 4.8 | 5.2 |
| **Total** | **153** | **0** | **3** | **4** | **0.956** | **0.932** | **7.8** | **8.15** |

† *Number of Deletion Errors*, ‡*Number of Insertion Errors*
††*Number of Substitution Errors*, ∗*Detection Ratio*, *'Reliability*

The system described in this work was developed as a robust software application that can be used on any standard Microsoft Windows based PC. The vision feature extraction component of the system was built using C++ and utilizes functions within OpenCV library. The gesture recognition engine was developed using C#. Supplementary material is provided in the form of a video to demonstrate the software application being used to recognize some continuous signs. The system we have proposed runs at near realtime speeds. Timing evaluations, which were performed during our experiments, showed that performing a full classification of a video clip took, on average, 1.32 times the length of the video (i.e. for a 7 second video clip, it would take our system 9.24 seconds to do a full classification of the signs in the video).

## 8. Conclusion

In this paper we have discussed current methods of continuous gesture recognition. The downside of these methods is that unnatural constraints are put on the signer, such as pauses between words, or the explicit training of models to handle movement epenthesis must be carried out. The method we have proposed in this work requires only that the dedicated gesture models be trained, and as a result of this training a single epenthesis model can be created.

The novelty of this system is that we have expanded on the work of Lee and Kim [7] to develop a HMM threshold model system which models continuous multidimensional gesture observations within a parallel HMM network to recognize two hand gesture and identify movement epenthe-

sis. Further to this, we have also developed a robust framework to utilize the results of our HMM threshold model system to spot and classify signs within continuous natural sign language sentences. A ROC analysis of the isolated gesture classification performance showed that the three dimensional feature vector $F_7$, defined in Table 1, was the best performing feature with an AUC measurement of $0.949$. The threshold model system showed a $5.2\%$ increase in performance when compared to a static threshold based system. Experiments carried out on continuous sentences show that our system had a detection ratio of $0.956$ and a reliability measure of $0.932$. Experiments also showed that the gesture spotter was able to flag gesture start points and end points within 312 milliseconds and 326 milliseconds respectively when compared to a human interpreter. Recognition of continuous sign language sentences without an explicit modeling of movement epenthesis is a difficult task and few researchers have dealt with this problem [20]. The contribution of this paper is that we have developed a robust pattern recognition framework for the recognition of motion based gestures and identification of movement epenthesis without the explicit modeling of these movement epenthesis. Although our framework was evaluated on sign language data, the system we propose is extendable to more general motion based gesture recognition for HCI.

### 8.1. Future Work

In this work we have proposed a spatiotemporal recognition framework, a next step in the overall goal of machine recognition of sign language is to integrate hand posture and non-manual information into the recognition process.

## 9. Acknowledgements

The Authors would like to acknowledge the financial support of the Irish Research Council for Science, Engineering and Technology (IRCSET).

## References

[1] M. Assan and K. Grobel. Video-based sign language recognition using hidden markov models. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 97–109, London, UK, 1998. Springer-Verlag.

[2] B. Bauer and K.-F. Kraiss. Towards an automatic sign language recognition system using subunits. In *GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, pages 64–75, London, UK, 2002. Springer-Verlag.

[3] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, 2:142–149 vol.2, 2000.

[4] W. Gao, G. Fang, D. Zhao, and Y. Chen. Transition movement models for large vocabulary continuous sign language recognition. *IEEE FG 2004*, pages 553–558, May 2004.

[5] D. Kelly, J. McDonald, T. Lysaght, and C. Markham. Analysis of sign language gestures using size functions and principal component analysis. In *IMVIP 2008*, 2008.

[6] Y.-J. Kim and A. Conkie. Automatic segmentation combining an hmm-based approach and spectral boundary correction. In *In ICSLP-2002*, pages 145–148, 2002.

[7] H. K. Lee and J. H. Kim. An hmm-based threshold model approach for gesture recognition. *IEEE PAMI*, 21(10):961–973, 1999.

[8] R. H. Liang and M. Ouhyoung. A real-time continuous gesture recognition system for sign language. In *IEEE FG 1998*, page 558, Washington, DC, USA, 1998. IEEE Computer Society.

[9] J. R. Liddell, S.K. American sign language: The phonological base. *Sign Langauge Studies*, 64.

[10] L. A.-C. M. Castrillon-Santana, O. Deniz-Suarez and J. Lorenzo-Navarro. Performance evaluation of public domain haar detectors for face and facial feature detection. *VISAPP 2008*, 2008.

[11] S. C. W. Ong and S. Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(6):873–891, 2005.

[12] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[13] H. Sagawa and M. Takeuchi. A method for recognizing a sequence of sign language words represented in a japanese sign language sentence. In *IEEE FG 2000*, page 434, Washington, DC, USA, 2000. IEEE Computer Society.

[14] T. Starner, A. Pentland, and J. Weaver. Real-time american sign language recognition using desk and wearable computer based video. *IEEE PAMI*, 20(12):1371–1375, 1998.

[15] J. Stokoe, William C. Sign language structure: An outline of the visual communication systems of the american deaf. *Journal of Deaf Studies and Deaf Education, v10 n1 p3-37 Win 2005*, 2005.

[16] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *CVPR, IEEE*, 1:511, 2001.

[17] C. Vogler and D. Metaxas. Parallel hidden markov models for american sign language recognition. In *In ICCV*, pages 116–122, 1999.

[18] C. Vogler and D. Metaxas. A framework for recognizing the simultaneous aspects of american sign language. *Computer Vision and Image Understanding*, 81:358–384, 2001.

[19] C. Wang, S. Shan, and W. Gao. An approach based on phonemes to large vocabulary chinese sign language recognition. In *IEEE FG 2002*, page 411, Washington, DC, USA, 2002. IEEE Computer Society.

[20] R. Yang, S. Sarkar, and B. Loeding. Enhanced level building algorithm for the movement epenthesis problem in sign language recognition. pages 1–8, 2007.