

# EVALUATION OF HMM TRAINING ALGORITHMS FOR LETTER HAND GESTURE RECOGNITION

Nianjun Liu Brian C. Lovell Peter J. Kootsookos

Intelligent Real-Time Imaging and Sensing (IRIS) Group  
School of Information Technology and Electrical Engineering  
The University of Queensland, Brisbane, Australia 4072  
Email: {nianjunl, lovell, kootsoop}@itee.uq.edu.au

**Abstract:** The paper introduces an application using computer vision for letter hand gesture recognition. A digital camera records a video stream of hand gestures. The hand is automatically segmented, the position of the hand centroid is calculated in each frame, and a trajectory of the hand is determined. After smoothing the trajectory, a sequence of angles of motion along the trajectory is calculated and quantized to form a discrete observation sequence. Hidden Markov Models (HMMs) are used to recognize the letters. Baum Welch and Viterbi Path Counting algorithms are applied for training the HMMs. Our system recognizes all 26 letters from A to Z and the database contains 30 example videos of each letter gesture. We achieve an average recognition rate of about 90 percent. A motivation for the development of this system is to provide an alternate text input mechanism for camera enabled handheld devices, such as video mobile phones and PDAs.

## 1. INTRODUCTION

Human-machine interfaces are playing a role of growing importance as computer technology continues to evolve. Keyboards have been replaced by handwriting recognition in Palm and Pocket PC PDAs [3]. Motivated by the desire to provide users with an intuitive gesture input system, we developed a system (Figure 1) to recognize hand gestures representing letters of the alphabet. Previous attempts to develop hand gesture recognition systems [1] employed geometric feature-based methods, template-based methods, and more recently active contour and active statistic models. Given the success of HMMs in speech and handwriting recognition, we use a variety HMM topologies to perform this gesture recognition role in our system [4].

## 2. SYSTEM OVERVIEW

The hand gesture is captured in a video stream. Each video sequence is composed of 25 frames. Skin colour segmentation based on a YUV colour space is applied to locate the hand. Pre-processing (morphological) operations is used to smooth the image and remove noise before tracking the hand with a modified Camshift algorithm [6]. After segmenting the hand, we calculate image moments to find the hand centroid in each frame. Typical extracted trajectories of the letters Z and W are shown in figure 2. Along the each trajectory, the orientation of each of the 25 hand movements is computed and quantized to one of 18

discrete symbols. This discrete observation sequence is input to a Hidden Markov Model classification module for training and testing. We evaluated the traditional Baum-Welch [7] and the Viterbi Path Counting algorithm (based on the Best-First method [8]) to train HMMs over a range of topologies from Fully Connected to Left-Right with the number of states varying from 4 to 10. Most letters are recognized with better than 90 percent accuracy. The best average percentage correct attained over all letters is 90 percent.

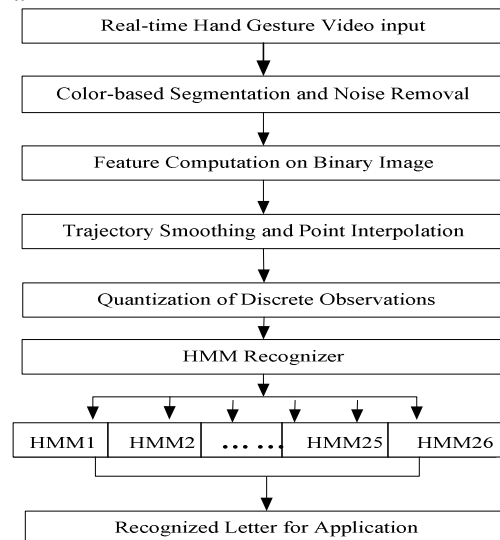


Figure 1 System Flow Chart.

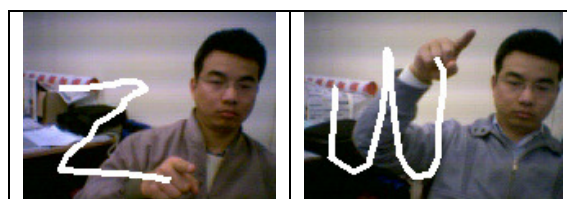


Figure 2 Hand gestures for letters Z and W.

## 3. HAND SEGMENTATION AND TRAJECTORY

Skin-colour based segmentation has proven to be an effective method for segmenting the hand in fairly unrestricted environment. It has been demonstrated [5], that regardless of ethnicity, the chrominance of skin is quite consistent. The major difference between skin tones is intensity, as dark-skinned people have greater skin saturation than light-skinned people. The YUV colour system is employed for separating chrominance and

intensity. The symbol  $Y$  denotes the intensity, while the UV components specify chrominance components. Human skin colours are distributed over a very small region in the chrominance plane. Therefore, a lookup table is employed to classify each pixel. If the pixel colour lies in the normal range of skin colour, it is mapped to a binary value one, and zero otherwise. This is done after the colour image has been mapped into a binary image of ones representing skin and zeros for non-skin regions.

During colour segmentation, it is common to return values that are close to skin but non-skin, or other skin-like regions that are not part of the body. These erroneous values are generally isolated pixels or group of pixels that are dramatically smaller than the total connected hand region in the binary image. We apply morphological opening (erosion followed by dilation) and closing (dilation followed by erosion) operations to remove the spurious components without drastically changing the shape of main objects (such as the hand and face). Following this, the Matlab<sup>®</sup> *imfill* function is used to fill the holes inside the hand. The next step is to automatically locate the hand region and remove all other skin-like parts.

An improved Continuously Adaptive Mean-shift algorithm (Camshift) [6] is applied to determine a region of interest surrounding the hand in each successive frame in order to track the hand and reduce the region for calculation. After calculating and smoothing the binary image in the search window, we sort the skin and skin-like regions by area. Because the hand is always the largest connected region, we only keep the largest one. Figure 3 shows the result of skin colour-based segmentation and the hand centroid location.

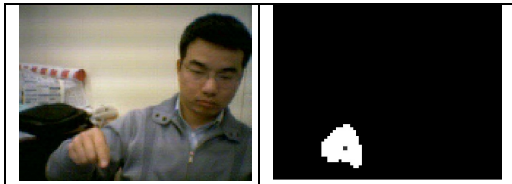


Figure 3 Hand Segmentation and centre location.

After obtaining the binary image, moments [6] are used to calculate the features of the hand. The center and size of the hand is calculated by the zeroth and first moments. In the following equations,  $I(x,y)$  is the pixel value at the position  $(x,y)$  of the image.

The zero<sup>th</sup> and 1<sup>st</sup> moment for  $x$  and  $y$  is defined by:

$$M_{00} = \sum_x \sum_y I(x,y) \quad M_{10} = \sum_x \sum_y xI(x,y); \quad M_{01} = \sum_x \sum_y yI(x,y) \quad (1)$$

The centroid is then found by:

$$x_c = \frac{M_{10}}{M_{00}}; \quad y_c = \frac{M_{01}}{M_{00}} \quad (2)$$

Figure 4 gives the flowchart of skin colour based segmentation, tracking and trajectory estimation.

After finding the coordinates of the hand centroid in each frame, we link them to form the initial trajectory estimate of the hand (Figure 5 first row). The initial trajectory is quite noisy. The major noise sources are due to 1) points too close together, 2) isolated points far away from the correct trajectory, 3) unclosed end points, and 4) curves which are not smooth.

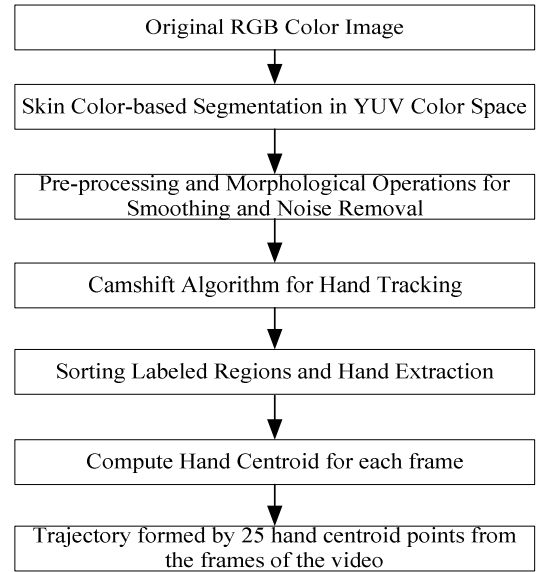


Figure 4 Hand Segmentation and Initial trajectory estimation

We designed the following procedure to smooth the trajectory curve and remove the noise.

- 1) Smooth the trajectory with light threshold to connect the nearby points below the threshold and remove unnecessary points.
- 2) Remove isolated noisy points with thresholding and resmooth.
- 3) Calculate and compare successive line segment orientation changes at the initial points. If differences are large, remove the noisy points at the start of the curve.
- 4) Use the interpolation method to insert points during the trajectory, recover the point number to 25 and distribute the points evenly.
- 5) Detect and connect the correct endpoints, remove redundant points and interpolate.

After performing the above steps, the smoothed trajectories are shown on the 2<sup>nd</sup> row of figure 5.

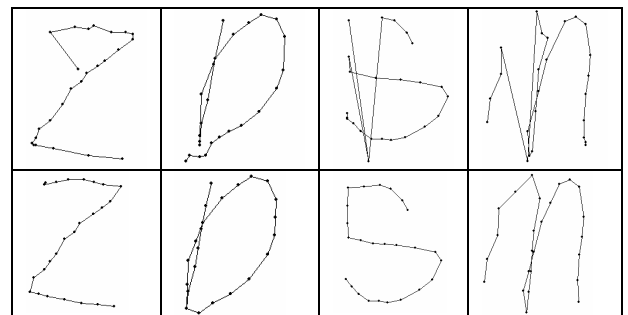


Figure 5 Trajectory smoothing

#### 4. QUANTIZATION FOR DISCRETE OUTPUT

Vector Quantization is widely used in applications such as image and voice compression, voice recognition, and statistical pattern recognition [2]. In our system, it provides a good way of reducing our set of continuous features to a single discrete representation suitable for the discrete HMM.

After getting the smoothed trajectory with 25 points, we calculated the orientation between consecutive points (Figure 6-1) to obtain a sequence of angles. The angle's

domain is [0,360] degrees. We divide this angle by 20 to quantize them to 18 directional codewords from 1 to 18. (Figure 6-2).

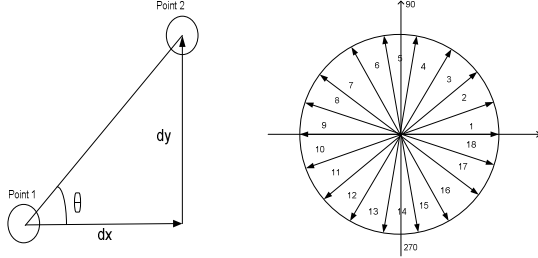


Figure 6 Angle and Quantizer

## 5. HIDDEN MARKOV MODEL IN GESTURE RECOGNITION

The gesture exists in both space and time — a spatio-temporal pattern. It is reasonable to assume that the temporal length of a gesture will vary amongst users. Thus we use a probabilistic approach, the Hidden Markov Model, to characterise a gesture in the recognition system.

We define a gesture letter as a sequence of directional angles which are the observation symbols. Each letter is mapped to one hidden Markov model. We adopted the traditional Baum-Welch [7] and the more recent Viterbi Path Counting algorithms to train the Hidden Markov Models over a range of model structures from Fully connected to Left Right with number of states ranging from 4 to 10.

### 5.1 Baum Welch Algorithm

In our system, we use 20 gestures of length=25 for training and reserve 10 for testing in a three-fold cross-validation methodology. Now consider the case where K observation sequences are known to be generated by the same HMM and the objective is to determine the HMM parameters that yield high probability of generating all K observed sequences. Rabiner [7] proposed just such a multi-sequence training algorithm using the K observation sequences at each stage of the re-estimation to iteratively update a single HMM parameter set. The re-estimation formulas are as follows:

$$\tilde{a}_{ij} = \frac{\sum_k W_k \sum_{t=1}^T \alpha_i^k a_{ij} b_j(O_{t+1}^{(k)}) \beta_{t+1}^{(k)}(j)}{\sum_k W_k \sum_{t=1}^T \alpha_i^k \beta_t^{(k)}(i)} \quad (3)$$

$$\tilde{b}_{ij} = \frac{\sum_k W_k \sum_{O_t^{(k)}=v} \alpha_i^k \beta_t^{(k)}(i)}{\sum_k W_k \sum_{t=1}^T \alpha_i^k \beta_t^{(k)}(i)} \quad (4)$$

Where  $W_k=1/P_k$ ,  $k \in [1, K]$  is the inverse of the probability of the current model estimate generating training sequence k, evaluated using the forward algorithm.

### 5.2 Viterbi Path Counting Algorithm

The VPC method is a simplification of the Best First method of Stolke and Omohundro [8] that has exhibited very good performance on simulated HMM data. The Viterbi Algorithm [7] is used to find the optimal state path for an observation sequence.

Step 1: Initialization:

$$\delta_1(i) = \pi_i b_i(o_1) \alpha \quad \psi_1(i) = 0 \quad \text{for } 1 \leq i \leq N$$

Step 2. Recursive computation. For  $2 \leq t \leq T$  for  $1 \leq j \leq N$

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t) \quad \psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}]$$

Step 3. Termination at T:  $P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \quad q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)]$

Step 4: Tracking back the optimal state sequence For  $t=T-1, T-2, \dots, 1$  by  $q_t^* = \psi_{t+1}(q_{t+1}^*)$ .

where  $P^*$  is the state-optimized likelihood function, and  $Q^* = \{q_1^*, q_2^*, \dots, q_T^*\}$  is the optimal state sequence.

The Viterbi Path Counting algorithm is as follows. For a vector of training sequence  $O_i(t)$  for time  $t \in [1, T]$  (T is the length of the sequence and i is the sequence index) and specified HMM model structure we perform the following training procedure:

1. Create a random HMM with the specified topology (e.g., Full connected or Left-Right) with parameters  $A, B, \pi$ .
2. Create counter matrices  $A_c, B_c, \pi_c$  by setting their elements to be 0 if the corresponding element in  $A, B, \pi$  is zero, otherwise set them to 1.
3. Repeat steps 4 to 8 as many times as desired to achieve good training.
4. Repeat steps 5-8 over all training sequences  $O_i$ .
5. Set the current estimate  $A', B', \pi'$  of the model parameters to be normalized counter matrices.
6. Use the Viterbi algorithm with current model parameter estimates  $A', B', \pi'$  to estimate the most likely path  $p(t)$  for the current observation sequence  $O_i$ .
7. Add 1 to  $\pi_c(p(1))$  to increment the counter for  $\pi$  at the first node  $p(1)$  in the most likely path.
8. Loop through the observation sequence, adding 1 to the values in  $A_c$  for every transition from  $p(t)$ , and adding 1 to  $B_c$  for every emission in  $p(t)$ .
9. Set the final estimate  $A_n, B_n, \pi_n$  of the model parameters to be the normalized counter matrices.

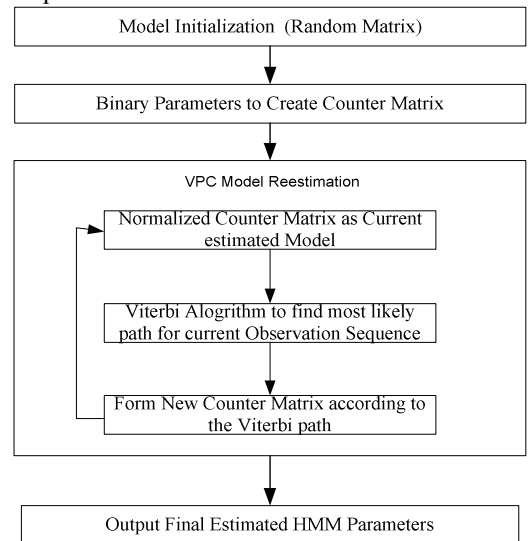


Figure 7 VPC Training Method

This VPC method simplifies the training process because it simply counts states and transitions, and estimates the model parameters from those total counts. The hard work of finding the best path is hidden with the call to the Viterbi algorithm. In addition, the dependence upon the initial

model is minimized because the model starts out a maximum entropy state, and the random seed only sets down the very first choice.

In contrast, Rabiner's method is effectively a hill-climbing method which makes it unlikely to find globally optimal solutions and there is a high degree of dependence upon the initial seed value.

## 6. EXPERIMENTAL RESULTS AND DISCUSSION

In the video database that we collected for this project, we have 30 videos of letter gestures for each of the 26 letters of the alphabet totalling 780 videos in all. The HMMs are trained on 20 videos and the other 10 are used for testing by way of three-fold cross validation.

We test Baum Welch (BW) and Viterbi Path Counting (VPC) algorithms for HMM training. HMM topologies include fully connected (FC) and Left-Right (LR) connected, and the number of states ranges from 4 to 10. After extracting the observation sequence from the input gesture video, we calculate the probability of the observation sequence for all 26 HMMs corresponding to the letters. We output the letter corresponding to the HMM with highest likelihood.

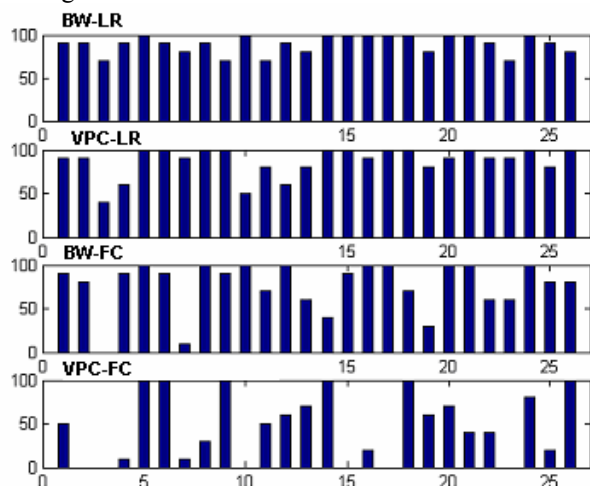


Figure 8 Percentage correct recognition for 26 letter gestures with HMM Number of States = 9.

Figure 8 presents the best recognition results for both methods and topologies for HMM number of states equal to 9. From all of the experiment results, we can draw the following observations:

- 1) The Baum-Welch algorithm, LR structure, 9 states, achieved the best overall accuracy (89.6%). For the VPC algorithm, the highest accuracy (86.4%) also occurred with LR topology and 9 states.
- 2) The average overall recognition performance of LR topologies is always better than FC. For 9 states, Baum-Welch LR achieves 89.6%, while FC gets 73%, and for VPC, LR achieves 86% while FC is 48%.
- 3) The recognition accuracy of LR HMMs trained with the Baum-Welch algorithm was dependent on the number of states and accuracies ranged between 89.6 and 83.2 percent. HMMs trained with the VPC algorithm were much less sensitive to the number of states, and accuracies ranged between 86.4 and 85.2 percent.

The recognition of letter C was poor being 70% at best; it is confused with G and O. Other poorly recognised letters are J, L, and X. In future work, we will use other discrete symbols and methods such as Coupled Hidden Markov Models and 2D VPC algorithm in an attempt to improve recognition.

## 7. CONCLUSION

This paper presents a system for hand gesture trajectory estimation and letter gesture recognition. An efficient segmentation and smoothing method is used to extract the trajectory of the hand movement. We compared two HMM training algorithms, traditional Baum-Welch and the more recently proposed Viterbi Path Counting method. With appropriate choice of parameters and HMM topology, both methods performed quite well on the 26 letters gestures yielding overall recognition performance of about 90%.

## 8 REFERENCES

- [1] J. Yang, Y. Xu, and C.S. Chen, "Gesture Interface: Modeling and Learning," IEEE International Conference on Robotics and Automation, Vol. 2, 1994, pp. 1747-1752.
- [2] A. Gersho and R.M.Gray. Vector Quantization and Signal Compression, Kluwer Academic Press,1991, ISBN 0-7923-9181-0.16
- [3] Palm Products-Ways to Enter Data into a Palm Handheld, Aug,2003, Available at (online): <http://www.palm.com/us/products/input/>
- [4] A. Kundu, Yang He and P. Bahl, "Recognition of Handwritten Words: First and Second Order Hidden Markov Model Based Approach, " Pattern Recognition, vol. 22, no. 3, p. 283, 1989.
- [5] D.Heckenberg and B. C. Lovell, "MIME: A Gesture-Driven Computer Interface", Proceedings of Visual Communications and Image Processing, SPIE, V 4067, pp 261-268, Perth 20-23 June, 2000
- [6] Intel® Software Products Open Source "Intel Open Source Computer Vision Library Reference Manual" Aug,2003, Available at (online): [www.intel.com/research/mrl/research/opency](http://www.intel.com/research/mrl/research/opency)
- [7] LR 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.
- [8] A. Stolcke and S. Omohundro. Hidden Markov Model induction by Bayesian model merging, Advances in Neural Information Processing Systems, pp. 11-18, Morgan Kaufmann, San Mateo, United States of America, CA1993.

## 9. APPENDIX: 26 Letter Gestures

A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z				