Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

An Online Optimal Path Decoder for HMM towards Connected Hand Gesture Recognition

Monalisa Mazumdar [**] **Mun-Ho Jeong** [***] **Bum-Jae You** [****]

Center for Cognitive Robotics Research
Korea Institute of Science and Technology
39-1 Hawolgok dong, Seongbuk ku, Seoul 136-791, Korea
[**] *(e-mail: monalisa@kist.re.kr)*
[***] *(e-mail: mhjeong@kist.re.kr)*
[****] *(e-mail: ybj@kist.re.kr)*

Vision, Pattern Recognition, Hand gesture, Hidden Markov Model, Optimal path

Abstract: In this paper, we model the recognition problem for hand gesture as that of finding an optimal path through a Hidden Markov Model (HMM) state graph. To determine this optimal path, we present a novel online method which decodes the observed gesture pattern and evaluates the optimal graph node at each time frame of the continuously deepening HMM state graph. The temporal signature is subsequently handled by introducing a rejection threshold which acts as a depth-wise sliding window for pruning the unnecessary graph nodes. The functional depth of the graph is defined by this depth rejection threshold. Experimental comparison of our algorithm with other HMM-based search algorithms demonstrates the effectiveness and robustness of our method.

1. INTRODUCTION

The wish to provide a more natural means of interaction with computers has led to a considerable interest in the field of hand gesture recognition. Pavlovic et al. [1997] provides a review on the existing techniques for hand gesture interpretation. In our aim to allow users to perform gestures in a natural and unencumbered manner we use vision-based techniques for gesture recognition.

While dealing with connected hand gesture recognition we are faced with two serious hurdles:
(1) Difficulty in reliably determining the boundaries between gesture models.
(2) Difficulty in establishing how many gesture models are contained in the sequence.

Hidden Markov Model (HMM), a stochastic process explained in detail by Rabiner [1989], provides a good framework for such a type of recognition. HMM has been extensively used in hand gesture recognition by Chen et al. [2003], Lee and Kim [1999], Ng and Ranganath [2002], Ramamoorthy et al. [2003]. HMM perform recognition by decoding this observed gesture pattern to find the underlying gesture sequence. For connected hand gesture recognition, we consider all the HMM gesture states $s = 1,\ 2,\ \dots N^{(M)}$, N is the total number of states of each reference gesture model M, together at every time frame to form one large HMM state graph as shown in Fig. 1.

Given an observed pattern $O_t = O_1,\ O_2,\ \dots O_T$, where T is the length of the observation pattern, the reference gesture states are then unfolded along the time axis of the test pattern to find the optimal state sequence with maximum joint likelihood probability as,

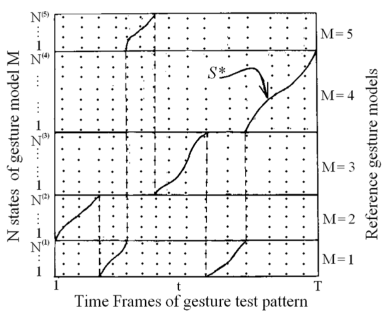

N states of gesture model M
Reference gesture models
M=5
M=4
M=3
M=2
M=1
S*
1
t
T
Time Frames of gesture test pattern

Fig. 1. Illustration of path finding problem in connected gesture recognition

$$s^* = argmax_s\ \ P(O, s|\Lambda), \tag{1}$$

where Λ is the set of HMM gesture models

The task now is to design an efficient search algorithm to deal with this huge search space and find the optimal state sequence.

Our research is concentrated towards this decoding process. For online recognition, with each advancement of time the HMM graph deepens gradually. In our work, we describe an algorithm to search online this ever deepening HMM state graph and find the optimal node at every time frame. The path corresponding to the optimal node represents the optimal state sequence and reveals the gestures enacted so far.

We implement the algorithm on the humanoid robot MAHRU, (You et al. [2005]). Trained hand gesture models are used to command the robot into specific actions.

The rest of the paper is organized as follows : in Section 2 we give a comparison with related works, in Section 3 we define the probabilistic model with mathematical formulation for our algorithm followed by the required procedural steps, and in Section 4 we describe the details of our real time Implementations along with the gesture recognition framework. Finally, in Section 5 we present an analysis of our experimental results and compare the performance with a couple of baseline methods, followed by a short conclusion in Section 6.

2. COMPARISON WITH RELATED WORKS

The problem of hand gesture recognition has been tackled by using different search strategies. The standard Viterbi algorithm is an exhaustive search procedure but a slight modification of it as the Viterbi Beam search is rather more effective. The N best search and A* search, also form the basis for most works in this domain. Some proposed improvements of these conventional methods can be found in Deshmukh et al. [1995], Hu and Brown [1996], Illina and Gong [1996], Nakamura and Heracleous [2002], Paul [1992], Richardson et al. [1995]. Brown and Rabiner [1982] employed the A* algorithm in offline speech recognition to show that it guarantees an optimal path with fewer computations than dynamic programming algorithms.

The significance of our algorithm is that we incorporate this idea as a time synchronous online algorithm, guaranteeing an optimal path at every instant of time. We find a significant reduction in computation time over algorithms of the like of Viterbi Beam and N-Best search. By restricting the search to investigate only the best path, we are able to keep the number of nodes examined low between $\frac{1}{3}$ and $\frac{1}{2}$ as compared to Viterbi Beam search and N-Best algorithms, thereby leading to a 60% reduction in computation.

Our online algorithm has more flexibility over conventional shortest path algorithm in that our source and destination nodes are not fixed but are arbitrarily chosen. This arbitrary destination acts as a temporary stop point thereby helping in identifying a momentary pause in an online sequence. The arbitrary source assists in selecting the correct source gesture node out of the various HMM gesture models. Further, we introduce a rejection threshold, δ_{th}, for online pruning of unlikely path candidates thus remaining within the memory capacity of the machine.

3. OPTIMAL PATH DECODER IN A HMM STATE DAG

3.1 Computation of the score of a path passing through a graph node

The HMM state graph of Fig. 1 is represented as directed acyclic graph (DAG) in the sense that the conditional probabilities that exist between HMM states and time is irreversible. A node, i, in the DAG represents a state instance of a gesture model, m, at a particular time, t, giving the coordinates as $i^{(m)} = (t, s_t)$. We denote the possibility of a path in the DAG to pass through the i^{th} node as the probability measure $P(o_1^T, s_t = i^{(m)})$, where $t \leq T$, T:current time. We compute this probability measure as,

$$\begin{aligned} P(o_1^T, s_t = i^{(m)}) &= P(o_1^t,\ s_t = i^{(m)},\ o_{t+1}^T) \\ &= P(o_1^t,\ s_t = i^{(m)})\ P(o_{t+1}^T | s_t = i^{(m)}) \\ &= \alpha_1^t(i^{(m)})\ \beta_t^T(i^{(m)}), \end{aligned} \tag{2}$$

considering $o_a^b = (o_t; a \leq t \leq b)$. Taking log on both sides to keep the calculations within the dynamic range of the machine,

$$\begin{aligned} \log[P(o_1^t, s_t = i^{(m)}, o_{t+1}^T)] = \log[\alpha_1^t(i^{(m)})] \\ + \log[\beta_t^T(i^{(m)})]. \end{aligned} \tag{3}$$

The probability measure of eq(3) gives the score for any path passing through the i^{th} node. The above eq(3) can be re-written in the form,

$$f_T(i^{(m)}) = g_1^t(i^{(m)})\ +\ h_t^T(i^{(m)}), \tag{4}$$

where $g_1^t(i^{(m)})$ is the score of the path of this node from the arbitrary source node, $h_t^T(i^{(m)})$ is the score from this node to the arbitrary destination node. $f_T(i^{(m)})$ is the evaluation function of the node giving the score along the path of the i^{th} node of the graph.

Definition 1. An Optimal node $i^{(m)} = (t, s_t)$ is the node with the highest score of $f_T(i^{(m)})$ selected at the current time T, such that $t = T$.

3.2 The Evaluation function

The efficiency of our algorithm strategy depends crucially on the specification of the evaluation function given in eq (4). We define $g_1^t(i^{(m)})$ as the probability of the observation sequence o_1^t to give the state $s_t = i^{(m)}$.

- Initially at time $t = 1$,

$$g_1^1(i^{(m)}) = \log[\pi_{s_1=i^{(m)}}\ b_{s_1=i^{(m)}}(O_1)]. \tag{5}$$

- At any gesture interior we reach the i^{th} node after a legal transition from its parent node j, hence

$$\begin{aligned} g_1^t(i^{(m)}) \ =\ & g_1^{t-1}(j^{(m)}) \\ & + log[\ a_{s_{t-1}=j^{(m)}, s_t=i^{(m)}} b_{s_t=i^{(m)}}(O_t)\ ]. \end{aligned} \tag{6}$$

- At any gesture boundary along the recognized optimal path, when there is a transition from the j^{th} node of any gesture M' to the i^{th} node of the new gesture M,

$$g_1^t(i^{(M)}) = g_1^{t-1}(j^{(M')}) + \log[\pi_{s_t=i^{(M)}}\ b_{s_t=i^{(M)}}(O_t)]. \tag{7}$$

To evaluate $h_t^T(i^{(m)})$ we determine the probability of observing the partial sequence from t to current time T given the state $s_t = i^{(m)}$. Now, to compute $h_t^T(i^{(m)})$, we need to be able to perceive the possible transitions made by the node $i^{(m)}$ to reach T. We assume that the expected route of the i^{th} node is made by transitions along the same state instance of current gesture, $m = M$, till it reaches T. This results in $T - t$ number of transitions along the same state instance of gesture M. Solving by induction for $h_t^T(i^{(m)})$ we have,

$$h_t^T(i^{(m)}) = \log[a^{T-t}_{s_t=i^{(M)},s_t=i^{(M)}} \prod_t^T b_{s_t=i^{(M)}}(O_T)]. \quad (8)$$

where $m = M$. At time $T = t$ we will have $h_T^T(i^{(m)}) = 0$.

By observing the value of $h_t^T(i^{(m)})$, the algorithm can predict whether the test and reference pattern are of the same gesture class or not.

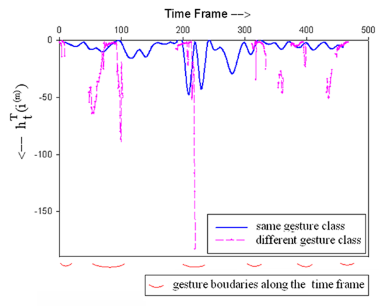


Fig. 2. Comparing the test and reference pattern at each time frame

On examining the Fig. 2 which matches the $h_t^T(i^{(m)})$ value of the test and reference gesture classes for an online sequence of 5 connected hand gestures, we find that,
(1) At gesture interior, nodes belonging to different gesture classes have a negligible value for $h_t^T(i^{(m)})$ which reduces the f-score to a minimum, thus eliminating the computation of these unlikely nodes. However, there exists a consistent high value for the same test and reference gesture class.
(2) At gesture boundary, the value of $h_t^T(i^{(m)})$ for same class is very high as compared to that of different gesture class. This results in fixing on the correct gesture promptly.

Therefore, the value of $h_t^T(i^{(m)})$ is the predominant factor during the calculation of the evaluation function $f_T(i^{(m)})$ of a node in the HMM state DAG.

3.3 The Optimal Path Decoder

At any instant of time t, the path through the node i^{th} node is given by eq(4) and the node is characterized by :
- the node coordinates $i^{(m)} = (t, s_t)$.
- the time τ upto which the node has been evaluated.
- $g_1^t(i^{(m)})$, which facilitates in computing the probability measure of a successor node.
- $h_t^\tau(i^{(m)})$, which is updated as we move forward in time thereby updating $f_\tau(i^{(m)})$.
- a path list from the arbitrary source node to itself.

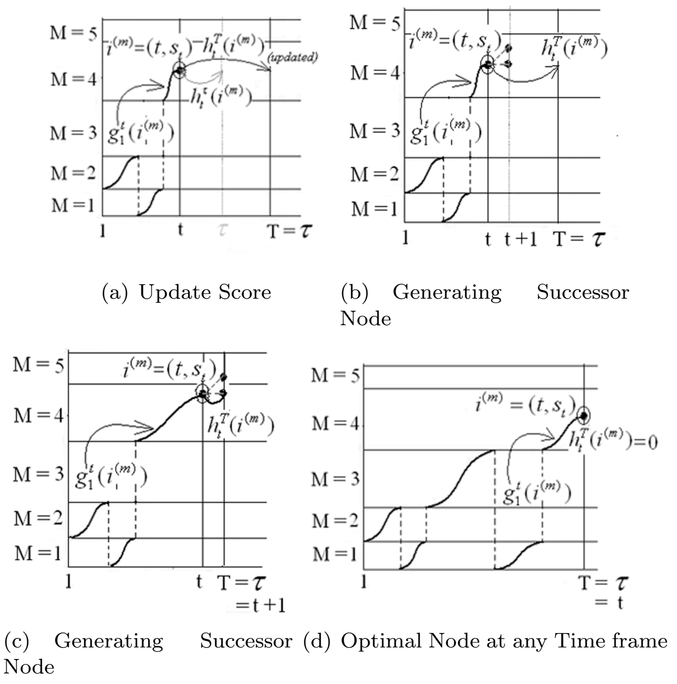
We introduce the concept of a virtual arbitrary source node for facilitating initialization of all the HMM gesture classes and a temporary termination on the selection of the optimal node $i^{(m)} = (t, s_t)$ at current time T such that $t = T$. Our optimal path decoder is summarized as below

1 Initialization :
at time $T = 1$ an arbitrary virtual source node initializes each gesture. These nodes are inserted in an "Open" list which maintains a list entry of all the potential Optimal nodes sorted in descending order of $f_T(i^{(m)})$. A Priority Queue (PQ) is used to retain this list.

2 Iteration at each time frame T
(a) The optimal node obtained at $T - 1$ is selected at current time frame T. Its evaluation function is updated, Fig. 3(a), and reinserted in the open list.
(b) From the open list the head node $i^{(m)} = (t, s_t)$ is popped and,
- if $t < T - \delta_{th}$, then the node is deleted.
- if $\tau < T$, then the evaluation score $f_\tau(i^{(m)})$ of the node is updated such that $\tau = T$ giving a new score $f_T(i^{(m)})$, Fig. 3(a).
- If $\tau = T$, then the node generates permitted successor nodes, Fig. 3(b) and 3(c). Each successor node inherits the path list of its parent node and we compute new path scores f_T. These are then inserted into the Open list. When a node i is successfully expanded then it is removed from the Open list and placed in the "Closed" list which contains a list of all nodes to which the optimal path has been found. We use a Hash Table to store this Closed list.
- if $t = \tau = T$, then i is the Optimal node in the current time frame T, Fig. 3(d). Its evaluation function need not be re-computed as it is already computed till T.

(c) Stop if end of time frame or iterate with $T = T + 1$.

Online trace back of the path of the Optimal node at time T gives the gestures traced out so far.



(a) Update Score (b) Generating Successor Node (c) Generating Successor Node (d) Optimal Node at any Time frame

Fig. 3. Algorithmic Steps

The path of the Optimal node is optimal only if it fulfills the following conditions:
1) the expansion is consistent for all nodes.
2) $g_1^t(i^{(m)}) \neq 0$, $\forall i$ and $g_1^t(i^{(m)})$ is monotonic.
3) $h_t^\tau(i^{(m)}) \neq 0$, $\forall i \neq T$, and $h_t^\tau(i^{(m)})$ is monotonic for all potential paths.

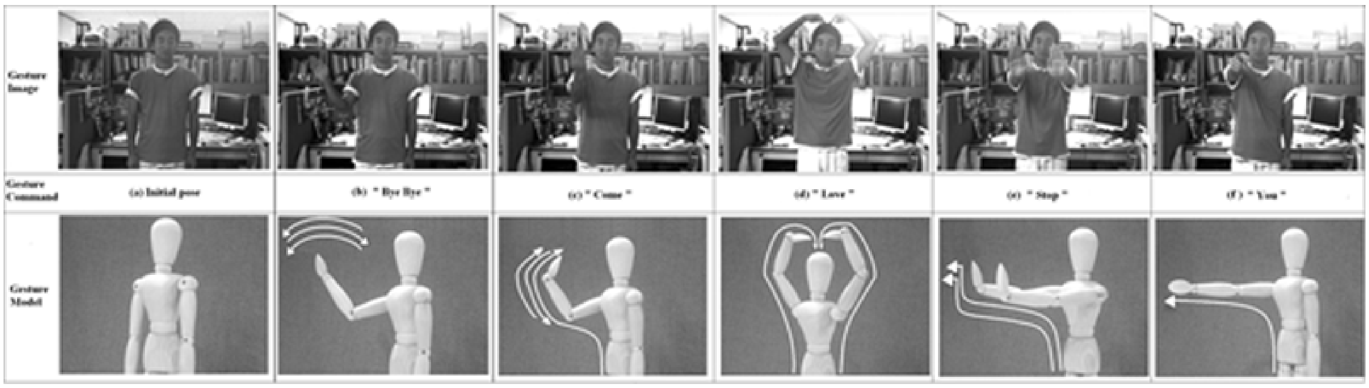


Fig. 4. Gesture Classification

4) $f_\tau(j^{(m)}) \leq f_\tau(i^{(m)})$, $\forall j \leq i$, with j lieing on the path of i.

These conditions are sufficient for the algorithm to find the Optimal path.

4. IMPLEMENTATION IN HAND GESTURE RECOGNITION

4.1 Observation Feature Vector

We adopt a depth feature space for analyzing the stereo images by employing a factory calibrated Pointgrey (Bumblebee) stereo vision camera with 4mm focal length lens producing disparity images at 15 fps with 320x240 resolution. As the actor faces the camera relatively closer than any other background objects, the foreground image which is the region of interest is extracted by performing human face detection using Open Source Computer Vision Library (OpenCV) face detector and estimate the foreground region (Oh and Lee [2004]) as,

$$F(x,y) = \begin{cases} 0 & , D(x,y) > D_f + c \\ F(x,y) & , D(x,y) \leq D_f + c \end{cases} \tag{9}$$

where $F(x,y)$ results in the foreground image having values for each pixel position (x,y), $D(x,y)$ is the depth of each pixel, D_f is the average depth of the human face and c is a constant of body thickness.

The values of each depth foreground image produces a high dimensional feature vector which is subjected to linear dimension reduction by using *Principal Components Analysis* (PCA). From the resulting eigenvectors, a subset n is chosen covering a desired amount of the data variance. This forms the linear subspace representing the observable feature vectors O_t, where $t = 1, 2, ...$, used for experimentation purpose.

4.2 Gesture Classification

Out of the wide variety of human gestures, for experimentation purposes, we selected five different dynamic gestures with which to command the robot into performing actions. The commands used in training the robot are - “ByeBye”, “Come”, “Love”, “Stop” and “You”. Each of the Gesture Models depict the patterns traced out by the hand motion of the Gesture Image. The arrow shows the direction of motion. The “initial pose” indicates the start and end of a gesture command.

For each gesture we train a left-right HMM, $\lambda = (A, O, \pi)$. 4 actors enacted separately each of the 5 gestures, to produce 5 sets of observation sequences $O = O_1, O_2, ...O_T$, one for each gesture respectively. These are used for offline training producing 5 user independent trained HMM gesture classes, each consisting of an unique identification number.

4.3 System Description

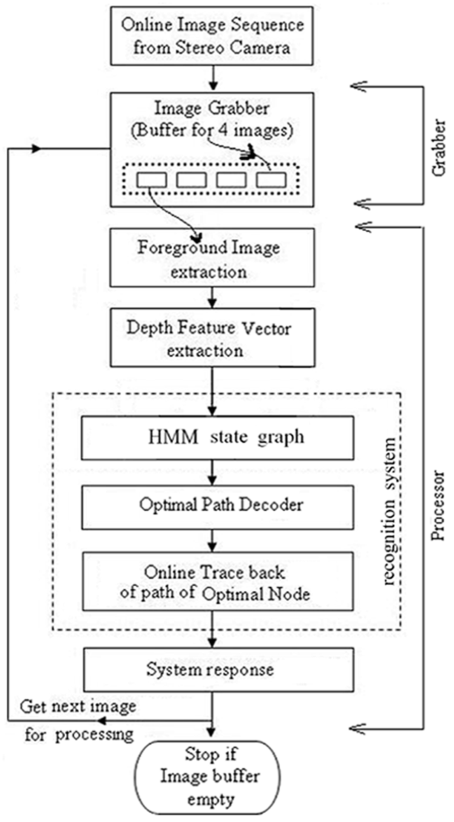
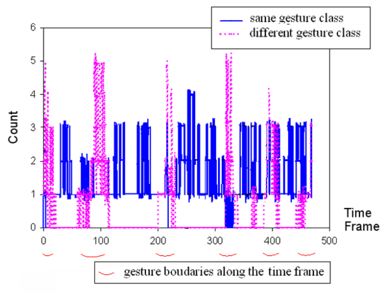
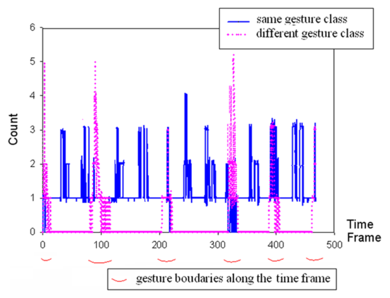

Online Image Sequence from Stereo Camera
Image Grabber (Buffer for 4 images)
Grabber
Foreground Image extraction
Depth Feature Vector extraction
HMM state graph
Optimal Path Decoder
Online Trace back of path of Optimal Node
recognition system
Processor
System response
Get next image for processing
Stop if Image buffer empty

Fig. 5. Overview of Processing Steps

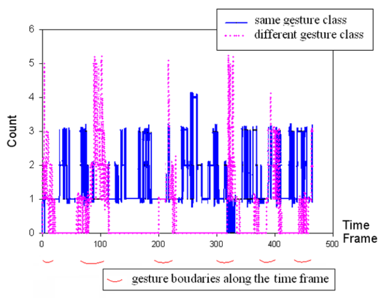
For real time implementation, we divide the gesture recognition task into two threads, grabber and processor operating as synchronized threads. The grabber acquires images online from the camera at a rate of 15 fps and



(a) Number of Nodes Selected



(b) Number of Nodes Expanded



(c) Number of Computations

Fig. 6. Plots matching test and reference patterns from the same (correct) and different(incorrect) gesture classes

stores the images in a buffer of 4 images; the processor reads the images sequentially from the buffer and performs recognition on the region of interest of the image.

Each image goes through the processing steps as shown in Fig. 5. The trained gesture classes are invoked by the recognition system, the results of which is output as system response. Using our proposed Optimal Path Decoder algorithm for HMM state DAG we performed experiments for online sequences of connected gestures. An analysis of experimental results are given in Section 5.

5. EXPERIMENTAL RESULTS AND ANALYSIS

5.1 Optimal Path Decoding Experimental Results

For our decoding experiments, 4 actors (different from training actors) performed different hand gestures taken from the training set, thereby producing a set of 4 online test patterns comprising of observable O_t where $t = 1, 2...$.

The experimental results are presented in Table 1 gives the statistics for each sequence.

Table 1. Recognition Statistics for Connected Gesture Sequences

	Sequence Number (Length of Sequence)			
	1 (290)	2 (320)	3 (492)	4 (400)
Number of connected gestures	3	3	5	4
Number of correct/incorrect gestures	3/0	3/0	5/0	4/0
statistics per gesture				
Average Time (ms)	200	150	170	170
Number of Nodes Selected	280	275	335	300
Number of Nodes Expanded	122	175	209	148
Number of Computations	292	300	337	305

The graphical results obtained from the third sequence are as shown in Fig. 6. Graphical Results from the other sequences showed similar behavior.

Examination of all the three graphs in the Fig. 6 shows,
(1) At gesture interiors, the number of incorrect references decreases to zero as compared to the number of correct references which is a constant of 1 node, or more when state boundaries occur (shown by the peaks in the blue plot).
(2) At gesture boundaries, although the number of incorrect references increases as compared to the correct gesture references, but it is seen that within a short period of time the algorithm fixes on the correct gesture path.
(3) All the nodes selected are not expanded indicating less memory consumption. At each time frame T, only if the selected node, $i^{(m)} = (t, s_t)$, has $t = \tau = T$ then it is expanded to generate child nodes.
(4) The number of computations per time frame is slightly less then the nodes selected, since only the nodes whose $t < \delta_{th}$, δ_{th} = rejection threshold time, are computed otherwise the selected node is pruned online.

Observations (1) and (2) indicates the accuracy of the algorithm while (3) and (4) gives an insight to the time and memory usage.

A reduction of 61% in memory consumption is observed by using online pruning using the rejection threshold value. The value of the rejection threshold is an important criteria for maintaining the size of the open list. For large values the size of the open list maybe too large for practical online recognition, and lower values might empty the open list. Using a static rejection threshold value of 25, Fig. 7 shows that the node count in the open list is maintained at an average of 46 throughout. While without the online pruning the size open list rises constantly reaching the limit of memory capacity very soon.

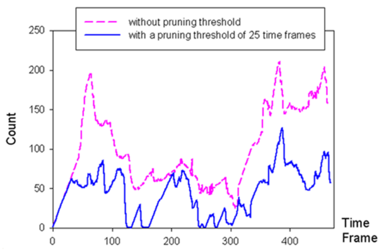


Fig. 7. Nodes in Open List

5.2 *Comparison with baseline methods*

The results obtained in Table 2 indicates that the Optimal Path Decoder achieves a reduction of almost $\frac{1}{2}$ in computation time without loss in accuracy. This is because the number of nodes considered by the Optimal Path Decoder is far less which also reduces the number of computations required per time frame. The data is based on the same sequence, as above, of length 492 and comprising of 5 connected gestures.

Table 2. Comparing Optimal Path Decoder with Viterbi Beam and N-Best algorithms.(values are recorded at the end of each recognized gesture)

Algorithm	Average time (ms)	Total Nodes	Total Number of computations	Accuracy
Viterbi Beam	350	700	742	100%
N-Best	280	610	670	100%
Optimal Path Decoder	170	336	337	100%

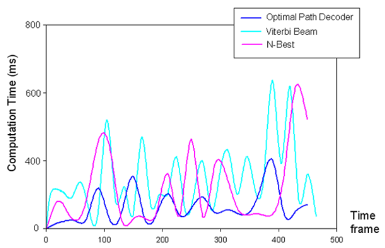


Fig. 8. Comparison of computation time at each time frame.

Figure 8 shows that the time required to fix on the optimal node by our algorithm at each time frame is comparable, if not better, to the time required to prune out the unwanted hypothesis in Viterbi Beam and N-Best algorithms. On closer examination, we see that Optimal path Decoder algorithm is able to obtain a reduction in computation time by a factor of 1.6 as compared to N-Best and approximately 2.0 compared to Viterbi beam search.

6. CONCLUSION

We have presented an online optimal path graph search algorithm which is
(a) computationally cheaper,
(b) reduces search space without overlooking the Optimal path.

We also introduced a rejection threshold time to monitor the nodes considered as we advance forward in DAG and prune online the unlikely path candidates.

The experimental results indicate that at least 60% fewer computations are required as compared with baseline methods like Viterbi Beam Search and N-Best algorithms. This leads to 15-20% speed up in recognition. The recognition experiments show that Online Optimal Path Decoder outperforms the baseline methods in both computation time and usage of search space.

REFERENCES

M. K. Brown and L. R. Rabiner. An adaptive, ordered, graph search technique for dynamic time warping for isolated word recognition. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 30(4):535–543, 1982.

F. S. Chen, C. M. Fu, and C. L. Huang. Hand gesture recognition using a real-time tracking method and hidden markov models. *Image and Vision Computing*, 21: 745–758, 2003.

N. Deshmukh, J. Picone, and Y. H. Kao. Efficient search strategies in hierarchical pattern recognition systems. *Proc. Southeastern Symposium on System Theory*, 1995.

J. Hu and M. K. Brown. On-line handwriting recognition with constrained n-best decoding. *Proc. ICPR International Conference on Pattern Recognition*, 1996.

I. Illina and Y. Gong. Improvement in n-best search for continuous speech recognition. *Proc. ICSLP International Conference on ASpoken Language Processing*, 1996.

H. K. Lee and J. H. Kim. An hmm-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10): 961–973, 1999.

S. Nakamura and P. Heracleous. 3-d n-best search for simultaneous recognition of distant-talking speech of multiple talkers. *Proc. ICMI IEEE International Conference on Multimodal Interfaces*, 2002.

C. W. Ng and S. Ranganath. A real-time gesture recognition system and application. *Image and Vision Computing*, 20:993–1007, 2002.

J. Y. Oh and C. W. Lee. Gesture recognition using shape and depth indformation of body for human-robot interaction. *Proc. ICAT International Conference on Artifial Reality and Telexistence*, 2004.

D. B. Paul. An efficient A* stack decoder algorithm for continuous speech recognition with a stochastic language model. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, 1992.

V. I. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

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

A. Ramamoorthy, N. Vaswani, S. Chaudhury, and S. Banerjee. Recognition of dynamic hand gestures. *Pattern Recognition*, 36:2069–2081, 2003.

F. Richardson, M. Ostendorf, and J. R. Rohlicek. Lattice-based search strategies for large vocabulary speech recognition. *Proc. ICASSP International Conference on Acoustics, Speech, and Signal Processing*, 1995.

B. J. You, Y. J. Choi, M. H. Jeong, D. Kim, Y. H. Oh, C. H. Kim, J. S. Cho, M. C. Park, and S. R. Oh. Network-based humanoids 'MAHRU' and 'AHRA'. *Proc. International Conference on Ubiquitous Robots and Ambient Intelligence*, 2005.