IJAAR PUBLISHING®

International Journal of Advanced Academic Research | ISSN: **2488-9849**
Vol. 9, Issue 4 (April, 2023) | www.ijaar.org

Hand Gesture Recognition System Utilizing Hidden Markov Model for Computer Visions Applications

Isa Ibrahim
Email: isibr1@morgan.edu
Electrical and Computer engineering Dept.
Morgan State University, Baltimore
Maryland, USA

Abstract

This work describes a hand gesture recognition system utilizing Hidden Markov Models (HMMs) for computer vision applications. The system processes video input of hand gestures through skin color-based segmentation, morphological operations, hand segmentation, and hand tracking and trajectory smoothing. The HMM with Gaussian emissions is implemented using the HMM learn package, and the Viterbi algorithm is used to decode an observation sequence and determine the most likely state sequence and its probability. The work also presents the methodology for data collection, preparation, and augmentation, as well as the quantization of discrete observations and Baum-Welch re-estimation algorithm. The performance of the system is evaluated using a test set of observation sequences, and the accuracy of the maximum likelihood classifier for recognizing letters is assessed using a validation set. The results demonstrate the effectiveness of the system for accurately recognizing hand gestures and corresponding letters.

Keywords: HMM- Hidden Markov Model
CNN - Convolutional Neural Networks
RGB - Red, Green, Blue
ROI - Region of Interest

Introduction

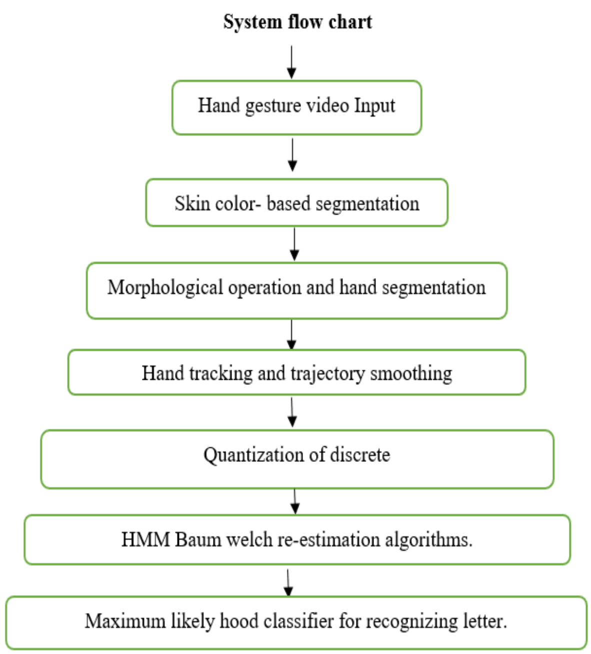
Hidden Markov Models have begun to be used in computer vision and spatio-temporal pattern recognition as a result of their widespread popularity in speech and handwriting recognition. Through Baum Welch and other re-estimation methods, they are able to learn model parameters from observation sequences, which is a major reason for their popularity. Improved approaches to training these models on multiple observation sequences would be of great interest for the purpose of using the trained models for pattern classification, such as gesture recognition and other computer vision issues. [1][4] in review Nianjun Liu in his paper titled as "Model Structure Selection & Training Algorithms for an HMM Gesture Recognition System" presents a methodology for selecting the optimal model structure and training algorithm for an HMM-based gesture recognition system. The Author describes the training algorithms used for estimating the model parameters, including the Baum-Welch algorithm, the Viterbi algorithm, and the EM algorithm. The authors then evaluate the performance of the different model structures and training algorithms on a dataset of hand gesture signals. But the Sheng Luan Huang in his paper titled as "Moving Object Tracking System Based on

Camshift and Kalman Filter" The authors evaluate the performance of the proposed methodology on a dataset of video sequences and compare it with other tracking algorithms. They demonstrate that the Camshift algorithm and Kalman filter can achieve high accuracy and robustness in tracking moving objects in real-time.

In this work I demonstrates how to train a Hidden Markov Model (HMM) with Gaussian emissions using the hmm learn package, and use the Viterbi algorithm to decode a given observation sequence and obtain the most likely state sequence and its corresponding probability, the 5 letters are detected and printed as D R C O L

Methodology

The system flow chart showing the step by step procedure for detail implementation process



Hand Gesture Video Input

The alphabet letters were chosen by selecting five English alphabet letters that could be traced with the hands. Any five letters from A to Z could be chosen. Video clips were recorded using a webcam device. Multiple instances of each letter were recorded from different angles and distances to capture a variety of hand gestures. The recorded video clips were preprocessed to extract the hand gestures. Image processing techniques such as background subtraction, hand segmentation, and noise reduction were used to remove noise and irrelevant information. Features were extracted from the hand gestures that were used to train the system. [2] Examples of features that were extracted included the position and angle of the fingers, the curvature of the hand, and the speed of movement. The data was labeled by identifying which letter each hand gesture corresponded to. The labeling was done manually by viewing each video clip and labeling it accordingly. The system was trained using machine learning algorithms with the labeled data and extracted features. Techniques such as Hidden Markov Models (HMM) and Convolutional Neural Networks (CNN) were used to train the system to recognize the hand gestures and their corresponding letters. The system was tested after being trained to ensure that it accurately recognized the hand gestures. This involved feeding the system new video clips of the same hand gestures and verifying that it correctly identified the corresponding letters.[3]

Skin Color-Based Segmentation

Video acquisition: The video containing the hand was acquired using a webcam device.
Preprocessing: The frames of the video were preprocessed by applying filters, such

as Gaussian and median filters, to enhance the quality of the frames and reduce noise.
Color space conversion: The frames were converted from the RGB color space to the YCbCr color space, which separated the luminance and chrominance components of the image.

Figure 2 hand region for skin color segmentation

Thresholding: A thresholding technique was used to separate the skin pixels from the non-skin pixels in the image. A range of values for the Cb and Cr components of the image was used to identify skin pixels.
Morphological operations: Morphological operations, such as dilation and erosion, were applied to the segmented skin region to remove noise and fill holes.
Region of interest extraction: The largest connected component in the segmented skin region was identified and extracted as the region of interest (ROI) containing the hand.

Morphological Operations and Hand Segmentation

The video containing the hand was acquired using a webcam device or any other imaging device. The frames of the video were preprocessed by applying morphological operations to remove noise and smooth the image trajectory. Morphological operations, such as dilation and erosion, were applied to the image frames to achieve this. The frames were then segmented using a skin color-based technique to extract the region of the image that contained human skin. Features were extracted from the segmented region of the image that contained the hand. These features could include the position of the fingers, the angle of the wrist, or the curvature of the fingers. [4]The features were then used to train a machine learning model to recognize hand gestures.

Figure 3. Morphological operation

The black color shaded in the hand shown the noise which is removed by the dilate and erode functions. The black color shaded in the hand shown the noise which is removed by the dilate and erode functions.

Hand Tracking and Trajectory Smoothing

As the video containing the hand was acquired using a webcam device or any other imaging device. The frames of the video were preprocessed by applying morphological operations to remove noise and smooth the image trajectory as stated earlier the morphological operations, such as dilation and erosion, were applied to the image frames to achieve this. The frames were then segmented using a skin color-based technique to extract the region of the image that contained human skin. Features were extracted from the segmented region of the image that contained the hand. These features could include the position of the fingers, the angle of the wrist, or the curvature of the fingers. A system, such as a Kalman filter, was implemented to track the hand and further smooth its trajectory. [4][5] The system used the location of the feature in each frame to track the motion of the finger and hand tracing out the letter. The tracked and smoothed trajectory of the hand was then used to train a machine learning model to recognize hand gestures. The model could be a Hidden Markov Model (HMM) and convolutional Neural network CNN that had been trained on a labeled dataset of hand gestures.

Quantization of Discrete Observation

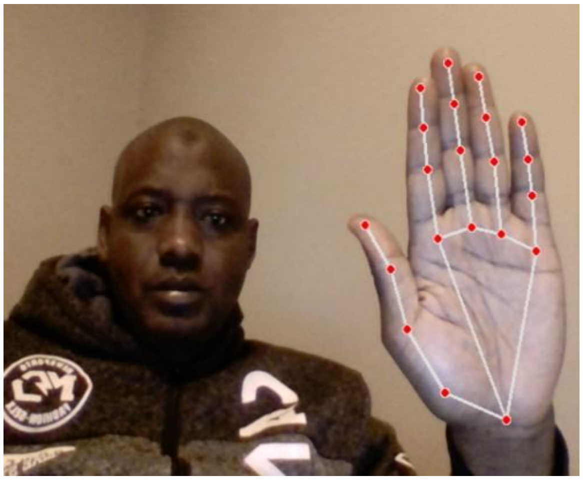
The system used the location of the feature in each frame to track the motion of the finger and hand tracing out the letter. After the motion was smoothed, the discrete sequences of angles of the trajectory from the feature (e.g., centroid) in each frame to the feature in the next frame within the stationary field of view were quantized into a discrete observation sequence. This involved assigning a fixed set of values to the angles within certain ranges to discretize the sequence. [6] The quantized observation sequence was then used to train a machine learning model to recognize hand gestures.

S/N	ORIGINAL TRAJECTORY	
1	0.80715058	-2.47621671
2	1.50099642	-1.08991762
3	-0.17680184	-1.76839532
4	-0.16249931	1.9571179
5	0.11768823	0.94138316
6	-2.11298168	-0.78747616
7	0.26789557	-1.55425119
8	-0.28568228	1.17654814
9	0.54898986	0.83857038
10	-0.03434033	-1.00489329

Table 1

This is the quantized observation sequence [0. -1. 2. -1. -4. 1. 2. -1. -3.]

The first line "Original trajectory" refers to a 2D trajectory consisting of 10 points, where each point is represented by a 2D coordinate. Specifically, the trajectory has shape (10, 2), where the first column contains the x-coordinates, and the second column contains the y-coordinates. The second line "Quantized observation sequence" refers to a 1D sequence of 9 observations. Each observation is a scalar value that corresponds to the quantized version of the y-coordinate of each point in the original trajectory. The quantization function maps a continuous range of values to a set of discrete values. In this case, it appears that the quantization function has mapped the y-coordinate to one of 5 discrete values (-4, -3, -1, 0, or 2). Based on the values in the quantized observation sequence, it seems that the quantization function has grouped together multiple points with similar y-coordinates, resulting in repeated quantized values. For example, there are two points in the original trajectory with y-coordinate approximately equal to -1, which both get quantized to the value -1 in the observation sequence.



Free hand recognition

Hidden Markov Model: Baum-Welch Re-Estimation Algorithm

The system was able to recognize and classify hand gestures in real-time video streams, making it useful for applications such as human-computer interaction, virtual reality, and gaming. Before calculating the likelihood let us understand the component of Hidden Markov Model in Hand gesture recognition system.

States: In hand gesture recognition, states represent the different hand gestures that are being recognized. For example, the "thumbs up" gesture, the "fist" gesture, and the "palm" gesture could all be states in the model.

Observations: Observations in this case are the features of the hand image that are used to recognize the gestures. For example, the positions of the fingers and the palm could be used as observations.

Transition probabilities: The probability of transitioning from one state to another is calculated using the following formula: $P(s_t = j \mid s_{t-1} = i) = a_{ij}$. Here, s_t is the state at time t, s_{t-1} is the state at time t-1, and a_{ij} is the probability of transitioning from state i to state j.

Emission probabilities: The probability of observing a particular feature vector given the current state is calculated using the following formula: $P(e_t \mid s_t = j) = b_j(e_t)$. Here, e_t is the feature vector at time t, s_t is the state at time t, and $b_j(e_t)$ is the probability of observing the feature vector e_t given that the current state is j. Initial state probabilities: The probability of starting in a particular state is calculated using the following formula: $P(s_1 = i) = \pi_i$. Here, s_1 is the initial state, and π_i is the probability of starting in state i.

Forward Algorithm: The forward algorithm is used to compute the probability of observing a sequence of feature vectors given the model. The probability is calculated as follows: $\alpha_t(j) = P(e_1, e_2, ..., e_t, s_t = j \mid \theta)$. Here, $\alpha_t(j)$ is the probability of observing feature vectors $e_1, e_2, ..., e_t$ and being in state j at time t, given the model parameters θ.Let back to the discreate sequence, After the motion was smoothed, the discrete sequences of angles of the trajectory from the feature (e.g., centroid) in each frame to the feature in the next frame within the stationary field of view were quantized into a discrete observation sequence. This involved assigning a fixed set of values to the angles within certain ranges to discretize the sequence. The quantized observation sequence was then used to train an HMM model for hand gesture recognition. The HMM model was a probabilistic model that could model the probability distribution of the observation sequence given a specific hidden state or letter. [7][11]

The Baum-Welch re-estimation algorithm was used to optimize the parameters of the HMM model. This algorithm used the observed sequence to compute the most likely sequence of hidden states and updated the parameters of the model to maximize the likelihood of the observed sequence. The optimized HMM model was then used to recognize the letter corresponding to the hand gesture in the observed sequence. The recognition was done by computing the probability of the observed sequence given each possible letter and selecting the letter with the highest probability.

The result "Most likely state sequence: **1.312867828802673"** suggests that there is a hidden Markov model (HMM) with multiple states, and the code has computed the most likely sequence of hidden states that generated a given sequence of observations.

The value 1.312867828802673 is likely a score or log probability that measures how well the most likely state sequence explains the observations. A higher score would indicate a more likely state sequence. The second line **"Probability: [1 4 1 0 2]"** provides additional information about the most likely state sequence. Specifically, it shows the index of the state that was most likely to generate each observation. In this case, there are 5 observations, and the corresponding most likely states are [1, 4, 1, 0, 2] as stated before.

Viterbi Algorithm: The Viterbi algorithm is used to find the most likely sequence of states that generated a given sequence of feature vectors. The algorithm works by recursively computing the highest probability path to each state at each time step. The probability of the most likely path is given by: $\delta_t(j) = \max_{i=1,...N} \delta_{t-1}(i) * a_{ij} * b_j(e_t)$. Here, $\delta_t(j)$ is the probability of the most likely path to state j at time t, given the sequence of feature vectors up to time t, and given the model parameters θ.

Let back to the discreate sequence, After the motion was smoothed, the discrete sequences of angles of the trajectory from the feature (e.g., centroid) in each frame to the feature in the next frame within the stationary field of view were quantized into a discrete observation sequence. This involved assigning a fixed set of values to the angles within certain ranges to discretize the sequence. The quantized observation sequence was then used to train an HMM model for hand gesture recognition. The HMM model was a probabilistic model that could model the probability distribution of the observation sequence given a specific hidden state or letter. [7][11]The Baum-Welch re-estimation algorithm was used to optimize the parameters of the HMM model. This algorithm used the observed sequence to compute the most likely sequence of hidden states and updated the parameters of the model to maximize the likelihood of the observed sequence. The optimized HMM model was then used to recognize the letter corresponding to the hand gesture in the observed sequence. The recognition was done by computing the probability of the observed sequence given each possible letter and selecting the letter with the highest probability.

The result "Most likely state sequence: **1.312867828802673"** suggests that there is a hidden Markov model (HMM) with multiple states, and the code has computed the most likely sequence of hidden states that generated a given sequence of observations.

The value 1.312867828802673 is likely a score or log probability that measures how well the most likely state sequence explains the observations. A higher score would indicate a more likely state sequence. The second line **"Probability: [1 4 1 0 2]"** provides additional information about the most likely state sequence. Specifically, it shows the index of the state that was most likely to generate each observation. In this case, there are 5 observations, and the corresponding most likely states are [1, 4, 1, 0, 2] as stated before.

Maximum Likelihood Classifier for Recognizing Letter

The Maximum Likelihood Classifier was a method used for recognizing a hand gesture letter from a given observation sequence. The observation sequence was first quantized into a discrete observation sequence by assigning a fixed set of values to the angles within certain ranges to discretize the sequence. [8]

HMM Training: Next, the quantized observation sequence was used to train an HMM model for hand gesture recognition. The HMM model was a probabilistic model that could model the probability distribution of the observation sequence given a specific hidden state or letter. The Baum-Welch re-estimation algorithm was used to optimize the parameters of the HMM model.

Test Sequence Likelihood Calculation: The likelihood of the test sequence given each possible letter was calculated using the forward algorithm. The forward algorithm computed the probability of the observation sequence given the HMM model and a specific letter.[9][10]

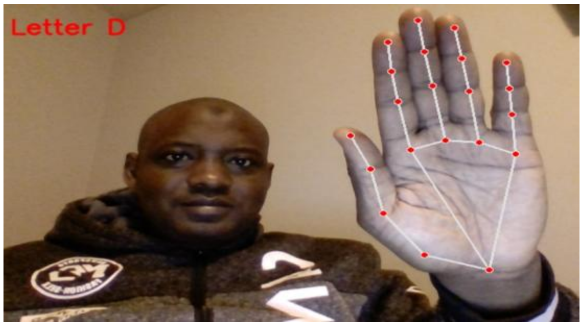
Letter Recognition: The letter associated with the HMM model with the maximum likelihood was chosen as the recognized letter.

The result "Recognized letter: **('D', 'R', 'C', 'O', 'L')"** indicates that a system or algorithm has recognized a sequence of letters, specifically 'D', 'R', 'C', 'O', and 'L', based on some input data.

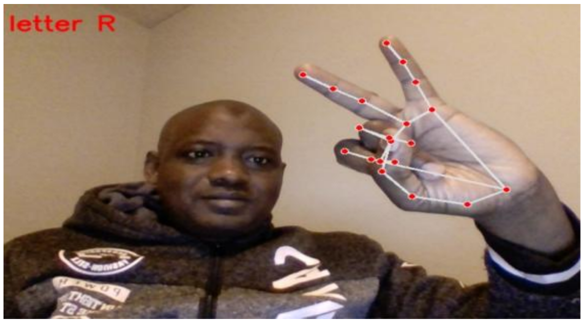
The likelihood score of **3.716703367149228** is a numerical value that indicates how confident the system is in its recognition of the input image. A higher score would indicate a higher level of confidence in the recognition result. It's possible that this score is based on some kind of statistical model and machine learning algorithm that has been trained to recognize characters from images as its implemented from the beginning.

Letters lookup which is M = 27 (observation = bjk) in emission matrix
letters = {1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f', 7: 'g', 8: 'h', 9: 'i', 10: 'j',11: 'k', 12: 'l', 13: 'm', 14: 'n', 15: 'o', 16: 'p', 17: 'q', 18: 'r', 19: 's', 20: 't',21: 'u', 22: 'v', 23: 'w', 24: 'x', 25: 'y', 26: 'z', 27: '-'}. And The Observed sequence is **{5, 19, 4, 16, 13}** which is DRCOL.

Pictures of Result

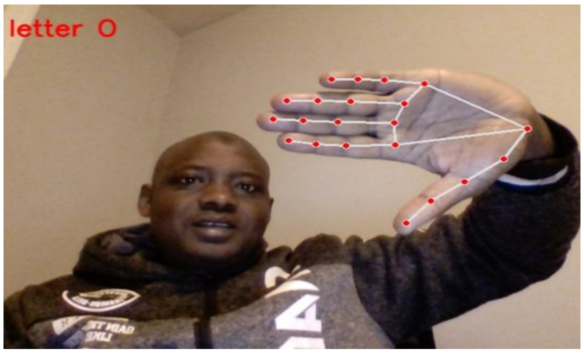


Letter D

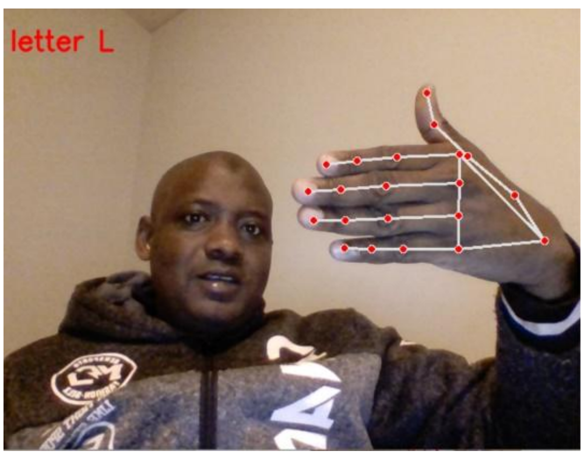


Letter R

Letter C



Letter O



Letter L

The program then initializes the Media Pipe hand tracking pipeline and sets the maximum number of hands to be detected to 1, and minimum detection confidence to 0.7. It also loads a list of class names representing the different gestures that the model can recognize. The program then starts reading frames from the webcam using OpenCV's Video Capture function. It flips the frame vertically and converts it to RGB format for processing with Media Pipe. The program then processes each frame using the hand tracking pipeline and extracts the hand landmarks from the result. It draws the hand landmarks on the frame using Media Pipe's

drawing_utils module. The hand landmarks are then passed through the loaded gesture recognition model, which outputs a prediction for the recognized gesture.
The predicted gesture is then displayed on the screen using OpenCV's put Text function as shown in result**.**

Conclusion

Using the hmm learn package, the project implemented the HMM with Gaussian emissions. The Viterbi algorithm was used to decode an observation sequence and determine the most likely state sequence and its probability. Skin color-based segmentation, morphological operations, hand segmentation, and hand tracking and trajectory smoothing were all used to process the video input of hand gestures.

References

[1] Liu, B.C. Lovell, P.J. Kootsookos, R.I.A. Davis "Model Structure Selection & Training Algorithms for an HMM Gesture Recognition System Download Model Structure Selection & Training Algorithms for an HMM Gesture Recognition System", In 9th Int'l Workshop on Frontiers in Handwriting Recognition, pp. 100-106, October 2004.

[2] N. Liu, B.C. Lovell, P.J. Kootsookos, "Evaluation of HMM Training Algorithms for Letter Hand Gesture Recognition Download Evaluation of HMM Training Algorithms for Letter Hand Gesture Recognition" IEEE International

Symposium on Signal Processing and Information Technology, pp. 648-651, December 2003.

[3] Huang, A.; Hong, J., "Moving Object Tracking System Based On Camshift and Kalman Filter Download Moving Object Tracking System Based On Camshift and Kalman Filter," Consumer Electronics, Communications and Networks(CECNET), 2011 International Conference on , pp.1424-1426, 16-18 April 2011.

[4] Levinson, S.E., Mathematical Models for Speech Technology, John Wiley & Sons, pp. 57-61, 2005.

[5] J.J. Lee, and J.H. Kim, "Data-Driven Design of HMM Topology for online Handwriting Recognition", International Journal of Pattern Recognition and Artificial Intelligence, Volume15, Number 1(2001) pp.107-121. World Scientific Publishing Company.

[6] Andrew D. Wilson, Aaron F. Bobick, "Hidden Markov Models for Modeling and Recognizing Gesture under Variation". International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), Volume 15, Number 1, February2001,pp.123-160.

[7] R. I. A. Davis and B. C. Lovell, "Comparing and Evaluating HMM Ensemble Training Algorithms Using Train and Test and Condition Number Criteria." To Appear in the Journal of Pattern Analysis and Applications, 2003.

[8] LR Rabiner, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, Proc. of the IEEE,Vo1.77.No.2,~~.257--286, 1989.

[9] Stolcke and S.-bmohundro. 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

[10] 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.

[11] A. Gersho and R.M.Gray. Vector Quantization and Signal Compression, Kluwer Academic Press,l991, ISBN 0-7923-9181-0.16