Detecting Objects of Variable Shape Structure With Hidden State Shape Models

Jingbin Wang, *Student Member, IEEE,* Vassilis Athitsos, *Member, IEEE,*
Stan Sclaroff, *Senior Member, IEEE,* Margrit Betke, *Member, IEEE*

Jingbin Wang, Vassilis Athitsos, Margrit Betke and Stan Sclaroff are with the Computer Science Department, Boston University, Boston, 02215, MA, U.S.A. E-mail: {jingbinw, athitsos, sclaroff, betke }@cs.bu.edu.

Digital Object Indentifier 10.1109/TPAMI.2007.1178

Abstract

This paper proposes a method for detecting object classes that exhibit variable shape structure in heavily cluttered images. The term "variable shape structure" is used to characterize object classes in which some shape parts can be repeated an arbitrary number of times, some parts can be optional, and some parts can have several alternative appearances. Hidden State Shape Models (HSSMs), a generalization of Hidden Markov Models (HMMs), are introduced to model object classes of variable shape structure using a probabilistic framework. A polynomial inference algorithm automatically determines object location, orientation, scale and structure by finding the globally optimal registration of model states with the image features, even in the presence of clutter. Experiments with real images demonstrate that the proposed method can localize objects of variable shape structure with high accuracy. For the task of hand shape localization and structure identification, the proposed method is significantly more accurate than previously proposed methods based on chamfer-distance matching. Furthermore, by integrating simple temporal constraints, the proposed method gains speed-ups of more than an order of magnitude, and produces highly accurate results in experiments on non-rigid hand motion tracking.

Index Terms

Object detection, shape modeling, probabilistic algorithms, dynamic programming.

I. INTRODUCTION

An important problem in computer vision is detecting objects in the presence of noise, clutter, and occlusions, and registering their shape with a model. It is desirable to use rich models that can capture a large range of possible object variations, and efficient methods that can register such models with an image. This paper introduces a detection algorithm that is explicitly designed for a large category of object classes where existing detection methods are not applicable: object classes that exhibit variable shape structure. In this paper we use the term "variable shape structure" to characterize object classes with any of the following properties:

- Some object parts can be repeated an arbitrary number of times, for example, the teeth of the hair combs and leaves on the branches in Fig. 1. The number of repetitions is not known a priori and can be different for different objects of the same class.
- Some object parts may be present or not present. For example, in the rightmost branch shown in Fig. 1, one of the leaves on the right side of the stem is missing. The leave does not appear in the image because it either does not exist or it is completely occluded. In

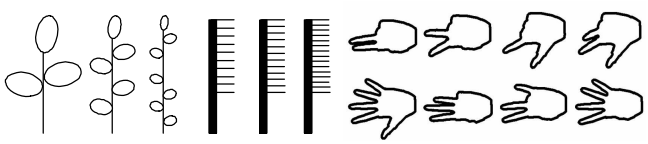

Fig. 1. Three object classes that exhibit variable shape structure: branches with leaves, hair combs, and hand contours. Such classes can be naturally modeled with a Hidden State Shape Model (HSSM).

both cases, the object belongs to the class "branch," even if typical instances of this class are branches without missing leaves.

- Some object parts can appear in alternative ways. Examples are the parts of articulated objects, such as the hands in Fig. 1, where each finger appears either totally extended, partially bent, or completely hidden.

Object classes of variable shape structure are frequently encountered in both man-made and natural objects. Blood vessels in the retina, airway ducts in the lung, and dendrites in nerve tissue are examples of biological objects with variable structure. Detecting such objects is important for tasks like diagnosing lung cancer or diseases of the retina. Roadways and waterways in aerial images are also examples of objects that have variable structure.

To model object classes of variable shape structure, we have introduced Hidden State Shape Models (HSSMs) [1], a generalization of Hidden Markov Models (HMMs) [2]. In HSSMs, alterative appearances of each object part are modeled by different model states. Using HSSMs, an object in an image can be localized and the shape structure of the object can be simultaneously identified by stepwise registration of these model states with the parts of the object. The computational complexity of this registration process is polynomial in terms of the total number of the model states and the total number of the observed features in the image, even in the presence of a significant amount of clutter.

The method proposed in this paper builds on top of the original HSSM method [1]. While the previous method [1] assumed that the scale of the object was known, the proposed method can handle more general realistic scenarios, where the object's scale in the image is not known a priori. In particular, the main contributions we make in this paper are summarized as follows: A unified probabilistic framework is formulated for object localization and structure identification

(Section IV). Within this framework, we demonstrate how the bias for the shortest registration sequence that is inherent in traditional HMM formulations is corrected by a novel "object-clutter model." In addition, a segmental HMM formulation ([3], [4]) is introduced to model the "duration probability" of each model state, so that the optimal combination of the object part scales is estimated. A polynomial inference algorithm automatically determines object location, orientation, scale and structure by finding the globally optimal registration of model states with the image features, even in the presence of clutter (Section V). Object scale is determined by evaluating registrations obtained at different scales. Our paper also provides a detailed approach for implementing the above framework (Section VI).

Experiments with real images of branches of leaves and hands demonstrate that the proposed method can localize objects of variable shape structure with high accuracy, improving upon the results we obtained using the original HSSM formulation [1]. For the task of hand shape localization and identification, the proposed method is also significantly more accurate than previously proposed methods based on chamfer-distance matching [5], [6]. Furthermore, by integrating simple temporal constraints, the proposed method gains speed-ups of more than an order of magnitude, and produces highly accurate results in experiments on non-rigid hand motion tracking.

II. Related Work

A large amount of work in computer vision addresses the issue of detecting deformable object shapes in images [7], [8]. Shock graphs [9] and FORMS [10] can be used to fit deformable models to silhouettes extracted from images, but these methods are sensitive to segmentation errors that change the topological properties of silhouettes. Such errors are frequent in the presence of noise and clutter. Another family of deformable models consists of active contour [11], [12] and shape [13] models. Even if prior information about object shape is incorporated into such models [14], [15], [16], [17], a deformable object in an image can typically not be detected automatically unless a good initial alignment between model and image object is provided.

Methods that rely on generative models, like graphical models, can be used to detect deformable shapes automatically, without requiring an initial guess [18], [19], [20]. When the graphical model is a sequence or a tree of nodes, dynamic programming can be used to find a globally optimal registration between the model and a set of possible part locations, even in the

presence of clutter [6], [21], [22], [23], [24], [25], [26]. A limitation of dynamic programming is that it cannot capture cyclical dependencies between shape parts. Graphical models that use iterative inference can capture such dependencies, with the tradeoff of not guaranteeing a globally optimal solution and of additional computational cost [18], [19], [20].

The main difference between our method and the above-mentioned methods is that our method can be used to model and detect object classes that exhibit *variable* shape structure. We should stress that "structure variation" is not synonymous with "deformation." Objects can be totally rigid and still exhibit variable structure as the hair combs in Fig. 1. Existing methods can model deformations of individual shape parts and deformations in the spatial arrangements of object parts; they cannot capture structure variations, for example, a shape part that is repeated an arbitrary number of times. Our method, in addition to modeling deformations, is explicitly designed to model variable shape structure.

Existing methods could only be used to detect objects with structure variation if each legal variation of the structure would be modeled separately. However, the approach of exhaustively modeling each variation as a fixed structure can be computationally intractable for many applications. For example, in the branch images shown in Fig. 1, a unique fixed structure is determined by specifying the number of leaves, and then specifying, for each leaf, if it occurs on the left or the right side of the stem. The number of possible fixed structures is exponential in the number of leaves, and thus, the time the above methods needed to check all these structures would also be exponential in the number of leaves. In contrast, our method captures shape variability with a *single* model, and thereby provides a polynomial-time solution for object detection.

HMMs[2] are simple and popular generative models, that are typically used to recognize temporal sequences of observations, but have also been used to recognize object shapes [27], [28], [29]. HMM-based shape recognition methods require object segmentation as a preprocessing step, and thus do not address the problem of localizing objects in clutter.

This paper describes a generative method for object detection that finds an optimal solution with an efficient inference algorithm. The method uses HSSMs, which can be viewed as a superclass of HMMs. HSSMs extend the functionality of HMMs in such a way that they can be applied to detect objects of variable shape structure in images with clutter and thus achieve object localization and structure identification simultaneously.

Complex and variable-structure shapes can also be modeled with shape grammars. Linden-

mayer systems (L-systems) have been used successfully in computer graphics for generating realistic images of biological shapes [30]. A generic shape grammar is used by Felzenszwalb [31] for the task of low-level image segmentation and grouping. A shape grammar can be used to improve the accuracy of rectangle detection in images [32]. The main difference between these methods [30], [31], [32] and the proposed method is that the latter, in addition to modeling object classes of variable shape structure, also addresses the issue of detecting instances of specific object classes in cluttered images.

III. Shape Modeling with HSSMs

This section starts with the formal definition of HSSMs (Section III-A). We then explain how HSSMs can be applied to locate an object and identify its shape structure (Section III-B). Finally, two important problems regarding object scale estimation are introduced (Section III-C).

A. Terminology and Notation

Designing an HSSM consists of the following two steps. First, a set of model states is specified, where each state corresponds to a possible part of the object shape. Second, probability functions are introduced to evaluate the likelihood of a match between a sequence of image features and a sequence of model states. More formally, an HSSM is specified by the following elements:

- $\mathbb{S} = \{s_1, \ldots, s_M\}$: a set of states that include M object part labels $\{s_1, \ldots, s_M\}$. Each state is associated with an object component.
- $\mathbb{E}$: a subset of $\mathbb{S}$ that defines legal end states of the HSSM.
- $\pi(s_i)$: the probability that state s_i is the initial state.
- $A(s_i, s_j) = p(s_j|s_i)$: the *state transition function* that represents the probability of transitioning from state s_i to state s_j.
- $B(f, s_i) = p(\phi|s_i)$: the *state observation function* that represents the probability of observing feature f with appearance ϕ in state s_i.
- $\tau(f', f, s_j, s_i) = p(y'|y, s_j, s_i)$: the *feature transition function* that represents the probability of observing some feature f' at location y' in state s_j given some other feature f was previously observed at location y in state s_i, where s_i could be equal to s_j if both f, f' belong to the same object part.

- $D(\ell, s_i, \omega) = p(\ell|s_i, \omega)$: the *state duration function* that represents the probability of continuously observing ℓ features in state s_i under the given object scale ω. Such a function D was used in the segmental HMM framework [3], [4]. The sequence of features observed in a state is called a *segment.*

Detailed definitions of probability functions π, A, B, τ and D will be given in Section IV.

Given a test image I, we assume that a set of K features $\mathbb{F} = \{f_1, \ldots, f_K\}$ has been extracted by some feature extraction method. We define $\mathbb{Q} = (q_1, \ldots, q_n)$ to be a sequence of n model states, and $\mathbb{O} = (o_1, \ldots, o_{\mathrm{L}})$ to be a sequence of L observations, where each $q_i \in \mathbb{S}$ and each $o_i \in \mathbb{F}$. We use the term "segment" for a subsequence of observations. Notation O_j^d denotes a segment $(o_j, ..., o_{j+d-1})$ of length d in observation sequence $\mathbb{O}$. We use notation $(O_j^d : q_i)$ to represent a segment-state pair, specifying that segment O_j^d is matched with model state q_i. A *registration* $R_{\mathbb{Q},\mathbb{O}}$ between the HSSM and the set $\mathbb{F}$ of image features is defined as an ordered sequence of segment-state pairs, i.e.,

$$\begin{aligned} R_{\mathbb{Q},\mathbb{O}} &= [(O_1^{d_1} : q_1), (O_{d_1+1}^{d_2} : q_2), \ldots, (O_{L-d_n+1}^{d_n} : q_n)] \\ &= [(o_1, .., o_{d_1} : q_1), (o_{d_1+1}, .., o_{d_1+d_2} : q_2), \ldots, (o_{L-d_n+1}, .., o_L : q_n)]. \end{aligned} \quad (1)$$

Since we model each object part with a specific model state, a registration thereby specifies which image features correspond to which object parts, and consequently, those features that are not matched with any model state are labeled as clutter. We use the term *registration length* for the length L of the sequence of features matching with states of the HSSM.

Our goal is to optimally register a sequence of image features of an object of unknown structure, location, and scale to a sequence of states of an HSSM that represents the object class. The "globally optimal registration," denoted as R_{opt}, is the most likely registration, which maximizes a joint probability function between image features $\mathbb{F}$ and the states of the HSSM. Since the number of all possible registrations is exponential, exhaustive search for this maximum is generally intractable. We give a detailed derivation of the joint probability function based on the definitions of probability functions π, A, B, τ and D and show how the joint probability can be maximized via a polynomial-time algorithm, using dynamic programming (Section V).

B. Some Examples of Shape Modeling Using HSSMs

Now we provide, using some examples, an intuitive description of how HSSMs can be used to model objects of variable shape structure. In particular, we provide example HSSMs for modeling object classes 'branch" and "hand." Their respective state transition diagrams are given in Figs. 2a and 3a, the features extracted from the input images are shown in Figs. 2b and 3b, and the registration results in Figs. 2c and 3c. For simplicity, each feature f_i corresponds to an edge pixel, and f_i stores the location and gradient orientation of that pixel.

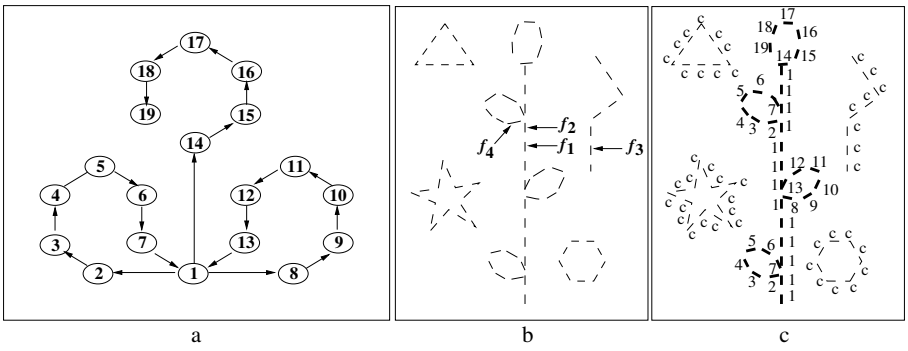

Fig. 2. Modeling the variable-structure class "branch with leaves" and detecting an instance of this class in an edge image. (a) State transition diagram of the HSSM "branch with leaves" that defines the states of the model and the legal transitions out of each state. State s_1 models the stem of the branch, states $s_2, \ldots, s_7$ model the leaves on the left side of the stem, states $s_8, \ldots, s_{13}$ the leaves on the right side, and states $s_{14}, \ldots, s_{19}$ the top leaf. State s_{19} is the only legal ending state. (b) Edge image that contains a branch of unknown structure, location, and scale, and some clutter (star, triangle). Edge features are illustrated by line segments, e.g., f_1, f_2, f_3 and f_4, etc. (c) An example registration of the model with the image features that results in detection of a branch with four alternate leaves. State labels are shown next to the features that the states in (a) were matched with. For example, features f_1, f_2 and f_4 in (b) can be modeled by object states 1 and 4, and feature f_3 as clutter (label c).

For both HSSMs, the initial probability $\pi(s_1)$ for state s_1 is one and for all other states is zero. The state transition probability $A(s_i, s_j)$ is set to a positive value for all their legal state transitions and to zero for all other transitions. The observation probability function $B(f_u, s_i)$ defines how likely it is to observe feature f_u in state s_i by measuring the difference between the observed edge orientation of feature f_u and the modeled orientation in state s_i. The feature transition function $\tau(f', f, s_j, s_i)$ captures the fact that, if feature f is matched to state s_i and then a transition from state s_i to state s_j is made, the feature f' matched to state s_j should appear in a position near f in the image, and the translation between f and f' should be compatible

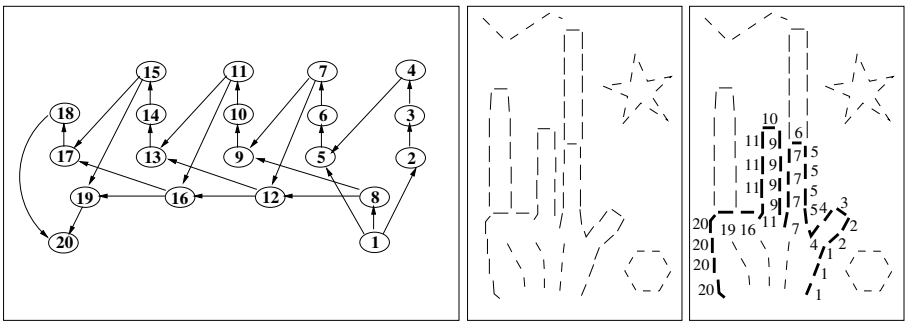

Fig. 3. Modeling the variable-structure class "hand" and detecting an instance of this class in an edge image. (a) State transition diagram of the HSSM "hand." State s_1 models the right boundary of the palm, states $s_2, \ldots, s_4$ model the outline of the thumb. States $\{s_5, s_6, s_7\}$, $\{s_9, s_{10}, s_{11}\}$, $\{s_{13}, s_{14}, s_{15}\}$, $\{s_{17}, s_{18}, s_{20}\}$, respectively model the shapes of other four fingers when they are extended, and states $s_8, s_{12}, s_{16}, s_{19}$ model the occluding boundary of each finger when they are hidden. State s_{20} models the left boundary of the palm and is the only legal ending state. (b) Edge image that contains a hand of unknown structure, location, and scale, and some clutter. c) An example registration of the model with the image features that results in detection of the hand with three extended fingers. State labels are shown next to the features that the states in a) were matched with (labels of clutter are not shown for clarity).

with the anticipated change between the modeled locations in states s_i and s_j. For the example shown in Fig. 2b, we have $\tau(f_2, f_1, s_1, s_1) > \tau(f_3, f_1, s_1, s_1)$ since f_2 is much closer to f_1 than f_3, and $\tau(f_4, f_2, s_2, s_1) > \tau(f_2, f_1, s_2, s_1)$ since the feature translation modeled by s_1 and s_2 is more compatible with the translation between f_2 and f_4 than between f_1 and f_2. Formal definitions for the above functions are provided in Section VI. They serve as an implementation example for the task of detecting the object classes "branch" and "hand."

We should stress that other HSSMs than the ones shown in Figs. 2a and 3a could be used to represent these classes if different types of features were chosen. For example, each f_i could correspond to the output of some object-part detector [24], [26], like a leaf detector if the objects are branches of leaves or a finger detector if the objects are hands. If we used such detectors, we would design HSSMs in which each model state would correspond to an entire leaf or finger.

C. Unknown-scale and Scale-dependency Problems

Since the specific shape structure of the object in an image is not known a priori, an HSSM-based object detection method may yield ambiguous detection hypotheses, for example, Figs. 2c and 4a–b, or 3c and Figs. 4c–d. The detection method needs to establish appropriate optimality

criteria, to identify the best out of all valid registration results. Here, two problems involving the scale estimation of the object in the image need to be addressed.

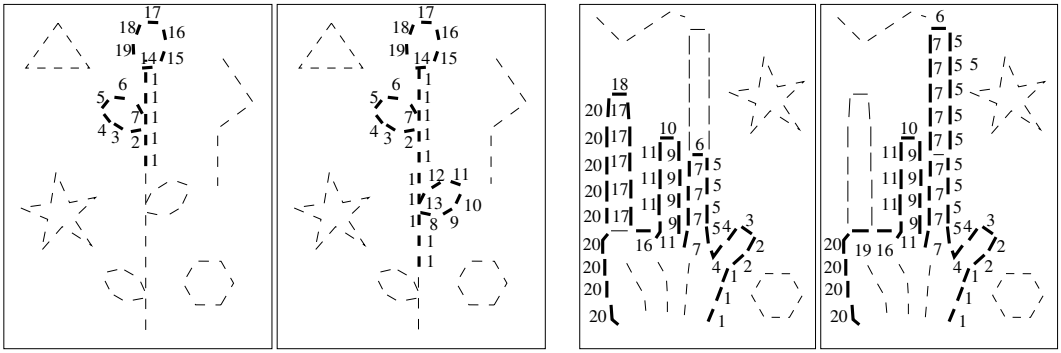

Fig. 4. An illustration of the unknown-scale problem (a,b), and the scale-dependency problem (c,d) using the HSSMs "branch" and "hand." (a–b): Registration results for a branch with two (a) and three (b) leaves. The proposed method resolves the ambiguity by determining which of these candidate registrations, including the branch with four leaves shown in Fig. 2c, corresponds to the optimal object scale. (c–d): Registration results describing a hand with four (c) and three (d) fingers. The proposed method resolves the ambiguity by determining which results, including the three-finger hand shape in Fig. 3c, corresponds to the most reasonable hand shape.

The first problem, called the *unknown-scale problem* (Fig. 4a–b), is that the number of image features that should match the model is not known a priori, i.e., the registration length is not known a priori. For example, considering the registration results shown in Figs. 4a–b and 2c, branches with two, three, or four leaves were detected by matching image feature sequences of different lengths to legal sequences of states. The unknown-scale problem consists of determining which of the valid registrations is optimal – here it is the four-leaf branch in Fig. 2c.

The second problem, called the *scale-dependency problem* (Fig. 4c–d), is that a registration may be composed of many locally well-matching features, but still not represent a globally meaningful result, because it contains extremely unlikely combinations of object part scales. Examples are given in Fig. 4c (little finger is too long) and Fig. 4d (forefinger is too long). The scale-dependency problem consists of correctly estimating the scales of different object parts and identifying their most likely configuration – here it is the two-finger hand in Fig. 3c.

When the shape structure of an object is known, traditional methods [33], [6] often applied an iterative and coarse-to-fine process to locate the object at the optimal scale. In our situation, the shape structure of the object is not known a priori. Shape structure variations may be coupled with

changes of the object scale (Fig. 4). Traditional multi-scale approaches and the accompanying normalization strategy cannot be applied here, since they would always favor the detection result that includes the fewest matching features, as we will explain later (Section IV-B). The above problems are therefore more general – and difficult – than they may appear initially.

IV. Object Localization and Structure Identification with HSSMs: A Probabilistic Framework

In this section, we formulate a unified probabilistic framework that can be used to find the optimal registration between model states and object features. The unknown-scale and scale-dependency problems are addressed by this framework.

A. Formulation of Probabilistic Framework

We describe our problem with the graphical model in Fig. 5, a Bayesian network [34] in which nodes represent random variables and arcs represent conditional dependencies. The first layer $\mathbb{Q}^+ = (\mathrm{q}_1, \ldots, \mathrm{q}_\mathrm{n}, q_c)$ consists of the sequence $\mathbb{Q}$ of n random variables that model states in $\mathbb{S}$ and a special random variable q_c that models clutter. [1] We require a state $s \in \mathbb{E}$ to be assigned to random variable q_n to guarantee that the last model state of a registration is a legal ending state. We also require $q_c = c$ where c is the label associated with clutter.

The second layer $\mathbb{O}^+ = (\mathrm{o}_1, \ldots, \mathrm{o}_\mathrm{L}, \{\mathrm{o}_{\mathrm{L}+1}, ..., \mathrm{o}_K\})$ consists of a sequence $\mathbb{O}$ of L observations of object features and a set of unordered observations $\mathbb{O}_c = \{\mathrm{o}_{\mathrm{L}+1}, \ldots, \mathrm{o}_K\}$ of clutter features, where each state o_i is assigned a feature $f \in \mathbb{F}$ and where $K = |\mathbb{F}|$ is the total number of image features. Each state q_i, for $1 \leq i \leq \mathrm{n}$, is associated with a random variable d_i that specifies the length of an observation segment. That is, each object component is composed of d_i object features and the total length of the object outline is $\mathrm{L} = \sum_{i=1}^{\mathrm{n}} \mathrm{d}_i$.

Given an instantiation of the above Bayesian network, each observation o_i can either be registered to an object state s or assigned the label c for clutter. Thus, the set of all features $\mathbb{F}$ is partitioned into the set of features $\mathbb{F}_o$ that belong to the object and the set of features $\mathbb{F}_c$ that are due to clutter, i.e., $\mathbb{F} = \mathbb{F}_o \cup \mathbb{F}_c$. The assignment process uniquely defines the registration

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

[1] We use font "roman" to denote random variables, font "italics" to indicate that the random variable has taken on a particular value.

by specifying the correspondence between states and observations with the set of parameters

$$\Lambda = \{[\mathrm{q}_1, ..., \mathrm{q}_\mathrm{n}], [\mathrm{d}_1, ..., \mathrm{d}_\mathrm{n}], [\mathrm{o}_1, ..., \mathrm{o}_\mathrm{L}]\}. \tag{2}$$

The registration can be computed by an algorithm (Section V) that optimizes the registration cost function derived within the probabilistic framework as follows.

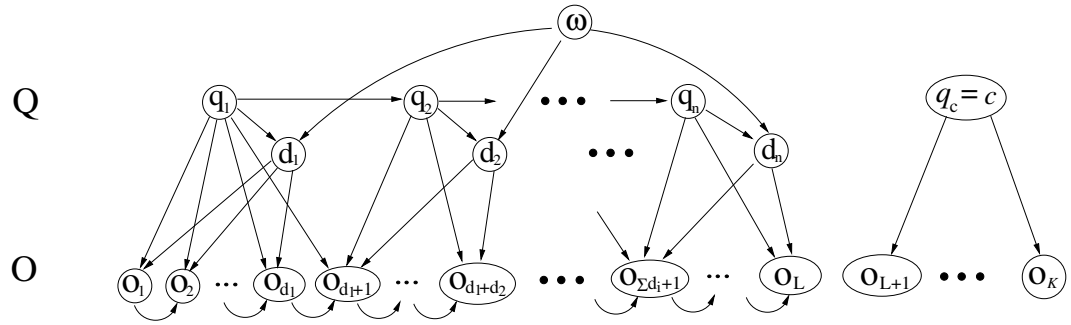

Fig. 5. Bayesian network for detecting objects of variable shape structure by HSSMs. Each image feature can be assigned to an object state or the "clutter" label. Each object state can model a segment of observations. The length of the segment is specified by d_i. A first-order Markov process is used to model the conditional dependencies between features that belong to the object. We assume each feature that is labeled as clutter to be conditionally independent.

Given the graphical model defined in Fig. 5 and the model parameters summarized by $\Omega = (\mathbb{S}, \pi, A, B, \tau, D)$, our goal is to maximize the conditional joint probability

$$\begin{aligned} p(\mathbb{Q}^+, \mathbb{O}^+) =& p(\mathbb{Q}, \mathbb{O})\ p(\mathrm{q}_c, \mathbb{O}_c) \\ =& p(\mathbb{Q}, \mathbb{O})\ [\prod_{\mathrm{o}_i \in \mathbb{O}_c} p(q_c, \mathrm{o}_i)\ ] \\ =& p(\mathbb{Q}, \mathbb{O})\ [\frac{\prod_{i=1}^{K} p(q_c, \mathrm{o}_i)}{\prod_{\mathrm{o}_i \in \mathbb{O}} p(q_c, \mathrm{o}_i)}\ ] \\ \propto& \frac{p(\mathbb{Q}, \mathbb{O})}{\prod_{\mathrm{o}_i \in \mathbb{O}}\ p(\mathrm{o}_i|q_c)\, p(q_c)}, \end{aligned} \tag{3}$$

where $\prod_{i=1}^{K} p(q_c, \mathrm{o}_i)$ is a constant and thus can be omitted, and where we considered each clutter feature to be conditionally independent. The foreground conditional probability is

$$\begin{aligned} p(\mathbb{Q}, \mathbb{O}) &= p(\mathrm{q}_1) \prod_{i=1}^{\mathrm{n}} [\ p(\mathrm{q}_i|\mathrm{q}_{i-1})\ p(\mathrm{d}_i|\mathrm{q}_i, \omega)\ p(\mathrm{O}_{j=\zeta(i)+1}^{\mathrm{d}_i}|\mathrm{q}_i, \mathrm{q}_{i-1})] \\ &= p(\mathrm{q}_1) \prod_{i=1}^{\mathrm{n}} [\ A(\mathrm{q}_{i-1}, \mathrm{q}_i)\ D(\mathrm{d}_i, \mathrm{q}_i, \omega)\ p(\mathrm{O}_{j=\zeta(i)+1}^{\mathrm{d}_i}|\mathrm{q}_i, \mathrm{q}_{i-1})], \end{aligned} \tag{4}$$

with segmental observation probability

$$p(\mathrm{O}_{j=\zeta(i)+1}^{\mathrm{d}_i}|\mathrm{q}_i,\mathrm{q}_{i-1}) = \tau(\mathrm{o}_{\zeta(i)+1},\mathrm{o}_{\zeta(i)},\mathrm{q}_i,\mathrm{q}_{i-1}) \prod_{j=\zeta(i)+1}^{\zeta(i)+\mathrm{d}_i} B(\mathrm{o}_j,\mathrm{q}_i) \prod_{j=\zeta(i)+2}^{\zeta(i)+\mathrm{d}_i} \tau(\mathrm{o}_j,\mathrm{o}_{j-1},\mathrm{q}_i,\mathrm{q}_i), \quad (5)$$

where function $\zeta(i) = \sum_{k=1}^{i-1} \mathrm{d}_k$ represents the length of the observation sequence before the i-th state is matched. Note that we introduce notations A, B, D and τ, as defined in Section III-A, to replace the corresponding probability terms in the above equation. Parameter ω specifies the object scale, as in Section III-A. Also, for notational convenience, we define $\zeta(1) = 0$ and $p(\mathrm{q}_1|\mathrm{q}_0) = p(\mathrm{o}_1|\mathrm{o}_0,\mathrm{q}_1,\mathrm{q}_0) = 1$. Appearance and location of feature o_j are assumed independent given state q_i. We require that q_i and q_{i-1} are instantiated with different model states.

By expanding Eq. 3 using Eqs. 4 and 5, we obtain

$$p(\mathbb{Q}^+,\mathbb{O}^+) \;=\; \pi(\mathrm{q}_1) \prod_{i=1}^{\mathrm{n}} \{A(\mathrm{q}_{i-1},\mathrm{q}_i) D(\mathrm{d}_i,\mathrm{q}_i,\omega) \prod_{j=\zeta(i)+1}^{\zeta(i)+\mathrm{d}_i} [\frac{B(\mathrm{o}_j,\mathrm{q}_i)}{B(\mathrm{o}_j,q_c)p(q_c)} \tau(\mathrm{o}_j,\mathrm{o}_{j-1},\mathrm{q}_i,\mathrm{q}_{i'})]\} \quad (6)$$

where $i' = i-1$ when $j = \zeta(i)+1$ and $i' = i$ otherwise.

By taking the negative logarithm of Eq. 6, we obtain a cost function for registration $R_{\mathbb{Q},\mathbb{O}}$:

$$C(R_{\mathbb{Q},\mathbb{O}}) = -\ln \pi(\mathrm{q}_1) - \sum_{i=1}^{\mathrm{n}} \{ \ln A(\mathrm{q}_{i-1},\mathrm{q}_i) + \ln D(\mathrm{d}_i,\mathrm{q}_i,\omega) + \xi(\mathrm{d}_i) + \sum_{j=\zeta(i)+1}^{\zeta(i)+\mathrm{d}_i} [\ln \frac{B(\mathrm{o}_j,\mathrm{q}_i)}{B(\mathrm{o}_j,q_c)} + \ln \tau(\mathrm{o}_j,\mathrm{o}_{j-1},\mathrm{q}_i,\mathrm{q}_{i'})]\}, \quad (7)$$

where function $\xi(\mathrm{d}_i) = \mathrm{d}_i \ln p(q_c)$ is a linear function of d_i.

Our optimization algorithm (Section V) finds optimal values for the set of parameters Λ so that Eq. 7 is minimized (or Eq. 6 is maximized). Accordingly, we define the globally optimal object registration R_{opt} as

$$R_{\mathrm{opt}} = \underset{R_{\mathbb{Q},\mathbb{O}}}{\operatorname{argmax}}\; p(\mathbb{Q}^+,\mathbb{O}^+) = \underset{R_{\mathbb{Q},\mathbb{O}}}{\operatorname{argmin}}\; C(R_{\mathbb{Q},\mathbb{O}}). \quad (8)$$

The optimal registration specifies the set of object features $\mathbb{F}_o$ and their ordering simultaneously.

When each $\mathrm{d}_1, \mathrm{d}_2, \ldots, \mathrm{d}_{\mathrm{n}} = 1$ in Eq. 7, we can define a simplified cost function for $R_{\mathbb{Q},\mathbb{O}}$:

$$C'(R_{\mathbb{Q},\mathbb{O}}) = -\ln \pi(\mathrm{q}_1) - \sum_{i=1}^{\mathrm{n}} \{\ln A(\mathrm{q}_{i-1},\mathrm{q}_i) + \ln \tfrac{B(\mathrm{o}_i,\mathrm{q}_i)}{B(\mathrm{o}_i,q_c)} + \ln \tau(\mathrm{o}_i,\mathrm{o}_{i-1},\mathrm{q}_i,\mathrm{q}_{i'})\}, \quad (9)$$

where q_{i-1} and q_i can be instantiated with any state $s \in \mathbb{S}$, possibly with the same state. This simpler cost function corresponds to an HSSM with no segmental components, and can be minimized with less expensive computations than Eq. 7, using the optimization algorithm that was described in the original HSSM method [1].

B. Discussion of Probabilistic Framework and Relation between HSSMs and HMMs

The above derivation of a probabilistic framework for object detection with HSSMs is aligned with traditional derivations involving HMMs, in particular, segmental HMMs. In the discussion of this framework we therefore focus on the differences to the traditional approach and stress our three main contributions. We explain how our design choices address the problems of 1) separating the object from clutter by determining the optimal registration length, 2) identifying the object shape from an unordered set of images features, and 3) determining the optimal scales of the object parts.

1) Handling Clutter: The traditional HMM-based recognition algorithm matches each observation to a model state [2]. In our object detection scenario, however, only a subset of the image features $\mathbb{F}$ may actually match the object model, while many (possibly most) observations will correspond to clutter. Note that once the optimal length of the registration is determined, we solve the unknown-scale problem and separate the object from clutter. This is because the unmatched features that are excluded in the registration are automatically considered clutter. However, we cannot use traditional HMMs for this problem because their probability of recognition is defined only for registered features, not for features that are excluded in the registration. Moreover, including an observation-state pair in a registration always decreases the overall recognition probability. Hence, the traditional HMM formulation [2] is inherently biased towards short registrations and cannot be applied here.

The key motivation in our formulation is that a correct probability formulation should explain the same number of input features, so that the costs that are associated with different registrations can be consistently defined. For the current application, instead of only explaining the features included in the registration, an ideal formulation should also explain the features that are excluded in the registration. In this way, the probability formulation always explains all features that were extracted from an image for different candidate registrations.

In our formulation (Eq. 6), the registration process either explicitly assigns each feature to an object state or gives to the feature the label "clutter." As opposed to maximizing $p(\mathrm{o}|\mathrm{q})$ in traditional HMM formulations, we maximize the likelihood ratio $p(\mathrm{o}|\mathrm{q})/p(\mathrm{o}|q_c)$. Incorrectly assigning an object feature to clutter or a clutter feature to an object state is penalized. For instance, when a feature is more likely to be explained by the object model than the clutter,

probability ratio $p(\mathrm{o}|\mathrm{q})/p(\mathrm{o}|q_c) > 1$ allows the overall registration probability defined in Eq. 6 to be increased; when a feature is more likely to be explained by the clutter than the object model, probability ratio $p(\mathrm{o}|\mathrm{q})/p(\mathrm{o}|q_c) < 1$ decreases the overall registration probability. In other words, with HSSMs, adding an observation-state pair to a registration may decrease or increase the overall registration cost, while, with traditional HMMs, this cost can never decrease. Therefore our method does not suffer from bias towards registrations that are short or registrations of a particular length.

Modeling of the background or clutter in an image has been useful in many detection systems. The work by Sidenbladh and Black [35] showed that modeling background-foreground statistics simultaneously can improve robustness and accuracy when tracking complex human motions. Paragios and Deriches [36] incorporated a likelihood ratio between background and foreground observations in a level set formulation and achieved promising image segmentation results. Our motivation here is different from the above works in the sense that, by accounting for the appearance of both the object and clutter, we solve the length-bias problem in traditional HMM formulations and determine the optimal registration length automatically.

2) Imposing an Ordering on Object Features: Traditional HMMs [2] were used to recognize temporal sequences of observations, where observations have a natural ordering based on the time they were observed. In our problem, however, the set $\mathbb{F}$ of features is *unordered*. Our probabilistic framework requires that the observations are instantiated by an ordered sequence of features. To provide a mechanism for comparing different possible feature orderings, we introduced a feature transition function τ, which is absent in the traditional HMM formulation [2].

The function τ is useful in situations where, given two consecutive states s_i and s_j, there may be two features f and f' such that probabilities $B(f, s_i)$ and $B(f, s_j)$ are very high, but features f and f' have some combined property that makes it unlikely for them to be consecutively observed in states s_i and s_j. To further explain this concept, we refer to the HSSM defined in Fig. 2. Function $p(\phi|s_1)$ measures the observation likelihood by comparing the edge orientation at f with the orientation modeled by state s_1, which corresponds to the upright orientation. Without the feature transition function τ, registration $[(f_1 : s_1), (f_2 : s_1)]$ is as likely as registration $[(f_1 : s_1), (f_3 : s_1)]$, since f_1, f_2, and f_3 all have the same upright orientation. However, with an appropriately designed feature transition probability function τ, the difference between successively observed features can be taken into account. A sequence

that involves an unlikely transition, e.g., $[(f_1 : s_1), (f_3 : s_1)]$, is penalized with a high registration cost. A transition between observed features that is compatible with the transition modeled by successive states, e.g., $[(f_1 : s_1), (f_2 : s_1)]$, is likely and thus yields a low registration cost.

We note that instead of using a function τ to model consecutive features, an alternative is to expand the state space, so that each state explicitly corresponds to a specific feature position and orientation.

3) Determining Scales of Object Parts: As we described in Section III-C, a desirable observation-state registration must comply with certain scale constraints between the shape parts of the object to be located (see Figs. 3c, and 4c–d). In order to solve the scale-dependency problem, we adopted the segmental HMM formulations in Eq. 6 to model the "duration behavior" of each object state, i.e., how many features may be continuously observed in the same state.

In a segmental HMM, a single state q_i models a sequence of (similar) observations, i.e., the segment, and their dependency relations. The modeling of such extra constraints is absent in traditional HMMs [2]. In Eq. 6, by specifying a common *reference scale* ω for the state duration probability $p(\mathrm{d}|\mathrm{q}, \omega; D)$, we explicitly model how likely an object part may appear at a particular scale. By this means we can incorporate the dependencies of the scales of different object parts as additional constraints into the registration process and thus achieve meaningful results.

A segment-based registration strategy has been shown more effective and accurate than traditional HMMs [2] in speech recognition applications [4]. However, to our knowledge, it has not been applied for shape modeling in object detection applications.

V. HSSM-Based Method for Finding Optimal Registration in Clutter

Given an HSSM model $\Omega = (\mathbb{S}, \pi, A, B, \tau, D)$ and a set of features $\{f_1, ..., f_K\}$ extracted from image I, our goal is to find the globally optimal registration R_{opt} (Eq. 8). This is equivalent to finding an optimal set of parameters $\Lambda = \{[\hat{\mathrm{q}}_1, ..., \hat{\mathrm{q}}_{\mathrm{n}}], [\hat{\mathrm{d}}_1, ..., \hat{\mathrm{d}}_{\mathrm{n}}], [\hat{\mathrm{o}}_1, ..., \hat{\mathrm{o}}_{\mathrm{L}}]\}$, i.e., states, state durations, and feature orderings, so that the registration cost as defined in Eq. 7 is minimized. We derived a dynamic programming method that finds the optimal sequence of states, as in the Viterbi algorithm, but also explicitly evaluated different state durations and feature orderings. The key difference between our algorithm and the standard Viterbi algorithm is that we added two extra dimensions to the search space.

We introduce notation $R_t(j, \ell, v)$ to represent a registration satisfying the following constraints:

1) The length of registration $R_t(j,\ell,v)$ is t.
2) The last observation segment $O_{t-\ell+1}^{\ell}$ in registration $R_t(j,\ell,v)$ has length ℓ and is matched with state s_j.
3) The last observation o_t in registration $R_t(j,\ell,v)$ is instantiated with feature f_v.

We then define the quantity

$$\delta_t(j,\ell,v) = \min_{R_t(j,\ell,v)} C(R_t(j,\ell,v)). \tag{10}$$

Note the notation differences: the traditional Viterbi algorithm [2] computes the recognition *probability* $\delta_t(j)$ as a function of registration index t and state j, while our algorithm computes the registration *cost* $\delta_t(j,\ell,v)$ as a function of registration index t, state j, state duration ℓ, and feature v. To simplify notation, we abbreviate the terms of Eq. 7 as follows: $\pi_i := -\ln \pi(s_i)$, $A_{ij} := -\ln A(s_i,s_j)$, $\tilde{B}_i(u) := -\ln \frac{B(f_u,s_i)}{B(f_u,c)}$, $\tau_{ij}(u,v) := -\ln \tau(f_v,f_u,s_j,s_i)$, and $D_i(\ell) := -\ln D(\ell,s_i)+\xi(\ell)$. We omit the scale parameter ω in the duration function D since it is provided as an input and thus is not part of the optimization task. To find the optimal registration R_{opt}, we need to keep track of the set of parameters Λ that minimizes Eq. 7, which is achieved via an array ψ. A variable λ is used to store intermediate costs. Note that M is the total number of model states, K the total number of features, ℓ_{max} the maximum number of features that can be observed in a state, and T_{max} the maximum registration length that we allow in practice.

HSSM-Based Registration Algorithm

- Initialization: For $t=1$, $1\le j\le M$, $1\le v\le K$:

$$\delta_1(j,1,v) = \begin{cases} \pi_j + \tilde{B}_j(v) + D_j(1), \text{ if } s_j \in \mathbb{E} \\ \infty, \text{ otherwise} \end{cases} \tag{11}$$
$$\psi_1(j,1,v) = (0,0,0)$$

- Recursion: For $1<t<T_{max}$, $1\le j\le M$, $1\le \ell\le \ell_{max}$, $\quad 1\le v\le K$:
  - if $t=\ell>1$

$$\delta_t(j,\ell,v) \;=\; \pi_j + D_j(\ell) + \tilde{B}_j(v) + \min_{1\le u\le K}[\tilde{B}_j(u) + \lambda_{jj}(1,u,t,v)],$$

  - if $t>\ell\ge 1$

$$\delta_t(j,\ell,v) = D_j(\ell)+\tilde{B}_j(v)+\min_{1\le i\le M, i\ne j}\{\min_{1<\gamma<t-\ell}\min_{1\le u\le K}[\delta_{t-\ell}(i,\gamma,u)+\lambda_{ij}(t-\ell,u,t,v)]+A_{ij}\}, \tag{12}$$

where

$$\lambda_{ij}(t',u,t,v)=\begin{cases}\tau_{ij}(u,v), & \text{if } t'=t-1\\ \min\limits_{1\le w\le K}[\lambda_{ij}(t',u,t-1,w)+\tilde{B}_j(w)+\tau_{jj}(w,v)], & \text{if } t'<t-1\end{cases} \tag{13}$$

Accordingly,

$$\psi_t(j,\ell,v)=\begin{cases}(i^*,\gamma^*,u^*), & \text{if } \ell=1\\ (j,\ell-1,w^*), & \text{if } \ell>1\end{cases}$$

where (i^*,γ^*,u^*) are the optimal values that minimize

Eq. 12, and w^* is the optimal value that minimizes Eq. 13.

- Termination: $t=T_{max}$

$$C(R_{opt})=\min_{t,j,\ell,v}\ \delta_t(j,\ell,v)\ \text{ s.t. }\ s_j\in\mathbb{E} \tag{14}$$

$$(t^*,j^*,\ell^*,v^*)=\operatorname*{argmin}_{t,j,\ell,v}\ \delta_t(j,\ell,v)\ \text{ s.t. }\ s_j\in\mathbb{E} \tag{15}$$

- Finding R_{opt} by backtracking:
  - $\mathrm{L}=t^*,\ \hat{\mathrm{q}}_{\mathrm{L}}=s_{j^*},\ \hat{\mathrm{o}}_{\mathrm{L}}=f_{v^*}$
  - For $t=\mathrm{L}-1,\mathrm{L}-2,...,1$:

$$(j',\ell',v')=\psi_{t+1}(j^*,\ell^*,v^*) \tag{16}$$

$$\hat{\mathrm{q}}_t=s_{j'},\ \hat{\mathrm{o}}_t=f_{v'} \tag{17}$$

$$j^*=j',\ \ell^*=\ell',\ v^*=v' \tag{18}$$

  - $R_{opt}=[(\hat{\mathrm{o}}_1:\hat{\mathrm{q}}_1),...,(\hat{\mathrm{o}}_{\mathrm{L}}:\hat{\mathrm{q}}_{\mathrm{L}})]$

A. Complexity Analysis of HSSM-Based Registration Algorithm

The computational complexity of the HSSM-Based registration algorithm is naturally higher than the standard HMM-based Viterbi algorithm [2], since it must also (1) evaluate observation segments of various length for each state, and (2) judge different orderings of feature sequences. In the initialization step, computing Eq. 11 in each iteration takes constant time. For the recursion step, we note an opportunity for precomputation so that computing Eq. 12 only requires at most $O(K)$ operations. In particular, when $t=2$,

$$\min_{1\le u\le K}[\tilde{B}_j(u)+\lambda_{jj}(1,u,2,v)]=\min_{1\le u\le K}[\tilde{B}_j(u)+\tau_{ij}(u,v)], \tag{19}$$

and when $t > 2$

$$\min_{1\le u\le K} \quad [\quad \tilde{B}_j(u) + \lambda_{jj}(1, u, t, v)] \tag{20}$$

$$= \min_{1\le w\le K} \quad \{\quad \min_{1\le u\le K}[\tilde{B}_j(u) + \lambda_{ij}(t', u, t-1, w)] + \tilde{B}_j(w) + \tau_{jj}(w, v)\}. \tag{21}$$

Quantity $\min_{1\le u\le K}[\tilde{B}_j(u) + \lambda_{ij}(t', u, t-1, w)]$ can be precomputed and stored in previous iterations for each w. For Eq. 12, only when $\ell = 1$, computing quantity

$$\min_{1\le i\le M, i\ne j}\{\min_{1<\gamma<t-\ell}\min_{1\le u\le K}[\delta_{t-\ell}(i, \gamma, u) + \lambda_{ij}(t-\ell, u, t, v)] + A_{ij}\} = \tag{22}$$

$$\min_{1\le i\le M, i\ne j}\{\min_{1<\gamma<t-1}\min_{1\le u\le K}[\delta_{t-1}(i, \gamma, u) + \tau_{ij}(u, v)] + A_{ij}\} \tag{23}$$

requires at most $O(MK\ell_{max})$ operations. For the cases when $\ell > 1$, we notice

$$\min_{1\le i\le M, i\ne j}\{\min_{1<\gamma<t-\ell}\min_{1\le u\le K}[\delta_{t-\ell}(i, \gamma, u) + \lambda_{ij}(t-\ell, u, t, v)] + A_{ij}\} = \tag{24}$$

$$\min_{1\le i\le M, i\ne j}\{\min_{1<w<K}\min_{1<\gamma<t-1}\min_{1\le u\le K}[\delta_{t-1-(\ell-1)}(i, \gamma, u) + \lambda_{ij}(t', u, t-1, w)] + \tag{25}$$

$$\tilde{B}_j(w) + \tau_{jj}(w, v) + A_{ij}\},$$

where quantity $\min_{1<\gamma<t-1}\min_{1\le u\le K}[\delta_{t-1-(\ell-1)}(i, \gamma, u) + \lambda_{ij}(t', u, t-1, w)]$ has been precomputed for each w in previous iterations. Computing the above equation thus requires $O(MK)$ operations.

Hence, within each iteration of computing $\delta_t(j, \ell, v)$ for each t, j, ℓ and v, $O(MK)$ operations are required, and $O(MK\ell_{max})$ operations only when $\ell = 1$. This results in an overall complexity of $O(M^2K^2\ell_{max}T_{max})$ for the registration algorithm. The complexity is polynomial and thus lower than the exponential complexity of the exhaustive search (see Section III-A).

It is important to note that the above complexity analysis is a worst-case analysis. In general, the time complexity is quadratic in the number of states, but for specific shape models the complexity can be linear. In the case of our models of hands and branches of leaves, the time complexity is linear in the number of states if the number of legal position transitions out of any state in the model is bounded by a constant. Similarly, in practice, the time complexity is linear in the number of features because the location of a feature severely constrains the location of the next feature in the registration. For example, in our implementation we define $\tau(f_v, f_u, s_j, s_i) = 0$ when the Euclidean distance between f_v and f_u is beyond a certain threshold. For each feature f_u, we can precompute a set of neighboring features $N_f(u)$ that can legally succeed f_u in a registration. If the number of neighboring features is bounded above by a constant, the complexity of the registration process becomes linear in the number of features.

In practice, our implementation of the proposed HSSM-based registration algorithm requires $O(MC_sKC_f\ell_{max}T_{max}) \ll O(2^{36})$ operations for $C_s = 2$ and $C_f = 20$, where C_s and C_f are constants that represent the average size of legal state transitions out of each state and legal feature transitions out of each feature, respectively. In addition to the optimization algorithm described here, we also applied a simplified and much faster optimization algorithm, as described in our preliminary work [1], to minimize the alternative cost function defined in Eq. 9. This faster algorithm is simply obtained by setting state duration variables $\ell = \gamma = 1$ in the registration algorithm, as described in Eqs. 11-18. As we will show later (Section VII), this faster optimization procedure produces slightly less accurate results while it requires only $O(MC_sKC_fT_{max}) \ll O(2^{31})$ operations.

VI. An Approach for Implementing HSSMs

In this section, we describe in detail an example implementation of the probabilistic framework we derived in Section IV. We should stress that the probabilistic framework and HSSM-based registration algorithm that we derived above are general and not restricted to a particular implementation. The feature model and probability functions we describe below are just particular choices we adopted in the current implementation. There could be several alternative ways to set up an HSSM model for specific object class, like we mentioned earlier in Section III-B.

In the current implementation, we assume that the state diagram of an HSSM, i.e., the number of states and their connectivity topology, was determined in advance, and our focus here is on how to define the probability functions π, A, B, τ, and D, either manually or automatically. We found it quite straightforward to define the initial function π and state transition function A manually. In particular, we require that π returns the same positive value for all legal starting states, and zero for all other states. Similarly, we define the state transition function A as a uniform function with respect to all legal transitions from the given state, and assign the transition probabilities to be zero for illegal state transitions. The following subsections describe how we defined probabilities $\frac{B(f,s)}{B(f,c)}$, $\tau(f, f', s, s')$ and $D(\ell, s, \omega)$ in our implementation.

A. Object-Clutter Observation Model

Each observation is a random variable o that can be assigned to some image feature f. As illustrated in Fig. 6, in the current implementation, we define an image feature $f \in \mathbb{F}$ as a

local image patch surrounding an edge pixel, and feature f is measured by its appearance ϕ and location y, i.e., $f = (\phi, y)$. Furthermore, the patch appearance ϕ is summarized by its color distribution, represented by vector ϕ_χ, and the intensity gradient orientation on the patch center, represented by scalar ϕ_g that ranges from 0 to 2π.

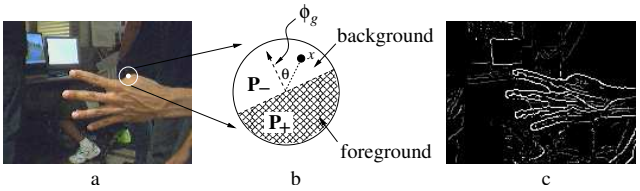

Fig. 6. An illustration of local feature patch. (a): Input image. (b): Close view of the local image feature patch. (c): SVM posterior ratio map, where the grayscale value of each edge pixel is computed by normalizing the posterior ratio (Eq. 29), for the associated local patch.

To define the observation probability function B, we need to determine how likely a feature f is observed in state s. For this purpose, we model the color distribution s_χ of a foreground boundary patch and the image gradient s_g on the center of the patch for each object state $\mathrm{q} = s \in \mathbb{S}$. We then define the observation probability

$$B(f, s) = p(\phi|s) = p((\phi_\chi, \phi_g)|(s_\chi, s_g)) = p(\phi_\chi|s_\chi)\, p(\phi_g|s_g), \tag{26}$$

where $p(\phi_g|s_g)$ is modeled by a Gaussian distribution with predefined mean s_g and variance σ_g^2, which can be easily learned from training data. We assume that the color ϕ_χ and intensity gradient ϕ_g are conditionally independent given the current model state s. We then simply measure the difference between the observed ϕ_χ, ϕ_g and the predefined s_χ, s_g.

Modeling the likelihood function $p(\phi_\chi|s_\chi)$ for Eq. 26 can be challenging because of the high dimensionality of the feature color distribution and limited training data. For many classification tasks [37], [38], [39], however, good discriminative models were developed that achieved satisfactory results. This observation motivates us to rewrite the likelihood ratio in Eq. 6 as

$$\frac{B(f, s)}{B(f, c)} = \frac{p(\phi_\chi|s_\chi)\, p(\phi_g|s_g)}{p(\phi_\chi|c_\chi)\, p(\phi_g|c_g)} \propto \frac{p(\mathrm{q} = s_\chi|\phi_\chi)\, p(\phi_g|s_g)}{p(\mathrm{q} = c_\chi|\phi_\chi)} \tag{27}$$

$$= \frac{p(\mathrm{q} = s|\phi_\chi)\, p(\phi_g|s_g)}{p(\mathrm{q} = c|\phi_\chi)}. \tag{28}$$

Probability $p(\phi_g|c_g)$ is defined as a constant, since we do not model any preference with regards to the gradient direction of a clutter feature. Given an input image patch described by ϕ_χ,

posterior probabilities $p(\mathrm{q} = s|\phi_\chi)$ or $p(\mathrm{q} = c|\phi_\chi)$ determines how likely the patch belongs to the object, $\mathrm{q} = s \in \mathbb{S}$, or to clutter, $\mathrm{q} = c$. In the current implementation, we have found that a simple two-class Support Vector Machine (SVM) classifier works well in approximating the above classification probabilities [40]. The resulting posterior ratio is defined as:

$$\frac{p(\mathrm{q} = s \,|\, \phi_\chi)}{p(\mathrm{q} = c \,|\, \phi_\chi)} \;\approx\; \exp(-\,\gamma\, h(\phi_\chi)), \tag{29}$$

where function h, computed by the SVM for given input ϕ_χ, outputs a decision value for classification purposes and scalar factor γ is determined experimentally.

In order to construct ϕ_χ to record sufficient discriminative information for the purpose of classification, we first split the input image patch into two half patches, i.e., $\mathbf{P}_+$ and $\mathbf{P}_-$, along the intensity gradient direction on the patch center (see Fig. 6). We then define vector ϕ_χ by arranging the weighted color intensities that distribute inside $\mathbf{P}_+$ and $\mathbf{P}_-$ in a certain order based on the patch gradient information, so that the definition of ϕ_χ is invariant to different orientations of the patch gradient. The weight of the color intensity of a pixel x is defined as $\cos\theta$, where θ is the angle between the vector that points to x from the center of the patch and the vector associated with the gradient orientation (see Fig. 6). By this means, ϕ_χ typically records enough foreground-background contrast information to identify a patch that is located on the object boundary or surrounds a clutter pixel, which makes the later classification task straightforward.

By the experiments described below, we have found that the two-class SVM works sufficiently well for the current application. More experimental details on learning the SVM classifier from training data are given in Section VII.

B. Feature Transition Model

Given an unordered set $\mathbb{F}$ of features as the input, the HSSM-based registration method needs to evaluate the ordering of features in a registration. We introduced the feature transition function to model the dependency between successive observations. In particular, we consider the locations and orientations of the features to be dependent only between successive observations in a registration. We define the feature transition probability

$$\begin{aligned} \tau(f, f', s, s') &= p(y|y', s, s') \\ &= \alpha\, e^{-\alpha(\|y'-y\| - \Delta(s_y, s'_y))} \end{aligned} \tag{30}$$

as an exponential distribution, where $\| y - y' \|$ represents the Euclidean distance between the centers of two patches f, f', and $\Delta(s_y, s'_y)$ defines the ideal distance translation between two features that are matched with states s and s'. The weighting scalar α can be learned from the training data via Maximum-Likelihood (ML) estimation.

The above feature transition model depends on the difference in position and orientation between f and f'. These definitions make the resulting HSSM models invariant to translation of the object in the image, since we use only relative feature location with respect to the location of the previous feature, instead of absolute feature locations.

C. State Duration Model

Given an object registration, the scale of an object part is represented by the number of features matched to the corresponding shape state. The scales of different object parts in a registration should be consistently maintained. For instance, for any particular hand shape, it is very unlikely to simultaneously observe a very long and a very short extended finger in the same image. We use the state duration function to capture this scale dependency of different shape parts.

Suppose that the scale of some object part, which is called as the *reference scale*, is known. We can then normalize the scale of each object part with respect to this reference scale. The statistics of the resulting relative scales is used to approximate the duration distribution of each shape state s, for which we apply the Gaussian mixture model

$$D(\ell, s, \omega) = p(\mathrm{d} = \ell | \mathrm{q} = s, \omega) = \sum_{i}^{n} p(\ell | \mu_i(\omega), s) p(\mu_i(\omega) | s), \tag{31}$$

where ω is provided as the input to specify the reference scale, $p(\ell|\mu_i(\omega), s)$ is a normal distribution with mean $\mu_i(\omega)$ and variance $\sigma_i^2(\omega)$, and $p(\mu_i(\omega)|s)$ is the conditional prior for the Gaussian distribution. Note that both mean μ_i and variance σ_i^2 are functions of reference scale ω. The value of n controls the degrees of freedom to which shape variation is allowed. In practice, we can learn the relative scale statistics of each object part from a training set using ground-truth registration results. An Expectation-Maximization (EM) procedure can be applied to find the optimal estimates for parameters μ_i and σ_i with respect to all states $s \in \mathbb{S}$.

The above approach provides an object detection mechanism that is transition invariant and can automatically estimate the object scale by determining the optimal registration length. In order to detect an object with an unknown orientation, we performed the registration processes

multiple times. In each registration process, we updated the states of the HSSM by increasing their predefined mean s_g (Eq. 26) by a certain degree, e.g., $\pi/4$ in the current implementation. This resulted in a total of eight registration processes per input image. Exhaustive search over these registration results then identified the orientation that gave the lowest registration cost. Furthermore, to determine the optimal combination of the object part scales without knowing the reference scale ω (Eq. 31), we performed a multi-scale registration process by testing different values of the reference scale ω.

VII. Experiments

We have implemented three versions of the HSSM method in C++ on an AMD Opteron 2.0 GHz processor. The first version (abbreviated HSSM) implemented the original HSSM method [1]. The second version (abbreviated HSSM+OC) minimized the cost function that includes only object-clutter modeling (Eq. 9), using the same optimization algorithm as desribed in the original HSSM method [1]. The third version (abbreviated HSSM+OC+SEG) minimized the cost function that includes both object-clutter modeling and state-duration modeling with segments (Eq. 7), using the optimization algorithm we described in Section V.

In both HSSM+OC and HSSM+OC+SEG implementations, we specified a fixed minimum and maximum registration length $L_{min} = 100$ and $L_{max} = 600$ for all input images. Both HSSM+OC and HSSM+OC+SEG implementations then identified an optimal solution based on the minimum cost stored in the dynamic programming table within this length range. As a result, the optimal registration length was determined automatically. In contrast, the original HSSM method assumed that the registration length, i.e., the scale of the object, was known.

The HSSM and HSSM+OC versions took 5–6 minutes to process each image, including trying all eight orientations. The memory size of the program is under 400 MB. For version HSSM+OC+SEG, it took about 25–30 minutes to process each image, including trying all eight orientations and different reference scales. The memory size of the program is under 1800 MB.

A. Detecting Objects with Variable Shape Structure

1) Dataset: We have evaluated the HSSM-based object detection method with two datasets of real images, containing two types of objects of variable structure. The first dataset consists of 100 images of branches of leaves. The second dataset consists of 353 hand images. Each test

image contains 120×160 pixels. Edges were extracted using a Canny edge detector [41]. There were between 2000 and 4000 edge pixels extracted from each image.

The following shape variations are present in the two datasets. Each branch image includes an unknown number of leaves, where a leaf occurs either at the left or right side of the branch stem. Each hand image shows the back of the palm. The camera viewing direction is perpendicular to the palm surface, and each of the five fingers can either be extended or hidden. Example images are shown in Figs. 8 and 9.

The task of our experiments can be summed up as follows: the system knows that there is a single object of the desired class in the image, and the goal is to locate the object and identify the orientation and shape structure of the object.

In order to provide quantitative measures of accuracy, we use the following terms:

- "Correct structure identification": the system has found the object in the correct orientation at the correct location, has correctly estimated the number of object parts, and has correctly registered each part.
- "Correct localization": the system has identified the correct object location and orientation. For branches, we require that 75% of the stem is registered correctly, and for hands, we require that the 75% of the palm edges are registered correctly. We allow incorrect estimation of the number and/or location of some shape parts, and incorrect registration of some shape parts. Note that "correct structure identification" is a subcase of "correct localization."
- "Incorrect localization": the method failed to find the correct object location and orientation.

The meaning of each of these accuracy measures is illustrated with examples in Figs. 8 and 9.

2) Learning: To learn the object-clutter model and its parameters σ_g and γ (Eqs. 26 and 29), we collected training sets of 40 hand images and 20 branch images, where the correct object boundary edges were localized by the original HSSM method [1]. The method labeled 68,000 edge patches that were due to clutter, and 25,000 edge patches on the object boundaries. Among these, 15,000 patches were on the hand boundaries and 10,000 patches on the boundaries of the branches. Given the object boundary image patches, a fixed value $\sigma_g = 0.13$ in Eq. 28 was learned by maximum-likelihood (ML) estimation. In order to classify a patch belonging to the object or clutter, we used the LIBSVM implementation [42] to build a linear SVM. We then used one fourth of the total number of patches to *train* the SVM and used the remaining patches for *testing*. In the testing stage, the correct classification rate was 87.2%. Once we trained the

SVM to compute $h(\phi_\chi)$ (Eq. 29), we determined the value $\gamma = 7.0$ experimentally.

Our strategy in tuning the SVM was to keep a high operating point, allowing high true and false positive detection rates. As a result, the SVM generally did not classify the object boundary edges as clutter, but may classify some edges that belong to clutter as the object boundaries. The latter will be identified correctly during the registration phase.

For the feature transition model (Eq. 30), we determined the values $\alpha = 0.40$ and $\beta = 0.14$ via ML estimation, based on the 25,000 boundary edges labeled by the original HSSM method [1]. Moreover, we did not allow transitions between features that are more than five pixels away.

The state duration models can be learned from the training data, as we demonstrate below for the hand detection application. In the HSSM for hand shapes, each shape state models the appearance of a certain part of the hand boundary. In the data collection process, two volunteers were asked to bend or extend each of their fingers (except the thumb), which resulted in 16 different poses. To facilitate accurate training of the state duration parameters, we asked the volunteers to vary their hand pose slightly, in particular, to vary the degree to which the fingers were bent without changing the set of fingers that were extended. We took a total of 1,600 images of two left hands against a neutral background. For each hand pose, 20 images were randomly chosen and added into a hand shape training set, which included $16 \times 20 = 320$ images in total. The labeled boundary edges were identified by the original HSSM [1]. Some example images are shown in Fig. 7. Afterwards, we first specified the number of Gaussian mixtures in Eq. 31. For instance, we chose $n = 2$ to model the scale statistics of a finger, so that the finger shape is allowed to present two modes, i.e., either the finger is partially bent at the first joint or it is fully extended. For the state associated with a finger tip, we chose $n = 1$ since the scale of the finger tip is almost unchanged (see Fig. 7). Second, we measured the scale of each finger part by the number of pixels that are associated with the same state. The relative scale of each object part is computed by normalizing its size with the reference scale ω, i.e., the scale of the thumb in this example. Given the relative scale statistics of shape parts in all images, we found the optimal estimates for parameters μ_i and σ_i by Expectation-Maximization (EM). For example, the two Gaussian means in the duration model of state 9 (second finger side in Fig. 3a) are 1.13 and 0.5, since the length of the second finger has either the same size of the thumb when it is extended (mean 1.13), or the half size of the thumb when it is bent (mean 0.5). Similarly, the Gaussian mean of state 3 (the first finger tip in Fig. 3a) is 0.19, since the width of the finger tip

is very small compared to the size of the thumb.

In our dataset, the size of the thumb ranges from 15 pixels to 35 pixels typically. In order to determine the optimal configuration of object part scales, the HSSM+OC+SEG method performed a multi-scale registration process with the reference scale ω from the set $\{15, 20, 25, 30, 35\}$. Each element of this set represents a different size of the thumb in pixel units. Exhaustive search over these scale-related registration results identified the optimal combination of the scales between object parts.

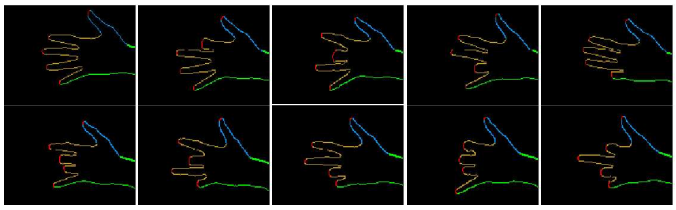

Fig. 7. Examples of the labeled hand boundary edges for learning state duration models. Each pixel was colored based on the state it is matched with (see Fig. 3a). For example, *green* pixels are those matched with the left or right side of the wrist and hand (states 1 and 20); *blue* pixels are those matched with both sides of the thumb (states 2 and 4); *orange* pixels are those matched with the both sides of a finger, (states 5, 7, 9, 11, 13, 15, and 17); *red* pixels are those matched with the finger tips (states 3, 6, 10, 14, and 18).

3) Testing: A quantitative evaluation of the proposed method on two real image datasets is reported in Table 1. Our method achieved accurate localization and structure identification results that were invariant to object translations, orientations and scales, even if the images included a large amount of clutter. We used the parameter values specified above in all experiments.

TABLE I

OBJECT DETECTION RESULTS OF HSSM ON BRANCH AND HAND DATASETS.

Dataset	Branches		Hands				
Methods	HSSM + OC	HSSM	HSSM +OC+SEG	HSSM + OC	HSSM	Chamfer dist. + orientations	Chamfer distance
Number of orientations	8	8	8	8	8	72	72
Correct structure identification	65.0%	43.0%	70.2%	59.3%	33.7%	21.8%	4.0%
Correct localization	95.0%	79.0%	95.2%	85.3%	59.5%	54.6%	35.2%
Incorrect localization	5.0%	21.0%	4.8%	14.7%	40.5%	45.4%	64.8%

In experiments with branch images, we compared the proposed HSSM+OC algorithm with the original HSSM [1]. The proposed method resulted in significant improvements, i.e., 65% correct structure identification rate and 95% correct localization rate, in comparison to 43% correct structure identification rate and 79% correct localization rate of the previous method. Some representative images are shown in Fig. 8.

In experiments with hand images, we compared both of the proposed HSSM+OC and HSSM+OC+SEG methods to the original HSSM [1], to the chamfer-distance method [5], and to the enhanced chamfer-distance method (denoted here as chamfer-distance + orientations) that was used by Thayananthan et al. [6] for hand localization. Some representative images are shown in Fig. 8.

Since there are five fingers in a hand, and each finger can be extended or hidden, we need 32 fixed-structure models in the chamfer-distance method to represent all valid hand shapes. In contrast, a single HSSM suffices for modeling the entire range of variations. For the chamfer matching method, "correct localization" means that the best response was obtained at the correct position (up to a displacement of half the size of the palm) and orientation (up to 45 degrees). "Correct structure identification" means that, in addition to obtaining correct localization, the best response was obtained by the correct fixed-structure model. Since the chamfer-distance method is not tolerant to large image-plane rotations, we evaluated it on 72 orientations of each test image. To achieve scale-invariant detection, each hand model was represented in 20 different scales. Hand localization using the chamfer-distance method took about 1–2 minute per image which included an exhaustive search over 72 orientations. As seen in Table I, our method was more accurate than both variants of the chamfer-distance method, in terms of both correct localization and correct structure identification.

In branch and hand experiments, the object-clutter modeling lead to a significant improvement in both the correct structure identification rate and the correct localization rate, compared to the original HSSM method [1]. We should emphasize that the proposed method is also more general than the original HSSM method [1]. This is because the results computed with the original method [1] were obtained by specifying, as input, the desired registration length for each image. In contrast, the proposed method performed a more difficult task, because the optimal registration length was unknown.

We also demonstrate that, the HSSM+OC+SEG method produced more accurate results than the proposed HSSM+OC method. As shown in Fig. 9e–j, by constraining how many features may

be successively observed in some state, the HSSM+OC+SEG method can improve over many incorrect structure identification or localization results of the proposed HSSM+OC method. Even for the correct structure identification results achieved by the HSSM+OC method in Fig. 9a–d, the HSSM+OC+SEG method produced more accurate results in identifying the locally matching features. For instance, the right side of the thumb in Fig. 9a–b and the right side of the wrist in Fig. 9c–d were localized more precisely by the HSSM+OC+SEG method.

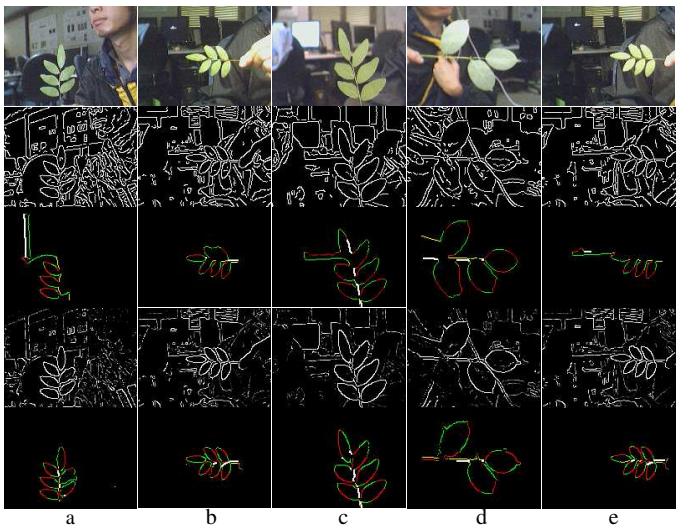

Fig. 8. Example results on images of the branch dataset. Row 1: Input images. Row 2: Canny edge images. Row 3: Results computed by the original HSSM method [1] based on Canny edge images. Results in columns b–d represent incorrect structure identification, results in columns a and e represent incorrect localization. Row 4: SVM posterior ratio map. Row 5: Results computed by the proposed HSSM+OC method based on posterior ratio maps. Results correspond to correct structure identification. Each pixel on the detected branch objects is colored based on its matching state in the branch HSSM (Fig. 2a). For example, *yellow* or *white* pixels match with state 1 that models the branch stem; *green* pixels match with *even* states in Fig. 2a; *red* pixels match with other *odd* states in Fig. 2a.

B. Tracking Non-rigid Motion

With simple implementation manipulations, the proposed method was extended to track non-rigid hand motion. Tracking was tested on five videos: one video (including 259 frames) with a simple background, one video with cluttered background but without partial occlusions (615 frames), two more challenging videos (including 633 and 588 frames each) with cluttered backgrounds, partial occlusions and large camera movements, and one most challenging video

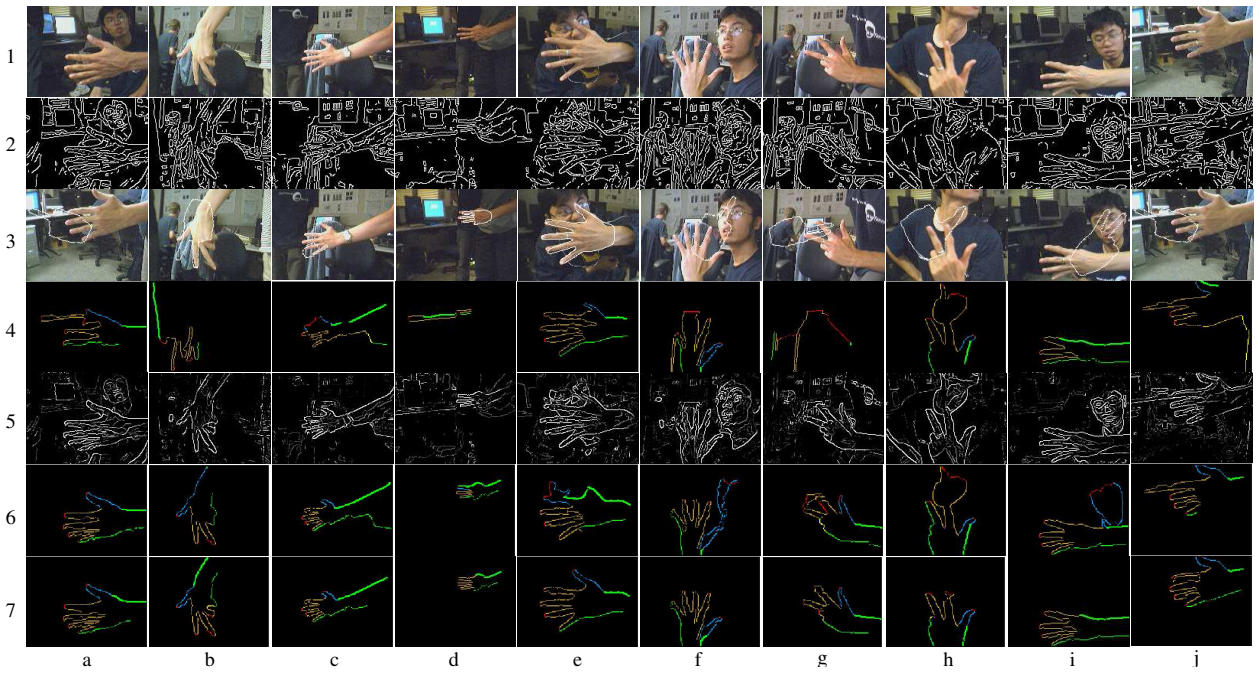

Fig. 9. Example results on images of the hand dataset. Row 1: Input images. Row 2: Canny edge images. Row 3: Results computed by the enhanced chamfer-distance method [6], where the matching hand contours are plotted in white. Note that the results in columns d–e correspond to correct structure identification, and the result in column b represents correct localization. Row 4: Results computed by the original HSSM method [1], where only the result in column e corresponds to correct structure identification, and results in columns a, c, and h–j correspond to correct localization. Row 5: SVM posterior ratio map. Row 6: Results computed by the proposed HSSM+OC method, where results in columns a–c correspond to correct structure identification, and results in other columns correspond to correct localization. Row 7: Results computed by theone proposed HSSM+OC+SEG method, where all results correspond to correct structure identification.

(including 510 frames) with hands placed above the subject's face [2]. Note that in all videos, only small variations of the camera's viewing direction, so that the back of the palm was relatively perpendicular to the viewing direction.

For efficiency purposes, we performed a simple frame-by-frame detection by the HSSM+OC method to track the hand in each frame. When the orientation of the hand is unknown, the system exhaustively evaluated the registration results on 8 different orientations only for the first frame. For each of the subsequent frames, the system updated the orientation that was associated with each state in the HSSM, based on the detected hand on the previous frame. Furthermore, since no large motions occurred in our experiments, we applied a local window that surrounded

[2]Results of tracking non-rigid hand motion are available at http://cs-people.bu.edu/athitsos/variable_structure/.

the previously detected hand to prune out many candidate edges in the current frame. With these implementation choices, our system tracked the hand pose at a speed of 3–5 seconds per frame. Note that no additional temporal constraints were employed in the current implementation. Incorporating recursive Bayesian filters like Kalman [43] or particle filters [44] remains a topic for future investigation.

We have achieved a correct structure identification rate of 75.0% and a correct localization rate of 95.0% over a total of 2,605 frames of 5 video sequences. There are three important results we have observed from our experiments (Fig. 10). First, because the HSSM+OC method can detect objects from heavily cluttered images, the tracking system is robust to camera motion, large background changes, and motion blur. Second, because the HSSM+OC method automatically estimated the optimal scale of the object, our tracking system is robust to the large changes in object scale. Lastly, even when the object was partially occluded, e.g., on frame 464 in Fig. 10, our system still produced reasonable interpretations of the observed features [3], which will be discussed next.

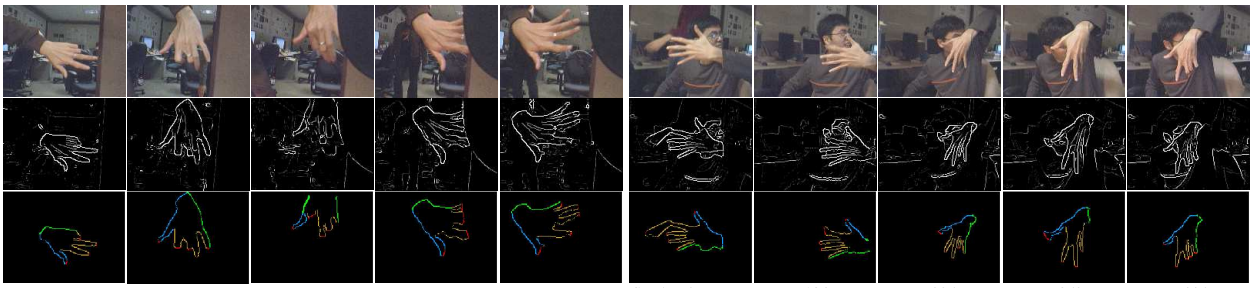

Fig. 10. Example results from tracking non-rigid hand motion. Row 1: Input images. Row 2: SVM posterior ratio map. Row 3: Results computed by the HSSM+OC method. The frame number in the sequence is given below each column. Incorrect structure identification results appear on frame 393 in sequence 4 and frame 97 in sequence 5.

VIII. Conclusions and Future Work

We have presented the first method for modeling object classes of variable shape structure and detecting instances of such classes in heavily cluttered images. The method used HSSMs

[3]Note that we do not claim the results we produced under partial occlusions are *correct*, only because the proposed method cannot infer the pose of the occluded parts of the object.

and dynamic programming to find the globally optimal registration between model and image object in polynomial time. The method can detect objects whose scale, orientation and even shape structure is not known a priori.

Partial occlusions are naturally handled by the proposed registration method, even though they are not explicitly modeled. For instance, when there are partial occlusions, which often result in small gaps on the object boundaries, the HSSM-based registration algorithm can still "bridge" separated features by matching states with those "incomplete" features (see Fig. 11a–b). Structure variability can be modeled by the object's HSSM even if an object part is completely occluded, for example, the absent fingers in frames 393 and 462 in Fig. 10 or the absent leaves in Fig. 11c. The registration process "skips" the state of the HSSM that models the occluded fingers and thus accounts for the structure variability of the hand. In cases where features are occluded that would otherwise (without the occlusion) correspond to some state of the HSSM, the registration process may be forced to assign other, less ideally matching features to this state. The process would then terminate with an optimal registration that included some mismatched features. It would result in a detection of the object that could be incorrect for some object parts.

In our experimental setting, there was exactly one object of interest, and the method tried to find the best registration hypothesis for that object. However, our method can also be applied in a more classical detection setting, where it is not known a priori if there are zero, one, or multiple instances of an object. Some preliminary results for multiple instance detection, which corresponded to the two registrations with lowest costs, are shown in Fig. 11d–e.

The generality of the proposed framework allows us to operate with different levels of features. In the experiments we demonstrated, when operating with low-level features, the proposed method can provide very accurate localization and structure identification results within the order of a pixel. Incorporating more descriptive features, like shape context [45] and SIFT features [46], may greatly improve registration accuracy and efficiency. More sophisticated outputs of object part detectors may be also used as the higher-level features. Such features could be leaves if the objects of interest were branches or fingers if a hand shape was to be detected. For each feature, information about its color, texture, or shape could be stored. Such enhancements remain a topic for future investigation.

In the current method, a registration is constrained to be a linearly ordered set of feature-state pairs. However, dynamic programming algorithms can also efficiently produce registrations that

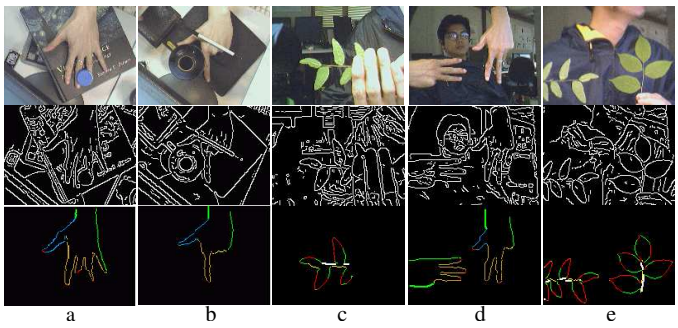

Fig. 11. Preliminary results illustrate the ability of our method to handle partial occlusions and detection of multiple objects. (a–c): Two hands with small occlusions and one partially occluded branch object. (d–e): Two branches and two hands were detected successfully, by finding the two highest scoring registrations for each input image.

are tree-ordered [24], [26]. Such registrations are more appropriate for branching objects like waterways, dendrites, and blood vessels. We are interested in extending our method to handle such cases.

Using our formulation, to find the optimal registration we need to search over multiple scales. While our formulation allows for rotation-invariant models in principle, our implemented models were not rotation invariant. Although these models could cover a broad orientation range of about 45 degrees, it was still necessary to search for the optimal orientation. Eliminating the need to search for optimal scale and orientation are important problems for future work. Another important problem is constructing shape models automatically, so that not only the probability distributions are learned from data, but the model topology itself.

The current method operates in a strictly bottom-up way, and the resulting global registration is simply the result of many local decisions. We expect that pairing our method with top-down mechanisms can significantly reduce false matches. Other future improvements include a method to apply machine learning techniques to identify discriminative features automatically for each state of the HSSM, or how to learn the structure of a HSSM automatically for a given object category.

ACKNOWLEDGMENTS

This research was supported in part through NSF grants IIS 0329009, IIS 0308213, IIS-0093367, EIA 0202067, EIA 0326483, and ONR grant N00014-03-1-0108.

REFERENCES

[1] V. Athitsos, J. Wang, S. Sclaroff, and M. Betke, "Detecting instances of shape classes that exhibit variable structure," in *Proc. European Conf. Computer Vision, Vol. 1*, Graz, Austria, 2006, pp. 121–134.

[2] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[3] M. J. F. Gales and S. J. Young, "The theory of segmental Hidden Markov Models," Engineering Dept., Cambridge Univ., Tech. Rep. CUED/F-INFENG/TR.133, 1993.

[4] M. Ostendorf, V. Digalakis, and O. Kimball, "From HMMs to segment models: a unified view of stochastic modeling for speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 4, no. 5, pp. 360–378, 1996.

[5] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, "Parametric correspondence and chamfer matching: Two new techniques for image matching," in *Int.l Joint Conf. Artificial Intelligence*, Cambridge, MA, 1977, pp. 659–663.

[6] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla, "Shape context and chamfer matching in cluttered scenes," in *Proc. Int.l Conf. Computer Vision Pattern Recognition*, vol. 1, Madison, WI, 2003, pp. 127–133.

[7] T. McInerney and D. Terzopoulos, "Deformable models in medical image analysis: a survey," *Medical Image Analysis*, vol. 1, no. 2, pp. 91–108, 1996.

[8] R. C. Veltkamp and M. Hagedoorn, "State of the art in shape matching," in *Principles of visual information retrieval*, M. S. Lew, Ed. Springer-Verlag, 2001, pp. 87–119.

[9] T. B. Sebastian, P. N. Klein, and B. B. Kimia, "Recognition of shapes by editing shock graphs," in *Proc. Int.l Conf. Computer Vision*, vol. 1, Vancouver, Canada, 2001, pp. 755–762.

[10] S. C. Zhu and A. L. Yuille, "FORMS: a flexible object recognition and modeling system," *Int.l J. Computer Vision*, vol. 20, no. 3, pp. 187–212, 1996.

[11] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int.l J. Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[12] T. McInerney and D. Terzopoulos, "Topologically adaptable snakes," in *Proc. Int.l Conf. Computer Vision*, Cambridge, MA, 1995, pp. 840–845.

[13] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active shape models - their training and application," *Computer Vision Image Understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[14] Y. Chen, H. Tagare, S. Thiruvenkadam, F. Huang, D. Wilson, K. S. Gopinath, R. W. Briggs, and E. A. Geiser, "Using prior shapes in geometric active contours in a variational framework," *Int.l J. Computer Vision*, vol. 50, no. 3, pp. 315–328, 2002.

[15] M. E. Leventon, W. E. L. Grimson, and O. Faugeras, "Statistical shape influence in geodesic active contours," in *Proc. Int.l Conf. Computer Vision Pattern Recognition*, vol. 1, Hilton Head, SC, 2000, pp. 316–323.

[16] D. Nain, A. Yezzi, and G. Turk, "Vessel segmentation using a shape driven flow," in *Proc. Int.l Conf. Medical Image Computing and Computer Assisted Intervention*, Saint-Malo, France, 2004, pp. 51–59.

[17] J. Wang, M. Betke, and J. P. Ko, "Pulmonary fissure segmentation on CT," *Medical Image Analysis*, vol. 10, no. 4, pp. 530–547, Aug. 2006.

[18] J. M. Coughlan and S. J. Ferreira, "Finding deformable shapes using loopy belief propagation," in *Proc. European Conf. Computer Vision*, vol. 3, Copenhagen, Denmark, 2002, pp. 453–468.

[19] L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black, "Attractive people: Assembling loose-limbed models using non-parametric belief propagation," in *Proc. Advance Neural Information Procesing System*, vol. 16, 2003, pp. 1539–1546.

[20] J. Zhang, R. Collins, and Y. Liu, "Representation and matching of articulated shapes," in *Proc. Int.l Conf. Computer Vision Pattern Recognition*, vol. 2, Washington D.C., 2004, pp. 342–349.

[21] A. A. Amini, T. E. Weymouth, and R. C. Jain, "Using dynamic programming for solving variational problems in vision," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 12, no. 9, pp. 855–867, 1990.

[22] J. Coughlan, A. Yuille, C. English, and D. Snow, "Efficient deformable template detection and localization without user initialization," *Computer Vision Image Understanding*, vol. 78, no. 3, pp. 303–319, 2000.

[23] P. F. Felzenszwalb, "Representation and detection of deformable shapes," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 27, no. 2, pp. 208–220, 2005.

[24] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *Int.l J. Computer Vision*, vol. 61, no. 1, pp. 55–79, 2005.

[25] D. Geiger, A. Gupta, L. A. Costa, and J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 17, no. 3, pp. 294–302, 1995.

[26] S. Ioffe and D. A. Forsyth, "Probabilistic methods for finding people," *Int.l J. Computer Vision*, vol. 43, no. 1, pp. 45–68, 2001.

[27] Y. He and A. Kundu, "2-D shape classification using Hidden Markov Model," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 13, no. 11, pp. 1172–1184, 1991.

[28] N. Arica and F. T. Yarman-Vural, "A shape descriptor based on circular Hidden Markov Model," in *Proc. Int.l Conf. Pattern Recognition*, vol. 1, Barcelona, Spain, 2000, pp. 1924–1927.

[29] M. Bicego and V. Murino, "Investigating Hidden Markov Models' capabilities in 2D shape classification," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 26, no. 2, pp. 281–286, 2004.

[30] P. Prusinkiewicz and A. Lindenmayer, *The algorithmic beauty of plants*. New York, NY: Springer-Verlag, 1990.

[31] P. F. Felzenszwalb, "Representation and detection of shapes in images," Ph.D. Thesis, MIT, Cambridge, MA, 2003.

[32] F. Han and S. C. Zhu, "Bottom-up/Top-down image parsing by attribute graph grammar," in *Proc. Int.l Conf. Computer Vision*, vol. 2, Beijing, China, 2005, pp. 1778–1785.

[33] G. Borgefors, "Hierarchical chamfer matching: A parametric edge matching algorithm," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 10, no. 6, pp. 849–865, 1988.

[34] J. Pearl, *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Mateo, California: Morgan Kaufmann, 1988.

[35] H. Sidenbladh and M. J. Black, "Learning image statistics for Bayesian tracking," in *Proc. Int.l Conf. Computer Vision*, vol. 2, Vancouver, Canada, 2001, pp. 709–716.

[36] N. Paragios and R. Deriche, "Geodesic active regions and level set methods for supervised texture segmentation," *Int.l J. Computer Vision*, vol. 46, no. 3, pp. 223–247, 2002.

[37] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 23, no. 4, pp. 349–361, 2001.

[38] P. Viola and M. J. Jones, "Robust real-time face detection," *Int.l J. Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[39] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. Int.l Conf. Computer Vision Pattern Recognition*, vol. 1, San Diego, CA, 2005, pp. 886–893.

[40] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[41] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[42] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[43] S. M. Kay, *Statistical Signal Processing*. Prentice Hall, Englewood Cliffs, 1993.

[44] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *Int.l J. Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[45] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[46] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int.l J. Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.