To appear in Proceedings of the European Conference on Computer Vision (ECCV),
October 2008.

Tracking with Dynamic Hidden-State Shape Models

Zheng Wu[1], Margrit Betke[1], Jingbin Wang[2], Vassilis Athitsos[3], and Stan Sclaroff[1]

[1] Computer Science Department, Boston University, Boston, MA, USA
[2] Google Inc.,Mountain View, CA, USA
[3] Computer Science and Engineering Department, University of Texas at Arlington, Arlington, Texas, USA

Abstract. Hidden State Shape Models (HSSMs) were previously proposed to represent and detect objects in images that exhibit not just deformation of their shape but also variation in their structure. In this paper, we introduce Dynamic Hidden-State Shape Models (DHSSMs) to track and recognize the non-rigid motion of such objects, for example, human hands. Our recursive Bayesian filtering method, called DP-Tracking, combines an exhaustive local search for a match between image features and model states with a dynamic programming approach to find a global registration between the model and the object in the image. Our contribution is a technique to exploit the hierarchical structure of the dynamic programming approach that on average considerably speeds up the search for matches. We also propose to embed an online learning approach into the tracking mechanism that updates the DHSSM dynamically. The learning approach ensures that the DHSSM accurately represents the tracked object and distinguishes any clutter potentially present in the image. Our experiments show that our method can recognize the digits of a hand while the fingers are being moved and curled to various degrees. The method is robust to various illumination conditions, the presence of clutter, occlusions, and some types of self-occlusions. The experiments demonstrate a significant improvement in both efficiency and accuracy of recognition compared to the non-recursive way of frame-by-frame detection.

1 Introduction

The radar literature describes mature algorithms for tracking targets that estimate the state of the targets within a dynamic system using recursive Bayesian filters [1]. The computer vision community has extended these algorithms to track objects in images – a much more difficult problem because the appearance of deformable and articulated objects in images can vary enormously. Our work moves beyond the limit of what has been accomplished so far by formulating and solving the task of tracking objects that have a variable shape structure (in addition to being rigid or deformable, and/or articulated). We present a model-based, recursive Bayesian estimation algorithm that tracks such objects in images with heavy clutter and simultaneously recognizes the various components of the object. Since the object has a variable structure, our method does not know a priori if any of these components appear or disappear in the video and what shape they will exhibit. An example of an object with variable structure is a hand, whose components, the fingers, may or may not appear in an image and may be curled to various degrees, resulting in self-occlusion (Fig. 1(a)). Our tracking method uniquely identifies and outlines each finger in the video, regardless of the degree of curling motion. Moreover, it concurrently handles dense clutter and illumination changes, and recovers automatically from occlusions by other objects. Our contributions are:

- We introduce dynamic hidden-state shape models (DHSSMs) to model classes of moving objects of variable structure (Fig. 1). DHSSMs are a generalization of (static) hidden-state shape models (HSSMs) [2, 3], where the model description will be updated through time.
- We propose a dynamic programming (DP) approach, called DP-Tracking, to find an optimal registration between the model states of a DHSSM and the features extracted from each video frame. DP-Tracking speeds up the model registration by two orders of magnitude compared to Wang et al.'s approach [3] and tracks hands in near real time. We achieve this by introducing a grouping technique that takes advantage of the spatial coherence of features in the DP table during tracking.
- We embed an online learning approach into the tracking mechanism that captures appearance changes through time and produces tracking results that are significantly more accurate than the results obtained with DP-Tracking without online learning or Wang et al.'s approach [3].

Our data contains a large number of candidate features (about 3,000 edge points per frame). We need to process both the contours of the object to be tracked and background objects (clutter) or occluding foreground objects (also clutter). Our method tracks each feature at position p on the contour of the moving object by assigning it to a feature in the next frame that is near to p. This straightforward recursive Bayesian tracking approach is particularly suited for such data, more so than the Kalman, kernel-based, or particle filters [1, 4, 5]. It is challenging to use the latter in real time processing, since the number of particles required would increase exponentially with the dimension of the state space. The hand DHSSM, for example, has 20 states including position, orientation, scale and feature duration (i.e., finger's length). Nevertheless, our framework for tracking objects with DHSSMs in principle allows adoption of any Bayesian algorithm, and others may explore this in the future.

Related Work. Tracking algorithms typically assume smoothness: the state or the appearance of an object component does not change much from one image frame to the next. This assumption helps to overcome problems with tracking

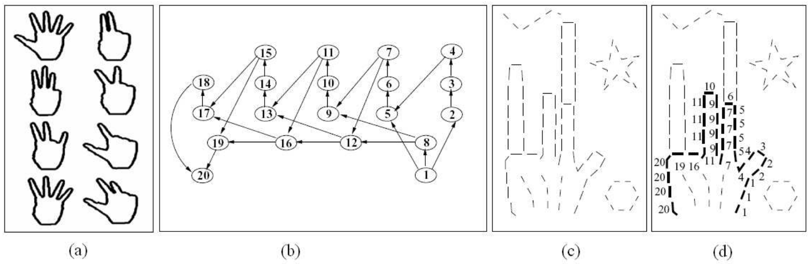

Fig. 1. Recognizing hands in images using HSSMs. (a) Hand contours with variable shape structure. (b) State transition diagram of the HSSM "hand." State s_1 models the right boundary of the palm, s_{20} the left. States s_2,...,s_4 model the thumb; in particular, state s_2 the right side of the thumb; state s_3 the thumb tip, and state s_4 the left side of the thumb. States s_5,...,s_7, s_9,...,s_{11}, s_{13},...,s_{15}, and s_{17}, s_{18} respectively model the contour of the other four fingers. States $s_8, s_{12}, s_{16}, s_{19}$ model the occluding boundary when the fingers are totally bent. (c) An edge image with segments of the hand contour and clutter. (d) Registration of hand model states to contour segments, resulting in the recognition of the hand structure. Figure courtesy of Wang et al. [3]

when the interpretation of image data is ambiguous. Successful systems have used graphical models for modeling non-rigid 3D objects [6, 7]. These systems, however, require a manual initialization of the object to be tracked (our system does not), and there is no guarantee that the systems can recover when the previous frame is interpreted falsely. An alternative way to overcome the problem of losing track is to treat tracking as the problem of detecting the object in each frame [8–10], also known as *tracking by detection* [11]. Thus, poor tracking in one frame will not affect any subsequent results, and the problem of drifting is also prevented [12]. Our DHSSM-based method extends the tracking-by-detection idea by adding "gating" [1], and is able to recover from tracking errors as we show in the experiments.

A popular approach for tracking-by-detection is template matching (e.g., [10]). To deal with object deformation, multiple templates can be organized in a tree structure so that exhaustive search for the best match can be replaced by hierarchical matching [10]. Although template matching is efficient, it may be difficult to use it to identify clutter or deal with large within-class variations in object shape. Recognition approaches that exploit contour-based information recently received considerable attention in the object detection literature (e.g., [13–15]). These algorithms either use static boundary templates or codebooks of appearance parts. Note the difference to our contour-based work: these representations model standard object configurations and do not explicitly account for the variable structure of the objects (e.g., fingers that are totally extended, partially bent, or completely hidden, as in Fig. 1). Another classic work of tracking 2D silhouette deals with the problem of discontinuous shape changes by explicitly modelling *wormholes* in shape space [16], but the contour extraction is not designed to handle cluttered images.

It is desirable to use rich models that can capture a large range of possible object variations and efficient methods that can register the image data to such models in tracking scenarios. In this paper, we propose such models, DHSSMs, and apply them with an efficient DP search algorithm to perform detection, recognition, and tracking of human hand in heavily cluttered images.

2 Dynamic Hidden-State Shape Models

We extend (static) Hidden-State Shape Models (HSSMs) [3], which are based on HMMs [17], to Dynamic Hidden-State Shape Models (DHSSMs) to include the index of the current frame, denoted by superscript t. A DHSSM is defined by

- a set $\mathbb{S} = \{s_1, \ldots, s_M\}$ of states modeling object components.
- a subset $\mathbb{E}$ of $\mathbb{S}$ that defines legal end states,
- the probability $\pi^{(t)}(s_i)$ that state s_i is the initial state at time t,
- the *state transition function* $A^{(t)}(s_i, s_j) = p^{(t)}(s_j|s_i)$ that represents the probability of transiting from state s_i to state s_j at time t,
- the *state observation function* $B^{(t)}(f_u, s_i) = p^{(t)}(f_u|s_i)$ that represents the probability of observing feature f_u in state s_i at time t,
- the *feature transition function* $\tau^{(t)}(f_v, f_u, s_j, s_i) = p^{(t)}(f_v|f_u, s_j, s_i)$ that represents the probability of observing feature f_v in state s_j given some other feature f_u was previously observed in state s_i at time t, where s_i could be equal to s_j if both f_u and f_v belong to the same object part, and
- the *state duration function* $D^{(t)}(\ell, s_i, \omega) = p^{(t)}(\ell|s_i, \omega)$ that represents the probability of continuously observing ℓ features in state s_i at time t, given the prior object scale ω.

To recognize the structure of an object modeled by a DHSSM in a frame $I^{(t)}$, we use the approach by Wang et al. [3] to extract K features $\mathbb{O}^+$ from the image, recognize the L features $\mathbb{O} \subseteq \mathbb{O}^+$ that represent the object contour, and separate them from the $K-L$ features $\mathbb{O}_c \subset \mathbb{O}^+$ that represent clutter. This is achieved by finding an *ordered* match between the object features $\mathbb{O} = (o_1, ..., o_L)$ and their corresponding DHSSM states and a match between the features $\mathbb{O}_c = (o_{L+1}, ..., o_K)$ that are not on the object contour and a special state $q_c \notin \mathbb{S}$ that is introduced to represent clutter. By $(o_j : q_i)$, we denote the match between an observed feature $o_j \in \mathbb{O}^+$ and a state $q_i \in \mathbb{Q}^+ = \mathbb{S} \cup \{q_c\}$. Since several features on the contour may be assigned to the same state, we use the notation $(O_j^{(d)} : q_i)$ to describe the ordered match between d observed features $O_j^{(d)} = (o_j, ..., o_{j+d})$, forming a single contour segment (e.g., side of a finger), and state q_i. A registration R of n contour segments to a sequence of n states is then

$$R^{(t)}_{\mathbb{O},\mathbb{Q}} = [(\mathrm{O}_1^{(\mathrm{d}_1)} : \mathrm{q}_1),\ (\mathrm{O}_{\mathrm{d}_1+1}^{(\mathrm{d}_2)} : \mathrm{q}_2),\ \ldots,\ (\mathrm{O}_{\mathrm{L}-\mathrm{d}_\mathrm{n}+1}^{(\mathrm{d}_\mathrm{n})} : \mathrm{q}_\mathrm{n})]^{(t)},$$

where $\mathrm{O}_j, \mathrm{q}_\mathrm{i}, \mathrm{d}_\mathrm{i}, \mathrm{n}$, and L are considered random variables. To recognize the object in the t-th video frame, the registration algorithm must find values for these random variables that maximize the joint probability $p(\mathbb{O}^+, \mathbb{Q}^+)$ of observations and states. By extending the derivations of Wang et al. [3] in a straightforward manner to include the index t, we can show this to be equivalent to minimizing the registration cost function

$$\begin{aligned} C^{(t)}(R^{(t)}_{\mathbb{O},\mathbb{Q}}) &= -\ln \pi^{(t)}(\mathrm{q}_1) - \textstyle\sum_{i=1}^{\mathrm{n}}\{\ln A^{(t)}(\mathrm{q}_{i-1},\mathrm{q}_i) + \ln D^{(t)}(\mathrm{d}_i,\mathrm{q}_i,\omega) \\ &\quad -\xi(\mathrm{d}_i) + \textstyle\sum_{j=\zeta(i)+1}^{\zeta(i)+\mathrm{d}_i}[\ln \frac{B^{(t)}(\mathrm{o}_j,\mathrm{q}_i)}{B^{(t)}(\mathrm{o}_j,q_c)} + \ln \tau^{(t)}(\mathrm{o}_j,\mathrm{o}_{j-1},\mathrm{q}_i,\mathrm{q}_{i'})]\}, \quad (1) \end{aligned}$$

where $A^{(t)}, B^{(t)}, D^{(t)}$ and $\tau^{(t)}$ were introduced above, $\xi(\mathrm{d}_i) = \mathrm{d}_i \ln p(q_c)$, $\zeta(i) = \sum_{k=1}^{i-1}\mathrm{d}_\mathrm{k}$, and $i' = i-1$ when $j = \zeta(i)+1$ and $i' = i$ otherwise. Recognition of the object in the presence of clutter is achieved by finding the globally optimal registration $R^{(t)}_{\mathrm{opt}} = \arg\min\, C^{(t)}(R^{(t)}_{\mathbb{O},\mathbb{Q}})$. More details about HSSMs can be found in Wang et al. [3].

For (static) HSSMs, a DP algorithm for minimizing the cost function C was proposed [3], which has the computational complexity of $O(MC_sK^2\ell_{\max})$, where M and K are the number of states and features, C_s is the average number of legal state transitions out of each state, and $\ell_{\max}$ is the maximum number of features that can be observed in a state. The `C++` implementation of this algorithm processed a 160x120 image in about 25 minutes (AMD Opteron 2.0 GHz processor) to determine the size and pose of a single object of variable structure. This included an exhaustive search among eight possible object orientations and no feature pruning (K is up to 3000). Thus, a frame-by-frame application of Wang et al.'s HSSM-based algorithm [3] to track objects in long video sequences is computationally inefficient. Here we show how DHSSMs and a new DP-based registration algorithm can be combined into a fast and robust object tracking system. The essential steps of our DHSSM-based method comprise hierarchical dynamic programming and online model updates.

2.1 Specification of a DHSSM for the Hand

We define the hand DHSSM similar to the hand HSSM introduced by Wang et al. [3] (Fig. 1). An image feature $f \in \mathbb{O}^+$ is a local image patch surrounding an edge pixel, measured by its appearance ϕ and location y, i.e., $f = (\phi, y)$. The appearance ϕ is specified by the color distribution ϕ_χ of the 5×5 pixels patch (a vector that stores weighted average rgb values for each of the two half patches separated along the edge direction in the patch center) and the intensity gradient orientation ϕ_g at the patch center (a scalar that ranges from 0 to 2π). For each object state $s \in \mathbb{S}$, we model the color distribution $s^{(t)}_\chi$ of an object boundary patch and the gradient $s^{(t)}_g$ at the center of the patch at t.

The state transition function $A^{(t)}$ can be simplified as a uniform distribution. We define the object/clutter observation likelihood ratio as

$$\frac{B^{(t)}(f,s)}{B^{(t)}(f,c)} = \frac{p^{(t)}(\mathrm{o} = f \mid \mathrm{q} = s)}{p^{(t)}(\mathrm{o} = f \mid \mathrm{q_c} = c)} \approx e^{-\gamma\, h^{(t)}(\phi_\chi)} p(\phi_g | s^{(t)}_g), \quad (2)$$

where $p(\phi_g|s^{(t)}_g)$ is a Gaussian distribution with mean $s^{(t)}_g$ and variance σ^2_g. Function $h^{(t)}$, for given input ϕ_χ, outputs a decision value, which is approximated by a two-class Support Vector Machine (SVM). The scalar factor γ is determined experimentally. The feature transition probability

$$\tau^{(t)}(f, f', s, s') = e^{-\alpha(\|y'-y\|)} e^{-\beta|\Delta(\phi_g, \phi_{g'}) - \Delta(s_g^{(t)}, s_{g'}^{(t)})|} \tag{3}$$

is an exponential distribution, where $\|y - y'\|$ represents the Euclidean distance between the centers of the two patches f and $f,'$ and $\Delta(\phi, \phi')$ represents the angle in radians between orientations ϕ and ϕ'. The weighting scalars α and β can be learned from the training data by maximum likelihood estimation. The state duration probability $D^{(t)}$ is defined as the Gaussian model

$$D^{(t)}(\ell, s, \omega) = p(\ell \mid \mu^{(t)}(\omega), \sigma^{(t)}(\omega))\, p(\mu^{(t)}(\omega)|s) \tag{4}$$

where ω is an input parameter to specify the reference scale, $p(\ell \mid \mu^{(t)}(\omega), \sigma^{(t)}(\omega))$ is a normal distribution with the mean $\mu^{(t)}(\omega)$ and covariance $\sigma^{(t)}(\omega)$, and $p(\mu^{(t)}(\omega) \mid s)$ is the conditional prior for the Gaussian distribution.

3 Tracking with DHSSMs

An overview of our tracking system for the application of hand and finger tracking is given in Fig. 2. The system contains the hand DHSSM that is updated to capture appearance changes through time, for example, due to the occluding motion of curling fingers or some other objects (short-term update) or changes in illumination and background (long-term update). Pseudo-code of our DP-Tracking algorithm is given in Sec. 3.2.

3.1 Updating the DHSSM during Tracking

We first explain how to update DHSSM in tracking. The temporal information is used to update the DHSSM in the short term, i.e., after processing each frame, and the SVM in the long term, i.e., after processing a number of frames.

The appearance description $s_g^{(t)}, \mu^{(t)}(\omega), \sigma^{(t)}(\omega)$ of the DHSSM is updated for each state s by sampling from its posterior distribution. For each object state q $= s \in \mathbb{S}$ of the DHSSM, the posterior distribution of the image gradient s_g at the center of an object boundary patch is estimated from the optimal registration in the previous frame $t-1$, denoted as $R_{\text{opt}}^{(t-1)}$:

$$p(s_g^{(t)}|R_{\text{opt}}^{(t-1)}) \sim \mathcal{N}(\mu(\phi_g^{(t-1)}), \sigma^2(\phi_g^{(t-1)})) \tag{5}$$

where $\mu(\phi_g^{(t-1)})$ is the mean gradient direction of feature set $\{o_j^{(t-1)}, \ldots, o_{j+d}^{(t-1)}\}$ that mapped onto state s and $\sigma^2(\phi_g^{(t-1)})$ the variance. This step captures the change of orientation of each state.

The posterior distribution of the duration mean $\mu(\omega)$ for each state s is estimated from the optimal registration during the past N frames,

$$p(\mu^{(t)}(\omega)|\underline{R_{\text{opt}}^{(t-1)}}) \sim \mathcal{N}(\mu(\underline{d}^{(t-1)}), \sigma^2(\underline{d}^{(t-1)})) \tag{6}$$

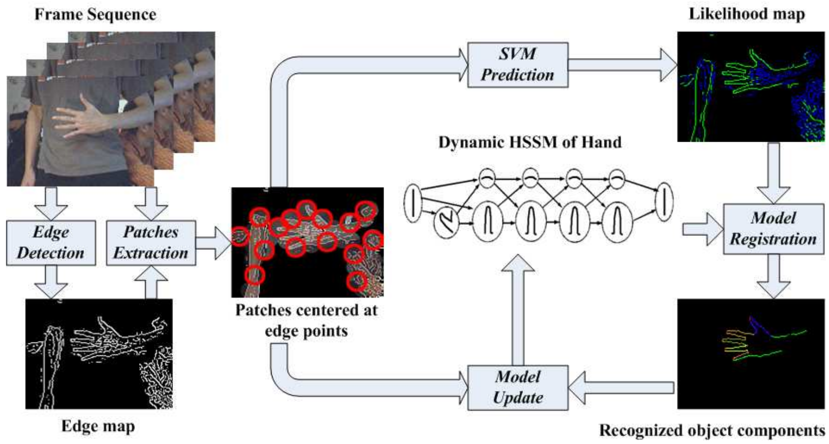

Fig. 2. Overview of tracking system. Each input image is first processed to extract features (edges) and feature patches (image regions centered around edges), pruned by a learned skin color model. The two-class SVM determines which features are likely to belong to the object contour (foreground) and which not (background clutter or occluding foreground clutter). Hand and finger detection is achieved by finding a globally optimal registration between DHSSM states and likely features. The locations of recognized object components (finger tips, sides, palm, etc.) are the frame-by-frame output of the tracking system and used to update the DHSSM online. After each frame, the probability densities of each state of the DHSSM are updated by feeding back the estimated orientation of the hand, and the relative orientation and amount of curling of each finger (the latter via updating the state duration variable). From time to time, a new collection of object and clutter patches are sampled to train a new SVM classifier that better models the current imaging scenario.

where $\underline{R_{\text{opt}}^{(t-1)}}$ is the history of the past N optimal registrations, $\mu(\underline{d}^{(t-1)})$ is the mean number of contour points mapped onto state s during the past N optimal registrations, and $\sigma^2(\underline{d}^{(t-1)})$ is the variance. We set $\sigma^{(t)}(\omega)$ to be $\sigma(\underline{d}^{(t-1)})$ for simplicity.

The long-term update of the tracking system focuses on the color distribution s_χ of a boundary patch for each object state s. As mentioned in Sec. 2.1, we approximate this log-likelihood ratio by the output of a two-class SVM. However, it is challenging to build a color model that is robust and sufficiently discriminative under various illumination conditions and background scenes [18]; defining a robust skin color model remains a research topic by itself. It is therefore natural to perform ONLINE LEARNING, where the training set comprises of "positive patches" sampled along the detected hand boundary, and "negative patches," sampled from clutter. Note we only train a new classifier every few dozens of frames, so that training time is still acceptable for a classifier like an SVM. We adopted two conservative strategies when we rebuild the classifier:

(1) Avoid sampling negative patches located close to the object boundary, where they could be misclassified due to shadows, occlusion, noise, etc.

(2) Add a validation step before training samples are collected to ensure that the samples are labelled correctly. To validate our registration result in the current frame, we assume that the registration result of the previous frame is trustworthy and create a hand contour template from it. We then perform chamfer matching between this contour template and the contour found in the current frame. A matching cost above a certain threshold is considered suspicious, and it is then decided to suspend all model updates.

3.2 Exploiting the Hierarchical Structure in DP

Our hand DHSSM resembles a left-to-right HMM (no state-transition loops are allowed) and we can build a 2D dynamic programming table to organize the matching process (Note the general DHSSM can have state-transition loops). The table has two axes. The state axis represents an ordered sequence of model states and each state contains multiple rows showing the state duration; the feature axis represents an *unordered* sequence of input features. We do not have ordered sequences in both axes as for Dynamic Time Warping [19].

To ease the explanation, we assume the state transition is a single Markov chain. The DP process can be seen as finding the shortest path between s_1 and s_M (e.g., right and left side of the palm) in a directed graph. Each node in the graph represents a matching pair $(f_j : s_{i,l})$, where feature f_j is observed in state s_i with duration l. Each edge that links two matching pairs $(f_j : s_{i,l_1})$ and $(f_n : s_{m,l_2})$ represents a possible jump from the observation f_j in s_i with duration l_1 to the next observation f_n in s_m with duration l_2. The weight of an edge is defined by the matching cost. To avoid the case of a loop, i.e., explaining one feature by more than one state, we restrict the selection of a set of features in the next jump, excluding those features that have already been chosen to belong to the shortest path. A DP algorithm to find the shortest path in this table generally has to consider all possible paths through the graph and therefore requires $O(MK^2\ell_{\max})$ operations, where M and K are the number of states and features, and $\ell_{\max}$ is the length of longest segment that can be observed. Since we use patches around edge points to represent image features, K typically is quite large, even in moderate sized images (up to 3,000 in the 160x120 video frames in our experiments, see Sec. 4). Gating [1] in the next frame will reduce the number of candidate features. In addition, we can exploit information from the spatial arrangement of the edge points in the local search neighborhood to help speed up the dynamic programming approach.

A benefit from tracking is that we can group neighboring edge points in the current frame based on the registration result in the previous frame. This yields groups of unordered input features. For example, in Fig. 3, the locations of feature points possibly matching onto states s_1 and s_2 are constrained by their respective local neighborhoods or gates (light and dark gray regions). A state transition can only happen in the intersection of the light and dark gray regions. when constructing the DP table, all the feature points are now sorted into groups where the group order corresponds to the state order. During the matching process, only feature transitions within the same group will be considered. Any

DHSSM-BASED TRACKING ALGORITHM:

Given Initial DHSSM description: $\{\mathbb{S}, \mathbb{E}, \pi^{(1)}, A^{(1)}, B^{(1)}, \tau^{(1)}, D^{(1)}\}$, skin color model, SVM classifier, and optimal registration in first frame $R_{opt}^{(1)}$ computed by [3].
For frame $t = 2...N$:

1. **Data Preparation**
   - Extract and prune features $\mathbb{O}^+$ from frame t by gating and skin color model;
   - Grouping based on $R_{opt}^{(t-1)}$;
   - Construct DP table.
2. **Model Registration**
   - For each group of features, running DP to find k candidate shortest paths;
   - For each group, detect occlusion by comparing histograms of gradients along candidate paths with corresponding segment in $R_{opt}^{(t-1)}$. If occlusion exists, label such groups invalid;
   - Run DP by linking one candidate valid path from each group to find the optimal shortest path $R_{opt}^{(t)}$.
3. **DHSSM Update**
   - Short-term update:
     Sample $s_g^{(t)}, \mu^{(t)}(\omega)$ according to Eqs. 5 and 6 and set $\sigma^{(t)}(\omega)$ to be $\sigma^2(\underline{d}^{(t-1)})$;
   - Long-term update: Collect $\mathbb{Q}^+, \mathbb{O}^+$ from $\underline{R_{\text{opt}}^{(t-1)}}$ to train a new SVM.

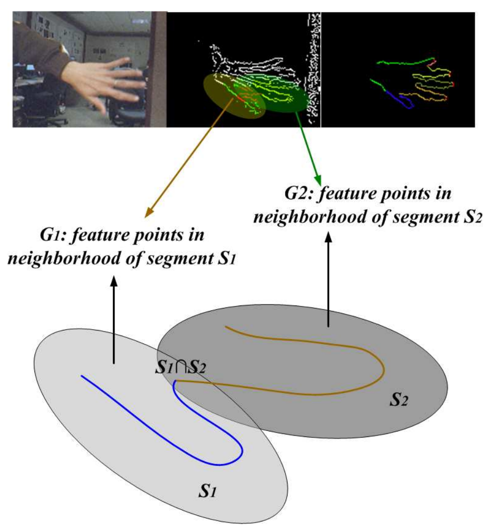

Fig. 3. Feature candidates mapped onto state s_1 are constrained by the light gray search region (gate), while feature candidates mapped onto state s_2 are constrained by the dark gray region. State transition can only happen in the intersection $s_1 \cap s_2$ of the two regions

valid shortest path must pass through all the intersection regions. In other words, within each group, we constrain the valid start and end points to be located in intersection regions.

If there are G groups, the average number of feature points within each group is $\frac{K}{G}$, and we need $O(\frac{MK^2 l_{max}}{G^2})$ operations to find the optimal registration. A careful design of the DHSSM is critical to achieve the proposed reduction in computation. If more than two group regions overlap each other, there is no guarantee that the optimal path will not contain feature transitions between non-adjacent groups. One way to deal with multiple overlapping regions is to insert duplicated features when grouping, which causes the input feature size to increase. Usually a proper choice of the size of the gate or neighborhood and an appropriate design of the states that correspond to large segments of the object boundary can alleviate the problem of redundant representation of feature points that fall into multiple overlapping regions.

Another benefit from the proposed grouping is that dynamic programming can be applied within each group independently. This step produces a number of candidate paths that correspond to boundary segments in the image. Then DP can be applied one more time by selecting one segment from each group and linking them together to form the final shortest path corresponding to the whole object boundary. The hierarchical DP algorithm allows the freedom to model the deformation within each segment locally and enforces spatial coherence between segments globally.

Moreover, it provides the opportunity to detect occlusions and help solve the problem of partial matches. While running DP within each group, our method compares histograms of gradients along each of those candidate paths with that of corresponding segments detected in previous frame. Intuitively, when occlusion occurs in some group, the candidate paths are noisy shapes so that the gradient histogram is quite different from that in the previous frame. If our method determines that occlusion occurs in some groups, candidate paths within these groups are excluded from participating in the second DP calculation. Then an incomplete hand contour can still be fitted well to the DHSSM model.

4 Experiments and Results

We implemented our DHSSM-based method in `C++` and tested it on an AMD Opteron 2.0 GHz processor. The datasets are challenging: 1) the background contains dense clutter so that the edges of the hand boundary and edges of clutter objects are difficult to distinguish; 2) the background is affected by illumination changes and other moving objects; 3) some scenes contain faces which means that skin-color-based hand detection must overcome ambiguities; 4) the shape structure of the hands in the videos changes due to the curling motion of the fingers, 5) the hand might be occluded so that there is no guarantee that a complete hand boundary exists in the scene. We assert that our datasets are sufficiently representative to demonstrate the performance of our system in tracking hands and fingers in real environments.

We used two quantitative measures of detection accuracy: (1) **Hand localization**: the algorithm correctly located the hand's position and orientation. This can be considered as the minimum requirement for all tracking systems. (2)

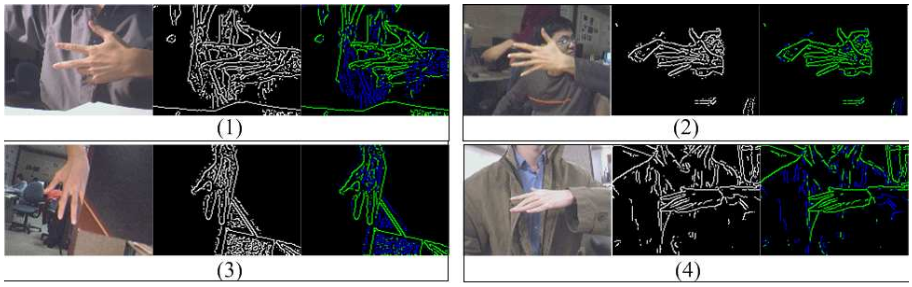

Fig. 4. Four 160x120 input images (left), edge maps after pruning with the skin color model (middle), and likelihood ratio maps predicted by the SVM classifier (right) with likely hand contour edges in green and clutter in blue. Samples include curled fingers (1), skin-color clutter (2), occlusion (3), and out-of-plane rotation (4).

Finger identification: the algorithm not only correctly located the hand's position and orientation, but also identified the state of each finger. We required that every visible finger was registered correctly to the respective states of the hand DHSSM. Such registration information would be valuable for solving advanced image understanding tasks, such as sign language recognition. Four datasets, collected in a laboratory environment, were used in our experiments:
(1) Data with large motion of the hand and fingers (260 frames): the hand appeared in the video in different positions, orientations, and scales because of its translation, rotation, and movements towards and away from the camera. Deformation due to different degrees of curling of fingers was included.
(2) Data with dense clutter (510 frames): the background contained other moving objects (walking people) and skin-color patches (faces).
(3) Data with illumination changes (182 frames): the scene and object color changed significantly due to newly-appearing light sources.
(4) Data with occlusions (167 frames): a part of the hand was occluded by the motion of a background object so that partial matching of the hand was required.

Our DHSSM-based method correctly localized the hand in almost all tested situations that involved a large amount of motion and clutter (98% and 92% among all frames in test dataset 1 and 2, respectively). The shape of each of the five fingers was also detected in these situations (89% and 85%, respectively). Illumination changes and occlusion reduced the recognition percentages by about 20 percentage points (Table 1). Online learning contributes to increasing the accuracy of finger identification (85%) by 10 percentage points when the skin color varies due to changing lighting conditions (Table 1, Fig. 5, middle). Our method was particularly useful in interpreting the hand in the presence of occluding objects whose appearance strongly interfered with feature detection, e.g., the corner of the notebook in Fig. 5, right.

Our DHSSM-based method processes each video frame in near real time. The running time is proportional to the number of edge points that must be processed, e.g., for datasets with less than 2,000 edge points on average per frame, it was 1 s and with 3,000 edge points 1.5 s (Table 1).

Table 1. Comparison of our DHSSM-based method with DP-Tracking (DPT) and with and without Online Learning (OL), and Wang et al.'s [3] method (HSSM).

Dataset	Large Motion		Dense Clutter		Partial Occlusion		Illumination Change	
Avg. # of features	1,200		1,800		3,000		3,000	
Method	HSSM	DHSSM +DPT	HSSM	DHSSM +DPT	HSSM	DHSSM +DPT	DHSSM +DPT	DHSSM +DPT +OL
Hand Localization	96%	98%	92%	92%	55%	75%	83%	83%
Finger Identification	75%	89%	79%	85%	40%	65%	58%	68%
Avg. time/frame	160 s	1 s	160 s	1 s	250 s	1.5 s	1.5 s	2 s

To compare our method to a previously published method, we reimplemented Wang et al.'s [3] HSSM-based method. For a fair comparison of both methods, we used the same size of the local search window for each feature point (40×40 pixels). Using Wang et al.'s method, we registered the static HSSM of the hand to the image data in each frame separately. Only the global orientation of the hand was passed from frame to frame. The processing time of our method compared to Wang et al.'s method [3] was more than two orders of magnitude shorter (Table 1). The improvement in localizing the hand was up to 20 percentage points and in identifying the fingers up to 25 percentage points (Table 1 and Fig. 5). The system has the ability to recover from tracking errors, which can occur due to severe occlusions or self-occlusions.

5 Discussion and Conclusion

We developed a near-real-time system that not only tracks a fast-moving hand in a cluttered environment but, at the same time, recognizes and labels the state of each moving finger. These tasks would be extremely difficult for a simple hand or finger blob tracker. Our experiments highlighted the strengths of our system:
- **Robustness to cluttered backgrounds**. Our system finds the optimal model registration between image features and states in the DHSSM or the unique clutter state q_c. By incorporating q_c into the registration formulation, our detection method becomes powerful and can handle heavy clutter. Success in scenes with clutter was not demonstrated for an existing real-time hand tracker [10] that was based on matching between boundary templates and object silhouettes.
- **Robustness to illumination and background changes**. Our online learning step updates the classifier so that it can best discriminate the object boundary from clutter. The need for online learning was shown most succinctly in our experiments where the appearance of the object color changed due to lighting changes. Such changes are captured by the DHSSM update.
- **Handling occlusions**. The DP registration process is decomposed into two stages. The first DP stage aims at detecting boundary segments that correspond to states of the DHSSM. Missing segments can be detected during this stage,

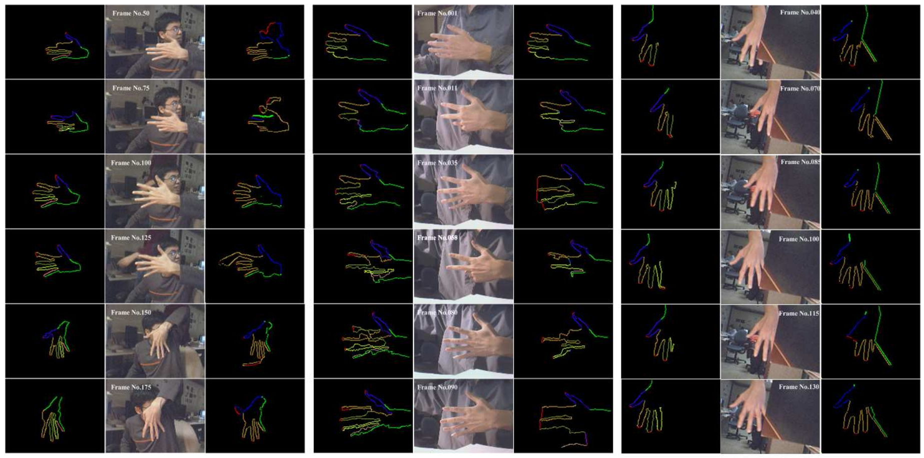

Fig. 5. Tracking results for images with clutter (left) due to a face, a chair, and a person moving in the background, and illumination changes (middle), and occlusions (right). The colors of the boundary pixels indicate which part of the boundary was detected as the finger tips (red), thumb (blue), index finger (orange), middle finger (olive), ring finger (yellow), little finger (pink), and outer boundary of the little finger and wrist (green). DP-TRACKING with ONLINE LEARNING produced the best recognition results.

so that they are not considered in the second stage during which boundary segments are connected to create the boundary of the whole hand.

- **Computational efficiency**. We exploit the spatial coherence between features obtained from temporal information so that we can significantly reduce the size of the DP table. The average processing time per frame for 160×120 images is 1–2 s and the SVM training step takes typically 3–5 s. This compares favorably to other tracking approaches that use graphical models and for which running times of several minutes per frame were reported [6].

Our future work will extend the DHSSM framework to enable us to insert or delete model states through time. Unlike the current version, where every possible structure change is described by the DHSSM, a more powerful DHSSM should be able to learn unpredicted object deformations online. This may allow us to apply the model to complicated deformable objects, such as cells in microscopic images.

Acknowledgment This research was supported in part by NSF grants IIS-0705749 and IIS-0713229.

References

1. Bar-Shalom, Y., Fortmann, T.: Tracking and Data Association. Academic Press (1988)
2. Athitsos, V., Wang, J., Sclaroff, S., Betke, M.: Detecting instances of shape classes that exhibit variable structure. In: ECCV. (2006) 121–134

3. Wang, J., Athitsos, V., Sclaroff, S., Betke, M.: Detecting objects of variable shape structure with hidden state shape models. IEEE T PAMI **30**(3) (2008) 477–492
4. Han, B., Zhu, Y., Comaniciu, D., Davis, L.: Kernel-based Bayesian filtering for object tracking. In: CVPR. Volume 1. (2005) 227–234
5. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface-quality hand tracking. In: ECCV (2). (2000) 3–19
6. Sudderth, E., Mandel, M., Freeman, W., Willsky, A.: Distributed occlusion reasoning for tracking with nonparametric belief propagation. In: NIPS. (2004)
7. Sigal, L., Bhatia, S., Roth, S., Black, M.J., Isard, M.: Tracking loose-limbed people. In: CVPR. (2004) 421–428
8. Athitsos, V., Sclaroff, S.: Estimating 3d hand pose from a cluttered image. In: CVPR. (2003) 432–440
9. Shakhnarovich, G., Viola, P., Darrell, T.: Fast pose estimation with parameter-sensitive hashing. In: ICCV. (2003) 750–758
10. Stenger, B., Thayananthan, A., Torr, P., Cipolla, R.: Hand pose estimation using hierarchical detection. In: International Workshop on Human-Computer Interaction, proceeding, LNCS. (2004)
11. Forsyth, D., Arikan, O., Ikemoto, L., O'Brien, J., Ramanan, D.: Computational Studies of Human Motion: Part 1, Tracking and Motion Synthesis. (2006)
12. Ramanan, D., Forsyth, D.A., Zisserman, A.: Strike a pose: Tracking people by finding stylized poses. In: CVPR. (2005) 20–25
13. Felzenszwalb, P.F., Schwartz, J.D.: Hierarchical matching of deformable shapes. In: CVPR. (2007) 1–8
14. Opelt, A., Pinz, A., Zisserman, A.: A boundary-fragment-model for object detection. In: ECCV. (2006) 575–588
15. Shotton, J., Blake, A., Cipolla, R.: Contour-based learning for object detection. In: ICCV. (2005) 503–510
16. Heap, T., Hogg, D.: Wormholes in shape space: Tracking through discontinuous changes in shape. In: ICCV. (1998) 344–349
17. Rabiner, L.R.: A tutorial on Hidden Markov Models and selected applications in speech recognition. Proceedings of the IEEE **77**(2) (1989) 257–286
18. Collins, R., Liu, Y.: On-line selection of discriminative tracking features. In: ICCV. (2003) 346–354
19. Schmidt, F.R., Farin, D., Cremers, D.: Fast matching of planar shapes in sub-cubic runtime. In: ICCV. (October 2007) 1–6