Coupled grouping and matching for sign and gesture recognition

Ruiduo Yang*, Sudeep Sarkar

Computer Science and Engineering, University of South Florida, Tampa, FL 33620, USA

ABSTRACT

Matching an image sequence to a model is a core problem in gesture or sign recognition. In this paper, we consider such a matching problem, without requiring a perfect segmentation of the scene. Instead of requiring that low- and mid-level processes produce near-perfect segmentation, we take into account that such processes can only produce uncertain information and use an intermediate grouping module to generate multiple candidates. From the set of low-level image primitives, such as constant color region patches found in each image, a ranked set of salient, overlapping, groups of these primitives are formed, based on low-level cues such as region shape, proximity, or color. These groups correspond to underlying object parts of interest, such as the hands. The sequence of these frame-wise group hypotheses are then matched to a model by casting it into a minimization problem. We show the coupling of these hypotheses with both non-statistical matching (match to sample-based modeling of signs) and statistical matching (match to HMM models) are possible. Our algorithm not only produces a matching score, but also selects the best group in each image frame, i.e. recognition and final segmentation of the scene are coupled. In addition, there is no need for tracking of features across sequences, which is known to be a hard task. We demonstrate our method using data from sign language recognition and gesture recognition, we compare our results with the ground truth hand groups, and achieved less than 5% performance loss for both two models. We also tested our algorithm on a sports video dataset that has moving background.

1. Introduction

Gesture recognition, and the related area of automated sign language recognition, is a rich area of research (see [1,2] for reviews) with many different applications and approaches, but sharing some common problems and solutions. Vision-based approaches all share the problem related to the vagaries of low-level segmentation. The states in a state space based gesture representations, such as the Hidden Markov Model [3–5], Dynamic Time Warping [6] or Finite State Machine (FSM) [7] approaches are based on the low-level features detected in the image. Motion tracks in trajectory based gesture recognition approaches [8,11,12] are dependent on the robustness of the tracking process which, in turn, is dependent on the stability of the low-level segmentation. This problem of low-level segmentation is sometimes addressed by engineering the imaging setup so as to ease the segmentation of hands by using controlled lighting, colored gloves or even non-vision based aids such as magnetic or optical markers. Pure vision-based solutions usually rely on skin color [13] and/or motion information [12] to detect hands. However, approaches based on predefined skin color models suffer from sensitivity with respect to changing illumination conditions. Motion-based hand segmentation approaches rely on the assumption that the features important for gesture will be associated with motion. This is not always true for sign recognition, which includes movement and hold phases. Fusion [14,15], multi-modal [16], Haar-like Features [17], accelerometers [18] and 3D [19,20] approaches can be used to arrive at better segmentation and detection. However, segmentation will never be perfect. Not only will there be missed detections, but there will also be false alarms. There is danger that these errors are propagated to the recognition stage. In this work, we advocate using an intermediate grouping module, coupled with the recognition module, to handle low-level segmentation errors. Such grouping processes have been found to be useful for object recognition tasks [21–23], but have not been used for gesture and sign recognition. The combination of top-down and bottom-up approaches in gesture sequence recognition can be found in [24] and [25]. Although these approaches can handle multiple candidate observations, there are no grouping processes incorporated. For example, a sliding window is used along with skin color model in both [25] and [24] to obtain the position of the moving hands. However, in real world application, bad lighting conditions may cause problems for skin color approaches, and a sliding window cannot be sufficient in some applications where exact hand shapes are needed. Apart from hand gesture, Srinivasan et al. also proposed grouping method [26] to classify human bodies, but their approach works for single images. Bar-Shalom et al. [9,10] used


* Corresponding author.
E-mail addresses: ryang@csee.usf.edu (R. Yang), sarkar@csee.usf.edu (S. Sarkar).

ARTICLE IN PRESS

Computer Vision and Image Understanding xxx (2009) xxx–xxx

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

© 2008 Elsevier Inc. All rights reserved.
data association techniques to track multiple targets, but their approach does not couple with a recognition module, which is necessary for recognizing an articulated object such as hand gesture and body gesture.

Our approach is depicted in Fig. 1. We use a top-down recognition process and bottom-up grouping process, integrated in a dynamic programming framework. First, we segment the image into a collection of disjoint regions. These disjoint regions are our grouping primitives. Some of these primitives are selected as our seed patches. Then we use a greedy search based grouping approach to generate groups representing possible hands. We start from the seed patches, progressively adding new adjacent primitives, followed by checking to prune out the bad groups. We generate layers of groups, with each layer based on one attribute such as color, proximity, or boundary gradient. The generated groups are not disjoint. Notice the color attributes are used as a similarity measure, that absolute value of color is not used to detect hands. The generated groups are then linked across adjacent frames to generate a set of candidate group sequences. Finally, we match each model sequence to the linked group structure to find the best match and, simultaneously, a matching score between the model sequence and the input sequence. We show this matching can be conducted for both deterministic and statistical models. Based on the matching score, we use a simple nearest neighbor rule for recognition. By using this approach, we significantly reduce the need for perfect segmentation at the first step.

The models we use in our system includes both statistical models (HMMs) and deterministic models. For the deterministic model, we simply store sequences of training signs in the database and match them with time warping techniques, which is essentially a dynamic programming process. A similar matching process can be seen at [25], with multiple observations but with no grouping. For HMMs, the structure is depicted in Fig. 2. We match each gesture HMM to the linked group structure to simultaneously compute the matching score and the best possible grouping for each frame. This HMM part of the work is an expanded version of our work presented in [27]. The current paper offers an unified framework for handling segmentation uncertainties for both deterministic and statistical models.

We demonstrate our method using three different datasets. The first one is an American Sign Language (ASL) dataset that was collected by us. This dataset was collected without the aid of any gloves or markers. The number of training samples in this dataset is limited. We demonstrate the deterministic approach on this data. The second dataset we consider is a two-view hand gesture dataset that was collected by Just and Marcel [33]. This dataset has overhead images of hand gestures against a complex background, which makes hand segmentation hard. Although the dataset has been collected with color gloves, we do not use the color information for each hand to construct a hand color model. We use color information only as a similarity measure for grouping.

![Fig. 1. Architecture of the recognition algorithm. First, the original frames are segmented, and the groups in each frame as well as the links between frames are produced. Then the sequences of linked candidate groups are matched to the model groups in the database.](image1)

![Fig. 2. For HMM, we do not have a unique observation sequence to match. Rather we have a collection of possible observation sequences, implied by the sequence of multiple observations at each frame.](image2)
For the third test, we used sport videos that were used in [34]. We showed how our approach can be used to detect a particular sports actions, such as “diving action”. We choose this action as the corresponding data was collected with moving camera and this would help demonstrate that our approach can be used with moving backgrounds too. Along with recognition, we show the ability to segment the diver from the moving background.

The rest of the paper is organized as follows. Section 2 describes the grouping process. Coupling with a deterministic approach is described in Section 3. Coupling with a statistical approach is described in Section 4. Experimental results are shown in Section 5, and we conclude with Section 6.

2. Grouping of low-level primitives

Low-level processes are never perfect. Skin color is the most commonly used cue for segmenting image parts from the hand or face in gesture analysis. Our low-level primitives of the grouping process are constant color (or intensity for gray level images) region patches. The nature of the algorithm used to detected these patches are not particularly important. We used the mean shift segmentation algorithm [29], which is fast and effective, to generate these patches based on color or intensity. However, this does not always produce perfect segmentation, with over segmentation being a particularly hard problem to handle. Fig. 3 shows an example of the illumination and shading change that can be found in a gesture sequence. If parts of the image from the finger and the palm are not grouped together, high-level matching will be starved of crucial information related to recognizing finger-spelled words in sign language recognition. To help overcome this problem of over segmentation, we use an intermediate grouping process. To shortchange the subsequent recognition process by insisting on disjoint groups, as is usually the practice in grouping, we allow for overlapping groups, resulting in redundant sets of groups. Some region patches are selected as seeds based on their size. This redundancy should help us counter grouping errors. We then grow these seeds with adjacent regions to generate larger groups. As the seeds are grown, groups are checked for being possible hands based on size and shape. Grouping can be conducted based on color, position, boundary smoothness or boundary gradient. These basic similarity cues resemble those adopted by Hoogs and Mundy to group region patches [28] for object recognition, where they used spatial intensity, parallelism and perimeter to form an object hypothesis. However, unlike them we perform the grouping based on each criterion independently of the other. Each criteria result in a set of groups, which we refer to as a grouping layer. Thus, we have grouping layers: color grouping layer, proximity grouping layer and boundary smoothness grouping layer. The grouping strategy used is the same for each layer and is discussed next.

2.1. Grouping process

Let the set of low-level primitives detected in the k-th image frame be denoted by $S_k = \{p_{k1}^1, \ldots, p_{nk}^k\}$. A grouping, $C_i^k$ of these region primitives will represent a subset of the primitives, $\{p_{k1}^i, \ldots, p_{nk}^i\}$.

We adopt a greedy approach to form the groups, outlined in the flowchart in Fig. 4. From the initial set of primitives $S_k$, we select a subset of primitives that are likely to come from hands, based on the size of the patch. These are our seed patches. Given some knowledge of the approximate size of hands in the sequence, we can eliminate large, non-homogeneous region patches from further consideration. We use a list $L$ to store the possible groups. This list is initialized by choosing each selected primitive to be a singleton group. These groups would be merged to form larger agglomerates.

$$L = \{(p_x^i) | a_x(p_x^i) \leq \text{size}, x = 1, \ldots, N_k\}$$

Here, $a_x$ is the operator that returns the size of $p_x$. For the entries in $L$, we maintain an adjacency graph, whose nodes are the groups in $L$, and links exist between groups that share a boundary. This graph is incrementally updated at each iteration. Fig. 5 shows an example local adjacency graph (Look ahead to Fig. 6 for a sequence of iterations of this graph.).

The grouping process starts by picking the first group in $L$, denoted here by $p$, and searches its neighbors ($N_p$). Each neighbor $N_p$ is considered for grouping with $p$ to generate a tentative larger grouping. We select the best local grouping, and denote it as $g$. In color layer, the best neighbor is the one that has smallest Euclidean distance with the base group in the RGB space; in the proximity layer, we choose the neighbor that is nearest to the base group according to the image coordinates of their centers; in boundary

![Fig. 3. Appearance changes over time. Notice the brightness variation of the left hand of the subject.](image-url)
layer, the neighbor that yields the smallest curvature score when grouping with the base group is selected as the best.

The group $g$ is further tested to see if it can possibly represent a hand. This test is based on three attributes: $[a_n, a_c, a_{cur}]$, where $a_n$ is the number of primitives in the group, $a_{cur}$ is the boundary curvature of the group, and $a_c$ is the size of the bounding box.

$$ (a_n \leq t_{size}) \land (a_{cur} \leq t_{curvature}) \land (a_c \leq t_{num}) $$

The test is conducted based on the result of Eq. 2, where $t_{curvature}$, $t_{num}$, and $t_{size}$ are the corresponding thresholds. Here the boundary curvature is approximated as the integral of the squared root of second order derivative along the curve. If the group $g$ passes this test, it is inserted into the final candidate group list, $C$, else if $a_c \leq t_{size}$ it is inserted at the end of the list $L$, to be considered for further grouping.

Fig. 6 shows us the grouping process based on the adjacency graph, where the mechanism is essentially a greedy search process on the adjacency graph, starting from a chosen seed. In Fig. 6, a solid link represents a grouping between two nodes. Starting from the seed patch $S$, a decision is made to group $S$ with the best neighbor, denoted by $N_c$ according to a layer $c$. Then we generate the new group $G_c$. After the grouping the adjacent graph is updated, where the new neighborhood will be the neighborhood of $G_c$, and the process starts again to group one of $G_o$'s neighbors with $G_c$ based on the same layer criteria used to form the seed primitives. The above process is used to generate 3 grouping layers based on color, position and boundary gradient, respectively. This process reduces the possibility that the group corresponding to the hand will not be generated. One may also just generate groups based on one layer, typically the color layer. It is a trade off between getting correct group and the time/space spent on it.

Note that the low-level primitives and the groups are formed on a frame by frame basis; there is no tracking or frame-to-frame correspondence. Fig. 7 shows us the grouping results for 3 different frames at the color grouping layer and the proximity grouping layer. For frame 1 in Fig. 7a, the color grouping layer in Fig. 7c includes the real hand group (shown with white circle), while the proximity grouping layer at Fig. 7d failed to include it. For frame 2, however, the proximity grouping layer gives us the true hand, the color grouping layer does not. For frame 3, both color grouping layer and proximity grouping layer have the true hand group in their list. Also we can see in Fig. 7 how the groups differ from each other in terms of missing fingers or added extraneous regions; this can confound the sign recognition process. Also note that we do not restrict ourselves to disjoint groups; thus we might have $G_i \cap G_j \neq \emptyset$. This is different from the usually employed disjoint groups constraint employed in segmentation and grouping. Allowing for overlapping groups allows us to avoid making hard decisions about group boundaries.

2.2. Associating groups across frames

We denote the $j$th group detected in $k$th frame as $G^k_j$. The groups detected in each frame are associated with those detected in previous frames to result in a linked sequence of groups spanning all of the frames. This structure help us propagate constraints during the matching process and restrict considering exponentially large numbers of possible observation sequences. We define the predecessors sets of each element in each groups set as

$$ Pre(G^k_j) = [G^{k-1}_{j-1}, \ldots, G^{k-1}_{j+1}] $$

where $G^{k-1}_{j+1}$ is one possible predecessor of $G^k_j$. The predecessor relationship between the groups from different time instants is based on feature similarity. It captures how likely the groups are from the same underlying cause in the image. Specifically, we test the difference in feature size and location between the two groups, with a liberally chosen threshold value.
3. Coupling groups with deterministic matching algorithm

We perform recognition based on the fragmented observations based on both a deterministic approach and a statistical approach. For a deterministic approach, recognition is conducted by matching groups found in any given sequence to each model sequence. In the model sequence, the real hand group is extracted manually frame by frame. The goal of the matching is to find one candidate group sequences (out of the many available, directed by the linked structure), which can be best mapped to the model sequence. This process also allows for time warping, and is shown to be solvable by dynamic programming. After matching to each model sequence, the ones with lower distance scores are considered as the recognition results.

3.1. Formulation of the matching process

Let the $i$th candidate group in the $k$th frame be represented as $C^i_k$. Also let $K$ be the number of frames in the test sequence. Similarly, the motion model will consist of a sequence of feature vectors, $M = \{m^1, ..., m^K\}$ that will have to be matched to the sequence of candidate groups, with each model feature vector mapped to one group. The matching score is represented with a 3D Matrix $S$, where the $(ijg)$ element of $S$ denotes the Mahalanobis distance with diagonal covariance matrix between the $i$th model feature vector and the $g$th candidate group's feature vector in the $j$th frame.

The warping path is a sequence of elements of $S$ denoting the matching. Since the model group can be mapped to one of the

---

Fig. 6. Example of the generation process of candidate groups (hands) starting from one seed. The process is repeated for each primitive as a seed.
candidate groups in one of the test sequence frames, the warping is conducted in both the time domain and the candidate group domain. If the cardinality of the candidate group’s feature vector set is one, then of course this correspondent establishment is trivial (only time warping is needed). Otherwise, we have to select between the possible candidate groups. We cast this problem as a minimization problem that we solve using dynamic programming.

Formally, we have to find a sequence of elements, one from each candidate set, that best matches the model sequence of feature vectors. Let,

$$(1) \ k(t) = (i, j, g)$$

be a multi-valued function that maps the indices of the warping path, denoted by $t$, to the 3D coordinates in $S$ where the model’s $i$th frame is matched with the $g$th candidate group of the $j$th frame in the test sequence.

$$(2) \ d(m^i, C^g_j)$$

represent the cost of matching the model group feature vector from the $i$-th image frame, $m^i$ with the $g$-th group feature vector from the $j$-th image frame, $C^g_j$.

Then the total matching cost can be cast as a minimization problem, more formally,

$$\min_{k(t)} \left( \sum_t d(m^i, C^g_j) \right) \quad (4)$$

Fig. 8 illustrates the minimization space. It is a 3D space spanned by the model sequence time index, $i$, the given image sequence time index, $j$, and the feature vector index into the candidate group sets, $g$. Each point in that space is associated with a cost defined between the corresponding image and model groups.
We seek a curve, defined by \( k(t) \), which minimizes the total cost function over this curve, with the following constraint in both the candidate group domain and the time domain. When we match a gesture to a gesture, where a model sign with length \( T_m \) is matched with a test sign with length \( K \), this curve starts at \( <i = 1, j = 1> \) and ends at \( <i = T_m, j = K> \). When we match a sign/gesture to a sign sentence, where a model sign with length \( T_m \) is matched with a sentence with length \( K \), this curve can start at any place with \( i = 1 \) and end at any place with \( i = T_m \). We also enforce a constraint when associating adjacent frames. This constraint defines all of the possible predecessors of a node on the warping path.

### 3.2. Dynamic programming

The dynamic programming can be used to obtain the optimal warping path in our problem. In a 3D matrix \( D \), let \( D(i,j,g) \) represent the minimum cumulative cost of matching the model sequence, \((m^i, \ldots, m^j)\), to the candidate group set sequence up to \((i,j,g)\). The optimal substructure of the problem allows the following recursive formula.

\[
D(i,j,g) = d(m^i, G^j) + \min \left\{ \begin{array}{l}
D(i, j-1, r) \\
D(i-1, j, r) \\
D(i-1, j-1, r)
\end{array} \right.
\]

(5)

Here we used a constraint that the coordinate \((i,j,g)\) in the dynamic programming is dependent on the locations, \((i-1,j,g), (i,j-1,g), \) and \((i-1,j-1,g)\). This is based on the general local constraints [31]. The solution to \( D(i,j,g) \) is the solution to our problem.

### 4. Coupling with Hidden Markov Models

In this section, we show the grouped candidate sequence can also be matched to a statistical model like HMM. While the structure and the training of the HMM is a fairly standard one, the decoding process, i.e. computing the likelihood of an image sequence to the HMM, is significantly different and new. Each gesture is modeled using an HMM \( \lambda_i \) over \( N \) states. The state at frame \( k \) is denoted as \( q_k \), where \( q_k \in \{1, \ldots, N\} \) is the state-transition matrix. The initial state distribution is denoted as \( \pi_i = P(q_1 = i) \) is the probability that state is \( i \) at the first frame. The observation probability is modeled as a mixture of Gaussian, the observation vector is denoted as \( O = [O_1, \ldots, O_D] \) with \( K \) to be the length of \( O \), its probability at state \( j \) is computed as \( b_j(O) = \sum_{c=1}^{M} c_j(O, H_j, \sigma_j) \) where \( \Omega \) is a Gaussian with \( \mu_j \) as the mean vector and \( \sigma_j \) as the covariance matrix, \( c_j \) is the mixture factor and \( M \) is the number of mixture components. At training, we have observation sequences \( O = O_k, j = 1, \ldots, K \), the above parameters \( [a_{ij}, \pi_i, c_j, H_j, \sigma_j] \) are found to maximize the likelihood \( P(O, \lambda) \). We use the Baum–Welch estimation process to train the HMM.

The decoding or matching process is different from conventional HMMs. In conventional HMM, the actual state sequence is unknown, but the observation sequence is unique. However, in vision gesture application, we consider the observation sequence to be non unique. In conventional HMM, the input observation feature vector \( O = [O_1, \ldots, O_D] \) is known for each frame and the likelihood \( P(O|\lambda) \) can be computed using an iterative forward pass process. In our framework, however, we do not assume that we know the exact observation vector \( O_k \) at each frame \( k \). Instead, we allow for multiple hypotheses about the observation. At frame \( k \) we have the group set \( G = \{G_1, \ldots, G_K\} \), where each element in \( G \) is one possible observation and \( c_k \) denotes the total number of groups in frame \( k \). We assume only one element in the observation set is the true observation. We do not decide upon the best group for each frame independently of the others. The entire sequence of group sets is used as the input. We will discuss the problem related to the optimal observation sequence and proposed three approaches to compute the matching score with such an input.

#### 4.1. Maximal observation, summed state

We are given a sequence of group sets \( G = \{G^1, \ldots, G^K\} \), where \( G^k = \{G^i_k, \ldots, G^K_k\} \), \( 1 \leq k \leq K \) is the group set at frame \( k \). The optimal observation sequence problem is to find one group sequence \( \psi \) that maximizes the likelihood, summed over possible HMM state transitions, \( P_{sum}(\psi|\lambda) \), where \( \lambda \) is the HMM and.

![Fig. 9. An illustration of the indexed forward process, the horizontal line represents the time, the vertical line corresponds to the candidate observations and the sub-vertical line denotes the N states. Note at each time step, only one best observation is selected based on the previous selected observations and the forwarding results. In this example, the optimally selected observations (circled ones) are <1, 2, 2, 3>.

\[
\psi = (\psi_1, \ldots, \psi_k), \psi_i \in G^j, 1 \leq k \leq K, \psi_{k-1} \in \text{Pre}(\psi_k)
\]

We denote the maximum value of likelihood by

\[
P_{\text{max,sum}}(G|\lambda) = \max_{k=1,\ldots,K} P_{\text{sum}}(\psi^k|\lambda)
\]

where \(K\) is the number of all possible sequence of groups. The probability \(P_{\text{sum}}(\psi^k|\lambda)\) represents the likelihood of the group sequence, summed over all possible HMM state sequences. For each sequence of groups the computation of \(P_{\text{sum}}(\psi^k|\lambda)\) can be done using the standard forward–backward algorithm used for HMMs.

A brute force solution for Eq. 7 will be to enumerate across the sets \(G^1, \ldots, G^K\) to get all possible observation sequences \(\psi^1, \ldots, \psi^K\), compute likelihood for each of the observation sequences, and select the maximum value. Obviously, exhaustive enumeration is computationally expensive; hence we resort to approximation based on incremental construction of the optimal sequence. Note we will use the superscript of \(\psi\) to refer to an instance of sequence, while we will use subscript of \(\psi\) to refer to an index of one frame in the sequence.

To find the best group at frame \(k\), suppose the observation sequence at frame \(1, \ldots, k-1\) has been recovered as \(\psi_1, \ldots, \psi_{k-1}\). We define the indexed forward variable \(a^k(i)\) as:

\[
a^k(i) = P(\psi_1, \ldots, \psi_k, q_k = i, \psi_k = G^j|\lambda)
\]

that is, the probability of the partial observation sequence \((\psi_1, \ldots, \psi_k)\) at frame \(k\) the state is \(i\) and the observation vector is \(G^j\), and \((\psi_1, \ldots, \psi_{k-1})\) is the observation vectors we have found at time \(1, \ldots, k-1\).

The initialization of the variable is:

\[
a^1(i) = \pi_i b_i(G^1);
\]

![Figure 10](image_url)

**Fig. 10.** Candidate groups from one frame in the dataset. While grouping we set \(i_{\text{num}} = 10\). In (d), there are 125 groups generated, the groups with a circle are the real left and right hands.
and we have

\[ \psi_1 = G_p, p = \arg \max \sum_{i=1}^{N} \alpha^1_i (i) \]  

(10)

The induction solution is

\[ \alpha^k_{i+1}(i) = \left[ \sum_{t=1}^{N} b_i (G^k_{t+1}) \right] b_i (G^k_{p}), \psi_k = G^k_p \]  

(11)

and then \( \psi_{k+1} \) is selected as:

\[ \psi_{k+1} = G_{p}, p = \arg \max \sum_{i=1}^{N} \alpha^k_{i+1}(i) \]  

(12)

At frame \( K \), the observation vector sequence is computed as \( \langle \psi_1, \ldots, \psi_K \rangle \). At the same time, the probability of this observation sequence given the HMM, can be computed as:

\[ \text{(d) Candidate Groups without grouping} \]

\[ \text{Fig. 11. Candidate groups from one frame in the dataset. While grouping we set } r_{num} = 1, \text{ basically no grouping. In (d), there is no grouping process, just a segmentation, we can see the hand are highly fragmented, without grouping we cannot get the real hand in the list.} \]

**Table 1**

List of matched scores for three test sentences. The signs with \( \checkmark \) are actually in the sentence, with the scores listing beside them.

<table>
<thead>
<tr>
<th>Test</th>
<th>TICKET BUY FINISH</th>
<th>PEOPLE LONGLINE WAIT ANGRY</th>
<th>GATE WHERE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank</td>
<td>In</td>
<td>Signs</td>
<td>Scores</td>
</tr>
<tr>
<td>1</td>
<td>/</td>
<td>Buy</td>
<td>1.51</td>
</tr>
<tr>
<td>2</td>
<td>/</td>
<td>I</td>
<td>2.24</td>
</tr>
<tr>
<td>3</td>
<td>Wait</td>
<td>2.35</td>
<td>Buy</td>
</tr>
<tr>
<td>4</td>
<td>/</td>
<td>Ticket</td>
<td>2.44</td>
</tr>
<tr>
<td>5</td>
<td>/</td>
<td>Finish</td>
<td>3.23</td>
</tr>
<tr>
<td>6</td>
<td>/</td>
<td>You</td>
<td>6.89</td>
</tr>
<tr>
<td>7</td>
<td>Have</td>
<td>7.07</td>
<td>Not</td>
</tr>
<tr>
<td>8</td>
<td>/</td>
<td>Have</td>
<td>4.26</td>
</tr>
<tr>
<td>9</td>
<td>Need</td>
<td>7.5</td>
<td>Again</td>
</tr>
<tr>
<td>10</td>
<td>Phone</td>
<td>8.38</td>
<td>Airplane</td>
</tr>
<tr>
<td>11</td>
<td>Mean</td>
<td>8.87</td>
<td>Ticket</td>
</tr>
<tr>
<td>12</td>
<td>Suitcase</td>
<td>9.11</td>
<td>Lipread</td>
</tr>
<tr>
<td>13</td>
<td>/</td>
<td>Gave</td>
<td>5.11</td>
</tr>
<tr>
<td>14</td>
<td>It</td>
<td>10.1</td>
<td>Phone</td>
</tr>
<tr>
<td>15</td>
<td>Can</td>
<td>10.52</td>
<td>Finish</td>
</tr>
<tr>
<td>16</td>
<td>People</td>
<td>11.17</td>
<td>Just</td>
</tr>
<tr>
<td>17</td>
<td>Again</td>
<td>11.58</td>
<td>Key</td>
</tr>
<tr>
<td>18</td>
<td>Gave</td>
<td>11.95</td>
<td>Gate</td>
</tr>
<tr>
<td>19</td>
<td>/</td>
<td>Not</td>
<td>12.62</td>
</tr>
<tr>
<td>20</td>
<td>Gate</td>
<td>13.68</td>
<td>Need</td>
</tr>
</tbody>
</table>

**Table 2**

List of matched position and the manual recognized position (in the bracket) in the test sequence.

<table>
<thead>
<tr>
<th>Test</th>
<th>TICKET BUY FINISH</th>
<th>PEOPLE LONGLINE WAIT ANGRY</th>
<th>GATE WHERE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank</td>
<td>Signs</td>
<td>Start</td>
<td>End</td>
</tr>
<tr>
<td>1</td>
<td>Ticket</td>
<td>18(18)</td>
<td>21(21)</td>
</tr>
<tr>
<td>2</td>
<td>Buy</td>
<td>28(28)</td>
<td>31(31)</td>
</tr>
<tr>
<td>3</td>
<td>Finish</td>
<td>42(42)</td>
<td>47(47)</td>
</tr>
<tr>
<td>4</td>
<td>Angry</td>
<td>99(100)</td>
<td>107(108)</td>
</tr>
</tbody>
</table>

\[ P(\psi_1, \ldots, \psi_K | \lambda) = \max_{i} \sum_{j=1}^{N} \varphi_j(i) \] (13)

Fig. 9 illustrates us the indexed forward process. The summation of the product of the forward variables and the observation probabilities remain the same as in the conventional HMM. The difference is that we take the observation vector dynamically depending on the previously decided observations. Note the result of Eq. 13 is not an exact solution for Eq. 7. Instead it is the solution to select the best current observation based on a certain selected partial observation sequence.

4.2. Summed observation, summed state

Instead of considering the maximum probability over all possible group sequences, we could consider the summation over all possible group sequences. Thus, the probability of interest is

\[ P_{\text{sum}, \text{sum}}(G | \lambda) = \sum_{k=1}^{K} P_{\text{sum}}(w_k | \lambda) \] (14)

where the possible sequence of groups are \( \psi_1, \ldots, \psi_K \). The probability \( P_{\text{sum}}(w_k | \lambda) \) represents the likelihood of the group sequence, summed over all possible HMM state sequences. As before, for each sequence of groups the computation of \( P_{\text{sum}}(w_k | \lambda) \) can be done using the standard forward–backward algorithm used for HMMs. However, we found the process of summing over all group sequences and over all state sequence can be effectively merged in the dynamic programming process. To do this, we defined the grouping forward variable \( k'_{j}(i) \) as:

\[ k'_{j}(i) = \sum_{\psi_1, \ldots, \psi_{i-1}} P(\psi_1, \ldots, \psi_{i-1}, \psi_i = i, \psi_k = G_j | \lambda) \] (15)

That is, the summation of the partial probability of all the group sequences that have \( \psi_k = G_j \) and \( q_k = i \). The initialization is

\[ k'_{1}(i) = \pi_i b_i(G_1) \] (16)

The induction is

\[ k'_{j+1}(i) = \sum_{p \in \text{Prev}(G_j)} \sum_{t=1}^{N} k^p(t) a_{ij} b_i(G_{j+1}) \] (17)

And, the result of Eq. 14 is obtained at the end of the process:

\[ P_{\text{sum}, \text{sum}}(G | \lambda) = \sum_{p \in \text{Prev}(G_j)} \sum_{t=1}^{N} k^p(t) \] (18)

4.3. Maximal observation, maximal state

The third quantity of interest is maximum probability over all possible group sequences and HMM state sequences. Thus, the probability of interest is

\[ \text{Fig. 12.} \quad (a) \text{The matched sequences in the candidate-time space of the test sentence: (PEOPLE LONGLINE WAIT ANGRY)} \quad (b) \text{The matched sequences (time warped) of the test sentence} \quad (c) \text{The recovered hand position (x coordinates).} \]

(a) PEOPLE

(b) The recovered right hand position (x coordinates) for sign PEOPLE

(c) LONGLINE

(d) The recovered right hand position (x coordinates) for sign LONGLINE

(e) WAIT

(f) The recovered right hand position (x coordinates) for sign WAIT

(g) ANGRY

(h) The recovered right hand position (x coordinates) for sign ANGRY

Fig. 13. The recovered right hand position and their corresponding hand movements in the test sequence.
\[ P_{\text{max}} \max (G|\lambda) = \max_{\psi_1 ... \psi_k} \max_{q_1 ... q_k} P(\psi_1, ..., \psi_k; q_1, ..., q_k|\lambda) \]

where the possible sequence of groups are \( \psi^1, ..., \psi^K \) and \( q_1, ..., q_k \) is a HMM state sequence. This quantity can again be computed using dynamic programming. We define the max-forward variable \( z^*_k(i) \) as

\[ z^*_k(i) = \max_{\psi_1,...,\psi_i} P(\psi_1, ..., \psi_i, q_k = i, \psi_k = G_k^i|\lambda) \]

This is the maximum partial probability among all the group sequences that have \( \psi_k = G_k^i \) and \( q_k = i \). The variable \( z^*_k \) represents the backtrack index of the observations for the corresponding max-backward process. The initialization is

\[ z^*_1(i) = p_i(C_1^i) \]

The induction is given by

\[ z^*_k+1(i) = \max_{p(C_k^j)} [z^*_k(j)|a_i(C_k^j)] \]

\[ z_k = \arg \max_{p(C_k^j)} [z^*_k(j)|a_i(C_k^j)] \]

\[ z_1, z_2, ..., z_K \] is obtained for the best group sequence (over the best state sequence) and this group sequence can be used to get the matching score.

5. Experiments and results

We have conducted experiments on three datasets using the grouping approach, coupling with deterministic and probabilistic models. For sign recognition, since the number of signs is large while the number of instances of each sign is relatively small, we use the deterministic sample-based model to match. For the gesture recognition dataset, where we have sufficient number of instances of each gesture, we use Hidden Markov Models with multiple candidate groups to recognize. For the sports action dataset, we demonstrate the ability of the matching process to work in the presence of camera motion.

5.1. Experiments with American Sign Language dataset

The vocabulary of the ASL dataset consists of words a hearing-impaired person would need to communicate with security personnel at airports. The video data is taken at 30 fps, with image resolution by 460 by 290. There are 39 different signs that are articulated in 25 different continuous sentences. Note that some signs appear more than once in some sentences. There are five samples of each sentence. The total number of individual sign instances in the dataset is 68–70 for each sample. We use four samples as model instances and the other to test. There are some variations between multiple instances of the same sentence. This is introduced by signing the same sentences differently. For example, the English sentence ‘if the plane is delayed, I’ll be mad’ can be signed as ‘AIRPLANE POSTPONE AGAIN, MAD I’ as well as ‘AIRPLANE AGAIN POSTPONE, MAD I’. The dataset consists of the same one subject in the test and training set.

The model sign dataset is formed from four of the five instances of each sentence. Specifically, for each sign we have four examples. We manually select the groups of region patches that are from each hand frame by frame. The sequence of these manually selected groups form the model sequences. Since the number of training samples is limited and it will be hard to estimate the HMMs accurately, we use the deterministic matching in this experiment.

For feature vectors, we fit each hand group with an ellipse in a least square error manner. The ellipse has a major axis \( a \), minor axis \( b \), the angle between major axis and \( x \) axis is \( \theta \). We then have a 10 dimensional feature vector to represent the hand group: \( x \) image coordinate of the centroid of the group, \( y \) image coordinate of the centroid of the group, motion displacement in \( x \) direction, motion displacement in \( y \) direction, length of major axis \( a \), length of minor axis \( b \), sine of \( 2\theta \), cosine of \( 2\theta \), eccentricity of the ellipse, and area of the ellipse. We choose all the groups which has a size smaller than 80% of the image height and which ranked top 500 based on the curvature score.

This gives us a very liberal threshold and allow most groups to be fed into the recognition algorithm.

Fig. 10 shows us an example of the generated groups for one frame. As we can see, even for the simple background and simple clothes, the hand can be very fragmented. Fig. 10d has more than 100 candidate groups, where the real hands can be generated during the grouping process. Without grouping, we cannot guarantee to have the real hand in the candidate list, as shown in Fig. 11d. We match each word model to the entire test sentence. The matched signs are ranked according to their matching scores. The first rank is the lowest score, and so on. Table 1 shows us the actual list after ranking the matching score for a few sentences.

with √ are actually in the sentence, with the scores listed beside them. The correct signs are towards lower rank, which is what we want. We use these ranks to estimate performance. In a full system the output of our algorithm would be integrated with grammar model to generate complete interpretation of the sentence. Ideally, the output of our approach should have the right signs at the top ranks. That is, if there are three signs in a sentence then the top three ranked sign should be the correct ones; the relative ordering between these three do not matter at this stage. Note that this result is without using higher level grammar. Table 2 shows the result for the same sentences, but with the matched starting and ending point listed. Ground truth starting and ending points are in the brackets.

Fig. 12 shows us one match result for the test sequence: (PEOPLE LONGLINE WAIT ANGRY). Fig. 12a is the warped path in the 3D space where the warping is from both the candidate hands selection and the time warping. Fig. 12b shows the projection of the same data onto TestSequence − TrainSequence plane. Fig. 12c, is the projection of the same data onto TestSequence−CandidateGroups plane, which reveals the detected hand’s x coordinates. Fig. 13 shows us the recovered position of the hand x coordinates and their hand movements in the test sequence. The result is shown in four parts, each of which corresponds to a sign in the sentence. These results show the hand position is recovered as a byproduct of the recognition even when the hand is crossing the face. This is a particularly hard problem to overcome in gesture recognition.

The overall recognition result for this database is shown at Fig. 14. Each model word is matched to the test sequence and the results are ranked according to their matching score. Those words which are in the original sentence and have a smaller or equal ranking than 1–10 will be counted as one correct recognition.
for the ranking 1–10, respectively. Fig. 14 shows us the performance under a different number of primitives allowed in one group, from 1, 5 to 10, 20. The upper curve is the recognition rates achieved using manually selected hands.

We can see the recognition results increase when the number of primitives increases from 1, 5 to 10. There is a slight drop on 20, which is due to the introduced noisy groups. The overall performance drop comparing with the manually selected hands at rank 5 is within 5%.

Table 3

<table>
<thead>
<tr>
<th>Low-level matching</th>
<th>Grouping sum–sum</th>
<th>No grouping sum–sum</th>
<th>Grouping sum–sum</th>
<th>No grouping sum–sum</th>
<th>Grouping sum–sum</th>
<th>No grouping sum–sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of total test frames</td>
<td>7249</td>
<td>7249</td>
<td>7249</td>
<td>7249</td>
<td>7249</td>
<td>7249</td>
</tr>
<tr>
<td>No. of groups/frame/view</td>
<td>94</td>
<td>22</td>
<td>94</td>
<td>22</td>
<td>94</td>
<td>22</td>
</tr>
<tr>
<td>No. of total sample seq.</td>
<td>210</td>
<td>210</td>
<td>210</td>
<td>210</td>
<td>210</td>
<td>210</td>
</tr>
<tr>
<td>No. of correct samples seq.</td>
<td>193</td>
<td>58</td>
<td>190</td>
<td>60</td>
<td>191</td>
<td>61</td>
</tr>
</tbody>
</table>

Fig. 17. Recognition performance of 210 instances of seven hand gestures for five different approaches. The first two are based on manual and color based segmentation of the hands. The next three does not use the knowledge of the hand color and take into account fragmented observations. The three corresponds to the three different kinds of probabilities that can be computed, \( P_{\text{sum, sum}} \), \( P_{\text{sum, sum}} \), and \( P_{\text{max, max}} \) using the HMM proposed in this paper.

Notice the modeling of sign dynamics is very weak with the simple nearest neighbor rules. At rank 7 we achieved a recognition rate around 90%. Associated errors can be handled in the higher level using grammar constraints. We have started to consider this in [32, 30].

5.2. Experiments with hand gesture sequences

We also present results on the publicly available Human Computer Interaction (HCI) dataset that has been recently collected by another research group, i.e. Just and Marcel [33]. The dataset is for recognizing seven hand actions: push, rotate front, rotate
back, rotate left, rotate right, rotate up, and rotate down. The authors of the data have explicitly separated the training and test data, where the training data consists of four subjects, each of whom performed the seven actions 10 times, with five of them at one session time and five of them at the other, and the test data has the same gestures but with three different subjects.

The total number of test sequences is 210. The dataset has samples from two fixed cameras, one sample from the left side and the other from the right side. We fused the scores from two views for each gesture by using the sum of the log likelihood from the HMM matches to arrive at one score based on which we compute the recognition performance. We will match the hand that is closer to the camera at each view. For this experiment, since we have a sufficient number of training data, we use the HMM to conduct the matching instead. We used a HMM with a 18 state ergodic model. We use one mixture component in each Gaussian.

Since this dataset was collected with yellow and blue colored gloves, it allows us to make comparisons with color-based hand segmentation schemes. As baseline performance comparison, we consider (i) manually segmented hands, and (ii) hands segmented using the information about the color of the gloves. For color-based hand segmentation, each glove color is modeled as mixture of three Gaussians in the RGB color space. For the proposed approach, we consider just region segmentation patches, detected as outlined earlier. Note that although we use color for segmentation and grouping, we do not use the knowledge that a specific color corresponds to the hand. We use the same feature vector as we use in Section 5.1.

Fig. 15 shows us all the candidate groups for one frame. We can see in Fig. 15d that the list consists of the real hand group and the group that generates it. In Fig. 16d, these groups do not exist because no grouping is done. Table 3 shows us the number of groups per frame and the number of correctly recognized gestures. We see that performance degrades significantly when grouping is not used. Without grouping, all of the candidate groups are singleton groups, the real hand normally is not included because the hand area is fragmented. Grouping is a necessary process, even for a dataset with colored gloves.

We consider recognition with each of the three probabilistic measures outlined earlier. The correct recognition rates are shown in Fig. 17. The five approaches—the two baselines and the three HMM ones, give us recognition rates: 79%, 94%, 91%, 92%, and 91%, respectively. From this result we can observe:

(1) For each frame, above 95% of the groups generated were noisy, with some being just random patches. However, their contribution to the final overall sequence is quite small, since they were not well linked across frames. Our approach allows us to recover from such errors. For the commonly used color-based hand segmentation approach, if any one frame has noisy hands, the recognition might fail. This is the reason why the recognition with hands segmented using just color information results in low performance.

Fig. 19. Recognition performance of each subjects separately, using summed-summed approach and manual segmentation.

Fig. 20. Recognition performance of each hand gestures, using DTW with proposed method and manual segmentation.

Fig. 21. Recognition performance of each subjects separately, using DTW with proposed method and manual segmentation.

Fig. 22. Sample frames for the first subject performing front action. Notice the right hands in three frames. The first frame shows one training example for rotate front. The second frame shows how the first subject is performing the same action. The third frame shows a rotate left frame is matched to this frame.
(2) Our approach which accommodates imperfect segmentation, is within 2% of recognition possible with manual segmentation.

Figs. 18 and 19 show the recognition rate on a per-gesture and per-subjects basis. We can see the majority of errors comes from one subject and the three gestures that can be easily mixed up. Subject 1 performed each gesture with larger motion than the other subjects in the training data. Such a case is hard to improve by using only the position features, hence subject 1 produced the majority of the errors. Among the gestures, Rotate Front, Push and Rotate Right all have motions moving forward and backward. There are only subtle orientation change in the palm. Hence these actions produced majority of the errors. However, the performance

![Tracking Results](image-url)

Fig. 23. The tracking results for the first test instance (defined by the author of the database) of the seven gestures. The first column is the first frame of the sequence. The second column is the tracking of the first part, followed by the tracking of the second part in the third column, and finally arrive at the last frame of the sequence in column 4.
measure of interest for this work is how well the recognition rate with grouped observation match that with perfect segmentation. On this account, the performance is quite strong.

Figs. 20 and 21 show the recognition rate on a per-gesture and per-subjects basis using the DTW with grouping method and DTW method with manually selected groups. The overall result (90% and 93%) is very close to the HMM result. Subject 3 has the best result (98%). Subject 1 and the action front has the worse result. Fig. 22 shows us some sample frames for Subject 1 performing the Front action, where we can see a over-rotate comparing to the training data (right hand). This makes the rotate front action the hardest one to recognize. Fig. 22 also shows us the frame in rotate left action that is matched to the rotate front because of the over rotating.

Fig. 23 shows visual examples of the optimal groups selected for the best match corresponding to all the seven gestures. The gestures are separated into two parts, where we show the tracking results separately. In each row, we first show the first frame in the sequence, then we show the first part tracking, which is the decoded groups by using the max–max method. Then we show the second part of tracking, followed by the last frame of the sequence. The shown sequences are just the first test sequence in each gesture, which is defined by the authors of the dataset [33]. Although

Fig. 24. Sample frames for the diving sequence in the sports video dataset from [34]. Note the moving background.

Fig. 25. The test result of detecting a signature subsequence of diving. The 10 crosses denote the matches to the 10 diving sequences. The 10 circles denote the match to other sports actions. The horizontal line is the matching distance while the vertical line is the start position of the match in the sequence, normalized by the sequence length.

Fig. 26. The test result of detecting a signature subsequence of diving. The 7 images in the first row is the training data where the diver is manually selected. The second and third row show the automatically decoded sequence of the body shape, notice that the background is moving.
we have many possible candidate group sequences, we can still retrieve the best group sequence with the proposed decoding method. This, make the need for perfect low-level segmentation not a necessary step any more.

5.3. Experiments with sports action dataset

We also experiment with a sports action dataset which is available online from Rodriguez et al. [34]. We consider the task of automatic indexing a specific type of sports video. In this dataset, some of the sports have a relatively still background, such as lifting, swing, etc. While some are shot with moving cameras, such as diving, soccer, which makes the indexing task hard. Fig. 24 shows us some sample frames in this dataset. We focus our test on detecting a diving sports video because the diving has a clear shape change of the human body, where the “sequence of shapes” is relatively unique and not easy to be found in other sports video.

Currently the dataset only has 14 sequences for diving. Some sample frames can be seen at Fig. 24. This number of video will not allow us to model the whole diving sequence since there are many styles of diving. Hence, we have modeled a subsequence of diving that is just before the diver get into the water. At this moment the diver usually unbends the body to dive into the water. We call this the signature subsequence for our task. Modeling this signature subsequence makes it possible for us to use the limited number of training data. Specifically, we choose four out of the 14 sequences as training data that is available. The signature subsequence of these four sequences are used to match to the other unknown entire video sequences, which is performed like matching a sign to a sentence. We use the 10 diving sequences as the positive test samples and all the other sport sequence as the negative test samples. Since the camera is moving fast, we cannot use the position or the motion of the body as the feature vectors, instead we added seven Hu moments [35] in the feature vector to better describe the shapes.

In Fig. 25 we illustrated the matching distances obtained for the 10 positive sequences and 10 negative sequences with lowest distance scores. The figure shows the matching distances on horizontal line and the start position of the match on the vertical line. The cross denotes the match of a positive sequence while a circle denotes the match to a negative sample. We can see that the match to diving sequences has a relatively lower score and the match starts all at the later part of the sequence (close to 1), which is where the signature sequence usually starts. While the negative sequences have a more uniformly distributed matched position and the distances are higher.

Fig. 26 shows us one signature subsequence (the first row) in the training data and two subsequences in the test data (the second and third rows) that are matched to it.

6. Conclusion

We proposed a new framework for gesture and sign recognition from video that does not rely on skin color models and can work with imperfect segmentation of scenes. We addressed the hard problem of hand segmentation by coupling it with recognition, via an intermediate grouping process. The grouping process generated layers of overlapping groups that are linked across time in a graph structure. We showed how the search for the optional sequence of groups can be arrived at different matching models. As a byproduct of the recognition problem we also segment the hand in each frame. We demonstrated its efficiency for sign recognition, HCI hand action recognition and a diving action indexing tasks. As our results show, using the coupled framework we are also able to provide an overall solution based on the segmentation and matching results and can improve the results when hand segmentation is not successful. For the HMM based matching, we share the source code as well as the examples on the website http://marathon.csee.usf.edu/ASL/.

Acknowledgement

This work was supported in part by the National Science Foundation under grant IIS 0312993. Any Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

References


