Boston University Computer Science Tech. Report No. 2005-021, June 13, 2005.
Revised February 17, 2006.

To appear in Proc. of the European Conference on Computer Vision (ECCV), May 2006.

Detecting Instances of Shape Classes That Exhibit Variable Structure

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

[1] Integrated Data Systems Department
Siemens Corporate Research
755 College Road East
Princeton, NJ 08540, USA

[2] Computer Science Department
Boston University
111 Cummington Street
Boston, MA 02215, USA

Abstract. This paper proposes a method for detecting shapes of variable structure in images with clutter. The term "variable structure" means that some shape parts can be repeated an arbitrary number of times, some parts can be optional, and some parts can have several alternative appearances. The particular variation of the shape structure that occurs in a given image is not known a priori. Existing computer vision methods, including deformable model methods, were not designed to detect shapes of variable structure; they may only be used to detect shapes that can be decomposed into a fixed, a priori known, number of parts. The proposed method can handle both variations in shape structure and variations in the appearance of individual shape parts. A new class of shape models is introduced, called Hidden State Shape Models, that can naturally represent shapes of variable structure. A detection algorithm is described that finds instances of such shapes in images with large amounts of clutter by finding globally optimal correspondences between image features and shape models. Experiments with real images demonstrate that our method can localize plant branches that consist of an a priori unknown number of leaves and can detect hands more accurately than a hand detector based on the chamfer distance.

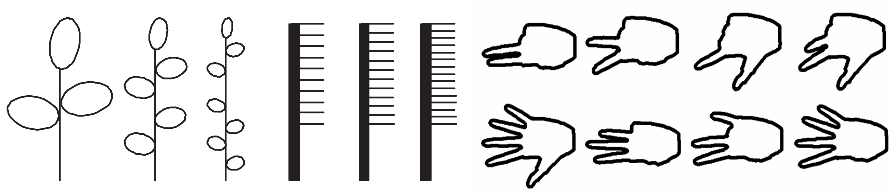

Fig. 1. Three shape classes that exhibit variable structure: branches with leaves, hair combs, and hand contours. Such classes can be naturally modeled with a Hidden State Shape Model (HSSM).

1 Introduction

An important problem in computer vision is detecting shapes in the presence of noise, clutter, and occlusions, and registering such shapes with a model. Ideally we want rich models that can capture a large range of possible object variations, and efficient methods for registering shapes with such models. This paper introduces a detection algorithm that is explicitly designed for a large category of shape classes where existing detection methods are not applicable: shape classes that exhibit variable structure. The term “variable structure” is used to characterize shape classes with the following properties:

- Some shape parts can be repeated an arbitrary number of times, like the teeth in the hair combs of Fig. 1.
- Some shape parts may be missing. For example, in the rightmost branch shown on Fig. 1, one of the leaves on the right side of the branch is missing.
- Some parts can appear in alternative ways. For example, in the hand shapes shown on Fig. 1, a finger can appear totally extended, partially bent, or completely bent. The degree of bending results in different levels of occlusion and thus different 2D hand shapes.

Natural, biological and man-made objects may have variable structures that result in large differences in shape. Blood vessels in the retina, airway ducts in the lung, and dendrites are examples of biological objets with variable structure. Detecting and recognizing such objects is important for tasks like diagnosing diseases of the retina or detecting nodules in the lung. Roadways and waterways in aerial images are also examples of object classes with variable structure.

In order to model shape classes of variable structure, we introduce Hidden State Shape Models (HSSMs), a generalization of Hidden Markov Models (HMMs) [1]. Using HSSMs, shapes can be detected in polynomial time, even in the presence of a significant amount of clutter. We describe an algorithm that performs detection-by-registration, and finds globally optimal correspondences between the HSSM model and image features. In experiments with real images, our method localizes branches of leaves with 79% accuracy, without prior knowledge of the number of leaves, and our method detects and recognizes hand shapes with higher accuracy than a method based on the chamfer distance.

2 Related Work

A large amount of literature in computer vision addresses the issue of detecting deformable shapes in images [2, 3]. Shock graphs [4] and FORMS [5] can be used for fitting 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 are active contours [6] and active shape models [7]. Shape priors can be incorporated into such models, as shown in [8, 9]. However active contours and active shapes cannot be used for automatically detecting deformable shapes in an image, unless a good initial alignment between the model and the image is provided.

Graphical models can be used to detect deformable shapes automatically, without requiring an initial guess [10–12]. When the graphical model is a sequence of parts, or a tree, Dynamic Programming (DP) can be used to find a globally optimal registration between the model and a set of possible shape part locations, even in the presence of clutter [13–18]. A limitation of DP is that it cannot capture cyclical dependencies between shape parts. Graphical models using iterative inference can capture such dependencies, at the cost of not guaranteeing a globally optimal solution [10–12].

The main difference between the method we introduce in this paper and all abovementioned methods is that our method can be used for modeling and detection of shape classes that exhibit *variable* structure. We should stress that "structure variation" is not synonymous with "deformation." Objects can be totally rigid and still exhibit variable structure, like the hair combs in Fig. 1. Deformable model methods [4–18] can model deformations of individual shape parts and deformations in the spatial arrangements between shape parts; they cannot capture structure variations, like the possibility that a shape part may be repeated an arbitrary number of times. Our method, in addition to modeling deformations, is explicitly designed to model variable structure.

Using existing deformable model methods [4–18], the only way one can model a shape class of variable structure is by exhaustively defining one deformable model for each fixed structure that is a legal structure for that shape class. However, such an approach can quickly become computationally intractable. 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. Thus, the number of possible fixed structures is exponential to the number of leaves, and any of the approaches in [4–18] would require exponential time to detect such a shape class. In contrast, our method captures such shape variability with a *single* model, and thus provides polynomial-time detection.

The HSSM models that we introduce in this paper are a generalization of HMMs [1]. HMMs have been used for shape modeling in previous work [19–21]. However, in those methods, HMMs are used to recognize shapes, and object detection is required as preprocessing. Traditional HMMs [19–21] cannot be used for object detection in clutter, even for objects with fixed structure. Our method extends HMMs in a way that overcomes this limitation.

Complex and variable-structure shapes can also be modeled with shape grammars. Lindenmayer systems (L-systems) have been used successfully in computer graphics for generating realistic images of biological shapes [22]. A generic shape grammar is used in [23] for the task of low-level image segmentation and grouping. In [24] a shape grammar is used to improve the accuracy of rectangle detection in images. The main difference between the proposed method and the methods described in [22–24] is that our method, in addition to modeling shape classes of variable structure, also addresses the issue of detecting specific shape classes in cluttered images.

3 Modeling Shapes with HSSMs

First we introduce formal definitions and notation. Then, in Section 3.2, we provide an example of how an HSSM can be used to model a shape. In Section 3.3 we discuss how HSSMs are related to HMMs.

3.1 Terminology and Notation

At a high level, in order to design an HSSM for a specific shape class we need to perform two steps: first, specify a set of states, where each state corresponds to a shape part. Second, specify some cost functions, that can be used to evaluate how well a sequence of image features matches a sequence of states. More formally, an HSSM is defined by specifying the following elements:

1. A set of N states $\mathbb{S} = S_1, \ldots, S_N$.
2. A transition cost function A. $A(S_i, S_j)$ is a non-negative real number that represents the cost of transitioning from state S_i to state S_j.
3. An observation cost function B. $B(S_i, F_k)$ is a non-negative real number that represents the cost corresponding to observing feature F_k at state S_i.
4. A feature transition cost function D. $D(S_i, F_k, S_j, F_l)$ is a non-negative real number that represents the cost associated with consecutively matching feature F_k to state S_i and feature F_l to state S_j. This feature transition cost function is an important difference between an HSSM model and a classical HMM model, as explained in Sec. 3.3.
5. An initial cost function I. $I(S_i)$ is a non-negative real number that represents the cost corresponding to state S_i being the initial state of the shape. If S_i is not a legal initial state, then $I(S_i) = \infty$.
6. A subset $\mathbb{E} \subset \mathbb{S}$ of legal end states for the shape.

Given a test image J, we assume that, using some feature extraction method, a set of K features $\mathbb{F} = \{F_1, \ldots, F_K\}$ has been extracted. For example each F_i can correspond to an edge pixel, and F_i can store the location and orientation of that edge pixel.

A registration between the HSSM and the set $\mathbb{F}$ of image features is denoted as $R_{\mathbb{Q},\mathbb{O}} = ((Q_1, O_1), \ldots, (Q_T, O_T))$, where $\mathbb{Q} = (Q_1, \ldots, Q_T)$ is a sequence of T states (each $Q_i \in \mathbb{S}$), and $\mathbb{O} = (O_1, \ldots, O_T)$ is a sequence of T observations (each $O_i \in \mathbb{F}$). The pair (Q_i, O_i), which represents the i-th step of the registration, consists of the model being in state Q_i (where $Q_i = S_j$ for some j) and the corresponding feature at that step being O_i (where $O_i = F_k$ for some k). Intuitively, a registration specifies which image features correspond to which shape parts.

The cost $C(R_{\mathbb{Q},\mathbb{O}})$ of registration $R_{\mathbb{Q},\mathbb{O}}$ is defined as follows:

$$C(R_{\mathbb{Q},\mathbb{O}}) = I(Q_1) + \sum_{i=1}^{T} B(Q_i, O_i) + \sum_{i=1}^{T-1} A(Q_i, Q_{i+1}) + \sum_{i=1}^{T-1} D(Q_i, O_i, Q_{i+1}, O_{i+1}) \,. \quad (1)$$

We define an operation $\oplus$ that takes a registration $R_{\mathbb{Q},\mathbb{O}} = ((Q_1, O_1), \ldots, (Q_T, O_T))$ and a state-feature pair (Q, O) and returns a new registration that is the result of appending (Q, O) to the end of R:

$$R_{\mathbb{Q},\mathbb{O}} \oplus (Q, O) = ((Q_1, O_1), \ldots, (Q_T, O_T), (Q, O)) \,. \quad (2)$$

We define a registration $R_{\mathbb{Q},\mathbb{O}} = ((Q_1, O_1), \ldots, (Q_T, O_T))$ to be a *total registration* if $Q_T \in \mathbb{E}$, i.e., if the last state of the registration is a legal end state for the HSSM.

Suppose we are given a shape modeled as an HSSM, a registration length $T_{\max}$, and a set $\mathbb{F}$ of features extracted from image J. Detecting the shape in image J consists of finding the globally optimal total registration R_{opt}, i.e., the registration among all possible total registrations $R_{\mathbb{Q},\mathbb{O}}$ with length $T_{\max}$ that minimizes $C(R_{\mathbb{Q},\mathbb{O}})$. Although the set of all possible total registrations is exponential in $T_{\max}$, the algorithm described in Sec. 4 finds a globally optimal total registration in polynomial time, using DP.

3.2 An Example

Consider the class of branch shapes shown in Fig. 1. Fig. 2a displays the state topology of an HSSM model for this class. We actually use this model in the experiments, to detect branches of leaves. In Sec. 5 we quantitatively define the cost functions associated with this model. In the next paragraphs we describe at an intuitive level what we want to capture with the model topology and the cost functions.

In the model, the stem is modeled as a straight line, and the leaves are modeled as hexagons. From the input image we extract oriented edge pixels (Fig. 2b). State S_1 models the stem. We expect stem features to have an upright orientation, and observation cost $B(S_1, F_i)$ penalizes for deviations from that orientation. Similarly, the six states corresponding to each leaf have low observation costs for features whose orientations are similar to the orientations expected to be observed at those states.

The state transition cost $A(S_i, S_j)$ is set to zero for all the legal state transitions shown in Fig. 2a, and to infinity for all other transitions. The initial cost $I(S_1)$ for state S_1 is zero, and the initial cost for all other states is infinity. The feature transition cost function $D(S_i, F_k, S_j, F_l)$ reflects the expectation that, if we match state S_i with feature F_k and then we make the transition from state S_i to state S_j, then the feature F_l matched to state S_j should appear in a position near F_k, and the direction of the vector connecting F_k to F_l should be compatible with the transition from S_i to S_j.

Fig. 2c shows an example registration of the model shown in Fig. 2a with the edge image shown in Fig. 2b. We should stress that the model shown in Fig. 2a is simply one of many possible models for the class of branch shapes shown in Fig. 1. For example,

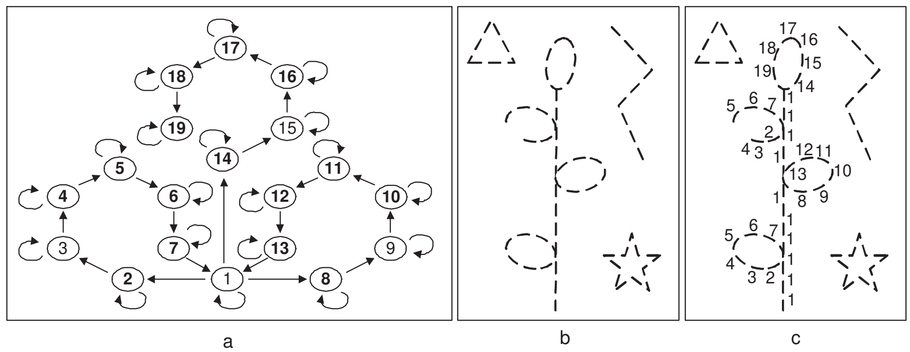

Fig. 2. An HSSM model of the branch class. a). The states of the model, and the allowed transitions out of each state. State S_1 models the stem, states $S_2, \dots, S_7$ model the left-side leaves, states $S_8, \dots, S_{13}$ model the right-side leaves, states $S_{14}, \dots, S_{19}$ model the top leaf. b). An edge image, containing a branch and some "clutter" objects. Each line and arc segment stand for an image feature. c). An example registration of the model with the image features: state labels are shown next to the features they were matched with. Note that the "clutter" features are not assigned to any state.

one could instead design leaf detectors, and model each leaf with a single state. The image features that would be matched to that state would correspond to locations where the detector response exceeds a threshold, and the observation cost of each feature would depend on the detector response at that feature location.

3.3 Relation to HMMs

HSSMs are a superclass of HMMs. An HMM is a special case of an HSSM, in which:

- Feature transition cost function D is set to zero.
- Function $A(S_i, S_j)$ is the negative logarithm of the transition probability of moving from state S_i to state S_j.
- Function $B(S, F)$ is the negative logarithm of the probability of observing feature F while at state S.
- Function $I(S)$ is the negative logarithm of the probability of S being the initial state.

Overall, if functions A, B, D and I are defined to be negative log likelihoods, then the HSSM model becomes probabilistic, and it provides a generative model that describes how to stochastically generate a set of image features given a shape class. At the same time, if the underlying probability distributions are not available, we can easily create HSSMs by constructing cost functions either manually or automatically. In our experiments we found it straightforward and intuitive to define those functions manually, as described in Sec. 5.

HMMs are typically used to recognize temporal sequences of observations. The traditional Viterbi algorithm employed in HMMs [1] optimally assigns a state to each

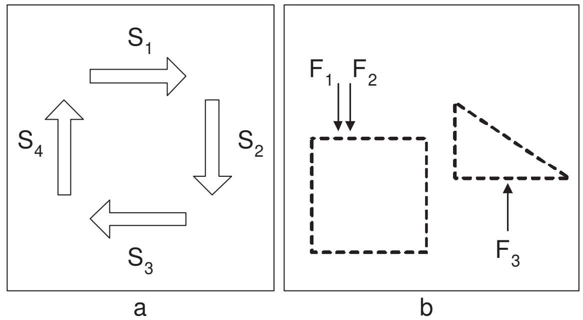

Fig. 3. An illustration of the need for a feature transition cost function. A square is modeled with four states, $S_1, \ldots, S_4$, as shown on the left. Suppose that $B(S_i, F_k)$ compares the edge orientation at F_k with the orientation corresponding to state S_i. Consider features F_1, F_2, F_3, shown on the right. Without a feature transition cost function, registration $((S_1, F_1), (S_1, F_2))$ is as good as registration $((S_1, F_1), (S_1, F_3))$, since F_1, F_2, and F_3 have the same orientation. The feature transition cost function D can penalize the transition from (S_1, F_1) to (S_1, F_3), since F_3 is so far from F_1.

observation, but relies on two key assumptions: first, that the observations are ordered (temporal sequences of observations are naturally ordered based on the time in which they were observed), and second, that each observation should be matched with the model. In our setting, we cannot use the standard Viterbi algorithm because neither of those two assumptions holds. The set $\mathbb{F}$ of features is an *unordered* set of observations, and only a subset of those observations may actually match the model, since many (possibly most) observations will correspond to clutter.

Since our system does not know a priori the order in which features must be registered, we need a feature transition cost function to evaluate different possible orderings. This function models the fact that, given two consecutive states S_i and $S_{i'}$, we may have two features F_k and $F_{k'}$ such that $B(S_i, F_k)$ and $B(S_{i'}, F_{k'})$ are very low, but the features F_k and $F_{k'}$ are located so far from each other or have some other combined property that makes them a really bad choice for consecutively matching S_i and $S_{i'}$. Fig. 3 illustrates an example.

4 Optimal Registration in Clutter

Suppose that we are given an HSSM model, a registration length $T_{\max}$, and a set $\mathbb{F}$ of features extracted from image J. We want to find a globally optimal total registration R_{opt}. In this section we describe how to find R_{opt} in polynomial time, using a modified version of the Viterbi algorithm.

As is typical in DP methods, we solve our problem by breaking it up into many subproblems whose solutions are related to each other. In particular, we define $W(i, j, k)$ to be the registration $R_{\mathbb{Q},\mathbb{O}}$ that achieves the smallest cost $C(R_{\mathbb{Q},\mathbb{O}})$ under the following constraints:

1. The length of $R_{\mathbb{Q},\mathbb{O}}$ is j.
2. $Q_j = S_i$. That is, the last state Q_j of $R_{\mathbb{Q},\mathbb{O}}$ is state S_i.
3. $O_j = F_k$. That is, the last feature O_j of $R_{\mathbb{Q},\mathbb{O}}$ is feature F_k.

If $j = 1$, then $W(i, j, k) = ((S_i, F_k))$. For $j > 1$, assume that we have already computed $W(i', j-1, k')$ for every $i' \in \{1, \dots, N\}$ and $k' \in \{1, \dots, K\}$, where N is the number of states and K is the number of features. Then, $W(i, j, k)$ can be found easily as follows: first, for notational convenience, for every i', k', we define registration $V(i', k', i, j, k)$ as:

$$V(i', k', i, j, k) = W(i', j-1, k') \oplus (S_i, F_k) \ . \tag{3}$$

Now, $W(i, j, k)$ is simply the $V(i', k', i, j, k)$ for which the cost $C(V(i', k', i, j, k))$ is minimized:

$$W(i, j, k) = \mathrm{argmin}_{V(i', k', i, j, k)} C(V(i', k', i, j, k)) \ . \tag{4}$$

Suppose that we have computed $W(i, j, k)$ for every combination of i, j, k. We want to find the globally optimal total registration R_{opt}, i.e., the total registration $R_{\mathbb{Q},\mathbb{O}}$ with the lowest cost $C(R_{\mathbb{Q},\mathbb{O}})$. First we define the set $\mathbb{W}$ of all registrations $W(i, T_{\max}, k)$ that are total, meaning that their last state is a legal end state:

$$\mathbb{W} = \{W(i, T_{\max}, k) | S_i \in \mathbb{E}\} \ . \tag{5}$$

The globally optimal total registration R_{opt} is simply the registration in $\mathbb{W}$ with the lowest cost:

$$R_{\mathrm{opt}} = \mathrm{argmin}_{R_{\mathbb{Q},\mathbb{O}} \in \mathbb{W}} C(R_{\mathbb{Q},\mathbb{O}}) \ . \tag{6}$$

Registration R_{opt} describes the optimal way to align the HSSM with the observed image features. It specifies where the shape is in the image, and also it specifies the actual structure of the shape, and the location of each individual shape part.

4.1 Complexity

In the worst case, to determine $W(i, j, k)$ for a specific combination of i, j, k we need to evaluate KN possible registrations $V(i', k', i, j, k)$, where K is the number of image features and N is the number of model states. Each of these possible registrations can be evaluated in constant time assuming that, for every i, j, k, when we compute $W(i, j, k)$ we save the cost $C(W(i, j, k))$ in an array $U(i, j, k)$. Then,

$$\begin{aligned} C(V(i', k', i, j, k)) = U(i', j-1, k') + A(S_{i'}, S_i) \\ + D(S_{i'}, F_{k'}, S_i, F_k) + B(S_i, F_k) \ . \end{aligned}$$

There are $O(KT_{\max}N)$ possible combinations of i, j, k. Therefore, the worst case cost of computing $W(i, j, k)$ for every combination of i, j, k is $O(K^2 T_{\max} N^2)$ operations. This cost is polynomial to all terms, which is much more efficient than the brute force method of simply evaluating every one of the exponentially many possible registrations between the model and the set of image features.

The complexity can be further reduced if we can impose some additional constraints. Constraints can be imposed in three different ways:

- By restricting the set of allowed state transitions. This restriction significantly reduces the number of registrations $V(i', k', i, j, k)$ that need to be evaluated in order to find $W(i, j, k)$, by requiring that $S_{i'}$ can be legally succeeded by S_i.
- By restricting the set of allowed feature transitions. If such a restriction is available, it can be used so that, when $W(i, j, k)$ is computed, the system only evaluates registrations $V(i', k', i, j, k)$ such that $F_{k'}$ can be legally succeeded by F_k.
- By restricting, for each state, the set of features that can legally be matched to that state. Then, $W(i, j, k)$ is evaluated only if F_k can be legally matched to S_i.

In the HSSM models used in our experiments we implemented two of those restrictions: first, there are at most four legal transitions for every state. Second, we do not allow a transition between any features f_k and f_l if the distance between f_k and f_l exceeds a threshold. With these two restrictions, the time complexity of the registration process is reduced from $O(K^2 T_{\max} N^2)$ to $O(K T_{\max} N)$.

5 Implementation

Given a shape class of variable structure, there are several alternative ways to set up an HSSM model for that class. For example, one can define specific detectors for individual shape parts and use the results of those detectors as features [16, 18]. For the implementation used in our experiments, we opted for a simpler solution, where every feature F is simply the location of an edge pixel. We denote with $L(F)$ the location of F, and with $\theta(F)$ the edge orientation of F, where the range of $\theta(F)$ is $[0, 2\pi)$.

Each state S simply models a line segment with orientation $\theta(S)$. To determine how well a feature F matches state S, we simply measure the difference between their orientations. We will denote with $\varDelta(\theta_1, \theta_2)$ the angle between orientations θ_1 and θ_2. The range of $\varDelta(\theta_1, \theta_2)$ is limited to $[0, \frac{\pi}{2}]$. Based on this notation, we define the observation cost function B between state S and feature F as follows:

$$B(S, F) = \varDelta(\theta(S), \theta(F)) \tag{7}$$

In all the models used for the experiments we set the transition cost function A to zero for state transitions that we define as legal, and to infinity for state transitions that we define as illegal. Every state is allowed to make a transition to itself. The observation transition cost function $D(S_i, F_k, S_j, F_l)$ depends on the difference in position and orientation between F_k and F_l. More formally, we denote by $V(\theta)$ the two-dimensional unit vector with orientation θ. Given a weight α that balances position and orientation information, the observation transition cost function $D(S_i, F_k, S_j, F_l)$ is defined as:

$$\begin{aligned} D(S_i, F_k, S_j, F_l) = {} & \left\| \frac{L(F_l) - L(F_k)}{\|L(F_l) - L(F_k)\|} - V(\theta(S_j)) \right\| + \\ & \alpha |\varDelta(\theta(S_i), \theta(S_j)) - \varDelta(\theta(F_k), \theta(F_l))| \,. \end{aligned} \tag{8}$$

Note that these definitions make the resulting HSSM models invariant to translation, since we do not use absolute feature location in any of the cost functions; we only use, in function D, relative feature location with respect to the location of the previous feature. The HSSM models used in the experiments are dependent on scale and orientation. We obtain the optimal value for α using a validation set, disjoint from the set of test images.

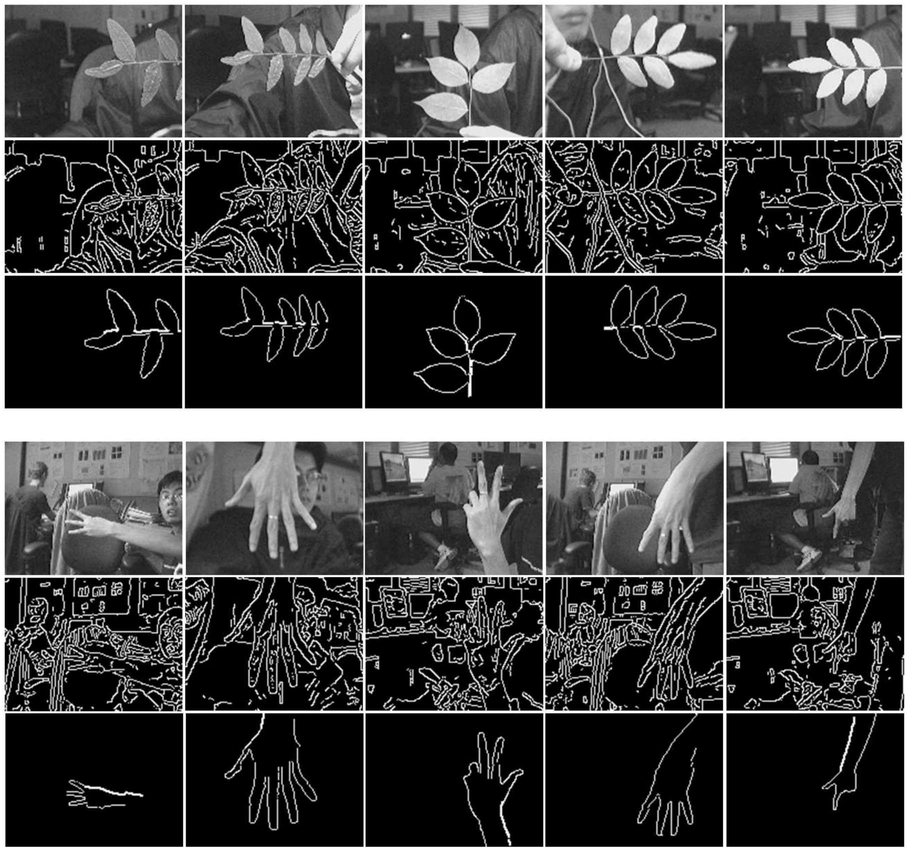

Fig. 4. Examples of "correct recognition" on images of branches of leaves (top half) and hand images (bottom half). For each test image, we show the actual image, the corresponding edge image, and the edge pixels registered to the HSSM model.

6 Experiments

We have evaluated our method on the task of object localization in two datasets of real images containing shapes of variable structure. The first dataset consists of 100 images of branches of leaves, and the second dataset consists of 353 hand images (Figs. 4, 5, 6). The task of object localization 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 successfully locate the object and identify the orientation and shape of the object.

In order to provide quantitative measures of accuracy, we will use the following terms to describe accuracy on a particular image:

– "Correct recognition": the system has found the shape at the correct location and orientation, has correctly estimated the number of shape parts, and has correctly registered each shape part.

– "Correct localization": the system has identified the correct object location and orientation. In particular, for the branches we require that 75% of the stem be registered correctly, and for hand images we require that the 75% of the palm edges be registered correctly. We allow incorrect estimation of the number and/or location of some shape parts, and incorrect registration of some shape parts.
– "Incorrect localization": the method failed to find the correct object location and orientation.

Figs. 4, 5, 6 illustrate the meaning of each of these terms with example images.

Exhaustive search was used to identify the orientation that gave the best registration score. For each image, eight different orientations were applied, sampled uniformly in the range from 0 to 2π. With respect to the scale of the object, we assume that $T_{\max}$ is known. The values used for $T_{\max}$ were from the set $\{200, 250, 300, 350, 400, 450, 500\}$.

The test images were 120×160 pixels. All images were converted to grayscale, no color information was available to the algorithm. Edges were extracted using a Canny edge detector. There were between 2000 and 4000 edge pixels extracted from each image. In the HSSMs used for these experiments we did not allow transitions between features that were more than five pixels away. It took about 5-6 minutes to process each image (including trying all eight orientations), with a C++ implementation, on an Opteron 2.0GHz processor. The memory size of the program was under 400MB.

6.1 Experiments on Branch Localization

We constructed an HSSM model for branches of leaves, where leaves occur at the left and right side of the stem (Fig. 2). We then applied the registration algorithm on 100 real images of branches. The intention of this experiment was to illustrate that our method extracts useful information from heavily cluttered edge images, and can be a useful complement to other sources of information, like color, motion, and background modeling.

Figs. 4, 5, 6 show example results of our method, and Table 1 provides a quantitative evaluation. In 79% of the images our method produced correct localization. Registration was correct in 43% of the images. We find these results promising, given that we only used edge information. Incorporating color information and more descriptive features, like shape context [25] and SIFT features [26], should greatly improve registration accuracy. Such enhancements remain a topic for future investigation.

6.2 Experiments on Hand Localization

We have also applied our method to the challenging problem of localizing hands in grayscale images using only edge information. We compared the detection and recognition accuracy of our method to results obtained using both the chamfer distance [27], and the modified chamfer distance (denoted here as chamfer distance + orientations) that takes edge orientations into account and was used in [28] for hand localization.

The class of hand contours that we modeled in this experiment is defined as follows: the back of the palm is visible, the camera viewing direction is perpendicular to the palm surface, and each of the five fingers can be either extended or hidden. Since a hand has

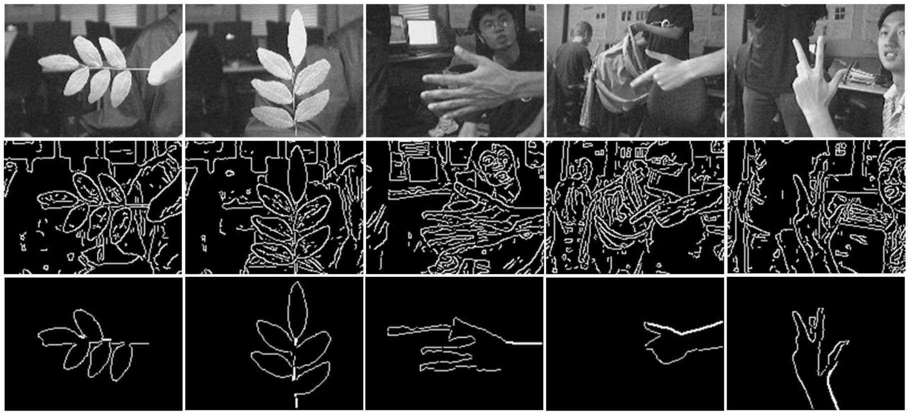

Fig. 5. Example images of branches and hands where the HSSM had "correct localization" but not "correct recognition." For each test image, we show the actual image, the corresponding edge image, and the edge pixels registered to the HSSM model.

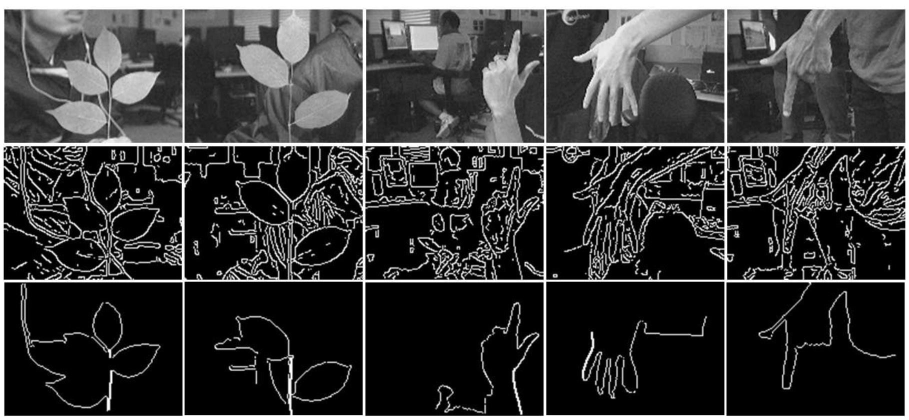

Fig. 6. Example images of branches and hands where the result was labeled as "incorrect". For each test image, we show the actual image, the corresponding edge image, and the edge pixels registered to the HSSM model.

five fingers, and each finger can be extended or hidden, we need 32 fixed-structure models to represent all valid fixed structures. Accordingly, we used 32 models for the chamfer distance. In contrast, a single HSSM was sufficient for modeling the entire range of variations.

We tested our method on 353 real images of hands, from seven different subjects. The images contained a significant amount of clutter. Figs. 4, 5, 6 show example results, and Table 1 quantitatively compares our method to the chamfer distance. For detection and recognition based on the chamfer distance, "correct localization" means that best

dataset:	branches	hands		
method:	HSSM	HSSM	chamfer distance + orientations	chamfer distance
number of orientations:	8	8	72	72
correct recognition	43.0%	33.7%	21.8%	4.0%
correct localization	79.0%	59.5%	54.6%	35.2%
incorrect localization	21.0%	40.5%	45.4%	64.8%

Table 1. Results of HSSM on images of branches and hands, as measured on 100 images of branches of leaves and 353 hand images. For hand images, we also show results using two version of the chamfer distance. Note that "correct recognition" is a subcase of "correct localization." Under each method we indicate the number of orientations at which the method was applied.

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 recognition" means that, in addition to obtaining correct localization, the best response was obtained by the correct fixed-structure model.

Since the chamfer distance is not tolerant to large image-plane rotations, we evaluated it on 72 orientations of each test image. To ensure a fair comparison to our method, the scale of the hand was available to the chamfer distance, so there was no need to search over a range of scales. The search was over all possible pixel locations, 72 orientations, and all 32 models. Hand localization using the chamfer distance took about 15 seconds/image.

As seen in Table 1, our method was more accurate than the results obtained using either variant of the chamfer distance, in terms of both correct localization and correct recognition. At the same time, we consider the accuracy reported here as the "lower bound" on hand pose matching accuracy with our approach, since color features, motion, etc. could be added to further improve localization and recognition rates [28–30]. We deliberately did not include these additional features, so that edge-based matching performance vs. the chamfer distance could be directly tested and compared.

7 Discussion and Future Work

We have described a novel method for detecting shapes of variable structure in cluttered images, using the proposed HSSM models. A globally optimal registration can be found in polynomial time, using Dynamic Programming. The HSSM models used in our experiments can be registered with a cluttered image using only easy-to-extract, low level features like edge pixel locations and orientations.

So far we have evaluated our method in a localization setting, where the system knows that there is exactly one object of interest, and the system tries to find the best registration hypothesis for that object. However, our method can also be applied in a more classical detection setting, where the system does not know a priori if there are zero, one, or multiple instances of an object. Fig. 7 shows some preliminary results

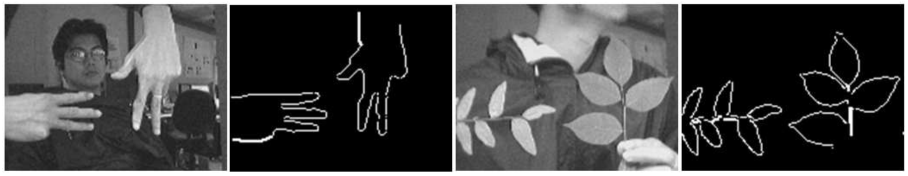

Fig. 7. Preliminary results illustrating the ability of our method to detect multiple objects in the same image. Two branches and two hands are detected successfully, by using, for each input image, the two highest scoring registrations found by the proposed registration algorithm.

for multiple instance detection. Those results correspond to the two highest scoring registrations found using the proposed registration algorithm.

In this paper, a registration is constrained to be a linearly ordered set of feature-state pairs. However, dynamic programming algorithms can also efficiently produce registrations that are tree-ordered [16, 18]. Such registrations are more appropriate for branching shapes like waterways, dendrites, and blood vessels. We are interested in extending our method to handle such cases.

It is interesting to note that our 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. We also believe that the accuracy of the method can be greatly improved by applying machine learning methods to optimize the cost functions, and to identify the most discriminative features for each state of the HSSM model. We are currently working on incorporating such methods into our framework.

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. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. In: Proc. of the IEEE. Volume 77:2. (1989) 257–286
2. McInerney, T., Terzopoulos, D.: Deformable models in medical image analysis: a survey. Medical Image Analysis **1** (1996) 91–108
3. Veltkamp, R.C., Hagedoorn, M.: State of the art in shape matching. In Lew, M.S., ed.: Principles of visual information retrieval. Springer-Verlag (2001) 87–119
4. Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of shapes by editing shock graphs. In: ICCV. (2001) 755–762
5. Zhu, S.C., Yuille, A.L.: FORMS: a flexible object recognition and modeling system. IJCV **20** (1996) 187–212
6. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. IJCV **1** (1988) 321–331

7. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models - their training and application. CVIU **61** (1995) 38–59
8. Chen, Y., Tagare, H., Thiruvenkadam, S., Huang, F., Wilson, D., Gopinath, K.S., Briggs, R.W., Geiser, E.A.: Using prior shapes in geometric active contours in a variational framework. IJCV **50** (2002) 315–328
9. Leventon, M.E., Grimson, W.E.L., Faugeras, O.: Statistical shape influence in geodesic active contours. In: CVPR. (2000) 316–323
10. Coughlan, J.M., Ferreira, S.J.: Finding deformable shapes using loopy belief propagation. In: ECCV. Volume 3. (2002) 453–468
11. Sigal, L., Isard, M., Sigelman, B.H., Black, M.J.: Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In: NIPS. (2003)
12. Zhang, J., Collins, R., Liu, Y.: Representation and matching of articulated shapes. In: CVPR. Volume 2. (2004) 342–349
13. Amini, A.A., Weymouth, T.E., Jain, R.C.: Using dynamic programming for solving variational problems in vision. PAMI **12** (1990) 855–867
14. Coughlan, J., Yuille, A., English, C., Snow, D.: Efficient deformable template detection and localization without user initialization. CVIU **78** (2000) 303–319
15. Felzenszwalb, P.F.: Representation and detection of deformable shapes. PAMI **27** (2005) 208–220
16. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV **61** (2005) 55–79
17. Geiger, D., Gupta, A., Costa, L.A., Vlontzos, J.: Dynamic programming for detecting, tracking, and matching deformable contours. PAMI **17** (1995) 294–302
18. Ioffe, S., Forsyth, D.A.: Probabilistic methods for finding people. IJCV **43** (2001) 45–68
19. He, Y., Kundu, A.: 2-D shape classification using Hidden Markov Model. PAMI **13** (1991) 1172–1184
20. Arica, N., Yarman-Vural, F.T.: A shape descriptor based on circular Hidden Markov Model. In: ICPR. (2000) 1924–1927
21. Bicego, M., Murino, V.: Investigating Hidden Markov Models' capabilities in 2D shape classification. PAMI **26** (2004) 281–286
22. Prusinkiewicz, P., Lindenmayer, A.: The algorithmic beauty of plants. Springer-Verlag New York, Inc., New York, NY, USA (1990)
23. Felzenszwalb, P.F.: Representation and Detection of Shapes in Images. PhD thesis, MIT (2003)
24. Han, F., Zhu, S.C.: Bottom-up/top-down image parsing by attribute graph grammar. In: ICCV. (2005) 1778–1785
25. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. PAMI **24** (2002) 509–522
26. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
27. Barrow, H.G., Tenenbaum, J.M., Bolles, R.C., Wolf, H.C.: Parametric correspondence and chamfer matching: Two new techniques for image matching. In: IJCAI. (1977) 659–663
28. Thayananthan, A., Stenger, B., Torr, P.H.S., Cipolla, R.: Shape context and chamfer matching in cluttered scenes. In: CVPR. (2003) 127–133
29. Martin, J., Devin, V., Crowley, J.: Active hand tracking. In: Face and Gesture Recognition. (1998) 573–578
30. Ong, E., Bowden, R.: A boosted classifier tree for hand shape detection. In: Face and Gesture Recognition. (2004) 889–894