Statistical Learning of Visual Feature Hierarchies

Fabien Scalzo and Justus H. Piater*
Montefiore Institute
University of Liège
B-4000 Liège, Belgium

Abstract

We propose an unsupervised, probabilistic method for learning visual feature hierarchies. Starting from local, low-level features computed at interest point locations, the method combines these primitives into high-level abstractions. Our appearance-based learning method uses local statistical analysis between features and Expectation-Maximization (EM) to identify and code spatial correlations. Spatial correlation is asserted when two features tend to occur at the same relative position of each other. This learning scheme results in a graphical model that allows a probabilistic representation of a flexible visual feature hierarchy. For feature detection, evidence is propagated using Nonparametric Belief Propagation (NBP), a recent generalization of particle filtering. In experiments, the proposed approach demonstrates efficient learning and robust detection of object models in the presence of clutter and occlusion and under view point changes.

1. Introduction

Feature representation is one of the most important issues for learning and recognition applications in computer vision. In the present work, we propose a new approach to representing and learning visual feature hierarchies in an unsupervised manner. Our hierarchical representation is inspired by the compositional nature of objects. Most objects encountered in the world, which can be either man-made or natural objects, are composed of a number of distinct constituent parts (e.g., a face contains a nose and two eyes, a phone possesses a keypad). If we examine these parts, it becomes obvious that they are themselves recursively composed of other subparts (e.g., an eye contains an iris and eyelashes, a keypad is composed of buttons). This ubiquitous observation constitutes our main motivation for arguing that a hierarchical representation must be taken into account to model objects in more flexible and realistic way.

Our long-term goal is thus to learn visual feature hierarchies that correspond to object/part hierarchies. The development of a hierarchical and probabilistic framework that is tractable in terms of complexity is a central problem for many computer vision applications such as visual tracking, object recognition and categorization, face recognition, stereo matching and image retrieval. In this paper, the principal objective is to obtain a probabilistic framework that allows the organization of complex feature models. The main idea is to use a graphical model to represent the hierarchical feature structure. In this representation, which is detailed in Section 2, the nodes correspond to the visual features. The edges model both the spatial arrangement and the statistical dependence between nodes. The formulation in terms of graphical models is attractive because it provides a statistical model of the variability of shape and appearance. These are specified separately by the edges and the low-level nodes of the graph, respectively.

An unsupervised feature learning method that allows the construction of a hierarchy of visual features is introduced in Section 4. The proposed framework accumulates statistical evidence from feature observations in order to find "conspicuous coincidences" of visual feature co-occurrences. The structure of the graph is iteratively built by combining correlated features into higher-level abstractions. Our learning method is best explained by first presenting the detection process, which is described in Section 3.

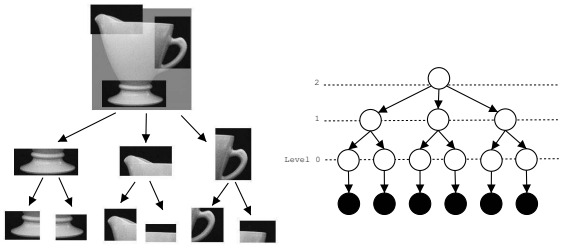

Figure 1: Object part decomposition (left) and corresponding graphical model (right). In our representation, each edge defines the relative location (defined by a distance and an orientation) from the subfeature to the center of the compound feature.

*This work was supported by the Wallonian Ministry of Research (D.G.T.R.E.) under Contract No. 03/1/5438.

For detection, we use Nonparametric Belief Propagation [7, 19], a message-passing algorithm, to propagate the observations in the graph, thus inferring the belief associated with higher-level features that are not directly observable. The functioning and the efficacy of our method are illustrated in Section 5. Finally, Section 6 provides a discussion of related work and biological analogies.

2. Representation

In this section, we introduce a new part-based and probabilistic representation of visual features. In the proposed graphical model, nodes represent visual features and are annotated with the detection information for a given scene. The edges represent two types of information: the relative spatial arrangement between features, and their hierarchical composition. We employ the term "visual feature" in two distinct contexts:

Primitive visual features are low-level features. They are represented by a local descriptor. For this work, we used simple descriptors constructed from normalized pixel values and located at Harris interest points [5], but our system does not rely on any particular feature detector. Any other feature detector can be used to detect and extract more robust information (such as SIFT keys [10]).

Compound visual features consist of flexible geometrical combinations of other subfeatures (primitive or compound features).

Formally, our graph $\mathcal{G}$ is a mathematical object made up of two sets: a vertex set $\mathcal{V}$, and a directed edge set $\overrightarrow{\mathcal{E}}$. For any node $s \in \mathcal{V}$, the set of parents and the set of children are respectively defined as $U(s) = \{u_i \in \mathcal{V} | (u_i, s) \in \overrightarrow{\mathcal{E}}\}$ and $C(s) = \{c_i \in \mathcal{V} | (s, c_i) \in \overrightarrow{\mathcal{E}}\}$. Information about feature types and their specific occurrences in an image will be represented in the graph by annotations of vertices and edges, as described next.

2.1. The Vertex Set

The vertices $s \in \mathcal{V}$ of the graph represent features. They contain the feature activations for a given image. Our graphical model associates each node with a hidden random variable $x \in \mathcal{R}^2$ representing the spatial probability distribution of the feature in the image. This random variable is continuous and is defined in a two-dimensional space where the dimensions $\mathcal{X}, \mathcal{Y}$ are the location in the image. For simplicity, we assume that the distribution of x can be approximated by a Gaussian mixture:

$$p(x; \Theta) = \sum_{i=1}^{N} w_i \mathcal{G}(x; \mu_i, \Sigma_i) \tag{1}$$

In order to retain information about feature orientation, we associate a mean orientation $\theta_i \in [0, 2\pi[$ to each component of the Gaussian mixture. All mixture components are collectively represented by the parameter vector $\Theta = (\mu_1, \ldots, \mu_N; \Sigma_1, \ldots, \Sigma_N; w_1, \ldots, w_N; \theta_1, \ldots, \theta_N)$.

Primitive feature classes lie at the first level of the graph. They are represented by a local descriptor and are associated with an observable random variable y, defined in the same two-dimensional space $(\mathcal{X}, \mathcal{Y})$ as the hidden variables x, and are likewise approximated by a Gaussian mixture.

2.2. The Edge Set

An edge $e \in \overrightarrow{\mathcal{E}}$ of the graph models two aspects of the feature structure. First, whenever an edge links two features it signifies that the destination feature is a part of the source feature. Second, an edge describes the spatial relation between the compound feature and its subfeature in terms of relative position. The annotation does not contain any specific information about a given scene but represents a static view of the feature hierarchy and shape.

Each edge $e = x_t, x_s$ is associated with a parameter $\mu_{st} = [\mu^x_{st}; \mu^y_{st}]$ that defines the relative, orientation-normalized, position of the compound feature x_t versus the other x_s.

We also define a potential function $\psi_{s,t}(x_s, x_t)$ that encodes the relative distribution of the compound feature with respect to the subfeature. The conditional density is approximated by a Gaussian mixture:

$$\psi_{s,t}(x_s, x_t) = \sum_{i=1}^{N_s} w^i_s \, \mathcal{G}(x_t; \vartheta_{s,t}(x^i_s), \Sigma^i_s) \tag{2}$$

where x^i_s is the i-th component of the mixture x_s, w^i_s is the component weight, ϑ is a function that computes the mean of the compound feature component x^i_t given the subfeature location μ^i_s and orientation θ^i_s:

$$\vartheta_{s,t}(x^{(i)}_s) = \mu^i_s + \begin{bmatrix} \cos(\theta^i_s) & -\sin(\theta^i_s) \\ \sin(\theta^i_s) & \cos(\theta^i_s) \end{bmatrix} \begin{bmatrix} \mu^x_{st} \\ \mu^y_{st} \end{bmatrix} \tag{3}$$

The use of Gaussian mixture models to represent spatial relations (Eq. 2) has been successfully applied in many works [17, 20]. It is especially useful for modeling relative position variability between features.

3. Nonparametric Belief Propagation

In the preceding section, we defined the structure of our graphical model used to represent visual features. We now describe the detection process of a visual feature, given its graphical-model representation. In our framework, inferring the probability density function $\hat{p}(x|y)$ of features conditioned on the primitives detected amounts to estimating the belief of the nodes.

Nonparametric Belief Propagation (NBP) [7, 19] is perfectly suitable for our continuous parameter space of each feature part and the non-Gaussian conditionals between nodes. NBP is an inference algorithm for graphical models that generalizes particle filtering. It has the advantage of allowing inference over general graphs and is not limited to Markov chains. The main idea of this inference algorithm is to propagate information via a series of local message-passing operations. Each message is represented using a sample-based density estimate which is defined nonparametrically as a Gaussian mixture. The conditional distribution (or belief) of each node is defined by the product of the incoming messages of the node. Due to the intractable complexity of an exact computation, this product must be approximated. Several techniques are possible. In this work, we use an efficient Gibbs sampling procedure with 300 samples for each message.

Following the conventional NBP notation, a message m_{ij} from node x_i to x_j is written

$$m_{i,j}(x_j) = \int \psi_{i,j}(x_i, x_j)\, \phi_i(x_i, y_i) \prod_{k \in \mathcal{N}_i \setminus j} m_{k,i}(x_i)\, dx_i$$

where $\mathcal{N}_i$ is the set of neighbors of node i, $\psi_{i,j}$ is the pairwise potential between nodes i, j, and $\phi_i(x_i, y_i)$ is the local likelihood associated with the node i and obtained via the observation function y_i of a low-level feature.

The message update algorithm is summarized in Algorithm 1 and illustrated in Figure 2. After a few iterations, depending on the size of the graph, each node i can produce an approximation of the conditional distribution $\hat{p}(x_i|y)$ using the product of incoming messages. For more details about Nonparametric Belief Propagation, consult the original research papers [7, 19].

Algorithm 1 Message update using NBP

Draw samples $\{\hat{x}_t^i, \hat{w}_t^i\}_{i=1}^M$ from the product of input messages $m_{kt}(x_t)$ using the Gibbs sampler,
Compute values for each outgoing message $m_{tu}(x_u)$
Means: $x_{tu}^i = \psi_{t,u}(\hat{x}_t^i, x_u^i)$
Orientations: θ_{tu}^i using $m_{kt}(x_t)$ (Eq. 4)

As we mentioned in Section 2.1, we associate an orientation to each sample. It is computed using the mean orientation of the incoming messages and local evidence at the location of the sample. More precisely, we compute the mean orientation $\overline{\theta}_{x^z}(l), z = 1, \ldots, M$ of the available mixture components weighted by their corresponding weights:

$$\begin{aligned} \tan \overline{\theta}_{x^z}(l) &= \frac{\mathcal{S}_{x^z}(l)}{\mathcal{C}_{x^z}(l)} \\ \mathcal{C}_x(l) &= \textstyle\sum_{i=1}^{N_x} v_i \times w_i \cos(\theta_x^i) \\ \mathcal{S}_x(l) &= \textstyle\sum_{i=1}^{N_x} v_i \times w_i \sin(\theta_x^i) \end{aligned} \tag{4}$$

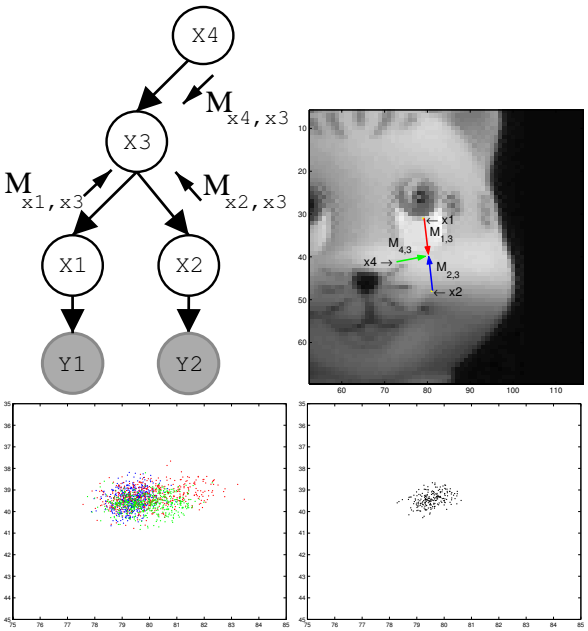

Figure 2: During an iteration of NBP, feature x_3 received messages from subfeatures x_1, x_2 and parent x_4. Even if individual messages may contain uncertain information about the location of feature (bottom left), the product of the incoming messages constrains the feature location (bottom right).

where l is a location of the sample, θ_x^i is the orientation of the i-th component of message x, N_x is the number of available components, $v_i = \mathcal{R}(l, x_i)$ is the response of component x_i at point l and w_i is the associated weight. This could have been integrated to NBP inference. To reduce the complexity and avoid the use of a linear-circular distribution, we compute it separately.

In our representation, higher-level features are not directly observable. Evidence is incorporated into the graph exclusively via the variables y_i representing primitives at the leaves of the graph as described next.

To initialize the observable variables y_i of the graph and to compute the local evidence $\phi_i(x_i, y_i)$, we match the detected local descriptors in an image with the most similar existing feature descriptor at the leaf nodes of the available graphs. We use the observed descriptor parameters (location, orientation, similarity) to annotate the variable y_i of the corresponding node $i \in \mathcal{V}$.

We illustrate the message-passing algorithm by presenting in Figure 3 the NBP detection process on an object. The object model was learned during our experiments. The product of incoming messages clearly allows a more precise localization of the features.

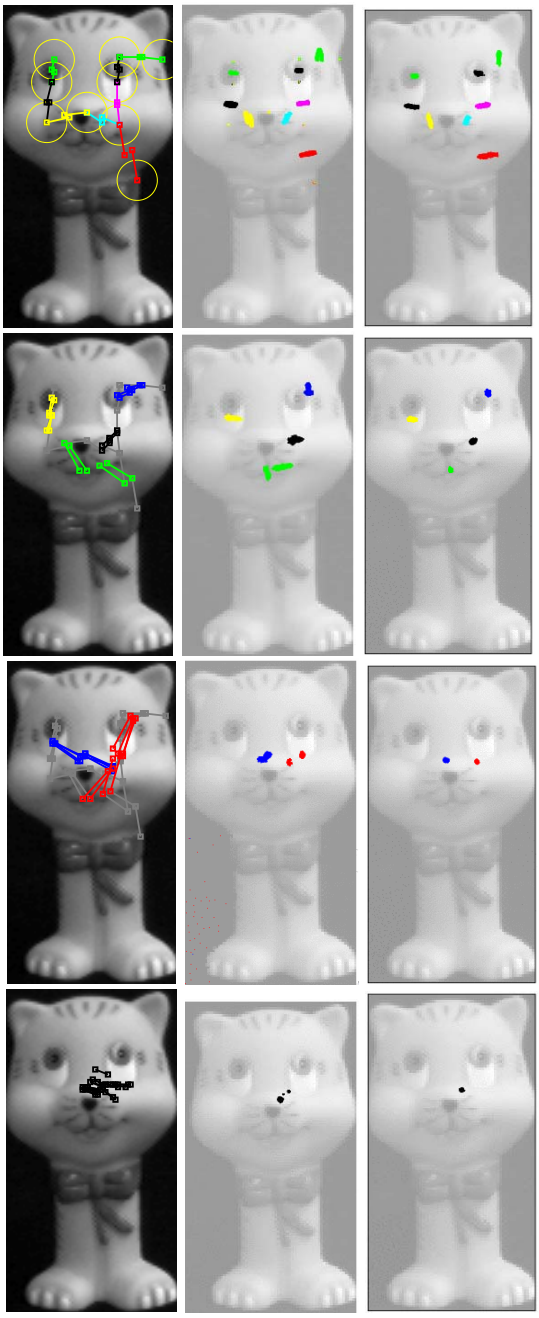

Figure 3: Illustration of an upward message-passing iteration during the NBP detection process. Starting from the first level (top left), the detection process uses the presence of primitives to predict the location of the second level features (top center). The product of these messages (on the right) refines the belief to a more precise localization. This product is then used for the next level. The geometrical model for each level is shown on the left. At the end of this simple example, we obtain the final set of samples that corresponds to the localization of the object model.

4. Visual Feature Learning

In this section, we introduce our unsupervised feature learning method that allows the construction of a hierarchy of visual features. The general idea of our algorithm is to accumulate statistical evidence from the relative positions of observed features in order to find "conspicuous coincidences" of visual feature co-occurrences. The structure of our graphical model is iteratively built by combining correlated features into new visual feature abstractions. First, the learning process votes to accumulate information on the relative position of features and extracts the feature pairs that tend to be located in the same neighborhood. Second, it estimates the parameters of the geometrical relations using either Expectation-Maximization (EM) or a voting scheme. It finally creates new feature nodes in the graph by combining spatially correlated features. In the following sections, we describe the three main steps of this unsupervised learning procedure.

4.1. Spatial Correlation

The objective of this first step of our learning process is to find spatially correlated features. A spatial correlation exists between two features if they are often detected in the same neighborhood. The size of the neighborhood to consider is set proportionally to the space covered by the feature parts. Co-occurrence statistics are collected from multiple feature occurrences within one or across many different images. The procedure to find correlated features is summarized in Algorithm 2. After its completion, we obtain a vote array $\mathcal{S}$ concerning the relative locations of correlated features. It is important to notice that before the first iteration we apply K-means clustering to the set of feature descriptors. This identifies primitive classes from the training set and is used to create the first level of the graphical model.

Algorithm 2 Spatial Correlation Extraction

Successively extract each image I from the training set
Apply NBP (Section 3) to detect all features $f_I = \{f_{i_0} \dots f_{i_n}\} \in \mathcal{G}$ in image I
for each pair $[f_i, f_j]$ where f_j is observed in the neighborhood of f_i **do**
  Compute the relative position $p_r \in \mathcal{R}^2$ of f_j given f_i
  Vote for the corresponding observation $[f_i, f_j, p_r]$
end for
Keep all pairs $[f_i, f_j]$ where $\sum_{p_r} \mathcal{S}[f_i, f_j, p_r] > t_c$

4.2. Spatial Relations

In our framework, spatial relations are defined in terms of relative position between features. We implemented two solutions to estimate this parameter. The first method uses the

Expectation-Maximization (EM) algorithm, and the second implements a fast discrete voting scheme to find location evidence. The estimated geometrical relations are used during feature generation (Section 4.3) in order to create new features. First, however, we give some details on both methods for the estimation of spatial relations.

4.2.1 Expectation-Maximization

In principle, a sample of observed spatial relations x_r between two given features can be approximated by a Gaussian mixture, where each component k represents a cluster of relative positions μ_k of one of the two features f_j with respect to the other, the *reference feature* f_i: $f(x_r;\Theta) = \sum_{k=1}^{K} w_k \mathcal{G}_k(x_r;(\mu_k,\Sigma_k))$.

EM can be used to estimate the parameters of the spatial relation between each correlated feature pair $[f_i, f_j] \in \mathcal{S}$. It maximizes the likelihood of the observed spatial relations over the model parameters $\Theta = (w_{1\ldots K}; \mu_{1\ldots K}; \Sigma_{1\ldots K})$. The Expectation (E) and Maximization (M) steps of each iteration of the algorithm are defined as follows:

Step E Compute the current expected values of the component indicators $t_k(x_i)$, $1 \le i \le n$, $1 \le k \le K$, where n is the number of observations, K is the number of components and q is the current iteration:

$$t_k^{(q)}(x_i) = \frac{\hat{w}_k^{(q)} \mathcal{G}\left(x_i; \hat{\mu}_k^{(q)}, \hat{\Sigma}_k^{(q)}\right)}{\sum_{l=1}^{K} \hat{w}_l^{(q)} \mathcal{G}\left(x_i; \hat{\mu}_l^{(q)}, \hat{\Sigma}_l^{(q)}\right)}$$

Step M Determine the value of parameters Θ^{q+1} containing the estimates $\hat{w}_k, \hat{\mu}_k, \hat{\Sigma}_k$ that maximize the likelihood of the data $\{x\}$ given the $t_k(x_i)$:

$$\hat{w}_k^{(q+1)} = \frac{\sum_{i=1}^{n} t_k^{(q)}}{n} \qquad \hat{\mu}_k^{(q+1)} = \frac{\sum_{i=1}^{n} t_k^{(q)}(x_i)}{\sum_{i=1}^{n} t_k^{(q)}}$$

$$\hat{\Sigma}_k^{(q+1)} = \frac{\sum_{i=1}^{n} t_k^{(q)} \left(x_i - \hat{\mu}_k^{(q+1)}\right)\left((x_i - \hat{\mu}_k^{(q+1)}\right)^T}{\sum_{i=1}^{n} t_k^{(q)}}$$

In our implementation, a mixture of only two Gaussian components ($K = 2$) is used to model spatial relations. The first component represents the most probable relative position, and the second is used to model the noise. When the location μ_1 and variance Σ_1 of the first component is estimated between features i and j, we store the corresponding information $[f_i, f_j, \mu_{i,j}, \Sigma_{i,j}]$ in a table $\mathcal{T}$.

4.2.2 Voting

A faster method to estimate spatial-relation parameters is to discretize distance and direction between features. The idea is to create a bi-dimensional histogram for every correlated feature pair $[f_i, f_j] \in \mathcal{S}$. The dimensions of these histograms are the distance d and the relative direction θ from features f_i to f_j. Each observation $[f_i, f_j, p_r]$ stored in table $\mathcal{S}$ is projected into a cylindrical space $[d, \theta]$ and votes for the corresponding entry $[d, \theta]$ of histogram $\mathcal{H}[f_i, f_j]$. After the completion of this voting procedure, we look for significant local maxima in the 2D histograms and store them in the table $\mathcal{T}$. In our implementation, the distances are expressed relative to the part size and are discretized into 36 bins, while the directions are discretized into 72 bins (5-degree precision).

Algorithm 3 Discrete Voting

for each vote $v_{i_{1\ldots n}}$ of entry $[f_i, f_j, p_r]$ in table $\mathcal{S}$ **do**
  Project $p_r(v_i) \in \mathcal{R}^2$ into a cylindrical space of the relative orientation $\theta(v_i)$ and the distance $d(v_i)$
  Vote for the Gaussian kernel at $d(v_i), \theta(v_i)$ in histogram $\mathcal{H}[f_i(v_i), f_j(v_i)]$
end for
Find maxima $[d, \theta]$ in all histograms and store corresponding relative positions $\mu_{i,j}$ in $\mathcal{T}$

4.3. Feature generation

When a reliable reciprocal spatial correlation is detected between two features $[f_i, f_j]$, the generation of a new feature in our model is straightforward. We combine these features to create a new higher-level feature by adding a new node f_n in the graph. We connect it to its subfeatures $[f_i, f_j]$ by two edges e_i, e_j that are added to $\overrightarrow{\mathcal{E}}$. Their parameters are computed using the spatial relation $\{\mu_{i,j}, \mu_{j,i}\}$ obtained from the preceding step, and are stored in table $\mathcal{T}$.

The generated feature is located at the midpoint between the subfeatures. Thus the distance from subfeatures to the new feature is set to half distance between subfeatures $[f_i, f_j]$; $\mu_1 = \mu_{i,j}/2$, $\mu_2 = \mu_{j,i}/2$ and is copied to the new edges; $e_i(f_i, f_n) = \{\mu_1, \Sigma_1\}$, $e_j(f_j, f_n) = \{\mu_2, \Sigma_2\}$.

5. Experiments

In this section, we illustrate our feature learning scheme on an object recognition task using several objects of the Columbia University Object Image Library (COIL-100) [12]. This library is very commonly used in object recognition research and contains color images of 100 different objects. Each image was captured by a fixed camera at pose intervals of 5 degrees. For our experiments, we used 5 views around both sides of the frontal pose of an object to build the object model. When the learning process is completed, the model is thus tuned for a given view of the object.

As we mentioned before, our system does not depend on any particular feature detector. We used Harris interest points and rotation invariant descriptors comprising 256 pixel values. Any other feature detector can be used to detect and extract more robust information (such as SIFT keys [10]). We deliberately used simple features to demonstrate the functioning of our method. To estimate the primitives of each object model, we used K-Means to cluster the feature space. The number of classes (between 16 and 60 in our experiments) was selected according to the BIC criterion [16].

During learning, in order to avoid excessive growth of the graph due to the feature combinatorics, we only kept the most salient spatial relations between features. In Figure 5, for each object, the first level of the learned graphical model is illustrated on the top center image. Each circle corresponds to a low-level feature and is associated to another one to create a new compound feature. On the bottom of each image, we illustrate the detection results of our models with NBP after only six iterations. During our experiments, each message is sampled with 300 values. Two graphical models are illustrated in Figure 8. They correspond to the first two objects appearing in Figure 5. We observe that some low-level features are connected to two compound features because they are involved in two different spatial relations.

A quantitative detection evaluation is presented in Figure 4. The first graph (on the left) illustrates the viewpoint invariance of the five object models presented in Figure 5. To generate the graph, we ran the detection process on a series of images differing in viewing angle by increments of 5 degrees. The models responded maximally around the training views. We observe that the response quickly falls at $\pm$ 40 degrees. This is caused by the loss of pertinent features in the view. We obtained an average viewpoint invariance over 80 degrees. These results are remarkable considering the fact that we did not use affine-invariant features at the leaf level. The second graph (right) demonstrates the convergence of the detection process using NBP. This graph was generated by measuring the uncertainty in the graph across 23 message-passing iterations. In our experiments, we observe that NBP converges to the optimal solution in less than 7 iterations. In general, the number of iterations required for convergence depends on the number of levels in the graph.

We also demonstrate the robustness of our model in a cluttered scene (Figure 6) and in the presence of occlusion (Figure 7). The first example contains two different learned object models. As we explained in Section 3, the detection process starts with low-level feature detection and then propagate evidence in graph. Many primitives are detected in the image. In this experiment, we add extra difficulty by assigning every interest point to the most similar feature class, without requiring a minimum degree of similarity (left of Figure 6). This results in noisy detection data. However, the use of geometric relations to infer the presence of higher-level features allows an unambiguous detection. As we can see for the objects of the Figure 6, only a few features are needed to detect the objects. In the second example (Figure 7), NBP correctly infers the localization of the occluded features.

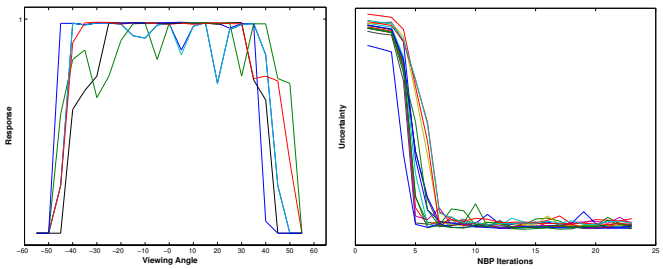

Figure 4: Maximum response of the five object models presented in Figure 5 on a series of images differing in viewing angle (left). Convergence of NBP to the optimal solution during the detection on different views of the objects (right).

6. Discussion

During the past years, great attention has been paid to unsupervised model learning methods applied to object models [13]. In theses techniques, objects are represented by parts, each modeled in terms of both shape and appearance by Gaussian probability density functions. This concept, which originally operated in batch mode, has been improved by introducing incremental learning [6] and by introducing variational Bayesian methods incorporating information obtained from previously learned models [3]. In parallel, Agarwal *et al.* [1] presented a method for learning to detect objects that is based on sparse, part-based representations. The main limitation of these schemes lies in the representation because it only contains two levels, the features and the models.

A hierarchical model of recognition in the visual cortex was proposed by Riesenhuber and Poggio [15] where spatial combinations of view-tuned units are constructed by alternately employing a maximum and sum pooling operation. Wersing and Körner [21] used a similar hierarchy for object recognition where invariance is achieved through pooling over increasing receptive fields. Unfortunately, in such models there exists no explicit representation of object structure.

A scheme for constructing visual features by hierarchical composition of primitive features [14] has been introduced. It uses an incremental procedure for learning discriminative composite features using a Bayesian network classifier. In this framework, the spatial arrangement of primitives was rigid and limits the robustness of the system.

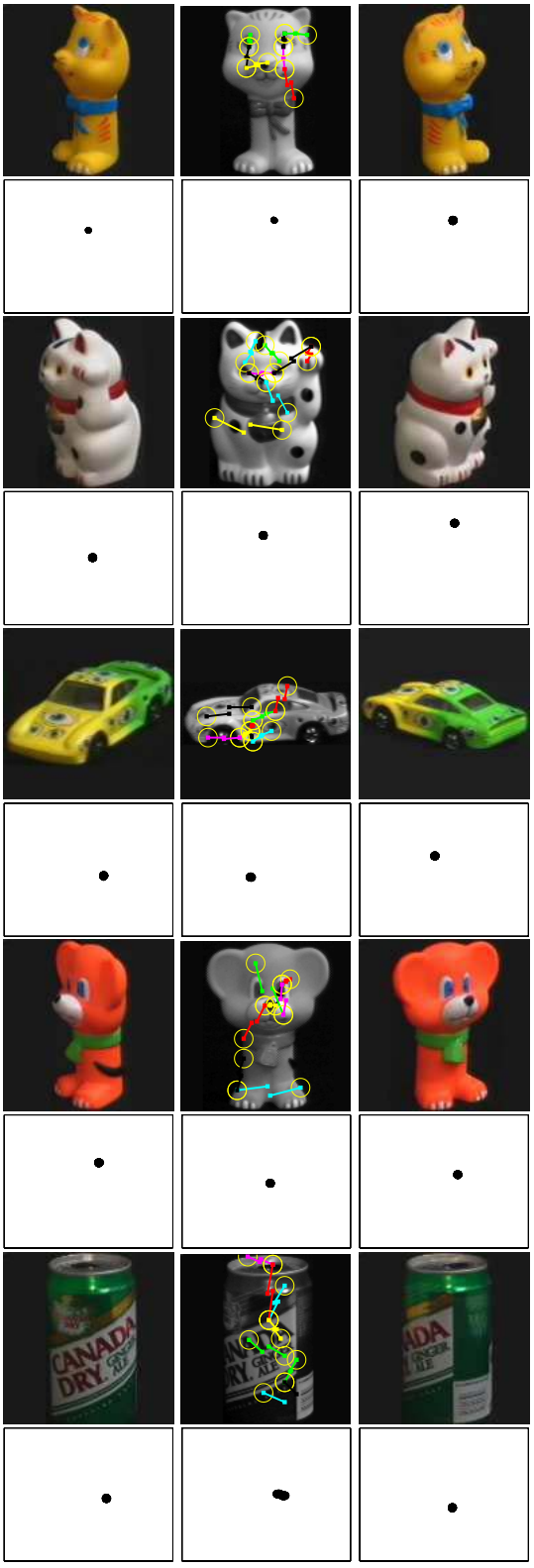

Figure 5: Detection results on a series of COIL-100 objects. The geometrical models between low-level features are shown in the top center images. The detection results with NBP are presented in the bottom images.

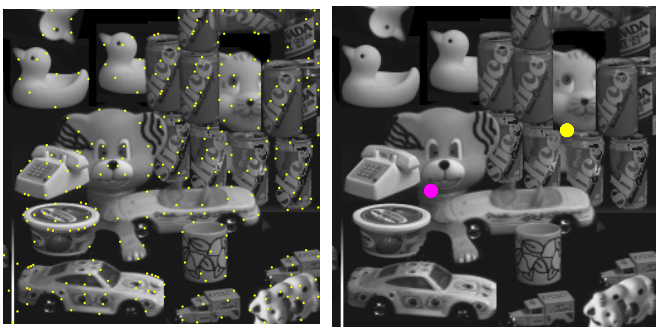

Figure 6: Cluttered scene containing two previously learned objects. Low-level features detected in the scene (left) and detection responses for both object models (right).

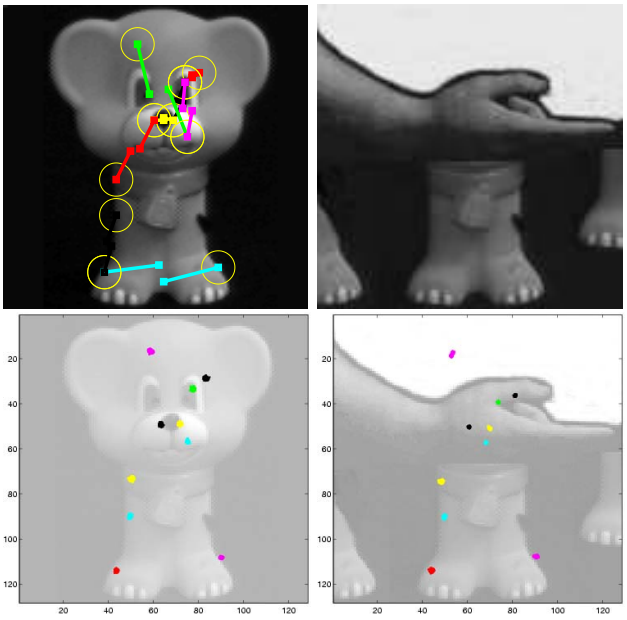

Figure 7: Spatial relations between the features of the first level are presented in the top left image. The results of the detection process after 6 iterations can be observed (for the first level) in the bottom images. For the occluded object, NBP correctly inferred the presence of missing first level features.

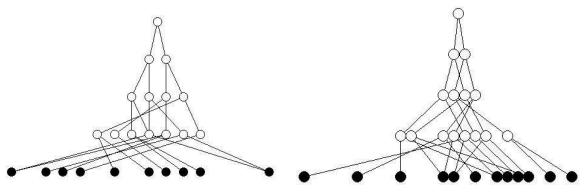

Figure 8: Two graphical models learned on five different views of the first two objects presented in Figure 5.

Recently, Felzenszwalb *et al.* [4] and Kumar *et al.* [8] presented two different frameworks for modeling objects both inspired by pictorial structure models. These frameworks have some similarities to our work in the sense that objects are modeled by different parts arranged in a graphical model. The appearance and shape of each part are also modeled separately. These techniques provide a good tool to represent articulated structures. However, Felzenszwalb's framework requires labeled example images to learn the models and both schemes are also limited to a single layer of features.

In neuroscience, recent evidence [2] reinforces the idea that the coding of geometrical relations in high-level visual features is essential. Moreover, recent work suggests that the visual cortex represents objects hierarchically by parts or subfeatures [9].

Even in the context of image structure analysis, hierarchical and probabilistic models such as HIP [18] demonstrate encouraging results. However, in the case of object recognition, it is not clear how to integrate rotation or translation invariance in such models.

The framework presented in this paper offers several significant improvements over current methods proposed in the literature. Taking advantage of graphical models, we represent shape and appearance separately. This allows us to deal with shape deformation and appearance variability. Moreover, our topology can deal with minor viewpoint changes without explicitly using affine-invariant features. Our scheme is also invariant to rotation and translation of the object. Scale invariance can easily be obtained by using scale invariant features [11] and by normalizing the distance between features with the intrinsic feature scale.

Finally, the use of Nonparametric Belief Propagation allows a robust and flexible detection of higher-level visual features. The proposed system spans the entire visual hierarchy from low-level features to object-level abstractions within a single, coherent framework. Learning occurs at all levels. We argue that a hierarchical model is essential for representing visual features. This appears to be a necessary step towards more flexible learning methods applied to object categories.

Future research will focus on two main directions: the unsupervised discovery of visual categories and the integration of the hierarchical model in a task-oriented learning environment. Many other applications are possible such as face recognition, object tracking, stereo matching and image retrieval.

References

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, volume 4, pages 113–130, 2002.

[2] E.E. Cooper and T.J. Wojan. Differences in the coding of spatial relations in face identification and basic-level object recognition. In *J. of Exp. Psychology*, volume 26, pages 470–488, 2000.

[3] L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *ICCV*, pages 1134–1141, 2003.

[4] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *Intl. Journal of Computer Vision*, 61(1):55–79, January 2005.

[5] C. Harris and M. Stephens. A combined corner and edge detector. In *ALVEY vision conference*, pages 147–151, 1988.

[6] S. Helmer and D. G. Lowe. Object recognition with many local features. In *Workshop on Generative Model Based Vision (GMBV)*, Washington, D.C., July 2004.

[7] M. Isard. Pampas: Real-valued graphical models for computer vision. In *CVPR*, pages 613–620, 2003.

[8] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Extending pictorial structures for object recognition. In *Proceedings of the British Machine Vision Conference*, 2004.

[9] Y. Lerner, T. Hendler, D. Ben-Bashat, M. Harel, and R. Malach. A hierarchical axis of object processing stages in the human visual cortex. In *Cerebral Cortex*, volume 4, pages 287–97, 2001.

[10] D.G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

[11] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[12] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100), 1996.

[13] P. Perona, R. Fergus, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume 2, page 264, Madison, Wisconsin, June 2003.

[14] J. H. Piater and R. A. Grupen. Distinctive features should be learned. In *BMCV*, pages 52–61, 2000.

[15] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. In *Nature Neuroscience*, volume 2, pages 1019–1025, 1999.

[16] G. Schwartz. Estimating the dimension of a model. *Ann. Stat.*, 6(2):461–464, 1978.

[17] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. Tracking loose-limbed people. In *CVPR*, pages 421–428, 2004.

[18] C. Spence, L. C. Parra, and P. Sajda. Hierarchical image probability (hip) models. In *ICIP*, 2000.

[19] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. In *CVPR*, pages 605–612, 2003.

[20] V. Vogelhuber and C. Schmid. Face detection based on generic local descriptors and spatial constraints. In *ICPR*, volume 1, pages 1084–1087, 2000.

[21] H. Wersing and E. Koerner. Unsupervised learning of combination features for hierarchical recognition models. In *ICANN*, pages 1225–1230, 2002.