Fingerspelling Alphabet Recognition Using A Two-level Hidden Markov Model

S. Lu, J. Picone, and S. Kong
Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA, USA

Abstract - *Fingerspelling is widely used for communication amongst signers. Signer-independent (SI) recognition of the American Sign Language alphabet is a very challenging task due to factors such as the large number of similar gestures, hand orientation and cluttered background. We propose a novel framework that uses a two-level hidden Markov model (HMM) that can recognize each gesture as a sequence of sub-units and performs integrated segmentation and recognition. Features based on the Histogram of Oriented Gradient (HOG) method are used. The focus of this work is optimization of this configuration with respect to two parameters: the number of sub-units and the number of states in the HMMs. Moreover, we evaluated the system on signer-dependent (SD) and signer-independent (SI) tasks for the ASL Fingerspelling Dataset and achieved error rates of 2.0% and 46.8% respectively. The SI results improved the best previously published results by 18.2% absolute (28.0% relative).*

Keywords: American Sign Language, Fingerspelling, Hand Gesture Recognition, HOG, Hidden Markov Models

1 Introduction

Developing automated recognition of American Sign Language (ASL) is important since ASL is the primary mode of communication for most deaf people. In North America alone it is estimated that as many as *500,000* people use ASL as their primary language for communication [1]. ASL consists of approximately *6,000* words with unique signs. Additional words are spelled using fingerspelling [2] of alphabet signs. In a typical communication, *10%* to *15%* of the words are signed by fingerspelling of alphabet signs. Similar to written English, the one-handed Latin alphabet in ASL consists of *26* hand gestures. The objective of this paper is to design a recognition system that can classify *24* ASL alphabet signs from a static 2D image. We excluded *"J"* and *"Z"* because they are dynamic hand gestures. Recognition of dynamic gestures is the subject of future research that is a straightforward extension of the work presented here.

A popular approach to ASL gesture recognition is to use sensory gloves [3][4]. Advanced sensors, such as Microsoft's Kinect, have become popular in recent years because they provide alternate information such as depth that can be very useful for gesture recognition applications. Such systems typically achieve better performance than a simple 2D camera but are often costly, intrusive and/or inconvenient. Therefore, the focus of our work is to only use intensity information, which can be collected from any single 2D camera.

Feris et al. [5] used a multi-flash image capture system, and achieved a signer-dependent (SD) recognition error rate of 4% on a dataset consisting of 72 images from a single signer. Pugeault et al. [6] developed an approach that used Gabor features and random forests and evaluated it on a dataset that consisted of 5 sets of signs from 4 signers, 24 gestures per subject, and over 500 samples per gesture for each signer (a total of over 60,000 images). This dataset is known as the ASL Fingerspelling (ASL-FS) Dataset, and is the basis for the work presented here. Their best recognition error rate was 25%, achieved by combining both color and depth information. It is important to note, however, that this error rate was achieved using a closed-loop evaluation – the evaluation dataset contained the same five signers as the training set.

Munib et al. [2] proposed a fingerspelling recognition system based on a Hough transform and neural networks. On a dataset consisting of 15 signers and 20 signs per signer, they achieved an error rate of 20%. Vieriu et al. [7] developed a static hand gesture recognition system using the angle distribution of a hand contour and a simple left to right hidden Markov model (HMM), and achieved an error rate of 3.8% for an SD test. However, the database they used contained only 9 hand gestures with simple backgrounds.

Our proposed method, shown in Figure 1, is based on HMMs also, delivers an error rate that is 18.2% lower than [6] on the same data, and represents the best published results on the ASL dataset for both SD and signer-independent (SI) evaluations. Because the dataset is relatively small, we used a cross-validation approach to measure SI performance. Our system features an unsupervised training mode that only requires a label for the overall image. Detailed transcriptions of subsections of the images, such as finger segments, are not required.

Scale invariance and environment adaptation are desired properties for any practical fingerspelling recognition system. Typical solutions include normalization and/or a priori segmentation. Normalization often causes distortion of the original data and also changes the geometric structure of the hand gesture. A priori segmentation is sub-optimal and often fails in the presence of noisy backgrounds such as complex

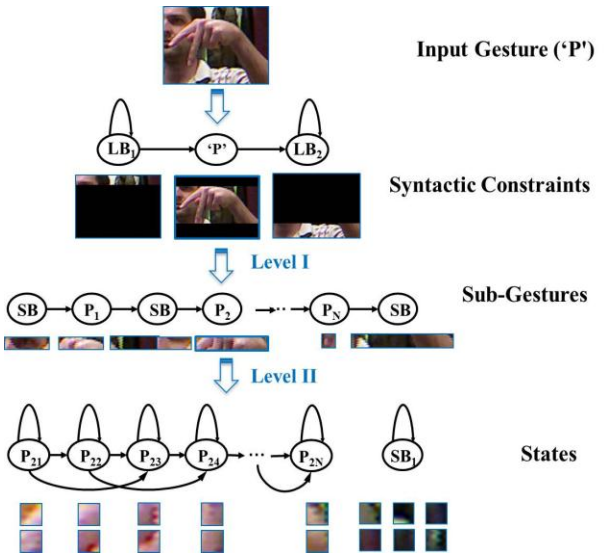

Figure 1: *A framework for fingerspelling that uses a two-level HMM architecture is shown.*

clutter. An HMM-based approach is attractive because it integrates the segmentation and recognition process, jointly optimizing both. However, the time-synchronous nature of the Markov model must be adjusted to deal with the 2D data presented by an image.

2 A two-level HMM architecture

A traditional HMM architecture typically involves three steps: (1) feature extraction, (2) parameter estimation via an iterative training process, and (3) recognition based on a Viterbi-style decoding [8]. In the architecture shown in Figure 1, each image is segmented into rectangular regions of $N \times N$ pixels, and an overlapping analysis window of $M \times M$ pixels is used to compute features. The image is scanned from left-to-right, top-to-bottom to facilitate real-time processing, as shown in Figure 2. The images in ASL-FS vary in size from *60x90* pixels to *170x130* pixels with the majority of the images approximately *80x100* pixels in dimension.

We ran an extensive series of baseline experiments on a subset of ASL-FS to optimize the frame and window sizes.

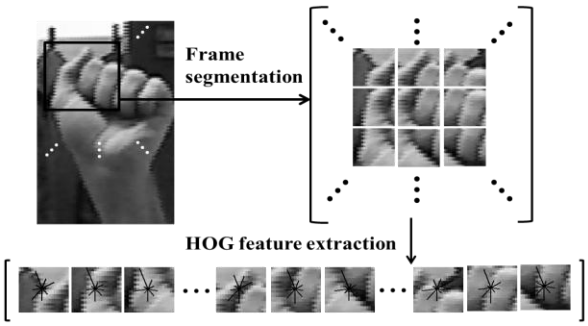

Figure 2: *A frame-based analysis was used to extract 9-dimensional HOG features.*

Selected results from these experiments are shown in Table 1. The best result was obtained with a combination of a frame size of *5* pixels and a window size of *30* pixels (referred to as *5/30*). The optimal number of states and model topology are related to these parameters. The *5/30* combination represents a good tradeoff between computational complexity, model complexity and recognition performance.

We selected Histogram of Oriented Gradient (HOG) features to use in our experiments [9][10] due to their popularity in imaging applications. To calculate HOG features, the image gradient magnitude, g, and angle, θ, for each pixel (u,v) are first calculated using a 1D filter and as shown below:

$$g(u,v) = \sqrt{g_x(u,v)^2 + g_y(u,v)^2} \quad (1)$$

$$\theta(u,v) = \arctan \frac{g_{y(u,v)}}{g_x(u,v)}. \quad (2)$$

For each frame, we compute a feature vector, f_i by quantizing the signed orientation into N orientation bins weighted by the gradient magnitude as defined by:

$$f_i = [\, f_i(n)]_{n\in[1,2,\ldots,N]}^{T} \quad (3)$$

$$f_i(n) = \sum_{(u,v)\in F_i} g(u,v)\delta[bin(u,v) - n]. \quad (4)$$

The function $bin(u,v)$ returns the index of the bin associated with the pixel (u,v). Parameter tuning experiments similar to those in Table 1 indicated that 9 bins were optimal, which is consistent with [10]. Since we used an overlapping analysis, we set each block to have only one cell when extracting HOG features.

These features were then used as input to the two-level HMM architecture shown in Figure 1. The top-level of the system applies syntactic constraints. This level serves two purposes. First, it restricts the system to output one hypothesis per image, which is consistent with the task definition – one sign is contained in each image. Any of the 24 signs can be output by the center node in the top-level HMM denoted "syntactic constraints" in Figure 1. This constraint avoids

Table 1. *A series of experiments were performed to optimize the value of the frame and window sizes.*

Frame (N)	Window (M)	% Overlap	% Error
5	20	75%	7.1%
5	30	83%	4.4%
10	20	50%	5.1%
10	30	67%	5.0%
10	60	83%	8.0%

insertion errors by preventing multiple hypotheses per image.

Second, it implicitly performs segmentation of the image by forcing each frame of the image to be classified as either background, which we refer to as long background (LB), or gesture. LB can only occur immediately preceding or following a hypothesis of an alphabet sign. Images are modeled as having an arbitrary number of frames classified as background, followed by one alphabet sign, followed by an arbitrary number of frames again classified as background. Hence, this level implements a coarse segmentation of the image as part of the recognition process.

The second level, labeled "Sub-Gestures" in Figure 1, models each alphabet sign as a left-to-right HMM that alternates between a model denoted short background (SB) followed by a model corresponding to a sub-gesture (SG). This level models each sign as a sequence of sub-gestures. The optimal number of SG models for each sign will be addressed in the next section. The SB model allows for sections of the image between sub-gestures to be modeled as background (e.g., the space between two vertically raised fingers). The function of this level is to decompose each sign into a unique sequence of sub-gestures. As we will see in the next section, training of these models is performed in a completely unsupervised manner.

Each sub-gesture is implemented as a traditional left-to-right HMM with skip states. This model is known as a Bakis model [8] and has been used extensively in other HMM applications [11][12]. Performance is not extremely sensitive to the topology of this model, though in pilot experiments we obtained a slight decrease in error rate using skip states. The main virtue of the skip state model is that it allows the system to map fewer frames of data to the model, thereby reducing the minimum number of states required by the model.

Unlike many systems that recognize gestures as a whole, we model each alphabet sign with a sequence that typically consists of LB, SB and a sequence of SG models, as shown in Figure 3. The reason we have two different types of background models is that SB is more focused on small background regions within fingers or at image boundaries, while the LB model is used for modeling large background areas preceding or following the sign. The SB model is a single-state HMM with a self-transition. The LB model is 11 states and also allows each state to self-transition.

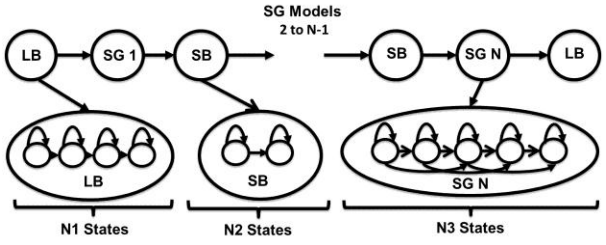

Figure 3: *Our HMM architecture uses a two-level approach that models an image as a combination of LB, SB and sub-gesture (SG) models.*

It is common practice in other applications to tie the SB model to the middle state of the LB model [13], which typically consists of only three states. It is not a wise choice in our case because tying the SB model to the center state of the 11-state LB model increases the chance that long stretches of a hand image will be mapped to LB. Therefore, LB and SB are trained as separate models in our experiments.

3 Unsupervised training

Unlike mature applications of HMMs in fields such as speech recognition, we do not have a universal underlying language for sub-gestures. The core units that compose an image such as an alphabet sign are not guided by something as structured as human language. Hence, it is highly desirable to let the system self-organize these units in a data-driven manner. Further, though supervised training is extremely effective, eliminating the need for transcribed training data significantly reduces the cost and development time of a pattern recognition system.

Therefore, we have implemented an unsupervised training process in which the LB, SB and SG models are learned without the need for any manual transcriptions other than the identity of the alphabet sign represented by the image. Each sample uses a generic transcript is "Start-LB, SG1, SB, SG2, ... , SGN, End-LB." Here, Start-LB refers to an LB model that must occur at the beginning of an image file. Both Start-LB and End-LB are tied, meaning they share the same HMM model.

Our training process follows procedures used extensively in speech recognition applications [11], and is based on the hidden Markov model toolkit, HTK [13]. We initialize our HMMs using a flat-start process in which all models are set to use a single Gaussian distribution with a mean and variance set to the global mean and variance computed across the whole training data [13]. The transition probabilities are manually initialized to favor a transition to the next state. For models with skip states, we initialize the self-transition, transition to the next state and the skip transition probabilities to be 0.1, 0.8 and 0.1, respectively. If there is no skip state, the self-transition and next state transition probabilities are set to 0.1 and 0.9.

We use the Expectation Maximization algorithm (EM), or more specifically, the Baum Welch (BW) algorithm, for HMM training. During the first three iterations of parameter reestimation, only LB and SG are trained so that we find a coarse segmentation of the data. Both the observation and transition probabilities are updated simultaneously within this procedure. Next, we add the SB model by introducing the transcriptions shown in Figure 3 and perform four iterations of reestimation. All HMMs in the system are updated simultaneously at each iteration.

Once we have a set of stable set of models that use a single Gaussian mixture model, we use a mixture-splitting approach to generate mixture models. At each stage, the algorithm repeatedly splits the mixture with the largest mixture weight into two mixture components [13]. This process is repeated until the required number of mixture components is obtained. We start with one mixture, and successively double the number of mixture components until we reach 16 mixture components. Four iterations of BW reestimation are run at each step of the process (e.g., after splitting 8 mixtures into 16 mixtures).

It is common practice in these types of HMM systems to use twice the number of mixture components for the LB and SB models as used in the SG models [14] to accommodate variation in background images. However, for ASL-FS, doubling the number of mixture components for LB and SB did not improve performance. Nevertheless, we followed this convention in our work here.

The decoding algorithm used for recognition is based on a Viterbi decoder [13] that utilizes a beam search algorithm. Pruning thresholds are set very high so that the decoder essentially only performs a dynamic programming-based pruning. The syntactic constraints are imposed using a nonprobabilistic model that constrains the output to one gesture per image and performs a forced-choice decision. Decoding one image typically requires *0.22* secs and *0.29* Mbytes of memory on a *2.67* GHz Pentium processor and is roughly $O(N_f \times N_s^2)$ in complexity, where N_f is the number of frames and N_s is the total number of states.

4 Experiments

Extensive tuning experiments were conducted to optimize the parameters of the system. Based on past experiences with other applications [14], we optimized these parameters on the SD task, for which performance was generally fairly high and relatively sensitive to changes in these parameter settings. Our experiments focused on optimizing the number of SG segments, states and mixture components. Since the data consists of at least 500 tokens per sign, the data for each sign for the SD task was partitioned into 10 randomly selected subsets. A cross-validation approach was used in which 9 subsets were selected for training and the remaining one was used for evaluation. We then rotated the sets so that each of the 10 partitions was used exactly once for evaluation.

The optimized parameter settings are shown in Table 2. The experiments for optimization of the frame size, window size and number of HOG bins were summarized in Section 2. The number of segments in each SG model was varied from 5 to 13, resulting in error rates that ranged from 10.9% to 9.9%. Optimal performance of 9.9% error was obtained with 11 sub-gesture segments.

Table 2. *A summary of the optimal parameter values for our two-level HMM system is shown.*

System Parameter	Value
Frame Size (pixels)	5
Window Size (pixels)	30
No. HOG Bins	9
No. Sub-gesture Segments	11
No. States Per Sub-gesture Model	21
No. States Long Background (LB)	11
No. States Short Background (SB)	1
No. Gaussian Mixtures (*SG* models)	16
No. Gaussian Mixtures (*LB/SB* models)	32

Once the number of segments was fixed, we explored the optimal number of states for each sub-gesture model under the constraint that all models should have the same number of states (this is a reasonable approach based on previous experience [13]). We also varied the number of states for LB (from 3 to 21) and SB (from 1 to 9). The optimal number of states for SB, LB and SG were found to be 1, 11 and 21, respectively. Note that the SG models include skip states, so a sub-gesture model with 21 states requires a minimum of 10 frames to be hypothesized. An SB model of one state is fairly common in HMM systems. Its self-transition probability is 0.42, which means the average duration of an SB model is 1.7 frames. Similarly, the LB model's transition probability structure indicates that the average duration of LB model is 22.3 frames, which is slightly larger than the width of an image in frames.

We explore optimization of the number of Gaussian mixtures for each state in Table 3. A significant reduction in error rate was achieved, and this is consistent with what we have seen in other applications. The mixture components for the SG models, not surprisingly, tend to model consistent variations in the data, such as lighting conditions. The mixture components for the LB model are close in weight, which implies the background was random and inconsistent.

Once the final system configuration was completed, we performed two sanity checks. First, we varied a number of parameters simultaneously to ensure that the final operating point was optimal. Since there is always a chance that parameter settings are coupled, we wanted to make sure our sequential optimization approach to optimization found at the

Table 3. *A summary of performance as function of the number of mixture components for the SG models.*

No. Mixtures	Error Rate (%)
1	9.9
2	6.8
4	4.4
8	2.9
16	2.0

very least a local optimum. Fortunately, we were able to verify that the operating point summarized in Table 2 was indeed optimal for this dataset.

Second, we examined how performance varied as a function of the cross-validation set. Error rates for each set are shown in Figure 4 (labeled Set 1 to Set 5) as a function of the index of the cross-validation subset (for the 10 randomly chosen subsets for each set). We note that Set 4 always had a higher than average error rate. This is due to a larger degree of hand rotation as shown in Figure 5. The average error rate was 2.0%, which represents the best published result on this task using only color information.

To remain consistent with previously published results, we also evaluated the system on a closed-loop test in which half of the data for a signer was used for training, while the remaining half was used for evaluation [6]. We refer this is as the "Shared" set in Table 4. The HMM system using only color information performs better than a color-only system and a system that uses both color and depth information.

To perform true open set evaluation, and yet maintain reasonable statistical significance, we also used a leave-one-out approach in which all but one of the signers was used for training and the remaining signer used for testing. We then rotated the set so that each signer was used exactly once as a test set. We refer to this as the "SI" set in Table 4. Our SI results with color only exceed the performance of the color+depth system reported in [6] and represent the best-published results on an SI task for the ASL-FS dataset.

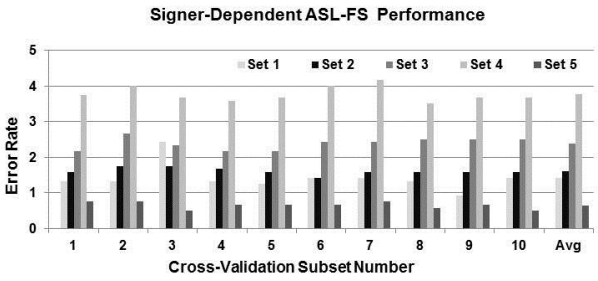

Figure 4: *SD results are shown as a function of the cross-validation set.*

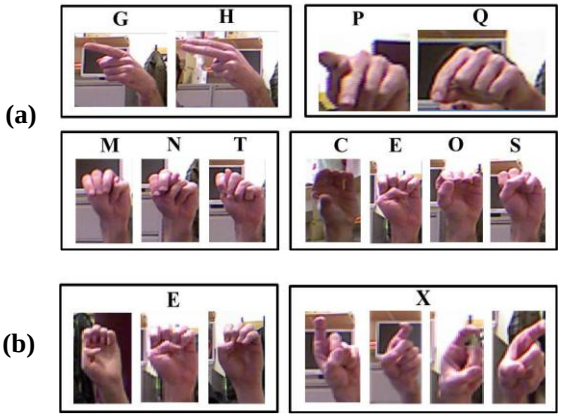

Figure 5: *Gestures with a high confusion error rate are shown in (a). Images with significant variations in background and hand rotation are shown in (b).*

Table 4. *Results for various training paradigms are shown for the ASL-FS dataset. Three tasks are included: signer-dependent (SD), multi-signer training in which the same subjects were used for training and evaluation (Shared), and signer-independent (SI).*

System	SD	Shared	SI
Pugeault (Color Only)	N/A	27.0%	65.0%
Pugeault (Color + Depth)	N/A	25.0%	53.0%
HMM (Color Only)	2.0%	7.8%	46.8%

An analysis of the common error modalities indicated that a major contributor to the error rate was confusions between similar gestures. Examples of this are shown in Figure 5(a). Another factor that results in a large portion of the errors is signer variations, such as angle rotations and scale changes, as shown in Figure 5(b). These have a significant influence on the overall error rate and require more sophisticated sub-gesture models or rotation-independent features.

Finally, a major goal in this work was a system that could accurately segment the data inside the recognition loop. Segmentation is crucial to high performance recognition and must be done in an integrated fashion to be robust to background and lighting variations. We examined segmentations of some typical images as shown in Figure 6. Frames marked as LB are shown in yellow, are generally recognized well because they are constrained to occur at the beginning and end of an input image. However, the SB model, which is used to model small regions of background images, is not always correctly used. Lighting and changes in the background color seem to cause the SB model to incorrectly identify hand images. Additional work on segmentation is clearly needed.

5 Conclusions

We have presented a two-level HMM-based ASL fingerspelling alphabet recognition system that trains gesture and background noise models automatically. Five essential parameters were tuned by cross-validation. Our best system configuration achieved a 2.0% error rate on an SD task, and 46.8% error on an SI task. Both represent the best published results on this data. An analysis of the confusion data suggests that gestures that are visually similar are, in fact, contributing most to the error rate. Hand rotations and scale changes are also challenges that need to be addressed.

As reliable detection by the SB model is crucial to a high performance system, we are currently developing new architectures that perform improved segmentation. Both

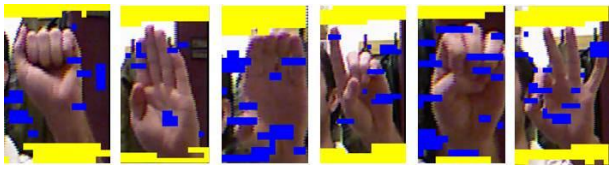

Figure 6: *Examples of segmentation results.*

supervised and unsupervised methods will be employed. Once that architecture definition is complete, we believe we will have a very stable platform from which we can experiment with better features and statistical models.

Finally, all scripts, models, and data related to these experiments are available from our project web site: http://www.isip.piconepress.com/projects/asl_fs. The ASL-FS data is publicly available from [6].

6 Acknowledgements

This research was supported in part by the National Science Foundation through Major Research Instrumentation Grant No. CNS-09-58854. We also wish to thank Nicolas Pugeault and his colleagues for developing and releasing the ASL-FS Database.

7 References

[1] K. Li, K. Lothrop, E. Gill, and S. Lau, "A Web-Based Sign Language Translator Using 3D Video Processing," in Proceedings of the International Conference on Network-Based Information Systems, 2011, pp. 356–361.

[2] Q. Munib, M. Habeeb, B. Takruri, and H. Al-Malik, "American Sign Language (ASL) Recognition Based on Hough Transform and Neural Networks," Expert Systems with Applications, vol. 32, no. 1, pp. 24–37, Jan. 2007.

[3] J. M. Allen, P. K. Asselin, and R. Foulds, "American Sign Language Finger Spelling Recognition System," in Proceedings of the IEEE Bioengineering Conference, 2003, pp. 285–286.

[4] C. Oz and M. Leu, "American Sign Language Word Recognition with a Sensory Glove Using Artificial Neural Networks," Engineering Applications of Artificial Intelligence, vol. 24, no. 7, pp. 1204–1213, Oct. 2011.

[5] R. Feris, M. Turk, R. Raskar, K. Tan, and G. Ohashi, "Exploiting Depth Discontinuities for Vision-Based Fingerspelling Recognition," in IEEE Workshop on Real-time Vision for Human-Computer Interaction, 2004, pp. 155–162.

[6] N. Pugeault and R. Bowden, "Spelling It Out: Real-time ASL Fingerspelling Recognition," in Proceedings of the IEEE International Conference on Computer Vision Workshops, 2011, pp. 1114–1119 (available at http://info.ee.surrey.ac.uk/Personal/N.Pugeault/index.php?section=FingerSpellingDataset).

[7] R.-L. Vieriu, B. Goras, and L. Goras, "On HMM Static Hand Gesture Recognition," in Proceedings of International Symposium on Signals, Circuits and Systems, 2011, pp. 1–4.

[8] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. of the IEEE, vol. 77, no. 2, pp. 257–286, 1989.

[9] M. B. Kaaniche and F. Bremond, "Tracking HOG Descriptors for Gesture Recognition," in Proceedings of the Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, 2009, pp. 140–145.

[10] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, vol. 1, pp. 886–893.

[11] J. Picone, "Continuous Speech Recognition Using Hidden Markov Models," IEEE ASSP Magazine, vol. 7, no. 3, pp. 26–41, Jul. 1990.

[12] M. Zimmermann and H. Bunke, "Hidden Markov Model Length Optimization for Handwriting Recognition Systems," in Proceedings of the International Workshop on Frontiers in Handwriting Recognition, 2001, pp. 369–374.

[13] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollagson, D. Povey, V. Valtchev, and P. Woodland, "The HTK Book," Cambridge, UK, 2006.

[14] I. Alphonso and J. Picone, "Network Training For Continuous Speech Recognition," in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 2004.

[15] J. Matoušek, D. Tihelka, and J. Psutka, "Automatic Segmentation for Czech Concatenative Speech Synthesis Using Statistical Approach with Boundary-specific Correction," in proceedings of the European Conference on Speech Communication and Technology, 2003, pp. 301–304.

[16] Y. Kim and A. Conkie, "Automatic Segmentation Combining an HMM-based Approach and Spectral Boundary Correction," in Proceedings of the International Conference on Spoken Language Processing, 2002, pp. 145–148.

[17] V. Khachaturyan, "Handwritten Signature Verification Using Hidden Markov Models," in Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition, 2012, pp. 347–350.