Spatial Random Tree Grammars for Modeling Hierarchical Structure in Images with Regions of Arbitrary Shape

Jeffrey M. Siskind, *Senior Member*, *IEEE*, James J. Sherman Jr., *Student Member*, *IEEE*,
Ilya Pollak, *Senior Member*, *IEEE*, Mary P. Harper, *Senior Member*, *IEEE*, and
Charles A. Bouman, *Fellow*, *IEEE*

Abstract—We present a novel probabilistic model for the hierarchical structure of an image and its regions. We call this model *spatial random tree grammars* (SRTGs). We develop algorithms for the exact computation of likelihood and maximum a posteriori (MAP) estimates and the exact expectation-maximization (EM) updates for model-parameter estimation. We collectively call these algorithms the *center-surround algorithm*. We use the center-surround algorithm to automatically estimate the maximum likelihood (ML) parameters of SRTGs and classify images based on their likelihood and based on the MAP estimate of the associated hierarchical structure. We apply our method to the task of classifying natural images and demonstrate that the addition of hierarchical structure significantly improves upon the performance of a baseline model that lacks such structure.

Index Terms—Bayesian methods for image understanding, multiscale analysis.

1 Introduction

Hierarchical organization can be very valuable in image analysis and classification. Recursive bipartitioning [14], [49], [53] and region merging [33], [37], using features such as intensity gradient, region shape, region color, and region texture, can yield the hierarchy of segmentations of an image. However, these methods are feature data driven and, generally, are not based on an overall explicit model of the hierarchical structure in the image. An alternative to these approaches is the use of hierarchical Bayesian statistical models of image structure.

Researchers in the fields of speech and computational linguistics have long exploited the value of structural priors in language processing. Hidden Markov models (HMMs) are a nonhierarchical prior that has had an enormous impact in speech processing [43]. Context-free grammars (CFGs) have been used to model hierarchical sentence structure [10], [11], [12]. Probabilistic CFGs (PCFGs) [47] have been used to construct probabilistic hidden tree models for sentences and to robustly parse naturally occurring text [31], [36]. An important property of these tree models is that the structure of the tree is random and can adapt to the hidden structure of the observations. As with most priors, HMMs and PCFGs require the estimation of numerous parameters. This can be done using the expectation-maximization (EM) algorithm [16] or, equivalently, the Baum-Welch reestimation procedure [2]. In both cases, the E and M steps can be computed exactly. However, in the case of HMMs, the E step is computed using the forward-backward algorithm [43], whereas in the case of PCFGs, it is computed using the more complex inside-outside algorithm [1], [26]. Variants of both algorithms can be used to compute the most probable state sequence, in the case of HMMs, or the most probable parse, in the case of PCFGs [52].

Structural priors are also valuable for the analysis and processing of images and complex scenes. For example, HMMs have been used for document image recognition [22] and layout analysis [20], and PCFGs have been applied to the event-recognition task by processing a one-dimensional (1D) aspect of a video stream [5]. However, the generalization of 1D priors to 2D has often proved to be very difficult. Markov random fields (MRFs) [3] are a nonhierarchical prior that is the most common extension of HMMs to 2D. For example, an MRF model of image regions was used in [32] for image interpretation. Although effective methods have been developed to approximately compute the maximum a posteriori (MAP) estimates [18] and maximum likelihood (ML) parameter estimates [4], [41], the exact computation of these estimates for MRFs is fundamentally intractable. HMMs have also been generalized to tree structures used for probabilistic reasoning [35] and multiscale image modeling [6], [8], [9], [15], [29], [28], [30], [45]. However, an important distinction between these models and PCFGs is that, in the former case, the tree structure is fixed and nonrandom.

Conditional random fields (CRFs) [25] model the posterior distribution directly without imposing a prior. Although recent adaptations of CRFs to image analysis [19], [24], [50] offer interesting and promising alternatives to MRFs, they still result in intractable estimation problems that can only be solved approximately.

- *J.M. Siskind, I. Pollak, M.P. Harper, and C.A. Bouman are with the School of Electrical and Computer Engineering, Purdue University, 465 Northwestern Ave., West Lafayette, IN 47907-2035.*
  E-mail: {qobi, harper}@purdue.edu, {ipollak, bouman}@ecn.purdue.edu.
- *J.J. Sherman Jr. is with the Department of Electrical and Computer Engineering, University of Maryland, 4444 AV Williams Building, College Park, MD 20742. E-mail: shermanj@umd.edu.*

Manuscript received 22 Sept. 2005; revised 11 July 2006; accepted 2 Nov. 2006; published online 17 Jan. 2007.
Recommended for acceptance by J. Goutsias.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0518-0905.
Digital Object Identifier no. 10.1109/TPAMI.2007.1169.

0162-8828/07/$25.00 © 2007 IEEE Published by the IEEE Computer Society

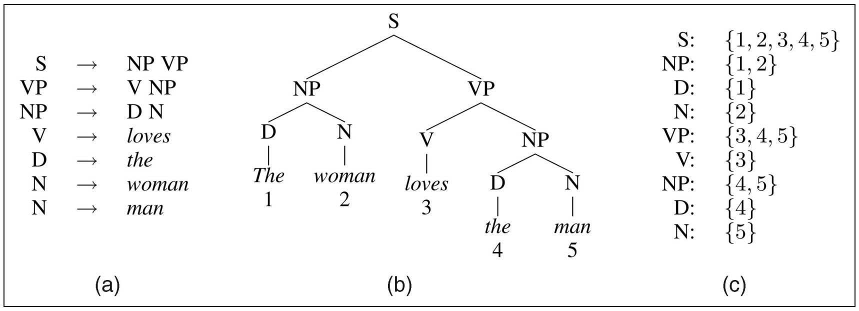


Fig. 1. The CFG (a) imposes the parse (b) on the sentence *The woman loves the man*. Note that the parse tree unambiguously specifies an ordering of the words from left to right, which leads to (c) the unique constituency function.

Various strategies have been used to apply grammar formalisms to images [17], [46], [48]. For example, Fu's research in syntactic pattern recognition incorporated the use of PCFGs for the classification of 2D structures [17]. Deterministic CFGs called *X-Y trees* and associated algorithms were constructed in [23], [34] to model the 2D layout of document images. Probabilistic grammars have more recently been applied to optical character recognition [42].

In this paper, we present a novel probabilistic model for the hierarchical structure of an image and its constituent components. Specifically, we present a novel method for reformulating PCFGs to model 2D images and other multidimensional data. We call this method *spatial random tree grammars* (SRTGs) and the associated algorithms the *center-surround algorithm*. The center-surround algorithm plays an analogous role to the inside-outside algorithm in that it allows for the exact computation of likelihood and MAP estimates and the exact EM updates for model-parameter estimation. We use SRTGs to formulate priors on image-region hierarchies, and we use the associated center-surround algorithm to automatically estimate the ML parameters of such priors, compute the MAP estimate of the associated parse tree, and classify images based on their likelihood. We extend prior work [38], [39], [40], [54] in several important directions. The algorithms in [38], [39], [40], [54] are based on constructing probabilistic models for pixel intensities and grouping image pixels into nonoverlapping rectangular regions. The models and algorithms proposed in this paper are more flexible in that they are able to handle feature images where the features are extracted from possibly overlapping regions of arbitrary shapes. This results in both more modeling choices and improved computational efficiency. In addition, there are important theoretical distinctions between the construction in [54] and the framework introduced in this paper. Whereas the model in [54] hardwires image regions to grammar symbols, we introduce a more flexible framework that is more similar to the traditional 1D PCFG models. In our current framework, the problem of uniquely associating a tree with an image is not straightforward. To solve this problem, we introduce in Section 3 a number of theoretical tools that ensure that each tree has at most one associated image, much like in the standard 1D PCFGs. In addition, the methods in both [54] and the present paper extend PCFGs to continuous observations, much like the work in [27] extended HMMs.

In order to show the potential value of our methods, we apply them to the special case of classification of natural images and demonstrate a significant increase in the average classification accuracy over baseline nonhierarchical models. We also show that substantial information is contained in the MAP image parses extracted by our methods.

The remainder of this paper is organized as follows: Section 2 presents a nontechnical overview of our method. Section 3 presents the technical details of our method. Section 4 presents experiments that illustrate our implementation of classifying the images. Section 6 concludes with a summary of the contributions of this paper.

2 Overview

CFGs can be used to impose a hierarchical structure on sentences, that is, 1D word strings. For example, the CFG in Fig. 1a imposes the parse in Fig. 1b on the sentence *The woman loves the man*. Nodes of such a parse tree correspond to *constituents*, where a constituent is a contiguous substring. Internal nodes might correspond to substrings that contain multiple words, whereas leaf nodes correspond to singleton constituents consisting of a single word.

For this 1D problem, the parse tree unambiguously specifies an ordering of the words from left to right. This implies that there is a unique mapping from the nodes of the parse tree to the constituents of the sentence. We call this mapping the *constituency function*. Moreover, the unambiguous ordering of the nodes implies that each parse tree produces a unique sentence. This property allows a probability distribution over parse trees to induce a probability distribution over strings. We elaborate on this in Section 3. When a constituency function that maps the nodes of the parse tree to parts of a sentence (or, later, an image) exists, we say that the parse tree is a *parse* of the sentence (respectively, image).

The fundamental difficulty in extending CFGs to 2D lies in the unambiguous specification of the constituency functions so that each parse tree produces a unique image. To see that this property does not automatically hold in 2D, consider the example in Fig. 2. Fig. 2b illustrates a parse tree generated by the CFG in Fig. 2a. Figs. 2c and 2e illustrate two different possible images O and O' parsed by this parse tree, with the corresponding constituency functions illustrated in Figs. 2d and 2f, respectively. Both images O and O' have the same set of regions: 1, 2, and 3. Note that the regions in the two images are not assumed to form a regular 2D lattice.

In this paper, we introduce a new class of grammars called *spatial tree grammars* (STGs) that ensures that no two distinct images have the same parse. STGs augment each branching production in a CFG with a *production class*. An STG generates parse trees whose branching nodes are *tagged* with production classes. The tags constrain the allowed relationships

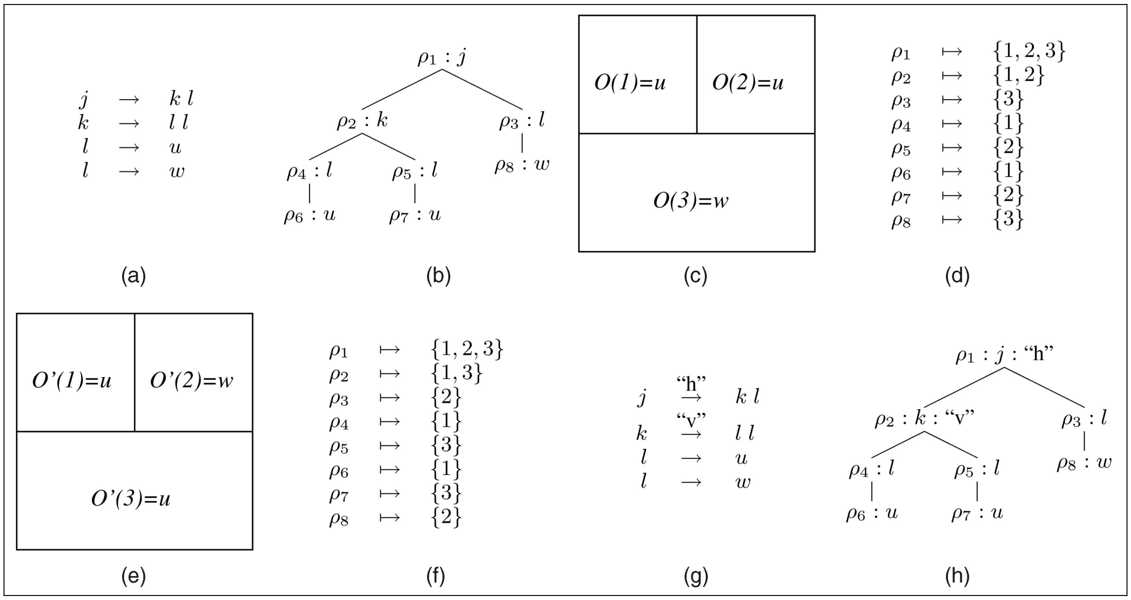


Fig. 2. (a) A CFG. (b) A parse tree generated by (a). (c) An image O with regions 1, 2, and 3 and region intensities $O(1) = u$, $O(2) = u$, and $O(3) = w$. (d) A constituency function that maps the nodes of (b) to the constituents of (c). (e) Another image O' with the same regions 1, 2, and 3 as image O but with different intensities: $O'(1) = u$, $O'(2) = w$, and $O'(3) = u$. (f) A constituency function that maps the nodes in (b) to the constituents of (e). (g) An STG derived from (a) by augmenting the branching productions with production classes. (h) A parse tree generated by (g). This parse tree is the same as (b) except that the branching nodes have been tagged with production classes.

between the constituents that are associated with parse-tree nodes and those that are associated with their children. The list of all valid partitions of a constituent tagged with a specific production class into two subconstituents is specified by a data structure called a *constituent hierarchy*. We formally define this data structure in Definition 1 of Section 3.3.2. Figs. 2g and 2h illustrate how production classes are used. In this example, we use two production classes: "h" (horizontal split) and "v" (vertical split). Intuitively, production class "h" constrains a parse-tree node to be associated with a constituent that is split into upper and lower subconstituents that are associated with the node's children, whereas production class "v" constrains a parse-tree node to be associated with a constituent that is split into left and right subconstituents that are associated with the node's children. Fig. 2g illustrates an STG derived from the CFG in Fig. 2a by augmenting the branching productions with these two production classes. Fig. 2h illustrates a parse tree generated by this STG. This new parse tree is the same as that in Fig. 2b, except that the branching nodes have been tagged with production classes. However, this parse tree does not parse the image in Fig. 2e. Fig. 2f is no longer a valid constituency function because the constituent $\{1, 3\}$ does not lie above the constituent $\{2\}$, as is required by the tag "h" on the parse-tree node ρ_1. It can be shown that, in fact, Fig. 2c is the only image parsed by Fig. 2h.

Note that, whereas the example in Fig. 2 contains only rectangular constituents and partitions image constituents only horizontally and vertically, our current method, described in Section 3, supports nonrectangular constituents and nonhorizontal, nonvertical constituent partitionings as well. In Section 3.4, we provide specific conditions that ensure that no two distinct images have the same parse, and in Section 4, we demonstrate a useful application of this more general case.

In the case of CFGs, one can interpret a branching production like S → NP VP as saying that one can construct an S by concatenating an NP with a VP. CFGs are called such because they are *context free*. One can concatenate *any* NP with *any* VP to construct an S. It has been shown that context-free languages can be parsed in polynomial time by using the Cocke-Younger-Kasami (CKY) algorithm [21], [56].

In the special case of STGs with rectangular constituents considered in this overview, one can interpret a branching production like $\mathrm{U} \overset{\text{"h"}}{\rightarrow} \mathrm{V\ Z}$ as concatenating matrices vertically and a branching production like $\mathrm{V} \overset{\text{"v"}}{\rightarrow} \mathrm{X\ Y}$ as concatenating matrices horizontally. However, unlike the case of CFGs, where it is possible to concatenate any two strings, it is only possible to vertically concatenate matrices of the same width and horizontally concatenate matrices of the same height. Analogous restrictions hold for the more general cases of arbitrary constituents and constituent partitionings that we consider in Section 3. This means that STGs are *context sensitive*. However, it can be easily shown that STGs are a special class of context-sensitive grammars; that is, that there exist context-sensitive languages that cannot be generated by an STG. Moreover, it is shown below that the parsing algorithm for STGs is polynomial in the number of constituents. One way to ensure that our methods use algorithms with polynomial complexity is therefore to limit the number of possible constituents to be polynomial in the size of the input. This can be done using the methods discussed in Section 4.1.

One may induce a probability distribution over the parse trees generated by a CFG by imposing probabilities on the productions. Such an augmented CFG is called a *PCFG*. In a similar fashion, one may induce a probability distribution over the parse trees generated by an STG by imposing probabilities on the productions. We call such an augmented STG an *SRTG*.

3 The Technical Details of Our Method

In this section, we formally introduce the concepts of *feature images* and *parse trees* and develop a framework for

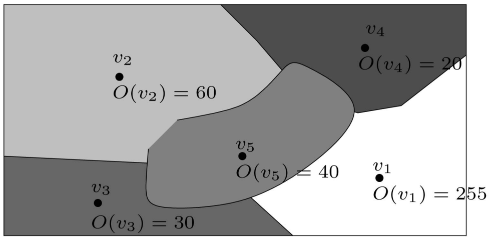


Fig. 3. Illustration of a feature image defined on domain $D = \{v_1, v_2, v_3, v_4, v_5\}$ of size $|D| = 5$ for feature dimension $q = 1$. Here, the locations $v_1, \cdots, v_5$ are the centroids of five constant-intensity regions of an image, and the feature $O(v_n)$ is the intensity at v_n.

associating parse trees with a feature image, given its spatial organization. We construct a probability distribution over parse trees and show that our framework allows this probability distribution to induce a probability distribution on the feature images given their spatial organization.

3.1 Feature Images

We start with an image or any other multidimensional data. From this data, we determine a set of *locations* $D = \{v_1, \cdots, v_{|D|}\}$, and at each of these locations v_n, we extract a *feature vector* $O(v_n) \in \mathbb{R}^q$. We define the *feature image* $\mathbf{O} \in \mathbb{R}^{q|D|}$ to be the vector consisting of all these feature vectors $\mathbf{O} \triangleq (O(v_1), \cdots, O(v_{|D|}))$.

For example, in our experiments reported in Section 4, each location corresponds to an image region obtained through a segmentation algorithm and the feature vector $O(v_n)$ for each location v_n is formed by extracting information from the corresponding region. An illustration of this procedure, for a feature image with five regions ($|D| = 5$) and one feature dimension ($q = 1$), is shown in Fig. 3. The preprocessing segmentation step has the advantage of reducing the number of nodes to be processed and, therefore, the total computation. However, it also imposes practical limits on how much local information can be passed to the SRTG model.

3.2 Parse Trees

Let $\mathcal{I}$ be a finite set of *production classes*, $\mathcal{J}$ be a finite set of *nonterminals*, and $\mathbb{R}^q$ be the set of *terminals*. We use i to denote production classes, u to denote terminals, and j, k, and l to denote nonterminals. We define a *parse tree* T to be a finite ordered tree, where

- leaf nodes (*terminal nodes*) are *labeled* with terminals,
- nonleaf nodes (*nonterminal nodes*) are *labeled* with nonterminals,
- nonterminal nodes have either one child (*nonbranching nonterminal nodes*) or two children (*branching nonterminal nodes*),
- branching nonterminal nodes are *tagged* with production classes,
- the child of every nonbranching nonterminal node is a terminal node, and
- the children of every branching nonterminal node are nonterminal nodes.

Fig. 2h shows an example of a parse tree with $\mathcal{I} = \{''\text{h}'', ''\text{v}''\}$, $\mathcal{J} = \{j, k, l\}$, branching nonterminal nodes ρ_1, ρ_2, nonbranching nonterminal nodes ρ_3, ρ_4, and ρ_5, and terminal nodes ρ_6, ρ_7, and ρ_8.

3.3 Constituents, Constituent Hierarchies, and Constituency Functions for SRTGs

3.3.1 Motivation

With PCFGs, one constructs a probability distribution over all trees generated by the PCFG and defines the probability[1] of each string to be the sum of probabilities of all its parses. This method of constructing probabilities is only correct if any two distinct strings have disjoint sets of parses. This is ensured by the fact that, for PCFGs, two distinct strings cannot have the same parse. The reason that two distinct strings cannot have the same parse is because the unique string corresponding to any parse tree is simply obtained by ordering the leaves of the tree from left to right.

However, a potential problem with SRTGs is that, if improperly designed, a single parse tree can parse two distinct images. This problem can result from the fact that there is no unique natural ordering of the elements of a domain D. This can make it impossible for a probability distribution over trees to induce a probability distribution over feature images in a manner similar to PCFGs. This difficulty is avoided by imposing an organization on the elements of D. Specifically, we introduce the novel concepts of *constituents* of D and *constituent hierarchies* that impose structure in ways that locations in a domain D can be combined. Section 3.3.2 introduces precise definitions of these concepts that ensure that a probability distribution on the feature images can be properly defined and moreover allow for computationally tractable estimation algorithms.

3.3.2 Constituents and Constituent Hierarchies

We designate some subsets of a domain D to be *constituents* and use $\mathcal{C}$ to denote the set of all constituents of D. As we stated earlier, not all subsets $V \subset D$ will be constituents, so $\mathcal{C}$ will generally be a proper subset of 2^D, the power set of D. Furthermore, for each production class $i \in \mathcal{I}$ and each constituent $V \in \mathcal{C}$, we will specify all valid two-way partitions of V into its subconstituents under production class i. To do this, we define the set $\mathcal{L}(i, V)$ of all *left i-subconstituents* of V, with the interpretation that a partition of V into V' and $V \setminus V'$ under the production class i is only valid if $V' \in \mathcal{L}(i, V)$. We call the quadruple $\mathcal{H} = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ a *constituent hierarchy*. We now formally define the concepts of constituents and constituent hierarchy.

Definition 1. *Let D be a finite domain and $\mathcal{I}$ a finite set of production classes. Let $\mathcal{C}$ be a set of subsets of D, and let $\mathcal{L}$ be a function from $\mathcal{I} \times \mathcal{C}$ to $2^{\mathcal{C}}$. The quadruple $\mathcal{H} \triangleq \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ is called a* constituent hierarchy *if it satisfies the four Conditions C1, C2, C3, andC4 given below. In this case, a subset $V \subset D$ is called a* constituent *of D iff $V \in \mathcal{C}$. A set V' is called a* subconstituent *of V if $V' \in \mathcal{L}(i, V)$ (in this case, V' is called a* left i-subconstituent *of V), or if $V \setminus V' \in \mathcal{L}(i, V)$ (in this case, V' is called a* right i-subconstituent *of V).*

C1. *The entire domain D is a constituent: $D \in \mathcal{C}$.*

C2. *Every subconstituent of every constituent V is a proper nonempty subset of V: $(\forall i \in \mathcal{I}, V \in \mathcal{C}, V' \in \mathcal{L}(i, V)) \Rightarrow (V' \subset V \text{ and } V' \neq \emptyset \text{ and } \mathrm{V}' \neq \mathrm{V})$.*

[1] If the terminals are continuously valued, as in Section 3.4, then the probability *density* of each string is the sum of probability densities of all its parses.

C3. *Every nonsingleton constituent can be partitioned into two in at least one way:*

$$(\forall V \in \mathcal{C} \text{ s.t. } |V| > 1) \Rightarrow (\exists i \in \mathcal{I} \text{ s.t. } \mathcal{L}(i, V) \neq \emptyset).$$

C4. *Every constituent has at most one left i-subconstituent of given cardinality:*

$$(\forall i \in \mathcal{I}, V \in \mathcal{C}, V', V'' \in \mathcal{L}(i, V) \\ \text{s.t. } |V'| = |V''|) \Rightarrow (V' = V'').$$

Given D and $\mathcal{I}$, it is typically easy to find $\mathcal{C}$ and $\mathcal{L}$ such that the quadruple $\mathcal{H} = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ meets C1, C2, and C3. Condition C4 is not as straightforward; however, we now describe a simple algorithm to modify any $\mathcal{H}$ satisfying C1, C2, and C3 to obtain a valid constituent hierarchy. Specifically, suppose that we have a quadruple $\mathcal{H} = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ that meets C1, C2, and C3 but not C4. The following algorithm constructs a quadruple $\mathcal{H}' = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L}' \rangle$ that meets all of the conditions.

Algorithm 1. *Let $\prec$ be a total ordering on constituents. Initialize $\mathcal{L}'$ to be the same as $\mathcal{L}$. Terminate $\mathcal{L}'$ if there are no violations of C4. Let V be the first constituent, according to $\prec$, that violates C4. That means that there exists a production class $i \in \mathcal{I}$ and two constituents V_1 and $V_2 \in \mathcal{L}'(i, V)$ such that $|V_1| = |V_2|$. Without loss of generality, assume that $V_1 \prec V_2$. Remove V_2 from $\mathcal{L}'(i, V)$. Repeat this process until there are no violations of C4.*

Section 4.1 shows how to construct $\mathcal{H}$ for a certain class of domains D, resulting in the number of constituents that is polynomial in $|D|$ and leading to polynomial-time inference algorithms.

3.3.3 Constituency Functions

To associate parse trees with feature images, we introduce the concept of constituency function. Suppose that $\mathcal{H} = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ is a constituent hierarchy, T is a parse tree generated using the set of production classes $\mathcal{I}$, and $\mathbf{O} \in \mathbb{R}^{q|D|}$ is a feature image. Consider a function F that maps the nodes of T to the constituents of D and has the following properties:

- For the root node ρ, $F(\rho) = D$.
- For every terminal node ρ labeled u, $F(\rho) = \{v\}$, where $O(v) = u$.
- For every nonbranching nonterminal node ρ with a child ρ_1, $F(\rho) = F(\rho_1)$.
- For every branching nonterminal node ρ, tagged i, with left and right children ρ_1 and ρ_2, $F(\rho_1) \in \mathcal{L}(i, F(\rho))$, and $F(\rho_2) = F(\rho) \setminus F(\rho_1)$.

In this case, we say that F is an *$\mathcal{H}$-constituency function* for T and $\mathbf{O}$, and we say that T is an *$\mathcal{H}$-parse* of $\mathbf{O}$. We show that Conditions C1, C2, C3, and C4 guarantee that any $\mathcal{H}$-parse T of $\mathbf{O} \in \mathbb{R}^{q|D|}$ cannot also be an $\mathcal{H}$-parse of another feature image $\mathbf{O}' \in \mathbb{R}^{q|D|}$, and that moreover, the $\mathcal{H}$-constituency function for T and $\mathbf{O}$ is unique.

Theorem 1. *Let $\mathcal{H} = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ be a constituent hierarchy. Let T be an $\mathcal{H}$-parse of $\mathbf{O} \in \mathbb{R}^{q|D|}$ and let F be an $\mathcal{H}$-constituency function for T and $\mathbf{O}$. Let T also be an $\mathcal{H}$-parse of $\mathbf{O}' \in \mathbb{R}^{q|D|}$ and let F' be an $\mathcal{H}$-constituency function for T and $\mathbf{O}'$. Then, $F = F'$ and $\mathbf{O} = \mathbf{O}'$.*

Proof. See the Appendix. □

3.4 Probabilistic Modeling with SRTGs

By using the framework described in the previous section, we can define a probability distribution over the feature images. We do this according to the following plan:

- Construct a probability distribution over trees, similar to PCFGs.
- Construct a quadruple $\mathcal{H}$ that satisfies Conditions C1, C2, and C3.
- If necessary, use Algorithm 1 to enforce C4 and form a constituent hierarchy.
- Construct the probability (or probability density) of a feature image $\mathbf{O} \in \mathbb{R}^{q|D|}$ by summing the probabilities (or probability densities) of all $\mathcal{H}$-parses of $\mathbf{O}$.

We first specify the method for generating random parse trees. With probability r_k, generate a root nonterminal node with label k. Next, repeatedly select a nonterminal node ρ that does not yet have children and denote its label by j. With conditional probability t_j, ρ will be a nonbranching node, and with conditional probability $1 - t_j$, it will be a branching node. If ρ is a nonbranching node, then its child must be a terminal node. In this case, the child's label u is chosen to be a conditionally normal random vector with mean μ_j and covariance Σ_j. If ρ is a branching nonterminal node, then it must have a production class tag that we call i, and left and right children whose labels we call k and l. In this case, choose $\langle i, k, l \rangle$ with conditional probability p_{jikl}. The distribution r and the conditional distributions p_j obey the following normalization constraints:

$$\sum_{j \in \mathcal{J}} r_j = 1, \tag{N1}$$

$$(\forall j \in \mathcal{J}) \sum_{i \in \mathcal{I}, k, l \in \mathcal{J}} p_{jikl} = 1. \tag{N2}$$

Note that, although we take u to be conditionally normal, it is possible to use any conditional distribution, discrete or continuous or hybrid, so long as one replaces the reestimation procedures for μ_j and Σ_j, which are presented in Fig. 8, with suitable variants for the alternate distribution. We take an SRTG G to be an octuple $\langle \mathcal{I}, \mathcal{J}, \mathbb{R}^q, r, t, p, \mu, \Sigma \rangle$, where $\mathcal{I}$ is the set of production classes, $\mathcal{J}$ is the set of nonterminals, and $\mathbb{R}^q$ is the set of terminals. The above process defines $P(T|G)$, the probability density over all trees generated by such a process, given an SRTG G.

Whenever we discuss an SRTG G and a constituent hierarchy $\mathcal{H}$, we assume that both use the same set $\mathcal{I}$ of production classes. In this case, we define, for any feature image $\mathbf{O} \in \mathbb{R}^{q|D|}$

$$P(\mathbf{O}|G, \mathcal{H}) \triangleq \sum_{T \text{ is an } \mathcal{H}\text{-parse of } \mathbf{O}} P(T|G).$$

Note that, strictly speaking, $P(\mathbf{O}|G, \mathcal{H})$ is not a probability density, since not every tree generated by the above stochastic process is necessarily an $\mathcal{H}$-parse of a feature image in $\mathbb{R}^{q|D|}$.[2] Therefore, it may happen that $\int_{\mathbb{R}^{q|D|}} P(\mathbf{O}|G, \mathcal{H}) d\mathbf{O} < 1$. A true probability density P' can then be obtained as follows:

[2] We expand upon this observation in Section 5, where we discuss the relationships between our framework and the traditional PCFGs.

$$\begin{array}{rcll}
c(j,\{v\}) & = & t_jN(O(v)|\mu_j,\Sigma_j) & & (1)\\
c(j,V) & = & \displaystyle\sum_{i,k,l}\ \sum_{V_1\in\mathcal{L}(i,V)}(1-t_j)p_{jikl}c(k,V_1)c(l,V\backslash V_1) & |V|>1 & (2)\\
s(j,D) & = & r_j & & (3)\\
s(j,V) & = & \displaystyle\sum_{i,k,l}\ \sum_{V_1:V\in\mathcal{L}(i,V_1)}(1-t_k)p_{kijl}s(k,V_1)c(l,V_1\backslash V)+ & V\neq D & (4)\\
 & & \displaystyle\sum_{i,k,l}\ \sum_{V_1:V_1\backslash V\in\mathcal{L}(i,V_1)}(1-t_k)p_{kilj}s(k,V_1)c(l,V_1\backslash V) & &
\end{array}$$

Fig. 4. The center and surround recursions for a single feature image $\mathbf{O}$. Equations (1) and (2): The center recursions. Equations (3) and (4): The surround recursions. Each summation over i, k, and l is performed on the entire range of these variables, that is, over $i\in\mathcal{I}$, k, $l\in\mathcal{J}$. Recall that $\mathcal{L}(i,V)$ is the set of left i-subconstituents of V and that t_j is the probability that a node labeled j is nonbranching. We use $N(u|\mu,\Sigma)$ to denote the normal density.

$$P'(\mathbf{O}|G,\mathcal{H})=\frac{P(\mathbf{O}|G,\mathcal{H})}{\int_{\mathbb{R}^{q|D|}}P(\mathbf{O}'|G,\mathcal{H})d\mathbf{O}'}. \quad \text{(N3)}$$

The probability density $P'(\mathbf{O}|G,\mathcal{H})$ is, in general, very difficult to compute because of the denominator in (N3). Fortunately, we will not need compute it in any of our algorithms.

Note that a single SRTG G can be used to define $P(\mathbf{O}|G,\mathcal{H})$ for *any* arbitrary constituent hierarchy $\mathcal{H}$ as long as G and $\mathcal{H}$ use the same set $\mathcal{I}$ of production classes. This is an important property, which is exploited in our algorithms.

3.5 Inference Algorithms

3.5.1 *Likelihood Calculation, Center Variables, and Surround Variables*

Given an SRTG G, a constituent hierarchy $\mathcal{H}$, and a feature image $\mathbf{O}\in\mathbb{R}^{q|D|}$, the likelihood $P(\mathbf{O}|G,\mathcal{H})$ can be computed using (1) and (2) shown in Fig. 4. For any nonterminal j and domain D, we let the *center variable* $c(j,D)$ be the conditional probability density of the feature image $\mathbf{O}$, given that the root label of its $\mathcal{H}$-parse is j:

$$c(j,D)=\sum_{T\text{ is an }\mathcal{H}\text{-parse of }\mathbf{O}\text{ with root label }j}P(T|G).$$

The center variable $c(j,V)$ for any other constituent V of D is defined similarly. Equation (2) in Fig. 4 relates the center variable $c(j,V)$ of any nonsingleton constituent V with the center variables of its subconstituents. Specifically, (2) takes into account all possible partitions of V into a left i-subconstituent V_1 and a right i-subconstituent $V\setminus V_1$ and sums over all such partitions, over all production classes i, and over all possible labels k and l of V_1 and $V\setminus V_1$. Equation (1) is the formula for the center variable of a singleton constituent. In this equation, $N(u|\mu,\Sigma)$ denotes the normal density. The center variables for all constituents and all nonterminals j can therefore be computed by first using (1) for each singleton constituent and then using (2) for every nonsingleton constituent. The feature image likelihood is then obtained as follows: $P(\mathbf{O}|G,\mathcal{H})=\prod_{j\in\mathcal{J}}r_jc(j,D)$.

The *surround variable* $s(j,V)$ for every nonterminal j and every constituent V is computed via the surround recursions shown in (3) and (4) of Fig. 4. Starting with the entire domain D, the surround variables for every constituent are recursively computed in terms of the surround variables of larger constituents and the center variables. These recursions, together with the center recursions, are used in the parameter estimation algorithm that is described below. Note that the center variables must be computed before calculating the surround variables. The center and surround recursions generalize the inside and outside recursions [1], [26].

3.5.2 *Estimation of the MAP Parse*

Given an SRTG G, a constituent hierarchy $\mathcal{H}$, and a feature image $\mathbf{O}\in\mathbb{R}^{q|D|}$, the MAP $\mathcal{H}$-parse of $\mathbf{O}$ is

$$\arg\max_{T\text{ is an }\mathcal{H}\text{-parse of }\mathbf{O}}P(T|G).$$

It can be found using (5), (6), and (7) of Fig. 7. Equations (6) and (7) recursively find the most likely partition of a constituent V labeled j and tagged i into two subconstituents V_1 and $V\backslash V_1$, labeled k and l. Once the most likely quadruple $\langle i,\widehat{k,l},V_1\rangle(j,V)$ for each j and each V is determined and stored, the MAP parse T for $\mathbf{O}$ can be constructed as follows:

1. Let ρ be the root node of T. Label ρ with $\arg\max_{j\in\mathcal{J}}\hat{c}(j,D)$ and let $F(\rho)=D$.
2. For each node ρ in T, labeled j, where $|F(\rho)|>1$, let $\langle i,k,l,V_1\rangle=\langle i,\widehat{k,l},V_1\rangle(j,F(\rho))$, tag ρ with i, add the left and right children ρ_1 and ρ_2 to ρ, label ρ_1 with k and ρ_2 with l, and let $F(\rho_1)=V_1$ and $F(\rho_2)=V\backslash V_1$.
3. For each node ρ in T, labeled j, where $F(\rho)=\{v\}$, add a single child ρ_1 to ρ, label ρ_1 with $O(v)$, and let $F(\rho_1)=\{v\}$.

This algorithm, based on (5), (6), and (7) of Fig. 7, generalizes the Viterbi [52] algorithm.

3.5.3 *Parameter Estimation*

Suppose we are given M training feature images $\mathbf{O}_1\in\mathbb{R}^{q|D_1|},\cdots,\mathbf{O}_M\in\mathbb{R}^{q|D_M|}$, and further suppose that each feature image has a corresponding constituent hierarchy $\mathcal{H}_m=\langle D_m,\mathcal{I},\mathcal{C},\mathcal{L}\rangle$ for $m=1,\cdots,M$. We seek $\arg\max_G\prod_{m=1}^{M}P(\mathbf{O}_m|G,\mathcal{H}_m)$. We address this problem via the algorithm in (8), (9), (10), (11), (12), (13), and (14) of Fig. 8, which uses the center and surround recursions given in (1), (2), (3), and (4) of Fig. 4.

The center and surround recursions, together with (8), (9), (10), (11), (12), (13), and (14) of Fig. 8, constitute the EM algorithm [2], [16]. Equations (8) and (9) constitute the E step, whereas (10), (11), (12), (13), and (14) constitute the M step, that is the reestimation formulas for r_j, t_j, p_{jikl}, μ_j, and Σ_j. We collectively refer to the equations in Figs. 4, 7, and 8 as the *center-surround algorithm.*

The standard implementation of the algorithm in Figs. 4, 7, and 8 memoizes the center and surround recursions. This leads to algorithms for likelihood calculation, MAP estimation, and parameter reestimation that are polynomial in $|D|$, so long as the number of c and s values to be memoized is polynomial in $|D|$, and the summations and maximizations range over sets of indices whose size is polynomial in $|D|$. These are both true, so long as the number of constituents is polynomial in $|D|$.

4 Experiments

4.1 Constituent Hierarchies: An Example

In our experiments, we obtain the domain D by segmenting an image. In this case, every location $v\in D$ corresponds to a region, that is, a set of pixels in an image. We let $\bar{x}(v)$ and $\bar{y}(v)$

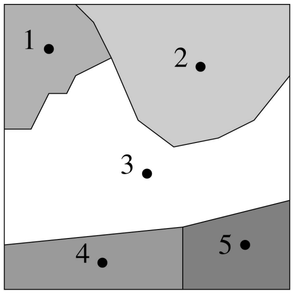


Fig. 5. Illustration of a domain D consisting of five regions. Their centroids are marked with black dots. According to the constituent hierarchy construction in Section 4.1, the following are valid constituents: $1 \cup 2$, $1 \cup 4$, $2 \cup 3$, $2 \cup 5$, $3 \cup 4$, $3 \cup 5$, $4 \cup 5$, $1 \cup 2 \cup 3$, $2 \cup 3 \cup 5$, $3 \cup 4 \cup 5$, $1 \cup 3 \cup 4$, $2 \cup 3 \cup 4$, $1 \cup 2 \cup 3 \cup 5$, $2 \cup 3 \cup 4 \cup 5$, $1 \cup 2 \cup 3 \cup 4$, and $1 \cup 2 \cup 3 \cup 4 \cup 5$.

denote the mean of x- and y-coordinates of the pixels in region v, respectively. We now present a method for constructing a constituent hierarchy $\mathcal{H}$ for any domain D obtained this way.

To define the set $\mathcal{C}$ of constituents of D, we take a nonempty subset $V \subset D$ to be a constituent iff there exists four numbers x_1, x_2, y_1, and y_2 such that for all $v \in V$, $x_1 \leq \bar{x}(v) \leq x_2$, and $y_1 \leq \bar{y}(v) \leq y_2$. Note that the number of constituents is $O(|D|^4)$. This results in a time complexity $O(|\mathcal{I}| \cdot |\mathcal{J}|^3 \cdot |D|^5)$ for the likelihood calculation, one step of EM, and a MAP tree estimation.

We take the set $\mathcal{I}$ of production classes to be $\{\text{"row"}, \text{"column"}\}$. In order to specify a constituent hierarchy, it remains to specify the sets $\mathcal{L}(i, V)$ of left i-subconstituents of V for both production classes i and every constituent V. Suppose V is a constituent. We take a proper nonempty subset V_1 of V to be a left "row"-subconstituent of V iff there exists a number x such that $\bar{x}(v) < x$ for all $v \in V_1$ and $\bar{x}(v) > x$ for all $v \in V \setminus V_1$. We take a proper nonempty subset V_1 of V to be a left "column"-subconstituent of V iff there exists a number y such that $\bar{y}(v) < y$ for all $v \in V_1$ and $\bar{y}(v) > y$ for all $v \in V \setminus V_1$. Formulating $\mathcal{H} = \langle D, \mathcal{I}, \mathcal{C}, \mathcal{L} \rangle$ in this fashion clearly meets Conditions C1-C4.

Fig. 5 illustrates these definitions. For example, 1 ∪ 2 is a valid constituent, since the centroids of regions 1 and 2 are adjacent vertically, and 3 ∪ 4 is a valid constituent, since the centroids of regions 3 and 4 are adjacent horizontally. For 1 ∪ 2, the only left "row"-subconstituent is 1, and the only left "column"-subconstituent is 1 as well. For 3 ∪ 4, the unique left "row"-subconstituent is 4, and the unique left "column"-subconstituent is 3. Note that, despite the fact that regions 1 and 3 are neighbors (that is, they share a common boundary), their union is not a valid constituent, since their centroids are separated by region 2's centroid in the vertical direction and by region 4's centroid in the horizontal direction. Thus, even though regions 1 and 4 are not neighbors, their union is a valid constituent.

4.2 Implementation of the Parameter Estimation Algorithm

When using SRTGs, it is sometimes convenient to designate a nonterminal j to only labeled nonbranching nonterminal nodes, that is, to have $t_j = 1$. This can be accomplished during training by initializing t_j to 1. Indeed, it follows from the equations in Fig. 8 that if $t_j = 1$, t_j will never change by reestimation. In this case, we refer to j as a *nonbranching nonterminal*. Similarly, we can initialize $t_j = 0$ and obtain a nonterminal j that can only label branching nonterminal nodes. In this case, we refer to j as a *branching nonterminal*. It is also sometimes useful to have the possible labels of the root node comprise only a small subset of $\mathcal{J}$. If the root node can be labeled with a nonterminal j, we refer to j as a *root nonterminal*. Any nonterminal j can be prevented from being a root nonterminal by initializing $r_j = 0$ during training.

4.3 Experiments with a House-Car Data Set

4.3.1 *Data Collection*

To evaluate our methods, we first applied them to the task of distinguishing images of houses from images of cars. We took 100 photographs each of houses and cars in the University Farms Subdivision, West Lafayette, Indiana. The houses were photographed from the front, the cars were photographed from the driver's side, and the target house or car was centered and sized to fill the field of view. The photographs were taken with an Intel Pocket Digital PC Camera at 640×480 resolution.[3] The original images were converted to 256-level gray scale and subsampled to 160×120 without filtering. All subsequent processing was performed on the subsampled gray-scale images. Some sample images from this data set are illustrated in Figs. 6a and 6c. Note that the house images often have (partially occluded) cars parked in front and the car images often have (partially occluded) houses in the background. Moreover, note that the images often have other occluding objects such as trees.

We segmented the images with Ratio Cut [53].[4] All 200 images in our data set contained exactly 10 regions after segmentation. For each image, we define the domain $D = \{v_1, \cdots, v_{10}\}$, where $v_1, \cdots, v_{10}$ are the 10 extracted regions. The results of segmenting the images in Figs. 6a and 6c are shown in Figs. 6b and 6d.

We constructed a four-element feature vector as follows: Let $\Lambda(v)$ denote the covariance matrix of the coordinates of the pixels in region v. We associated each region v with the following features:

- the area of v, that is, the number of pixels in v,
- the average intensity of the pixels in v,
- the orientation of the principle eigenvector of $\Lambda(v)$, and
- the ratio of the smallest to the largest eigenvalue of $\Lambda(v)$.

We normalized each component of the feature vector to be in [0, 1] by dividing the area by the total image area, the average pixel intensity by the maximal possible pixel intensity, and the principle eigenvector orientation (whose value was shifted to lie in $[0, \pi)$) by π. The eigenvalue ratio is already in [0, 1] since it is a ratio of the smallest to the largest eigenvalue.

[3] This data set, as well as all of the source codes and scripts used to perform the experiments reported in this section, are available in ftp://ftp.ecn.purdue.edu/qobi/pami2007.tgz.

[4] We used the iterated region-based segmentation technique with the blocking heuristic. We used a linear decreasing function g, a block size of 32×32, and a homogeneity threshold $H_T = 720$ for the first iteration. We postprocessed the results of the first iteration by using the area-merging technique with $A_T = 100$. We then performed a second iteration with $H_T = 700$ and postprocessed the results of this iteration by repeatedly merging the two adjacent regions with the largest cut ratio until the number of regions was 10.

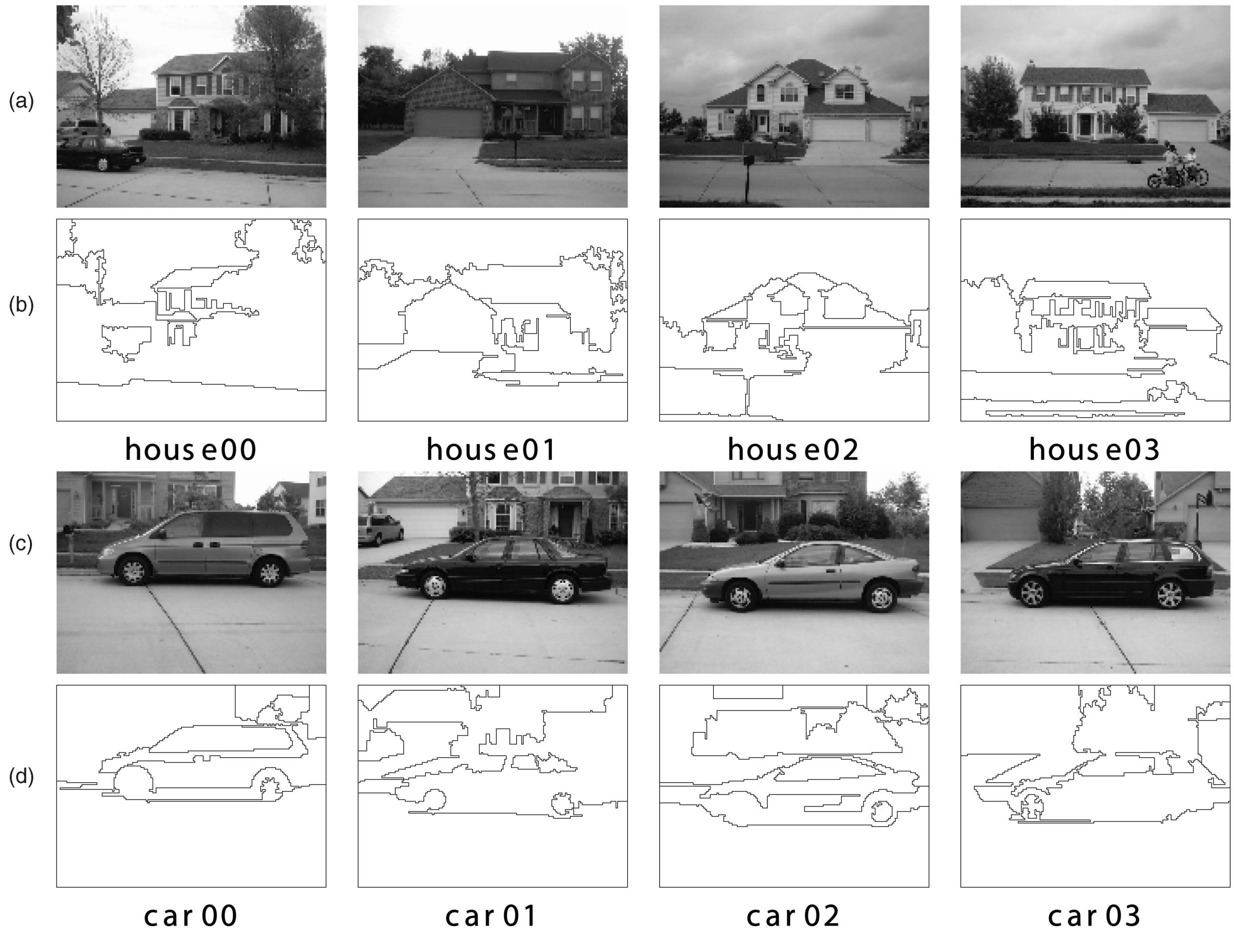


Fig. 6. (a) Sample images from our `house-car` data set. (b) The results of segmenting the images from (a) with Ratio Cut.

$$
\begin{aligned}
\hat{c}(j,\{v\}) &= t_j N(O(v)|\mu_j,\Sigma_j) && && (5)\\
\hat{c}(j,V) &= \max_{i,k,l} \max_{V_1\in\mathcal{L}(i,V)} (1-t_j)p_{jikl}\hat{c}(k,V_1)\hat{c}(l,V\backslash V_1) && |V|>1 && (6)\\
\langle i,\widehat{k,l,V_1}\rangle(j,V) &= \arg\max_{i,k,l} \max_{V_1\in\mathcal{L}(i,V)} (1-t_j)p_{jikl}\hat{c}(k,V_1)\hat{c}(l,V\backslash V_1) && |V|>1 && (7)
\end{aligned}
$$

Fig. 7. The Viterbi variant of the center recursions, used for computing the MAP $\mathcal{H}$-parse of a feature image $\mathbf{O}$ defined on a domain D.

4.3.2 Classifiers

We use two different methods for constructing a classifier with the algorithm in Figs. 4, 7, and 8:

- **Method A**. Train a distinct SRTG for each class on training feature images from that class and classify a test feature image $\mathbf{O}$ by asking which SRTG G maximizes $P(\mathbf{O}|G,\mathcal{H})$.
- **Method B**. Train a single SRTG G with distinct root nonterminals for each of the classes on all of the training feature images, where those feature images are labeled with their class, and classify a test feature image $\mathbf{O}$ by finding the MAP parse of $\mathbf{O}$ and examining the root-node label.

When training with method B, r_j is taken to be 1 when j corresponds to the root nonterminal for the feature image being trained on and is taken to be 0 otherwise. When classifying with method B, the single SRTG constructed jointly for all classes weights the r_j equally among the root nonterminals j for all classes. Method B allows a common vocabulary of terminals across different classes, reflected in the shared μ_j and Σ_j parameters, as well as a common vocabulary of nonterminals across different classes, reflected in the shared p_{jikl} parameters.[5]

4.3.3 Experimental Results

We conducted four experiments to compare the classification accuracy of SRTGs to that of a baseline. For each experiment, we performed a set of five round-robin training and classification runs. For each round-robin run, we partitioned our data set into a training set of 80 images for each class and a test set of 20 images for each class. The test sets partition the entire data set.

[5] When reestimating r_j, t_j, and p_{jikl}, we clip them to 0 when they are less than ϵ. Further, when reestimating t_j, we clip it to 1 when it is greater than $1-\epsilon$. We renormalize the distributions r and p_j after such clipping to maintain the normalization equations (N1) and (N2). When estimating the parameters Σ_j for our baseline model, we perform an eigenvalue/eigenvector decomposition of Σ_j, clip the eigenvalues to ϵ_Σ when they are less than ϵ_Σ, and recompose Σ_j from the original eigenvectors and the clipped eigenvalues. In our experiments, we found that our algorithms are not excessively sensitive to the values of the parameters ϵ and ϵ_Σ. In particular, we found that the values $\epsilon = 10^{-5}$ and $\epsilon_\Sigma = 10^{-3}$, which we used in all the experiments in Section 4, worked well.

$$f_m(j,i,k,l,V) = \frac{s_m(j,V)(1-t_j)p_{jikl}\sum_{V_1\in\mathcal{L}(i,V)} c_m(k,V_1)c_m(l,V\backslash V_1)}{\sum_{j\in\mathcal{J}} c_m(j,D_m)r_j} \tag{8}$$

$$g_m(j,V) = \frac{s_m(j,V)c_m(j,V)}{\sum_{j\in\mathcal{J}} c_m(j,D)r_j} \tag{9}$$

$$p'_{jikl} = \frac{\sum_{m=1}^{M}\sum_{V\in\mathcal{C}_m,|V|>1} f_m(j,i,k,l,V)}{\sum_{m=1}^{M}\sum_{V\in\mathcal{C}_m,|V|>1} g_m(j,V)} \tag{12}$$

$$r'_j = \frac{1}{M}\sum_{m=1}^{M} g_m(j,D_m) \tag{10}$$

$$\mu'_j = \frac{\sum_{m=1}^{M}\sum_{v\in D_m} g_m(j,\{v\})O_m(v)}{\sum_{m=1}^{M}\sum_{v\in D_m} g_m(j,\{v\})} \tag{13}$$

$$t'_j = \frac{\sum_{m=1}^{M}\sum_{v\in D_m} g_m(j,\{v\})}{\sum_{m=1}^{M}\sum_{V\in\mathcal{C}_m} g_m(j,V)} \tag{11}$$

$$\Sigma'_j = \frac{\sum_{m=1}^{M}\sum_{v\in D_m} g_m(j,\{v\})[O_m(v)-\mu'_j][O_m(v)-\mu'_j]^t}{\sum_{m=1}^{M}\sum_{v\in D_m} g_m(j,\{v\})} \tag{14}$$

Fig. 8. Parameter reestimation. Equations (8) and (9): The E step of the reestimation procedure. Equations (10)-(14): The M step of the reestimation procedure. We use the superscript t to denote the transpose of a vector.

TABLE 1
The Results of Experiment 1, Classifying the `house-car` Data Set Using Both the Baseline Models and SRTGs, for Various Round-Robin Runs

test set	nonzero p entries		accuracy on training set			accuracy on test set		
	house	car	baseline	SRTG	Δ	baseline	SRTG	Δ
00–19	108	95	95.6%	100.0%	4.4%	80.0%	92.5%	12.5%
20–39	114	110	95.0%	100.0%	5.0%	85.0%	95.0%	10.0%
40–59	113	108	93.8%	100.0%	6.3%	90.0%	100.0%	10.0%
60–79	108	107	95.6%	100.0%	4.4%	82.5%	100.0%	17.5%
80–99	107	102	95.0%	100.0%	5.0%	82.5%	90.0%	7.5%
mean			95.0%	100.0%	5.0%	84.0%	95.5%	11.5%

This experiment used the separate baseline models and Method A from Section 4.3.2.

We used two different baselines, both of which were Gaussian mixture models. One, the *separate baseline*, trained two separate mixtures of 10 Gaussians: one on houses and one on cars. Each model contained distinct μ_j, Σ_j, and π_j parameters for each class. The other, the *joint baseline*, trained a single mixture of 10 Gaussians on both houses and cars. Each model contained distinct π_j parameters for each class, but the models for both classes shared the same μ_j and Σ_j parameters.

Distinct separate and joint baseline models were trained for each round-robin run. The separate baseline model was trained in the traditional fashion. The joint baseline model was trained by first training a single set of μ_j, Σ_j, and π_j parameters on the combined set of house and car training images. The π_j mixture proportions were then discarded. New sets of π_j mixture proportions were then trained separately on just houses or just cars by using the same μ_j and Σ_j parameters for both houses and cars. Images were classified with the baseline models by selecting the model that best fits that image.

To focus the comparison between SRTGs and the baseline on the advantages of the hierarchical structure provided by SRTGs, we used the same μ_j and Σ_j parameters for both the baseline models and the SRTGs in each round-robin run in each experiment. To train the SRTGs for each round-robin run in each experiment, we took the μ_j and Σ_j parameters for that round-robin run from the baseline model, discarded the π_j mixture proportions, and trained just the p_{jikl} parameters of an SRTG. All SRTGs were trained with 20 nonterminals, in which 10 were constrained to be branching nonterminals, and the remaining 10 were constrained to be nonbranching nonterminals with the fixed μ_j and Σ_j parameters from the baseline model. To train each SRTG, we started the reestimation with two different random values for p_{jikl}, performed 300 iterations of reestimation on each by using (1), (2), (3), and (4), and (8), (9), (10), (11), and (12), and then selected the one that yielded the highest likelihood on the training set. Each initial p contained 8,000 nonzero entries. The number of nonzero entries in the final values of p is indicated in Tables 1, 2, 3, and 4. Note that, typically, less than 2 percent of the entries are nonzero after training.

TABLE 2
The Results of Classifying the `house-car` Data Set Using Support Vector Machines

	SVM accuracy	
test set	on training set	on test set
00–19	83.1%	75.0%
20–39	84.4%	65.0%
40–59	81.9%	75.0%
60–79	81.3%	77.5%
80–99	83.1%	77.5%
mean	82.8%	74.0%

TABLE 3
The Results of Experiment 2, Classifying the `house-car` Data Set Using Both the Baseline Models and SRTGs, for Various Round-Robin Runs

test set	nonzero p entries		accuracy on training set			accuracy on test set		
	house	car	baseline	SRTG	Δ	baseline	SRTG	Δ
00–19	106	106	75.6%	100.0%	24.4%	75.0%	95.0%	20.0%
20–39	117	109	75.6%	100.0%	24.4%	77.5%	95.0%	17.5%
40–59	108	112	75.6%	100.0%	24.4%	75.0%	100.0%	25.0%
60–79	113	118	80.0%	100.0%	20.0%	62.5%	95.0%	32.5%
80–99	98	105	72.5%	100.0%	27.5%	75.0%	100.0%	25.0%
mean			75.9%	100.0%	24.1%	73.0%	97.0%	24.0%

This experiment used the joint baseline models and Method A from Section 4.3.2.

TABLE 4
The Results of Experiment 3, Classifying the `house-car` Data Set Using Both the Baseline Models and SRTGs, for Various Round-Robin Runs

test set	nonzero p	accuracy on training set			accuracy on test set		
	entries	baseline	SRTG	Δ	baseline	SRTG	Δ
00–19	158	75.6%	99.4%	23.8%	75.0%	92.5%	17.5%
20–39	158	75.6%	100.0%	24.4%	77.5%	97.5%	20.0%
40–59	153	75.6%	98.1%	22.5%	75.0%	95.0%	20.0%
60–79	161	80.0%	100.0%	20.0%	62.5%	100.0%	37.5%
80–99	141	72.5%	98.8%	26.3%	75.0%	100.0%	25.0%
mean		75.9%	99.3%	23.4%	73.0%	97.0%	24.0%

This experiment used the joint baseline models and Method B from Section 4.3.2.

Experiment 1. In our first experiment, we compared the classification accuracy of SRTGs to the separate baseline models by using method A from Section 4.3.2. For each round-robin run, we trained both a separate baseline model and an SRTG for each class. We then classified both the training images and test images by using both the baseline models and the SRTGs. The results of this classification are shown in Table 1. SRTGs consistently outperformed the baseline, with only nine misclassifications on 200 test images, compared to 32 baseline misclassifications. It is also interesting to note that our training procedure drastically reduced the model order of SRTGs: although the initial number of nonzero p_{jikl}s was 8,000, the final number was usually about 100.

To provide another benchmark for our experiments, we list in Table 2 the classification performance of a support vector machine (SVM)[6] [51]. The SVM classification experiments used the same feature vectors as the baseline and SRTG models in Table 1. Comparing Tables 1 and 2, we see that SVMs performed substantially worse than both SRTGs and the Gaussian mixture baseline. In addition, SVMs performed substantially worse than SRTGs in Experiments 2 and 3 described below.

Experiment 2. Our second experiment differed from Experiment 1 in that it used the joint baseline models instead of the separate baseline models. We again compared the classification accuracy of SRTGs to the baseline models by using method A from Section 4.3.2. The results of this classification are shown in Table 3. Again, SRTGs consistently outperformed the baseline but, this time, misclassifying only six test images out of 200, compared to 54 baseline misclassifications.

Experiment 3. Experiments 1 and 2 both used method A from Section 4.3.2. Our third experiment differed from Experiment 2 in that it used method B instead of method A. We again compared the classification accuracy of SRTGs to the joint baseline models. We used the same joint baseline model for each class and round-robin run as the one from the corresponding class and round-robin run from Experiment 2. (This means that the baseline numbers in Tables 3 and 4 are identical.) For each round-robin run, we trained a single SRTG on both houses and cars in the corresponding training set. We constrained each SRTG to contain two root branching nonterminals corresponding to the training labels of the two image classes. We then classified both the training images and the test images by using both the baseline models and the SRTGs. The results of this classification are shown in Table 4. SRTGs again significantly outperformed the baseline, with six misclassifications on 200 test images, compared to 54 baseline misclassifications.

Experiment 4. We evaluated the parses produced by SRTGs with a further experiment. For each round-robin run in this experiment, we used the same joint baseline model and SRTG as the corresponding round-robin run of Experiment 3. We attempted to classify houses and cars by using just the shape of their parses without any image data, feature vectors, parse-tree node tags, and parse-tree node labels. For each round-robin run, we computed the set of all best parses for each image in our data set with the SRTG trained on the training set for that round-robin run. Note that there can be, and often are, multiple distinct parses with the same maximal probability. We computed the set of all such parses for each image in our data set. We then took the training set in each round-robin run as the exemplars and classified the test set in

[6] We used the SVM software of http://svmlight.joachims.org/.

TABLE 5
The Results of Experiment 4, Classifying the `house-car` Data Set, Using Both the Baseline Models and SRTGs for Various Round-Robin Runs and Values of k

	baseline	SRTG						
		$k=1$	$k=3$	$k=5$	$k=7$	$k=9$	$k=11$	$k=13$
00–19	75.0%	72.5%	77.5%	75.0%	72.5%	75.0%	82.5%	80.0%
	Δ	−2.5%	2.5%	0.0%	−2.5%	0.0%	7.5%	5.0%
20–39	77.5%	77.5%	70.0%	70.0%	65.0%	70.0%	67.5%	72.5%
	Δ	0.0%	−7.5%	−7.5%	−12.5%	−7.5%	−10.0%	−5.0%
40–59	75.0%	82.5%	85.0%	85.0%	82.5%	82.5%	82.5%	82.5%
	Δ	7.5%	10.0%	10.0%	7.5%	7.5%	7.5%	7.5%
60–79	62.5%	75.0%	72.5%	85.0%	82.5%	82.5%	82.5%	82.5%
	Δ	12.5%	10.0%	22.5%	20.0%	20.0%	20.0%	20.0%
80–99	75.0%	82.5%	75.0%	82.5%	72.5%	80.0%	80.0%	75.0%
	Δ	7.5%	0.0%	7.5%	−2.5%	5.0%	5.0%	0.0%
mean	73.0%	78.0%	76.0%	79.5%	75.0%	78.0%	79.0%	78.5%
	Δ	5.0%	3.0%	6.5%	2.0%	5.0%	6.0%	5.5%

For each round-robin run in this experiment, we used the same joint baseline model and SRTG as the corresponding round-robin run of Experiment 3. Classification with SRTGs was performed using a k-nearest-neighbor classifier between the entire training set taken as exemplars and each element of the test set. The minimal pivot distance between all best parses of an exemplar and all best parses of a test image was taken as the distance metric. Note that pivot distance measured only the shape of the parses and was not sensitive to any parse-tree node tags or labels.

each round-robin against these exemplars with a k-nearest-neighbor classifier by using pivot distance as the distance metric. Our definition of pivot distance is given in the Appendix. Since both exemplars and test images could have multiple best parses, we computed the pivot distance between all pairs consisting of an exemplar best parse and a test image best parse and selected the minimal pivot distance as the distance metric between the exemplar and the test image. Note that pivot distance measured only the shape of the parses and was not sensitive to any parse-tree node tags or labels. We repeated this for all odd values of $1 \leq k \leq 13$ and compared the classification accuracy against the baseline models. The results of this classification are shown in Table 5. Note that, on the average, the SRTG-based method that used only the extracted hierarchical structure outperformed the baseline method that directly used the region feature vectors.

4.3.4 Discussion of the Experiments

The experiments reported in Section 4.3.3 used the same μ_j and Σ_j parameters from the baseline models when constructing SRTGs and did not reestimate those parameters. This was done to focus the experiments on measuring the added leverage afforded by hierarchical structure. SRTGs are strictly more powerful than the baseline models because the baseline models are equivalent to degenerate SRTGs with a single branching nonterminal. Our methods can obviously be used to reestimate the μ_j and Σ_j parameters to yield SRTGs with an even better fit to the training data.

Experiments 1, 2, and 3 demonstrate the added leverage afforded by a hierarchical structure. The baseline models and SRTGs in each round-robin run of each experiment differ only in that the baselines use nonhierarchical mixture proportions reflected in the π_j parameters, whereas the SRTGs use a hierarchical structure reflected in the p_{jikl} parameters. In all three experiments, SRTGs consistently, and often significantly, outperform the corresponding baseline model. This indicates that the presence of a hierarchical structure significantly improves classification accuracy.

Experiment 4 demonstrates that the trained SRTGs allow a derivation of image parses that contain sufficient information to support classification. In other words, it is possible to classify images solely on the basis of their hierarchical structure reflected in the shape of their parses. One can imagine using this for image database retrieval. In fact, it is interesting to note that, in Experiment 4, a classification based solely on the extracted hierarchical structure outperforms, on the average, the baseline classification method which directly uses the region feature vectors.

4.4 Experiments with Additional Data Sets

To further illustrate our methods, we conducted two more experiments where we used different data sets. In addition, in order to illustrate the applicability of our classification methods in conjunction with a variety of preprocessing segmentation algorithms, we employed two segmentation algorithms that are different from the segmentation method used in Experiments 1, 2, 3, and 4: Normalized Cuts [49] and Mean Shift [13].

Experiment 5. In our first additional experiment, we used images from the California Institute of Technology (Caltech) house data set[7] and the Caltech car data set.[8] We used the Normalized Cuts algorithm [49] to segment each image, followed by a single pass of Ratio Cut in order to get 10 regions per image. Our basic setup was similar to Experiment 1. Specifically, we trained both a separate baseline mixture of 10 Gaussians and an SRTG for each class; the same μ_j and Σ_j parameters were used for the baseline and SRTG models. Out of the total of 441 car images and 237 house images used in the experiment, we

[7] http://www.vision.caltech.edu/Image_Datasets/Pasadena-Houses-2000.tar.

[8] http://www.vision.caltech.edu/Image_Datasets/cars_brad/cars_brad.tar.

TABLE 6
The Results of Experiment 5, Classifying the Caltech `house-car` Data Set, Using Both the Baseline Models and SRTGs

accuracy on training set			accuracy on test set		
baseline	SRTG	Δ	baseline	SRTG	Δ
95.0%	97.8%	2.8%	90.2%	95.4%	5.2%

randomly selected 160 images of each class as the training set and used the remaining images as the test set. The results of this experiment, summarized in Table 6, show that SRTGs significantly outperform the baseline, achieving misclassification rates that are more than twice as low as the baseline misclassification rates for both training data and test data.

Experiment 6. In our second additional experiment, we combined the house and car images used in Experiments 1-4 with another set of 200 images taken around the University Farms Subdivision, West Lafayette, Indiana: 100 images of mailboxes and 100 images of basketball hoops. We segmented the whole data set with the Mean Shift algorithm [13], followed by three passes of Ratio Cut in order to get 20 or fewer regions per image. We again used an experimental setup similar to that of Experiment 1. We conducted five round-robin runs, where 80 images in each class were used as a training set, and the remaining 20 images were used as a test set. The test sets for the five runs were disjoint. The correct classification rates for each pairwise classification task, averaged over the five round robins, are given in Table 7. On average, SRTGs reduce the misclassification rate, as compared to the baseline, by about a factor of 3 on the training set and a factor of 1.5 on the test set. Note that these average numbers are very similar to the numbers for the two newly added classes: the SRTG misclassification rate for hoops versus mailboxes is about three times smaller than the baseline's misclassification rate on the training set and about 1.5 times smaller than the baseline's misclassification rate on the test set. The correct classification rates for one-of-four classifications are given in Table 8, again showing that SRTGs achieve approximately three and 1.5 times smaller misclassification rates than the baseline on the training and test sets, respectively.

5 SRTGs, PCFGs, and Our Prior Work

Our notation for the probability distribution of images is different from the traditional notation for PCFGs in that, in our case, the probability distribution for $\mathbf{O}$ is conditioned not only on the grammar G but also on the constituent hierarchy $\mathcal{H}$. However, the traditional PCFGs implicitly assume the constituent hierarchy that corresponds to concatenating 1D strings and, therefore, the probabilities induced by a PCFG are implicitly conditioned on this constituent hierarchy.

More precisely, let $D_M = \{1, \cdots, M\}$, and let $\mathcal{I} = \{i\}$ consist of a single production class. Let us form a constituent hierarchy $\mathcal{H}_M = \langle D_M, \mathcal{I}, \mathcal{C}_M, \mathcal{L}_M \rangle$ as follows: We form the set $\mathcal{C}_M$ by taking every contiguous subset of D_M to be a constituent. We take any proper nonempty subset V_1 of V to be a left i-subconstituent of V iff there exists an integer n such that $v \leq n$ for all $v \in V_1$ and $v > n$ for all $v \in V \setminus V_1$. Formulating $\mathcal{H}_M$ in this fashion clearly meets Conditions C1, C2, C3, and C4. Let $\mathcal{H}^{\mathbb{Z}^+} = \{\mathcal{H}_1, \mathcal{H}_2, \cdots\}$ be the collection of these constituent hierarchies $\mathcal{H}_M$ for all M.

Now, let G be any SRTG that uses the same set $\mathcal{I} = \{i\}$ consisting of a single production class. Let G_{PCFG} be the PCFG that has the same sets of terminals and nonterminals as G and has the same parameters r_j, p_{jikl}, t_j, μ_j, and Σ_j. For simplicity, we assume that $q = 1$ (that is, the terminals are scalars) and that the PCFG G_{PCFG} is proper.[9] It can then be shown that the pair $\langle G, \mathcal{H}^{\mathbb{Z}^+} \rangle$ is equivalent to the PCFG G_{PCFG} in the following sense. For any $\mathbf{O} \in \mathbb{R}^M$, the quantity $P(\mathbf{O}|G, \mathcal{H}_M)$, as defined in Section 3.4, is equal to the probability of the string $\mathbf{O}$ induced by the PCFG G_{PCFG}. In this sense, PCFGs can be viewed as a special case of SRTGs that is obtained by using a singleton set of production classes and the collection $\mathcal{H}^{\mathbb{Z}^+}$ of constituent hierarchies.

A similar relationship can be shown to exist between SRTGs as formulated in the present paper and our prior work [38], [39], [40], where a specialized version of the center-surround algorithm was developed for the case when every constituent is a rectangular set of image pixels. Another version of spatial random tree models and the center-surround algorithm was developed in [54]. In that work, the problem of uniquely associating the leaves of a parse tree with image pixels was solved by hard-wiring image regions to nonterminals. This resulted in a CFG that models images specified on a fixed finite domain, which is a departure both from [38], [39], [40], and the framework developed in the present paper.

We finally point out an interesting distinction between PCFGs and our general formulation of SRTGs. Given an SRTG G and a constituent hierarchy $\mathcal{H}$ for a domain D, it is easy to see that a parse tree generated by G will not necessarily be an $\mathcal{H}$-parse of some feature image $\mathbf{O} \in \mathbb{R}^{q|D|}$, since the number of leaves of the parse tree may not necessarily be equal to $|D|$. This is similar to PCFGs. However, even if the number of leaves of a parse tree T is equal to $|D|$, it may happen that T is not an $\mathcal{H}$-parse of any feature image. This is in contrast to PCFGs where every parse tree generated by a grammar must necessarily be a parse for some string. This distinction, however, does not influence any of our algorithms.

6 Conclusion

We have presented a novel method for formulating priors on the hierarchical organization of a feature image and its constituents by using SRTGs. The center-surround algorithm can be used to estimate the parameters of SRTGs from sets of training feature images and to classify images based on their likelihood and on the MAP estimate of the associated

[9] A PCFG is called *proper* [7] if the total probability of all infinite trees generated by the PCFG is zero.

TABLE 7
Results for All Two-Way Classifications for the `hoop-mailbox-house-car` Data Set in Experiment 6

	accuracy on training set			accuracy on test set		
	baseline	SRTG	Δ	baseline	SRTG	Δ
hoop-mailbox	85.3%	95.5%	10.2%	83.0%	88.0%	5.0%
hoop-car	97.2%	99.1%	1.9%	97.5%	98.5%	1.0%
hoop-house	94.8%	96.8%	2.0%	93.5%	93.0%	−0.5%
mailbox-car	94.2%	98.7%	4.5%	92.0%	96.5%	4.5%
mailbox-house	95.1%	97.8%	2.7%	92.0%	95.0%	3.0%
car-house	95.1%	99.3%	4.2%	92.5%	96.0%	3.5%
mean	93.6%	97.9%	4.3%	91.75%	94.5%	2.75%

hierarchical structure. We have demonstrated the efficacy of these methods by training on and classifying natural images.

There are several open problems associated with our framework that we are currently investigating. We are developing alternative methodologies both for constructing constituent hierarchies [54] and feature selection to improve the overall classification performance. We are also investigating methods to improve parameter estimation, that is, to avoid convergence to poor local maxima.

Appendix A
Proof of Theorem 1

The proof is by contradiction. Let ρ be the first node in T in a preorder traversal from left to right for which these two functions differ, that is, for which $F(\rho) \neq F'(\rho)$. Since $\mathcal{H}$-constituency functions map the root node to D, ρ cannot be the root node.

Case 1. ρ is an only child. Since $\mathcal{H}$-constituency functions map every nonbranching nonterminal node to the same constituent as its child, F and F' must map the parent of ρ to different constituents. This contradicts the premise, since parents are traversed before their children.

Case 2. ρ is a right child. Let V_2, V, and V_1 denote the constituents to which F maps ρ, its parent, and its sibling. Let V_2', V', and V_1' denote the constituents to which F' maps them, respectively. By traversal, $V = V'$, and $V_1 = V_1'$. Since $\mathcal{H}$-constituency functions map the children of every branching nonterminal node to constituents that partition the constituent mapped to by their parent, $V_2 = V_2'$, which contradicts the premise.

Case 3. ρ is a left child. Let i be the tag of the parent of ρ. Let V_1 and V denote the constituents to which F maps ρ and its parent. Let V_1' and V' denote the constituents to which F' maps them respectively. By traversal, $V = V'$. Furthermore, $|V_1| = |V_1'|$, since both equal the number of leaf nodes dominated by ρ. Based on Condition C4, $V_1 = V_1'$, which contradicts the premise.

This proves that $F = F'$. It is easy to see that Conditions C1, C2, C3, and C4, together with our definition of a parse tree, imply that, for every $v \in D$, there exists a leaf node ρ of T such that $F(\rho) = \{v\}$. For this location v, we have, based on the definition of an $\mathcal{H}$-constituency function, $O(v) = u$, where u is the label of ρ. Since $F = F'$, it follows that $F'(\rho) = \{v\}$, and therefore, $O'(v) = u = O(v)$. Since this is true for every $v \in D$, it follows that $\mathbf{O} = \mathbf{O}'$. □

Appendix B
MAP Parse and Parameter Estimation Formulas

In Fig. 7, we give the recursive formulas for computing the MAP $\mathcal{H}$-parse of an image, as discussed in Section 3.5. Note that (5) is identical to (1) of Fig. 4, and (6) is obtained by replacing each "$\sum$" with a "$\max$" in (2).

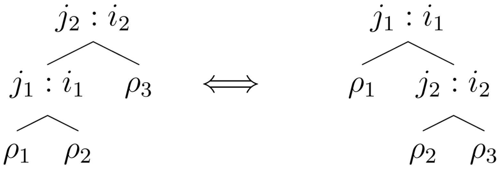
Equations (8), (9), (10), (11), (12), (13), and (14) of Fig. 8 are the formulas for the parameter updates, as discussed in Section 3.5. In (8), (9), (10), (11), (12), (13), and (14), the parameters corresponding to the mth training feature image $\mathbf{O}_m$ are indexed by m. In particular, c_m stands for the center variables computed by applying the center recursions to $\mathbf{O}_m$, and s_m stands for the surround variables computed by applying the surround recursions to $\mathbf{O}_m$. If we take G' to be $\langle \mathcal{I}, \mathcal{J}, \mathbb{R}^q, r', t', p', \mu', \Sigma' \rangle$, then it can be shown that $\prod_{m=1}^{M} P(\mathbf{O}_m|G', \mathcal{H}_m) \geq \prod_{m=1}^{M} P(\mathbf{O}_m|G, \mathcal{H}_m)$. Repeating this process converges to a local maximum [44], [55].

TABLE 8
Results for Four-Way Classifications for the `hoop-mailbox-house-car` Data Set in Experiment 6

class	accuracy on training set			accuracy on test set		
	baseline	SRTG	Δ	baseline	SRTG	Δ
mailbox	80.0%	97.7%	17.7%	73.0%	89.0%	16.0%
hoop	80.0%	89.2%	9.2%	78.0%	79.0%	1.0%
house	89.2%	96.2%	7.0%	82.0%	92.0%	10.0%
car	92.5%	97.2%	4.7%	91.0%	92.0%	1.0%
mean	85.4%	95.1%	9.7%	81.0%	88.0%	7.0%

Appendix C
Pivot Distance

Let us define the *pivot distance* between pairs of parse trees as follows: Let BRANCHING(ρ) be true if ρ is a branching nonterminal node and false if it is a nonbranching nonterminal node. If ρ is a branching nonterminal node, let LEFT(ρ) and RIGHT(ρ) denote the left and right children of ρ, respectively. Define a *left pivot* to be the transformation from the right to left and a *right pivot* to be the reverse transformation.



Note that it is only possible to left-pivot a branching node whose right child is a branching node. Similarly, note that it is only possible to right-pivot a branching node whose left child is a branching node. Let LEFTPIVOT(ρ) and RIGHTPIVOT(ρ) denote the left and right pivots, respectively, of ρ when they exist and ρ when they do not exist.

The pivot distance $||\rho_1, \rho_2||$ between ρ_1 and ρ_2 is defined as the minimal number of pivots that must be applied to ρ_1 and ρ_2 so that the resulting trees have the same shape (that is, of the same ignoring labels and tags). The pivot distance $||\rho_1, \rho_2||$ can be computed as follows:

$$\|\rho_1,\rho_2\|=\begin{cases} 0 & \begin{pmatrix}\neg\text{BRANCHING}(\rho_1),\\ \neg\text{BRANCHING}(\rho_2)\end{pmatrix}\\ \infty & \begin{pmatrix}\text{BRANCHING}(\rho_1),\\ \neg\text{BRANCHING}(\rho_2)\end{pmatrix}\\ \infty & \begin{pmatrix}\neg\text{BRANCHING}(\rho_1),\\ \text{BRANCHING}(\rho_2)\end{pmatrix}\\ \min\begin{pmatrix} 1+\min\begin{pmatrix}\|\text{LEFTPIVOT}(\rho_1),\rho_2\|,\\ \|\text{RIGHTPIVOT}(\rho_1),\rho_2\|,\\ \|\rho_1,\text{LEFTPIVOT}(\rho_2)\|\ ,\\ \|\rho_1,\text{RIGHTPIVOT}(\rho_2)\|\end{pmatrix},\\ \|\text{LEFT}(\rho_1),\text{LEFT}(\rho_2)\|+\\ \|\text{RIGHT}(\rho_1),\text{RIGHT}(\rho_2)\|\end{pmatrix} & \begin{pmatrix}\text{BRANCHING}(\rho_1),\\ \text{BRANCHING}(\rho_2)\end{pmatrix}.\end{cases}$$

This can be computed in polynomial time by memoizing $||\rho_1, \rho_2||$. Note that the pivot distance between trees with different numbers of nodes is infinite and that the pivot distance between trees with the same number of nodes is finite.

ACKNOWLEDGMENTS

This work was supported in part by the US National Science Foundation (NSF) Faculty Early Career Development (CAREER) Award CCR-0093105, US NSF Grants 9987576, 9980054, and 0329156IIS, a Purdue Research Foundation grant, a Xerox Foundation grant, an Advanced Research and Development Authority (ARDA) grant under Contract MDA904-03-C-1788, and a DARPA grant under Contract HR0011-06-C-0023. Part of this work was carried out while M.P. Harper was on leave at the NSF. Any opinions, findings, and conclusions expressed in this paper are those of the authors and do not necessarily reflect the views of ARDA, DARPA, and NSF. The authors would like to thank Shawn Brownfield, Bingrui Foo, Jacob Harris, Yan Huang, Eduardo Maia, William Nagel, and Eric Théa for the experimental work, as well as for stimulating and helpful discussion, leading to this paper. The experiments reported in this paper utilized the significant computational resources made available to us by the ITaP Research Computing Services Group.

REFERENCES

[1] J.K. Baker, "Trainable Grammars for Speech Recognition," *Proc. Speech Comm. Papers, 97th Meeting of the Acoustical Soc. Am.,* pp. 547-550, 1979.

[2] L.E. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Math. Statistics,* vol. 41, no. 1, pp. 164-171, 1970.

[3] J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems," *J. Royal Statistical Soc. B,* vol. 36, no. 2, pp. 192-236, 1974.

[4] J. Besag, "Efficiency of Pseudolikelihood Estimation for Simple Gaussian Fields," *Biometrika,* vol. 64, no. 3, pp. 616-618, 1977.

[5] A.F. Bobick and Y.A. Ivanov, "Action Recognition Using Probabilistic Parsing," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 196-202, June 1998.

[6] C.A. Bouman and M. Shapiro, "A Multiscale Random Field Model for Bayesian Image Segmentation," *IEEE Trans. Image Processing,* vol. 3, no. 2, pp. 162-177, Mar. 1994.

[7] Z. Chi and S. Geman, "Estimation of Probabilistic Context-Free Grammars," *Computational Linguistics,* vol. 24, no. 2, pp. 299-305, June 1998.

[8] H. Choi and R.G. Baraniuk, "Multiscale Document Segmentation Using Wavelet-Domain Hidden Markov Models," *Proc. Int'l Soc. Optical Eng./Soc. for Imaging, Science and Technology (SPIE/IS&T) 12th Ann. Int'l Symp.-Electronic Imaging,* Jan. 2000.

[9] H. Choi and R.G. Baraniuk, "Multiscale Image Segmentation Using Wavelet-Domain Hidden Markov Models," *IEEE Trans. Image Processing,* vol. 10, no. 9, pp. 1309-1321, Sept. 2001.

[10] N. Chomsky, *The Logical Structure of Linguistic Theory.* Plenum, 1955.

[11] N. Chomsky, *Syntactic Structures.* Mouton, The Hague, 1957.

[12] N. Chomsky, "On Certain Formal Properties of Grammars," *Information and Control,* vol. 2, pp. 137-167, 1959.

[13] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach toward Feature Space Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 5, pp. 603-619, May 2002.

[14] I.J. Cox, S.B. Rao, and Y. Zhong, "Ratio Regions: A Technique for Image Segmentation," *Proc. Int'l Conf. Pattern Recognition,* pp. 557-564, Aug. 1996.

[15] M.S. Crouse, R.D. Nowak, and R.G. Baraniuk, "Wavelet-Based Statistical Signal Processing Using Hidden Markov Models," *IEEE Trans. Signal Processing,* vol. 46, no. 4, pp. 886-902, Apr. 1998.

[16] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm (with Discussion)," *J. Royal Statistical Soc. B,* vol. 39, pp. 1-38, 1977.

[17] K.S. Fu, *Syntactic Pattern Recognition and Applications.* Prentice Hall, 1982.

[18] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, pp. 721-741, Nov. 1984.

[19] X. He, R.S. Zemel, and M.Á. Carreira-Perpi nán, "Multiscale Conditional Random Fields for Image Labeling," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2004.

[20] T. Kanungo and S. Mao, "Stochastic Language Models for Style-Directed Layout Analysis of Document Images," *IEEE Trans. Image Processing,* vol. 12, no. 5, pp. 583-596, May 2003.

[21] T. Kasami, "An Efficient Recognition and Syntax Algorithm for Context-Free Languages," Scientific Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, Mass., 1965.

[22] G.E. Kopec and P.A. Chou, "Document Image Decoding Using Markov Source Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 6, pp. 602-617, June 1994.

[23] M. Krishnamoorthy, G. Nagy, S.C. Seth, and M. Viswanathan, "Syntactic Segmentation and Labeling of Digitized Pages from Technical Journals," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 15, no. 7, pp. 737-747, July 1993.

[24] S. Kumar and M. Hebert, "A Hierarchical Field Framework for Unified Context-Based Classification," *Proc. IEEE Int'l Conf. Computer Vision,* Oct. 2005.

[25] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," *Proc. Int'l Conf. Machine Learning,* pp. 282-289, 2001.

[26] K. Lari and S.J. Young, "The Estimation of Stochastic Context-Free Grammars Using the Inside-Outside Algorithm," *Computer Speech and Language,* vol. 4, no. 1, pp. 35-56, 1990.

[27] S.E. Levinson, "Continuously Variable Duration Hidden Markov Models for Speech Analysis," *Proc. Int'l Conf. Acoustic and Speech Signal Processing,* pp. 1241-1244, 1986.

[28] J. Li and R.M. Gray, "Context-Based Multiscale Classification of Document Images Using Wavelet Coefficient Distributions," *IEEE Trans. Image Processing,* vol. 9, no. 9, pp. 1604-1616, Sept. 2000.

[29] J. Li, R.M. Gray, and R.A. Olshen, "Multiresolution Image Classification by Hierarchical Modeling with Two-Dimensional Hidden Markov Models," *IEEE Trans. Information Theory,* vol. 46, no. 5, pp. 1826-1841, Aug. 2000.

[30] M.R. Luettgen, W.C. Karl, A.S. Willsky, and R.R. Tenney, "Multiscale Representations of Markov Random Fields," *IEEE Trans. Signal Processing,* vol. 41, no. 12, Dec. 1993.

[31] C. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[32] J.W. Modestino and J. Zhang, "A Markov Random Field Model-Based Approach to Image Interpretation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 14, no. 6, pp. 606-615, June 1992.

[33] J.-M. Morel and S. Solimini, *Variational Methods in Image Segmentation.* Birkhauser, 1995.

[34] G. Nagy and S.C. Seth, "Hierarchical Image Representation with Application to Optically Scanned Documents," *Proc. Int'l Conf. Pattern Recognition,* pp. 347-349, July 1984.

[35] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[36] F. Pereira and Y. Schabes, "Inside-Outside Reestimation from Partially Bracketed Corpora," *Proc. 30th Ann. Meeting Assoc. Computational Linguistics,* pp. 128-135, 1992.

[37] I. Pollak, A.S. Willsky, and H. Krim, "Image Segmentation and Edge Enhancement with Stabilized Inverse Diffusion Equations," *IEEE Trans. Image Processing,* vol. 9, no. 2, Feb. 2000.

[38] I. Pollak, J.M. Siskind, M.P. Harper, and C.A. Bouman, "Modeling and Estimation of Spatial Random Trees with Application to Image Classification," *Proc. Int'l Conf. Acoustics, Speech, and Signal,* Apr. 2003.

[39] I. Pollak, J.M. Siskind, M.P. Harper, and C.A. Bouman, "Parameter Estimation for Spatial Random Trees Using the EM Algorithm," *Proc. Int'l Conf. Image Processing,* Sept. 2003.

[40] I. Pollak, J.M. Siskind, M.P. Harper, and C.A. Bouman, "Spatial Random Trees and the Center-Surround Algorithm," Technical Report TR-ECE-03-03, School of Electrical and Computer Eng., Purdue Univ., Jan. 2003.

[41] G.G. Potamianos and J.K. Goutsias, "Partition Function Estimation of Gibbs Random Filed Images Using Monte Carlo Simulation," *IEEE Trans. Information Theory,* vol. 39, no. 4, pp. 1322-1332, July 1993.

[42] D. Potter, "Compositional Pattern Recognition," PhD dissertation, Brown Univ., 1999, http://www.dam.brown.edu/people/dfp.

[43] L.R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE,* vol. 77, no. 2, pp. 257-286, 1989.

[44] R.A. Redner and H.F. Walker, "Mixture Densities, Maximum Likelihood and the EM Algorithm," *SIAM Rev.,* vol. 26, no. 2, pp. 195-239, Apr. 1984.

[45] J.K. Romberg, H. Choi, and R.G. Baraniuk, "Bayesian-Tree-Structured Image Modeling Using Wavelet-Domain Hidden Markov Models," *IEEE Trans. Image Processing,* vol. 10, no. 7, pp. 1056-1068, July 2001.

[46] A. Rosenfeld, *Picture Languages.* Kluwer Academic Press, 1979.

[47] D. Sankoff, "Branching Processes with Terminal Types: Application to Context-Free Grammars," *J. Applied Probability,* vol. 8, pp. 233-240, 1971.

[48] A.C. Shaw, "A Formal Picture Description Scheme as a Basis for Picture Processing Systems," *Information and Control,* vol. 14, pp. 9-52, 1969.

[49] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 888-905, Aug. 2000.

[50] A. Torralba, K.P. Murphy, and W.T. Freeman, "Contextual Models for Object Detection Using Boosted Random Fields," *Proc. 18th Ann. Conf. Neural Information Processing Systems,* 2004.

[51] V.N. Vapnik, *The Nature of Statistical Learning Theory.* Springer, 1995.

[52] A.J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Trans. Information Theory,* vol. 13, pp. 260-267, 1967.

[53] S. Wang and J.M. Siskind, "Image Segmentation with Ratio Cut," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 6, pp. 675-690, June 2003.

[54] W. Wang, I. Pollak, T.-S. Wong, C.A. Bouman, M.P. Harper, and J.M. Siskind, "Hierarchical Stochastic Image Grammars for Classification and Segmentation," *IEEE Trans. Image Processing,* vol. 15, no. 10, pp. 3033-3052, Oct. 2006.

[55] C.F.J. Wu, "On the Convergence Properties of the EM Algorithm," *Annals of Statistics,* vol. 11, no. 1, pp. 95-103, 1983.

[56] D.H. Younger, "Recognition and Parsing of Context-Free Languages in Time $O(n^3)$," *Information and Control,* vol. 10, no. 2, pp. 189-208, 1967.

Jeffrey M. Siskind received the BA degree in computer science from the Technion, Israel Institute of Technology, Haifa, in 1979, the SM degree in computer science from the Massachusetts Institute of Technology (MIT), Cambridge, in 1989, and the PhD degree in computer science from MIT in 1992. He did a postdoctoral fellowship at the University of Pennsylvania Institute for Research in Cognitive Science from 1992 to 1993. He was an assistant professor in the Department of Computer Science, University of Toronto, from 1993 to 1995, a senior lecturer in the Department of Electrical Engineering, Technion, in 1996, a visiting assistant professor in the Department of Computer Science and Electrical Engineering, University of Vermont, from 1996 to 1997, and a research scientist at NEC Research Institute from 1997 to 2001. He joined the Purdue University School of Electrical and Computer Engineering in 2002, where he is currently an associate professor. His research interests include machine vision, artificial intelligence, cognitive science, computational linguistics, child language acquisition, and programming languages and compilers. He is a senior member of the IEEE and the IEEE Computer Society.

James J. Sherman Jr. received the BS degree in electrical engineering and the BS degree in mathematics in 2003 from Purdue University, West Lafayette. In 2003, he joined the PhD program at the University of Maryland, College Park, where he is currently studying. His research interests include computer vision and pattern recognition. He is a student member of IEEE.

Ilya Pollak received the BS and MEng degrees in electrical engineering in 1995 and the PhD degree in electrical engineering in 1999, all from the Massachusetts Institute of Technology, Cambridge. From 1999 to 2000, he was a postdoctoral researcher at the Division of Applied Mathematics, Brown University, Providence, Rhode Island. Since 2000, he has been with Purdue University, West Lafayette, Indiana, where he is currently an associate professor of electrical and computer engineering. He has held short-term visiting positions at the Institut National de Recherche en Informatique et en Automatique in Sophia Antipolis, France, and at Tampere University of Technology, Finland. He is an associate editor of the *IEEE Transactions on Signal Processing*, a member of the IEEE Signal Processing Society's Technical Committee on Signal Processing Theory and Methods, and the chair of the Signal Processing Chapter of the Central Indiana Section of the IEEE. He is the cochair of the International Society for Optical Engineering/Society for Imaging, Science and Technology (SPIE/IS&T) Conference on Computational Imaging. His research interests include image and signal processing, specifically, hierarchical statistical models, fast estimation algorithms, nonlinear scale-spaces, and adaptive representations. He received a Faculty Early Career Development (CAREER) award from the US National Science Foundation in 2001. He is a senior member of the IEEE.

Mary P. Harper received the ScM and PhD degrees in computer science from Brown University, Providence, Rhode Island, in 1986 and 1990, respectively. In 1989, she joined the faculty of Purdue University, West Lafayette, Indiana, where she currently holds the rank of professor in the School of Electrical and Computer Engineering. She was a speaker for the IEEE Computer Society Distinguished Visitors Program and the Chapter Tutorial Program from 1997 to 2000. She is currently an associate editor of the *IEEE Transactions on Speech, Audio, and Language Processing*. She also sits on the Board of the North American Chapter of the Association for Computational Linguistics (NAACL). Her research focuses on computer modeling of human communication, with a focus on methods for incorporating multiple types of knowledge sources, including lexical, syntactic, prosodic, and, most recently, visual sources. Her recent research involves the integration of speech and natural language processing systems, the integration of speech, gesture, and gaze, and the utilization of hierarchical structure to improve the classification accuracy of documents and images. She is a senior member of the IEEE and the IEEE Computer Society.

Charles A. Bouman received the BSEE degree from the University of Pennsylvania, Philadelphia, in 1981, the MS degree in electrical engineering from the University of California, Berkeley, in 1982, and the PhD degree in electrical engineering from Princeton University, Princeton, New Jersey. From 1982 to 1985, he was a full staff member at MIT Lincoln Laboratory, Lexington, Massachusetts. In 1989, he joined the faculty of Purdue University, West Lafayette, Indiana, where he holds the rank of professor, with a primary appointment in the School of Electrical and Computer Engineering and a secondary appointment in the School of Biomedical Engineering. He is currently the editor-in-chief of the *IEEE Transactions on Image Processing* and a member of the Steering Committee of the *IEEE Transactions on Medical Imaging*. He has been an associate editor of the *IEEE Transactions on Image Processing* and the *IEEE Transactions on Pattern Analysis and Machine Intelligence*. He has also been cochair of the 2006 International Society for Optical Engineering/Society for Imaging, Science and Technology (SPIE/IS&T) Symposium on Electronic Imaging, cochair of the SPIE/IS&T conferences on Visual Communications, and Image Processing (VCIP) 2000, the vice president of Publications and a member of the Board of Directors for the IS&T Society. He is the founder and cochair of the SPIE/IS&T Conference on Computational Imaging. His research focuses on the use of statistical image models, multiscale techniques, and fast algorithms in applications including medical and electronic imaging. He is a fellow of the IEEE, a fellow of the American Institute for Medical and Biological Engineering (AIMBE), a fellow of IS&T, a member of the SPIE Professional Society. He is a recipient of IS&T's Raymond C. Bowman Award for outstanding contributions to digital imaging education and research and a University Faculty Scholar of Purdue University.