International Journal of Computer Vision 61(6), 55–79, 2005

Pictorial Structures for Object Recognition

Au: Mismatch file disk followed.

PEDRO F. FELZENSZWALB
Artificial Intelligence Lab, Massachusetts Institute of Technology
pff@ai.mit.edu

DANIEL P. HUTTENLOCHER
Computer Science Department, Cornell University
dph@cs.cornell.edu

Received November 1, 2002; Revised February 12, 2004; Accepted March 25, 2004

First online version published in September, 2004

Abstract. In this paper we present a computationally efficient framework for part-based modeling and recognition of objects. Our work is motivated by the pictorial structure models introduced by Fischler and Elschlager. The basic idea is to represent an object by a collection of parts arranged in a deformable configuration. The appearance of each part is modeled separately, and the deformable configuration is represented by spring-like connections between pairs of parts. These models allow for qualitative descriptions of visual appearance, and are suitable for generic recognition problems. We address the problem of using pictorial structure models to find instances of an object in an image as well as the problem of learning an object model from training examples, presenting efficient algorithms in both cases. We demonstrate the techniques by learning models that represent faces and human bodies and using the resulting models to locate the corresponding objects in novel images.

Keywords: part-based object recognition, statistical models, energy minimization

1. Introduction

Research in object recognition is increasingly concerned with the ability to recognize generic classes of objects rather than just specific instances. In this paper, we consider both the problem of recognizing objects using generic part-based models and the problem of learning such models from example images. Our work is motivated by the pictorial structure representation introduced by Fischler and Elschlager (1973) thirty years ago, where an object is modeled by a collection of parts arranged in a deformable configuration. Each part encodes local visual properties of the object, and the deformable configuration is characterized by spring-like connections between certain pairs of parts. The best match of such a model to an image is found by minimizing an energy function that measures both a match cost for each part and a deformation cost for each pair of connected parts.

While the pictorial structure formulation is appealing in its simplicity and generality, several shortcomings have limited its use: (i) the resulting energy minimization problem is hard to solve efficiently, (ii) the model has many parameters, and (iii) it is often desirable to find more than a single best (minimum energy) match. In this paper we address these limitations, providing techniques that are practical for a broad range of object recognition problems. We illustrate the method for two quite different generic recognition tasks, finding faces and finding people. For faces, the parts are features such as the eyes, nose and mouth, and the spring-like connections allow for variation in the relative

(a) (b)

Figure 1. Sample results for detection of a face (a); and a human body (b). Each image shows the globally best location for the corresponding object, as computed by our algorithms. The object models were learned from training examples.

locations of these features. For people, the parts are the limbs, torso and head, and the spring-like connections allow for articulation at the joints. Matching results with these two models are illustrated in Fig. 1

The main contributions of this paper are three-fold. First, we provide an efficient algorithm for the classical pictorial structure energy minimization problem described in Fischler and Elschlager (1973), for the case where the connections between parts do not form any cycles and are of a particular (but quite general) type. Many objects, including faces, people and animals can be represented by such acyclic multi-part models. Second, we introduce a method for learning these models from training examples. This method learns all the model parameters, including the structure of connections between parts. Third, we develop techniques for finding multiple good hypotheses for the location of an object in an image rather than just a single best solution. Finding multiple hypotheses is important for tasks where there may be several instances of an object in an image, as well as for cases where imprecision in the model may result in the desired match not being the one with the minimum energy. We address the problems of learning models from examples and of hypothesizing multiple matches by expressing the pictorial structure framework in a statistical setting.

1.1. Pictorial Structures

A pictorial structure model for an object is given by a collection of parts with connections between certain pairs of parts. The framework is quite general, in the sense that it is independent of the specific scheme used to model the appearance of each part as well as the type of connections between parts. A natural way to express such a model is in terms of an undirected graph $G = (V, E)$, where the vertices $V = \{v_1, \dots, v_n\}$ correspond to the n parts, and there is an edge $(v_i, v_j) \in E$ for each pair of connected parts v_i and v_j. An instance of the object is given by a configuration $L = (l_1, \dots, l_n)$, where each l_i specifies the location of part v_i. Sometimes we refer to L simply as the object location, but "configuration" emphasizes the part-based representation. The location of each part can simply specify its position in the image, but more complex parameterizations are also possible. For example, for the person model in Section 6 the location of a part specifies a position, orientation and an amount of foreshortening.

In Fischler and Elschlager (1973) the problem of matching a pictorial structure to an image is defined in terms of an energy function to be minimized. The cost or energy of a particular configuration depends both on how well each part matches the image data at its location, and how well the relative locations of the parts agree with the deformable model. Given an image, let $m_i(l_i)$ be a function measuring the degree of mismatch when part v_i is placed at location l_i in the image. For a given pair of connected parts let $d_{ij}(l_i, l_j)$ be a function measuring the degree of deformation of the model when part v_i is placed at location l_i and part v_j is placed at location l_j. Then an optimal match of the model to the image is naturally defined as

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i, v_j) \in E} d_{ij}(l_i, l_j) \right), \qquad (1)$$

which is a configuration minimizing the sum of the match costs m_i for each part and the deformation costs d_{ij} for connected pairs of parts. Generally the deformation costs are only a function of the relative position of one part with respect to another, making the model invariant to certain global transformations. Note that matching a pictorial structure model to an image does not involve making any initial decisions about locations of individual parts, rather an overall decision is made based on both the part match costs and the deformation costs together.

This energy function is simple and makes intuitive sense. However, previous methods have used heuristics or local search techniques that do not find an optimal solution and depend on having good initialization. In contrast we present an efficient algorithm that can find a global minimum of the energy function without any initialization.

Pictorial structures can be used to represent quite generic objects. For example, the appearance models for the individual parts can be a blob of some color and orientation, or capture the response of local oriented filters. The connections between parts can encode generic relationships such as "close to", "to the left of", or more precise geometrical constraints such as ideal joint angles. Since both the part models and the relationships between parts can be generic, pictorial structures provide a powerful framework. Suppose we want to model the appearance of the human body. It makes sense to represent the body as an articulated object, with joints connecting different body parts. With pictorial structures we can use a coarse model, consisting of a small number of parts connected by flexible joints. The combination of simple appearance models for the parts and structural relations between parts provides sufficient context to find the human body as a whole, even when it would be difficult to find generic parts such as "lower-leg" or "upper-arm" on their own.

1.2. *Efficient Algorithms*

Our primary goal is to take the pictorial structure framework, and use it to efficiently solve object recognition and model learning problems. We consider a natural class of pictorial structure models and present efficient algorithms both for matching such models to images and for learning models from examples. These efficient algorithms are based on two restrictions on the form of the pictorial structure models. First, our methods require that the graph G be acyclic (i.e., form a tree). Second the methods require that the relationships between connected pairs of parts be expressed in a particular form.

Restricting the connections between parts to a tree structure is natural for many classes of objects. For example, the connections between parts of many animate objects can form a tree corresponding to their skeletal structure. Many other kinds of objects can be represented using a tree such as a star-graph, where there is one central part to which all the other parts are connected. When the graph G is a tree it is possible to compute the best match of the model to an image in polynomial time. This is done using a generalization of the Viterbi algorithm (Rabiner and Juang, 1993). Related methods are known in the Bayesian Network community as belief propagation algorithms (Pearl, 1988). The fastest such polynomial time algorithms run in $O(h^2 n)$ time, where n is the number of object parts, and h is a discrete number of possible locations for each part. Unfortunately this is too slow in most cases because the number of possible locations for a single part is usually quite large – in the hundreds of thousands or millions.

The restriction that we impose on the form of connections between parts enables an improvement in the running time of the matching algorithms so that it becomes essentially linear rather than quadratic in the number of possible locations for each part. We require that $d_{ij}(l_i, l_j)$ be a Mahalanobis distance between transformed locations,

$$d_{ij}(l_i, l_j) = (T_{ij}(l_i) - T_{ji}(l_j))^T M_{ij}^{-1} (T_{ij}(l_i) - T_{ji}(l_j)), \tag{2}$$

The matrix M_{ij} should be diagonal, and for simplicity we will assume that T_{ij}, and T_{ji} are one-to-one. We further require that it be possible to represent the set of possible transformed locations $T_{ij}(l_i)$ and $T_{ji}(l_j)$ as positions in a grid. These functions capture the ideal relative locations for parts v_i and v_j. The distance between the transformed locations, weighted by M_{ij}^{-1}, measures the deformation of a "spring" connecting the two parts. This special form for the deformation costs allows for matching algorithms that run in time linear in the number of grid positions of the transformed space. Often this is the same as the number of possible locations for each part, but sometimes it may be slightly larger. As we will see, a broad class of interesting relationships can be represented in this form, including those illustrated in Sections 5 and 6.

The asymptotic running time of the matching algorithms that we develop is thus nearly optimal, in the sense that the methods run in essentially the same asymptotic time as simply matching each part to the image separately, without accounting for the connections between them. In practice, the algorithms are also quite fast, finding the globally best match of a pictorial structure to an image in just a few seconds using a desktop computer.

1.3. Statistical Formulation

In their original work, Fischler and Elschlager only considered the problem of finding the best match of a pictorial structure model to an image. As discussed above, they characterized this problem using the energy function in Eq. (1). While this energy function intuitively makes sense, it has many free parameters. For each different object, one has to construct a model, which includes picking appearance parameters for each part, a set of edges connecting pairs of parts and the characteristics of the connections. We are interested in automatically learning these parameters from examples. Moreover, the energy minimization formulation only characterizes the problem of finding the best match of a model to an image, whereas it is often desirable to find multiple good potential matches.

These questions are naturally addressed using a statistical framework for pictorial structure models which we describe in Section 2. In this framework, the energy minimization problem introduced by Fischler and Elschlager is equivalent to finding the maximum a posteriori estimate of the object configuration given an observed image. The statistical formulation can be used to learn the parameters of a model from examples. In fact, *all* model parameters can be learned from a few training examples using maximum likelihood estimation. This is of practical as well as theoretical interest, since it is generally not possible to find the best parameters for a deformable model by trial and error.

The statistical framework also provides a natural way of finding several good matches of a model to an image rather than finding just the best one. The idea is to consider primarily good matches without considering many bad ones. We can achieve this by sampling object configurations from their posterior probability distribution given an observed image. Sampling makes it possible to find many locations for which the posterior is high, and to subsequently select one or more of those using an independent method. This procedure lets us use imprecise models for generating hypotheses and can be seen as a mechanism for visual selection (see Amit and Geman, 1999).

1.4. Related Work

Research in object recognition has been dominated by approaches that separate processing into distinct stages of feature extraction and matching. In the first stage, discrete primitives, or “features” are detected. In the second stage, stored models are matched against those features. For instance, in the pioneering work of Roberts (1965) children’s blocks were recognized by first extracting edges and corners from images and then matching these features to polyhedral models of the blocks. The model-based recognition paradigm of the 1980’s similarly followed this approach. These methods focus largely on the problem of efficiently searching for correspondences between features that have been extracted from an image, and features of a stored model. Examples include interpretation tree search (Ayache and Faugeras, 1986; Grimson and Lozano-Perez, 1987), the alignment method (Huttenlocher and Ullman, 1990), RANSAC (Fischler and Bolles, 1981) and geometric hashing (Lamdan et al., 1990).

Limitations of the simple features used by most earlier model-based recognition techniques led to a quite different class of recognition methods, developed in the 1990’s, which operate directly on images rather than first extracting discrete features. These include both appearance-based methods (e.g., Turk and Pentland, 1991; Murase and Nayar, 1995) and template-based methods such as Hausdorff matching (Huttenlocher et al., 1993). Such approaches treat images as the entities to be recognized, rather than having more abstract models based on features or other primitives. One or more training images of an object are used to form a “template” that is used as a model. This model is then compared to new images to determine whether or not the target is present, generally by explicitly considering possible transformations of the template.

The matching of pictorial structures is an alternative approach that in many ways combines the appearance-based and geometric techniques. The energy minimization problem associated with these models as defined in Eq. (1) incorporates match costs for the individual parts and deformation costs for the geometric configuration into a single overall problem. Thus the approach provides a means of simultaneously using appearance and geometry, rather than first making binary decisions about the possible locations of parts or features. The main drawback of the pictorial structures approach has been the computational difficulty of the energy minimization problem, which we address here for a class of models.

There have been other part-based recognition methods, which like the pictorial structures approach are based on separately modeling the appearance of individual parts and the geometric relations between them. However most of these part-based methods make binary decisions about potential part locations (e.g., Pentland, 1987; Dickinson et al., 1993; Rivlin et al., 1995; Burl and Perona, 1996). Moreover, most part-based methods use some kind of search heuristics, such as first matching a particular "distinctive" part and then searching for other parts given that initial match, in order to avoid the combinatorial explosion of the configuration space. Such heuristics make it difficult to handle occlusion, particularly for those parts that are considered first in the search.

In Burl et al. (1998) models similar to pictorial structures were used to represent objects in terms of a constellation of local features. In these models, rather than there being connections between pairs of parts, all the parts are constrained with respect to a central coordinate system using a Gaussian distribution. Like the pictorial structures formulation, no binary decisions are made about part or feature locations. These models, however, are not well suited for representing articulated objects, as a joint Gaussian distribution cannot capture multiple articulation points. Moreover, in Burl et al. (1998) the matching algorithms use heuristics that don't necessarily find the optimal match of a model to an image.

The problem of finding people in images using coarse part-based two-dimensional models was considered in Ioffe and Forsyth (2001). This is one of two domains that we use to illustrate the pictorial structures approach. Two different methods are reported in Ioffe and Forsyth (2001). The first method makes binary decisions about the possible locations for individual parts and subsequently searches for groups of parts that match the overall model. The second method uses sequential importance sampling (particle filtering) to generate increasingly larger configurations of parts. We also describe a sampling-based technique, however rather than employing approximate distributions obtained via sequential importance sampling, our method is based on efficiently computing the *exact* (discrete) posterior distribution for the object configuration and then sampling from that posterior.

In illustrating the pictorial structures approach using the problem of finding people in images we employ simple part models based on binary images obtained by background subtraction. This suggests comparisons with silhouette-based deformable matching techniques (e.g., Gdalyahu and Weinshall, 1999; Sebastian et al., 2001). These approaches are quite different, however. First of all, silhouette-based methods generally operate using boundary contours, requiring good segmentation of the object from the background. In contrast, the models we use are not based on a boundary representation and operate directly on binary images. For example, a single part could match a region of the image that has several disconnected components. Secondly, deformable matching methods are generally based on two-dimensional shape representations rather than highly parameterized models. Thus they do not apply to cases such as an articulated body where in some configurations the parts can cross one another yielding vastly different shapes.

Finally we note that models similar to pictorial structures have recently been used for tracking people by matching models at each frame (Ramanan and Forsyth, 2003). In contrast, most work on tracking highly articulated objects such as people relies heavily on motion information (Bregler and Malik, 1998; Ju et al., 1996) and only performs incremental updates in the object configuration. In such approaches, some other method is used to find an initial match of the model to the image, and then tracking commences from that initial condition. Pictorial structures can be used to solve this track initialization problem, or as demonstrated in Ramanan and Forsyth (2003) can be used as a tracking method on their own.

2. Statistical Framework

As noted in the introduction, the pictorial structure energy minimization problem can be viewed in terms of statistical estimation. The statistical framework described here is useful for addressing two of the three questions that we consider in this paper, that of learning pictorial structure models from examples and that of finding multiple good matches of a model to an image. For the third question, that of efficiently minimizing the energy in Eq. (1), the statistical formulation provides relatively little insight, however it unifies the three questions in a common framework.

A standard way of approaching object recognition in a statistical setting is as follows. Let θ be a set of

P1: xxx
International Journal of Computer Vision KL3162-04/5277531 September 6, 2004 22:24

parameters that define an object model, I denote an image, and as before let L denote a configuration of the object (a location for each part). The distribution $p(I \mid L, \theta)$ captures the imaging process, and measures the likelihood of seeing a particular image given that an object is at some location. The distribution $p(L \mid \theta)$ measures the prior probability that an object is at a particular location. Finally, the posterior distribution, $p(L \mid I, \theta)$, characterizes the probability that the object configuration is L given the model θ and the image I. Using Bayes' rule the posterior can be written as,

$$p(L \mid I, \theta) \propto p(I \mid L, \theta) p(L \mid \theta) . \tag{3}$$

A common drawback of the Bayesian formulation is the difficulty of determining a prior distribution, $p(L \mid \theta)$, that is both informative and generally applicable. For instance, a uniform prior is general but provides no information. On the other hand a prior which says that the object is in the lower left corner of the image is highly informative but of little use in general. For pictorial structures, the prior over configurations encodes information about the *relative* positions of the parts, which can be both informative and general. For instance, for a human body model such a prior can capture which are likely relative orientations of two connected limbs.

A number of interesting problems can be characterized in terms of this statistical framework,

- MAP estimation—this is the problem of finding a location L with maximum posterior probability. In some sense, the MAP estimate is our best guess for the location of the object. In our framework this will be equivalent to the energy minimization problem defined by Eq. (1).
- Sampling from the posterior—sampling provides a natural way to hypothesize many good potential matches of a model to an image, rather than just finding the best one. This is useful to detect multiple instances of an object in an image and to find possible locations of an object with an imprecise model.
- Model estimation—this is the problem of finding θ which specifies a good model for a particular object. The statistical framework allows us to learn the model parameters from training examples using maximum likelihood estimation.

Our pictorial structure models are parametrized by $\theta = (u, E, c)$, where $u = \{u_1, \dots, u_n\}$ are appearance parameters, the set of edges E indicates which parts are connected, and $c = \{c_{ij} \mid (v_i, v_j) \in E\}$ are connection parameters. There is a separate appearance model for each part, but the exact method used to model the appearance of parts is not important at this point. In Section 5 we model appearance using image derivatives around a point, to represent local features of a face such as the tip of the nose or the corners of the mouth. In Section 6 we model appearance using rectangular shapes, to represent individual body parts. In practice, the appearance modeling scheme just needs to provide a distribution $p(I \mid l_i, u_i)$ up to a normalizing constant, which measures the likelihood of seeing a particular image, given that a part with appearance parameters u_i is at location l_i. This distribution does not have to be a precise generative model, an approximate measure is good enough in practice.

We model the likelihood of seeing an image given that the object is at some configuration by the product of the individual likelihoods,

$$p(I \mid L, \theta) = p(I \mid L, u) \propto \prod_{i=1}^{n} p(I \mid l_i, u_i). \tag{4}$$

This approximation is good if the parts do not overlap, as in this case they generate different portions of the image. But the approximation can be bad if one part occludes another. For the iconic models described in Section 5 the prior distribution over configurations enforces that the parts do not overlap (the probability of a configuration with overlap is very small). For the articulated models described in Section 6 there is much less constraint on the locations of parts, and parts can easily overlap. In this case we demonstrate that a good estimate of the object configuration can be found by obtaining multiple samples from the posterior distribution and then selecting one of them using an independent method. This shows that sampling from the posterior can be useful for handling modeling error.

The prior distribution over object configurations is captured by a tree-structured Markov random field with edge set E. In general, the joint distribution for a tree-structured prior can be expressed as,

$$p(L \mid \theta) = \frac{\prod_{(v_i, v_j) \in E} p(l_i, l_j \mid \theta)}{\prod_{v_i \in V} p(l_i \mid \theta)^{\deg v_i - 1}},$$

where $\deg v_i$ is the degree of vertex v_i in the graph defined by E. We do not model any preference over the absolute location of each part, only over their relative configuration. This means that $p(l_i \mid \theta)$ is constant, and we let it equal one for simplicity. The joint distributions

P1: xxx
International Journal of Computer Vision KL3162-04/5277531 September 6, 2004 22:24

for pairs of parts connected by edges are characterized by the parameters $c = \{c_{ij} \mid (v_i, v_j) \in E\}$. Since we let $p(l_i \mid \theta) = 1$, the prior distribution over object configurations is given by,

$$p(L \mid \theta) = p(L \mid E, c) = \prod_{(v_i, v_j) \in E} p(l_i, l_j \mid c_{ij}). \quad (5)$$

Note that both $p(l_i, l_j \mid c_{ij})$ and $p(L \mid E, c)$ are improper priors (see Berger, 1985). This is a consequence of using an uninformative prior over absolute locations for each part.

In Eq. (4) we defined the form of $p(I \mid L, \theta)$, the likelihood of seeing an image given that the object is at a some configuration, and in Eq. (5) we defined the form of $p(L \mid \theta)$, the prior probability that the object would assume a particular configuration. These can be substituted into Eq. (3) yielding,

$$P(L \mid I, \theta) \propto \left(\prod_{i=1}^{n} p(I \mid l_i, u_i) \prod_{(v_i, v_j) \in E} p(l_i, l_j \mid c_{ij}) \right).$$

Taking the negative logarithm of this equation yields the same energy function that is being minimized in Eq. (1), where $m_i(l_i) = -\log p(I \mid l_i, u_i)$ is a match cost measuring how well part v_i matches the image data at location l_i, and $d_{ij}(l_i, l_j) = -\log p(l_i, l_j \mid c_{ij})$ is a deformation cost measuring how well the relative locations for v_i and v_j agree with the prior model. Thus we see that the MAP estimation problem for the statistical models introduced in this section is equivalent to the original energy minimization problem for pictorial structures described in Fischler and Elschlager (1973).

As discussed in the introduction our efficient algorithms require that the deformation costs be expressed in a particular form as shown in Eq. (2). This requirement has a natural interpretation in terms of the statistical models. Since $d_{ij}(l_i, l_j) = -\log p(l_i, l_j \mid c_{ij})$, it is equivalent to assume that the joint prior distribution for the locations of a pair of connected parts is given by a Gaussian over the displacement between transformed locations,

$$p(l_i, l_j \mid c_{ij}) \propto \mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij}), \quad (6)$$

where T_{ij}, T_{ji}, and D_{ij} are the connection parameters encoded by c_{ij}. These parameters correspond to the ones in Eq. (2) where $D_{ij} = M_{ij}/2$ is a diagonal covariance matrix.

3. Learning Model Parameters

Suppose we are given a set of example images $\{I^1, \ldots, I^m\}$ and corresponding object configurations $\{L^1, \ldots, L^m\}$ for each image. We want to use the training examples to obtain estimates for the model parameters $\theta = (u, E, c)$, where $u = \{u_1, \ldots, u_n\}$ are the appearance parameters for each part, E is the set of connections between parts, and $c = \{c_{ij} \mid (v_i, v_j) \in E\}$ are the connection parameters. The maximum likelihood (ML) estimate of θ is, by definition, the value θ^* that maximizes

$$p(I^1, \ldots, I^m, L^1, \ldots, L^m \mid \theta) = \prod_{k=1}^{m} p(I^k, L^k \mid \theta),$$

where the right hand side is obtained by assuming that each example was generated independently. Since $p(I, L \mid \theta) = p(I \mid L, \theta) p(L \mid \theta)$, the ML estimate is

$$\theta^* = \arg\max_{\theta} \prod_{k=1}^{m} p(I^k \mid L^k, \theta) \prod_{k=1}^{m} p(L^k \mid \theta). \quad (7)$$

The first term in this equation depends only on the appearance of the parts, while the second term depends only on the set of connections and connection parameters. Below we show that one can independently solve for the appearance models of the individual parts and the structural model given by the connections and their parameters. As a consequence, any kind of part models can be used in this framework as long as there is a maximum likelihood estimation procedure for learning the model parameters for a single part from examples. We use quite simple part models in this paper because our focus is on developing a general framework and providing efficient algorithms that can be used with many different modeling schemes.

3.1. *Estimating the Appearance Parameters*

From Eq. (7) we get

$$u^* = \arg\max_{u} \prod_{k=1}^{m} p(I^k \mid L^k, u).$$

The likelihood of seeing image I^k, given the configuration L^k for the object is given by Eq. (4). Thus,

$$u^* = \arg\max_{u} \prod_{k=1}^{m} \prod_{i=1}^{n} p\left(I^k \mid l_i^k, u_i\right) = \arg\max_{u} \prod_{i=1}^{n} \prod_{k=1}^{m} p\left(I^k \mid l_i^k, u_i\right).$$

Looking at the right hand side we see that to find u^* we can independently solve for the u_i^*,

$$u_i^* = \arg\max_{u_i} \prod_{k=1}^{m} p\big(I^k \mid l_i^k, u_i\big).$$

This is exactly the ML estimate of the appearance parameters for part v_i, given independent examples $\{(I^1, l_i^1), \ldots, (I^m, l_i^m)\}$. Solving for u_i^* depends on picking a specific modeling scheme for the parts, and we return to this in Sections 5 and 6.

3.2. *Estimating the Dependencies*

From Eq. (7) we get

$$E^*, c^* = \arg\max_{E,c} \prod_{k=1}^{m} p(L^k \mid E, c). \tag{8}$$

We need to pick a set of edges that form a tree and the connection parameters for each edge. This can be done in a similar way to the algorithm of Chow and Liu (1968), which estimates a tree distribution for discrete random variables. Eq. (5) defines the prior probability of the object assuming configuration L^k as,

$$p(L^k \mid E, c) = \prod_{(v_i,v_j)\in E} p\big(l_i^k, l_j^k \mid c_{ij}\big).$$

Plugging this into Eq. (8) and re-ordering the factors we get,

$$E^*, c^* = \arg\max_{E,c} \prod_{(v_i,v_j)\in E} \prod_{k=1}^{m} p\big(l_i^k, l_j^k \mid c_{ij}\big). \tag{9}$$

We can estimate the parameters for each possible connection independently, even before we know which connections will actually be in E as,

$$c_{ij}^* = \arg\max_{c_{ij}} \prod_{k=1}^{m} p\big(l_i^k, l_j^k \mid c_{ij}\big).$$

This is the ML estimate for the joint distribution of l_i and l_j, given independent examples $\{(l_i^1, l_j^1), \ldots, (l_i^m, l_j^m)\}$. Solving for c_{ij}^* depends on picking a specific representation for the joint distributions. Independent of the exact form of $p(l_i, l_j \mid c_{ij})$, and how to compute c_{ij}^* (which we consider later, as it varies with different modeling schemes), we can characterize the "quality" of a connection between two parts as the probability of the examples under the ML estimate for their joint distribution,

$$q(v_i, v_j) = \prod_{k=1}^{m} p\big(l_i^k, l_j^k \mid c_{ij}^*\big).$$

Intuitively, the quality of a connection between two parts measures the extent to which their locations are related. These quantities can be used to estimate the connection set E^* as follows. We know that E^* should form a tree, and according to Eq. (9) we let,

$$\begin{aligned} E^* &= \arg\max_{E} \prod_{(v_i,v_j)\in E} q(v_i, v_j) \\ &= \arg\min_{E} \sum_{(v_i,v_j)\in E} -\log q(v_i, v_j). \end{aligned} \tag{10}$$

The right hand side is obtained by taking the negative logarithm of the quantity being maximized (and thus finding the argument minimizing the value, instead of maximizing it). Solving for E^* is equivalent to the problem of computing the minimum spanning tree (MST) of a graph. We build a complete graph on the vertices V, and associate a weight $-\log q(v_i, v_j)$ with each edge (v_i, v_j). The MST of this graph is the tree with minimum total weight, which is exactly the set of edges defined by Eq. (10). The MST problem is well known (see Cormen et al., 1996) and can be solved efficiently. Kruskal's algorithm can be used to compute the MST in $O(n^2 \log n)$ time, since we have a complete graph with n nodes.

4. Matching Algorithms

In this section we present two efficient algorithms for matching tree-structured models to images with connections of the form in Eqs. (2) and (6). The first algorithm solves the energy minimization problem in Eq. (1), which in the statistical framework is equivalent to finding the MAP estimate of the object location given an observed image. The second algorithm samples configurations from the posterior distribution. In Felzenszwalb and Huttenlocher (2000) we described a version of the energy minimization algorithm that uses a different restriction on the form of connections between parts. That form did not allow for efficient sampling from the posterior distribution.

P1: xxx
International Journal of Computer Vision KL3162-04/5277531 September 6, 2004 22:24

4.1. *Energy Minimization or MAP Estimate*

As discussed in Section 1.1, the problem of finding the best match of a pictorial structure model to an image is defined by the following equation,

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i, l_j) \right).$$

The form of this minimization is quite general, and it appears in a number of problems in computer vision, including MAP estimation of Markov random fields for low-level vision such as image restoration and stereo and optimization of active contour models (snakes). While the form of the minimization is shared with these other problems, the structure of the graph and space of possible solutions differ substantially. This changes the computational nature of the problem.

Solving this minimization for arbitrary graphs and arbitrary functions m_i, d_{ij} is an NP-hard problem (see Boykov et al., 2001). However, when the graph $G = (V, E)$ has a restricted form, the problem can be solved more efficiently. For instance, with first-order snakes the graph is simply a chain, which enables a dynamic programming solution that takes $O(h^2 n)$ time (see Amini et al., 1990), where as before we use n to denote the number of parts in the model and h is a discrete number of possible locations for each part. Moreover, with snakes the minimization is done over a small number of locations for each vertex (e.g., the current location plus the 8 neighbors on the image grid). This minimization is then iterated until the change in energy is small. The key to an efficient algorithm for snakes is that the number of possible locations for each part, h, is small in each iteration, as the dynamic programming solution is quadratic in this value. Another source of efficient algorithms has been in restricting d_{ij} to a particular form. This approach has been particularly fruitful in some recent work on MRFs for low-level vision (Boykov et al., 2001; Ishikawa and Geiger, 1998). Here we use constraints on both the structure of the graph and the form of d_{ij}.

By restricting the graphs to trees, a similar kind of dynamic programming can be applied as is done for chains, making the minimization problem polynomial rather than exponential time. The precise technique is described in Section 4.1.1. However, this $O(h^2 n)$ algorithm is not practical in most cases, because for pictorial structures the number of possible locations for each part is usually huge.

Recall our restricted form for d_{ij} shown in Eq. (2) in terms of a Mahalanobis distance between transformed locations,

$$d_{ij}(l_i, l_j) = (T_{ij}(l_i) - T_{ji}(l_j))^T M_{ij}^{-1} (T_{ij}(l_i) - T_{ji}(l_j)).$$

We will show how this restriction can be used to obtain a minimization algorithm that runs in $O(h'n)$ rather than $O(h^2 n)$ time, where h' is the number of grid locations in a discretization of the space of transformed locations given by T_{ij} and T_{ji}. The relationship between h' and h depends on the particular transformations being used, but in most cases the two quantities have similar value. This makes it quite practical to compute a *globally optimal match* of a pictorial structure model to an image, up to the discretization of the possible locations. We first discuss the overall minimization problem for tree-structured models and then turn to the method that exploits the form of d_{ij}.

4.1.1. Efficient Minimization. In this section, we describe an algorithm for finding a configuration $L^* = (l_1^*, \ldots, l_n^*)$ that minimizes Eq. (1) when the graph G is a tree, which is based on the well known Viterbi recurrence. Given $G = (V, E)$, let $v_r \in V$ be an arbitrarily chosen root vertex (this choice does not affect the results). From this root, each vertex $v_i \in V$ has a depth d_i which is the number of edges between it and v_r (and the depth of v_r is 0). The children, C_i, of vertex v_i are those neighboring vertices, if any, of depth $(d_i + 1)$. Every vertex v_i other than the root has a unique parent, which is the neighboring vertex of depth $(d_i - 1)$.

For any vertex v_j with no children (i.e., any leaf of the rooted tree), the best location l_j^* for that vertex can be computed as a function of the location of just its parent, v_i. The only edge incident on v_j is (v_i, v_j), thus the only contribution of l_j to the energy in (1) is $m_j(l_j) + d_{ij}(l_i, l_j)$. The quality of the best location for v_j given location l_i for v_i is

$$B_j(l_i) = \min_{l_j}(m_j(l_j) + d_{ij}(l_i, l_j)), \tag{11}$$

and the best location for v_j as a function of l_i can be obtained by replacing the min in the equation above with arg min.

For any vertex v_j other than the root, assume that the function $B_c(l_j)$ is known for each child $v_c \in C_j$. That is, the quality of the best location for each child is known with respect to the location of v_j. Then the quality of the best location for v_j given a location for its parent v_i is

$$B_j(l_i) = \min_{l_j} \left(m_j(l_j) + d_{ij}(l_i, l_j) + \sum_{v_c \in C_j} B_c(l_j) \right). \tag{12}$$

Again, the best location for v_j as a function of l_i can be obtained by replacing the min in the equation above with arg min. This equation subsumes (11) because for a leaf node the sum over its children is simply empty. Finally, for the root v_r, if $B_c(l_r)$ is known for each child $v_c \in C_r$ then the best location for the root is

$$l_r^* = \arg\min_{l_r} \left(m_r(l_r) + \sum_{v_c \in C_r} B_c(l_j) \right).$$

That is, the minimization in (1) can be expressed recursively in terms of the $(n-1)$ functions $B_j(l_i)$ for each vertex $v_j \in V$ (other than the root). These recursive equations suggest a simple algorithm. Let d be the maximum depth in the tree. For each node v_j with depth d, compute $B_j(l_i)$, where v_i is the parent of v_j. These are all leaf nodes, so clearly $B_j(l_i)$ can be computed as in (11). Next, for each node v_j with depth $(d-1)$ compute $B_j(l_i)$, where again v_i is the parent of v_j. Clearly, $B_c(l_j)$ has been computed for every child v_c of v_j, because the children have depth d. Thus $B_j(l_i)$ can be computed as in (12). Continue in this manner, decreasing the depth until reaching the root at depth zero. Besides computing each B_j we also compute B'_j, which indicates the best location of v_j as a function of its parent location (obtained by replacing the min in B_j with arg min). At this point, we compute the optimal location l_r^* for the root. The optimal location L^* for all the parts can be computed by tracing back from the root to each leaf. We know the optimal location of each node given the location of its parent, and the optimal location of each parent is now known starting from the root.

The overall running time of this algorithm is $O(Hn)$, where H reflects the time required to compute each $B_j(l_i)$ and $B'_j(l_i)$. In the general case this takes $O(h^2)$ time as it is necessary to consider every location of a child node for each possible location of the parent. In the next section, we show how to compute each $B_j(l_i)$ and $B'_j(l_i)$ more efficiently when d_{ij} is restricted to be in the form of Eq. (2).

4.1.2. Generalized Distance Transforms.

Traditional distance transforms are defined for sets of points on a grid. Suppose we have a grid $\mathcal{G}$, and $\rho(x, y)$ is some measure of distance between points on the grid. Given a point set $B \subseteq \mathcal{G}$, the distance transform of B specifies for each location in the grid, the distance to the closest point in the set,

$$\mathcal{D}_B(x) = \min_{y \in B} \rho(x, y).$$

In particular, $\mathcal{D}_B$ is zero at any point in B, and is small at nearby locations. The distance transform is commonly used for matching edge based models (see Borgefors, 1988; Huttenlocher et al., 1993). The trivial way to compute this function takes $O(k|B|)$ time, where k is the number of locations in the grid. On the other hand, efficient algorithms exist to compute the distance transform in $O(k)$ time, independent of the number of points in B (see Borgefors, 1986; Karzanov, 1992). These algorithms have small constants and are very fast in practice. In order to compute the distance transform, it is commonly expressed as

$$\mathcal{D}_B(x) = \min_{y \in \mathcal{G}} (\rho(x, y) + 1_B(y)),$$

where $1_B(y)$ is an indicator function for membership in the set B, that has value 0 when $y \in B$ and ∞ otherwise. This suggests a generalization of distance transforms where the indicator function is replaced with some arbitrary function over the grid $\mathcal{G}$,

$$\mathcal{D}_f(x) = \min_{y \in \mathcal{G}} (\rho(x, y) + f(y)).$$

Intuitively, for each grid location x, the transform finds a location y that is close to x and for which $f(y)$ is small. Note that if there is a location where $f(x)$ has a small value, $\mathcal{D}_f$ will have small value at x and nearby locations.

With the restricted form of d_{ij} in Eq. (2), the functions $B_j(l_i)$ that must be computed by the dynamic programming algorithm can be rewritten as generalized distance transforms, where the distance in the grid, $\rho(x, y)$, is given by the Mahalanobis distance defined by M_{ij},

$$B_j(l_i) = \mathcal{D}_f(T_{ij}(l_i)),$$

where

$$f(y) = \begin{cases} m_j\big(T_{ji}^{-1}(y)\big) + \sum_{v_c \in C_j} B_c\big(T_{ji}^{-1}(y)\big) & \text{if } y \in \text{range}(T_{ji}) \\ \infty & \text{otherwise} \end{cases}$$

The grid $\mathcal{G}$ specifies a discrete set of possible values for $T_{ji}(l_j)$ that are considered during the minimization. This in turn specifies a discrete set of locations l_j. There is an approximation being made, since the set of discrete values for $T_{ji}(l_j)$ (the locations in the grid) might not match the set of discrete values for $T_{ij}(l_i)$ (where we need the value of $\mathcal{D}_f$). We can simply define the value of the distance transform at a non-grid position to be the value of the closest grid point. The error introduced by this approximation is small (as the transform by definition changes slowly).

The same algorithms that efficiently compute the classical distance transform can be used to compute the generalized distance transform under different distances, by replacing the indicator function $1_B(x)$ with an arbitrary function $f(x)$. In particular we use the method of Karzanov (originally in Karzanov, 1992, but see Rucklidge, 1996) for a better description) to compute the transform of a function under a Mahalanobis distance with diagonal covariance matrix. This algorithm can also compute $B'_j(l_i)$, the best location for v_j as a function of its parent location, as it computes $B_j(l_i)$.

4.2. *Sampling from the Posterior*

We now turn to the problem of sampling from the posterior distribution of object configurations. The sampling problem can be solved with a very similar algorithm to the one described in the previous section. The relationship between the two cases is analogous to the relationship between the forward-backward and the Viterbi algorithms for hidden Markov models. Basically the sampling algorithm works directly with the probability distributions instead of their negative logarithms, and the maximizations in the recursive equations are replaced by summations.

As we saw in Section 2 the posterior distribution for our models is given by

$$p(L \mid I, \theta) \propto \left(\prod_{i=1}^{n} p(I \mid l_i, u_i) \prod_{(v_i, v_j) \in E} p(l_i, l_j \mid c_{ij}) \right).$$

Like before, let $v_r \in V$ be an arbitrarily chosen root vertex, and the children of v_i be C_i. The algorithm works by first computing $p(l_r \mid I, \theta)$. We then sample a location for the root from that distribution. Next we sample a location for each child, v_c, of the root from $p(l_c \mid l_r, I, \theta)$. We can continue in this manner until we have sampled a location for each part. The marginal distribution for the root location is,

$$p(l_r \mid I, \theta) \propto \sum_{l_1} \cdots \sum_{l_{r-1}} \sum_{l_{r+1}} \cdots \sum_{l_n} \times \left(\prod_{i=1}^{n} p(I \mid l_i, u_i) \prod_{(v_i, v_j) \in E} p(l_i, l_j \mid c_{ij}) \right).$$

Computing the distribution in this form would take exponential time. But since the set of dependencies between parts form a tree, we can rewrite the distribution as,

$$p(l_r \mid I, \theta) \propto p(I \mid l_r, u_r) \prod_{v_c \in C_r} S_c(l_r).$$

The functions $S_j(l_i)$ are similar to the $B_j(l_i)$ we used for the energy minimization algorithm,

$$S_j(l_i) \propto \sum_{l_j} \left(p(I \mid l_j, u_j) p(l_i, l_j \mid c_{ij}) \prod_{v_c \in C_j} S_c(l_j) \right). \tag{13}$$

These recursive functions already give a polynomial algorithm to compute $p(l_r \mid I, \theta)$ up to a normalizing constant. As in the energy minimization algorithm we can compute the S functions starting from the leaf vertices. The trivial way to compute each $S_j(l_i)$ takes $O(h^2)$ time. For each location l_i we evaluate the function by explicitly summing over all possible locations l_j. We will show how to compute each $S_j(l_i)$ more efficiently for the case where $p(l_i, l_j \mid c_{ij})$ is in the special form given by Eq. (6). But first let's see what we need to do after we sample a location for the root from its marginal distribution. If we have a location for the parent v_i of v_j we can write,

$$p(l_j \mid l_i, I, \theta) \propto p(I \mid l_j, u_j) p(l_i, l_j \mid c_{ij}) \prod_{v_c \in C_j} S_c(l_j). \tag{14}$$

If we have already computed the S functions we can compute this distribution in $O(h)$ time. So once we have sampled a location for the root, we can sample a location for each of its children. Next we sample a location for the nodes at the third level of the tree, and so on until we sample a location for every part. Note that if we want to sample multiple times we only need to compute the S functions once. And when the location of a parent node is fixed, we only need to compute the distribution in (14) for locations of the children where $p(l_i, l_j \mid c_{ij})$ is not too small. So sampling multiple times is not much more costly than sampling once.

4.2.1. Computing the S Functions. We want to efficiently compute the function in Eq. (13). We will do this by writing the function as a Gaussian convolution in the transformed space of locations given by T_{ij} and T_{ji}. Using the special form of $p(l_i, l_j \mid c_{ij})$ we can write,

$$S_j(l_i) \propto \sum_{l_j} \left(\mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij})\; p(I \mid l_j, u_j) \prod_{v_c \in C_j} S_c(l_j) \right).$$

This can be seen as a Gaussian convolution in the transformed space:

$$S_j(l_i) \propto (F \otimes f)\,(T_{ij}(l_i)),$$

where F is a Gaussian filter with covariance D_{ij}, $\otimes$ is the convolution operator, and

$$f(y) = \begin{cases} p(I \mid T_{ji}^{-1}(y), u_j) \displaystyle\prod_{v_c \in C_j} S_c(T_{ji}^{-1}(y)) & \text{if } y \in \text{range}(T_{ji}) \\ 0 & \text{otherwise} \end{cases}$$

Just like when computing the generalized distance transform, the convolution is done over a discrete grid which specifies possible values for $T_{ji}(l_j)$. The Gaussian filter F is separable since the covariance matrix is diagonal. We can compute a good approximation for the convolution in time linear in h', the set of grid locations, using the techniques from Wells, III (1986). This gives an overall $O(h'n)$ time algorithm for sampling a configuration from the posterior distribution.

5. Iconic Models

The framework presented so far is general in the sense that it doesn't fully specify how objects are represented. A particular modeling scheme must define the pose space for the object parts, the form of the appearance model for each part, and the type of connections between parts. In this section we describe models that represent objects by the appearance of local image patches and spatial relationships between those patches. This type of model has been popular in the context of face detection (see Fischler and Elschlager, 1973; Burl et al., 1998). We first describe how we model the appearance of a part, and later describe how we model spatial relationships between parts. Learning an iconic model involves picking labeled landmarks on a number of instances of the target object. From these training examples both the appearance models for each part and the spatial relationships between parts are automatically estimated, using the procedure described in Section 3. In Section 5.3 we show some experiments with face detection.

5.1. Parts

In this class of models the location of a part is specified by its (x, y) position in the image, so we have a two-dimensional pose space for each part. To model the appearance of each individual part we use the iconic representation introduced in Rao and Ballard (1995). The iconic representation is based on the response of Gaussian derivative filters of different orders, orientations and scales. An image patch centered at some position is represented by a high-dimensional vector that collects all the responses of a set of filters at that point. This vector is normalized and called the iconic index at that position. Figure 2 shows the nine filters used to build the iconic representation at a fixed scale. In practice, we use three scales, given by $\sigma_1 = 1$, $\sigma_2 = 2$, and $\sigma_3 = 4$, the standard deviations of the Gaussian filters. So we get a 27 dimensional vector. The iconic index is fairly insensitive to changes in lighting conditions. For example, it is invariant to gain and bias. Invariance to bias is a consequence of using image derivative filters,

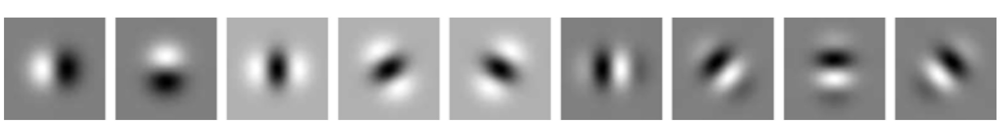

Figure 2. Gaussian derivative basis functions used in the iconic representation.

and the normalization provides the invariance to gain. Iconic indices are also relatively insensitive to small changes in scale and other image deformations. They can also be made invariant to image rotation, although we use an orientation-sensitive representation here.

The appearance of a part is modeled by a distribution over iconic indices. Specifically, we model the distribution of iconic indices at the location of a part as a Gaussian with diagonal covariance matrix. Using a diagonal covariance matrix makes it possible to estimate the distribution with a small number of examples. If many examples are available, a full Gaussian or even more complex distributions such as a mixture of Gaussians, or a non-parametric estimate could be used. Under the Gaussian model, the appearance parameters for each part are $u_i = (\mu_i, \Sigma_i)$, a mean vector and a covariance matrix. We have,

$$p(I \mid l_i, u_i) \propto \mathcal{N}(\alpha(l_i), \mu_i, \Sigma_i),$$

where $\alpha(l_i)$ is the iconic index at location l_i in the image. We can easily estimate the maximum likelihood parameters of this distribution, as required by the learning technique in Section 3, using the mean and covariance of the iconic indices corresponding to the positive examples of a particular part.

Note that we could use other methods to represent the appearance of image patches. In particular, we experimented with the eigenspace techniques from Moghaddam and Pentland (1997). With a small number of training examples the eigenspace methods are no better than the iconic representation, and the iconic representation can be computed more efficiently. In fact, the iconic representation can be computed very fast by convolving each level of a Gaussian pyramid with small x-y separable filters (see Freeman and Adelson, 1991).

5.2. *Spatial Relations*

The spatial configuration of the parts is modeled by a collection of springs connecting pairs of parts. Each connection (v_i, v_j) is characterized by the ideal relative location of the two connected parts s_{ij}, and a full covariance matrix Σ_{ij} which in some sense corresponds to the stiffness of the spring connecting the two parts. So the connection parameters are $c_{ij} = (s_{ij}, \Sigma_{ij})$. We model the distribution of the relative location of part v_i with respect to the location of part v_j as a Gaussian with mean s_{ij} and covariance Σ_{ij},

$$p(l_i, l_j \mid c_{ij}) = \mathcal{N}(l_i - l_j, s_{ij}, \Sigma_{ij}). \tag{15}$$

So, ideally the location of part v_i is the location of part v_j shifted by s_{ij}. Since the models are deformable, the location of v_i can vary by paying a cost that depends on the covariance matrix. This corresponds to stretching the spring. Because we have a full covariance matrix, stretching in different directions can have different costs. For example, two parts can be highly constrained to be at the same vertical position, while their relative horizontal position may be uncertain. As with the appearance models for the individual parts, the maximum likelihood parameters of these spatial distributions for pairs of parts can easily be estimated using training examples.

In practice, we need to write the joint distribution of l_i and l_j in the specific form required by our algorithms. It must be a Gaussian distribution with zero mean and diagonal covariance in a transformed space. To do this, we first compute the singular value decomposition of the covariance matrix $\Sigma_{ij} = U_{ij} D_{ij} U_{ij}^T$. Now the following transformations can be defined,

$$T_{ij}(l_i) = U_{ij}^T (l_i - s_{ij}), \quad \text{and} \quad T_{ji}(l_j) = U_{ij}^T (l_j),$$

which allow us to write Eq. (15) in the correct form,

$$p(l_i, l_j \mid c_{ij}) = \mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij}).$$

5.3. *Experiments*

To test the iconic modes just described we used the ML estimation procedure from Section 3 to train a model of frontal faces, and the MAP estimation technique from Section 4.1 to detect faces in novel images. Our first model has five parts, corresponding to the eyes, nose, and corners of the mouth. To generate training examples we labeled the location of each part in twenty different images (from the Yale face database). More training examples were automatically generated by scaling and rotating each training image by a small amount. This makes our model handle some variation in orientation and scale. Some of the training examples and the structure of the learned model are shown in Fig. 3. Remember that we never told the system which pairs of parts should be connected together. Determining the structure is part of the ML parameter estimation procedure.

We tested the resulting model by matching it to novel images using the energy minimization algorithm for finding the MAP estimate of the object location. Note

Figure 3. Three examples from the first training set showing the locations of the labeled features and the structure of the learned model.

that *all* model parameters were automatically estimated with the maximum likelihood procedure. Thus, there are no "knobs" to tune in the matching algorithm. Some matching results are shown in Fig. 4. Both the learning and matching algorithms are extremely fast. Using a desktop computer it took a few seconds to learn the model and less than a second to compute the MAP estimate in each image. These experiments demonstrate

Figure 4. Matching results.

Figure 5. Matching results on occluded faces. The top row shows some input images and the bottom row shows the corresponding matching results. The MAP estimate was a good match when the faces had up to two of five parts occluded and incorrect when three parts were occluded.

that we can learn a useful model from training examples.

Figure 5 illustrates matching results on images with partially occluded faces. The matching algorithm automatically handles such partial occlusion in a robust way, finding a good configuration of all the parts when up to two of the five parts are occluded. The occluded parts are placed at reasonable locations because of the constraints between parts. Moreover, it does not matter which parts are occluded because our matching algorithm finds the global minimum of the energy function, independent of the choice of root used by the dynamic programming approach. When three of the five parts are occluded the best match of the model to the image was incorrect.

Figure 6 illustrates matching results on an image that contains multiple faces. Recall that our energy minimization algorithm computes the optimal location for the model as a function of the location of a root part. To detect multiple faces we first find the best overall

Figure 6. Matching results on an image with multiple faces. See text for description.

location for the root. We then exclude nearby locations and find the best remaining one and so on for additional detections. Each root location yields an object configuration that is optimal with respect to that location of the root. In this example we simply found the best three locations for the model, alternatively a threshold could be used to find all matches above a certain quality. Multiple detections could also have been generated with the sampling techniques together with a separate verification technique.

We also learned a larger model, this one with nine parts. We now have three parts for each eye, one for the left corner, one for the right corner and one for the pupil. This is a useful model to detect gaze direction. Figure 7 shows one of the training examples and the learned model. Also, in Fig. 7, there is a detailed illustration of the connections to the left corner of the right eye (part 1). The ellipses illustrate the location uncertainty for the other parts, when this part is at some fixed location. They are level sets of the probability distribution for the location of parts 2, 3, and 4, given that part 1 is fixed. Note that the location of the pupil (part 2) is much more constrained with respect to the location of the eye corner than any other part, as would be expected intuitively. Also note that the distributions are not spherically symmetric, as they reflect the typical variation in the relative locations of parts. We see that the algorithm both learned an interesting structure for the model, and automatically determined a rich set of constraints between the locations of different pairs of parts.

6. Articulated Models

In this section we present a scheme to model articulated objects. Our main motivation is to construct a system that can estimate the pose of human bodies. We concentrate on detecting objects in binary images such as those obtained by background subtraction. Figure 8 shows an example input and matching result. Binary images characterize well the problem of pose estimation for an articulated object. We want to find an object configuration that covers the foreground pixels and leaves the background pixels uncovered. Our method works with very noisy input, including substantial occlusion which we illustrate with examples. Note that in order to detect articulated bodies we use the sampling techniques in Section 4.2 instead of computing the MAP estimate for the object location. This is important because the models for articulated bodies are imprecise rather than being accurate generative models.

6.1. Parts

For simplicity, we assume that the image of an object is generated by a scaled orthographic projection, so that parallel features in the model remain parallel in the image. For images of human forms this is generally a reasonable assumption. We further assume that the scale factor of the projection is known. We can easily add an extra parameter to our search space in order to relax this latter assumption.

Suppose that objects are composed of a number of rigid parts, connected by flexible joints. If a rigid part is more or less cylindrical, its projection can be approximated by a rectangle. The width of the rectangle comes from the diameter of the cylinder and is fixed, while the length of the rectangle depends on the length of the cylinder but can vary due to foreshortening. We model the projection of a part as a rectangle parameterized by (x, y, s, θ). The center of the rectangle is given in image coordinates (x, y), the length is defined by the amount of foreshortening $s \in [0, 1]$, and the orientation is given by θ. So we have a four-dimensional pose space for each part.

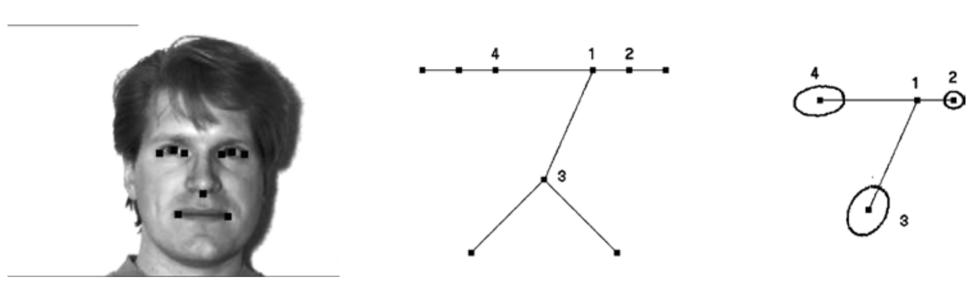

Figure 7. One example from the second training set, the structure of the learned model, and a pictorial illustration of the connections to one of the parts, showing the location uncertainty for parts 2, 3, and 4, when part 1 is at a fixed position.

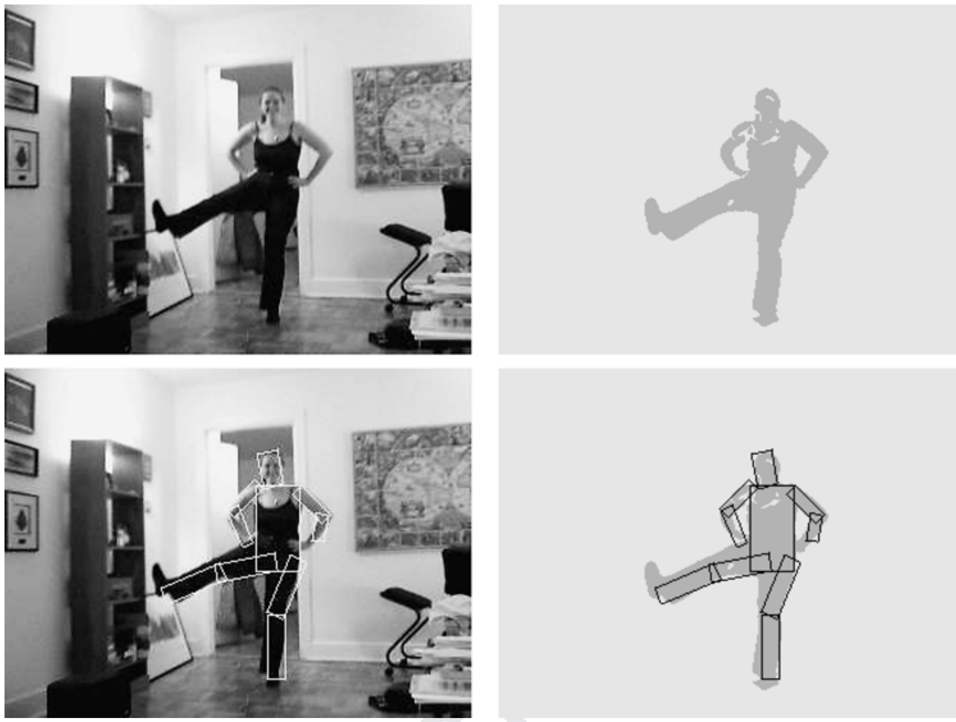

Figure 8. Input image, binary image obtained by background subtraction, and matching result superimposed on both images.

We model the likelihood of observing an image given a particular location for a part in the following way. First, each pixel in the image is generated independently. Pixels inside a part are foreground pixels with probability q_1. Intuitively, q_1 should be close to one, expressing the idea that parts occlude the background. We also model a border area around each part (see Fig. 9). In this area, pixels belong to the foreground with probability q_2. In practice, when we estimate q_2 from data we see that pixels around a part tend to be background. We assume that pixels outside both areas are equally likely to be background or foreground pixels. Thus,

$$p(I \mid l_i, u_i) = q_1^{\text{count}_1} \, (1 - q_1)^{(\text{area}_1 - \text{count}_1)} \, q_2^{\text{count}_2} \\ \times (1 - q_2)^{(\text{area}_2 - \text{count}_2)} \, 0.5^{(\text{t} - \text{area}_1 - \text{area}_2)},$$

where $count_1$ is the number of foreground pixels inside the rectangle, and $area_1$ is the area of the rectangle. $count_2$ and $area_2$ are similar measures corresponding to the border area, and t is the total number of pixels in the image. So the appearance parameters are $u_i = (q_1, q_2)$, and it is straightforward to estimate these parameters from training examples.

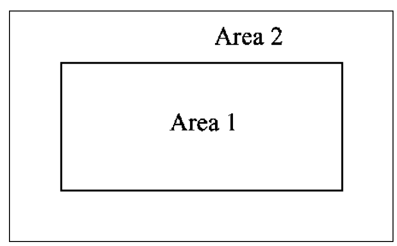

Figure 9. A rectangular part. $area_1$ is the area inside the part, and $area_2$ is the border area around it.

To make the probability measure robust we consider a slightly dilated version of the foreground when computing $count_1$, and to compute $count_2$ we erode the foreground (in practice we dilate and erode the binary images by two pixels). Computing the likelihood for every possible location of a part can be done efficiently by convolving the image with uniform filters. Each convolution counts the number of pixels inside a rectangle (specified by the filter) at every possible translation.

Intuitively, our model of $p(I \mid l_i, u_i)$ is reasonable for a single part. The likelihood favors large parts, as they explain a larger area of the image. But remember that we model $p(I \mid L, u)$ as a product of the individual likelihoods for each part. For a configuration with overlapping parts, this measure "over-counts" evidence. Suppose we have an object with two parts. The likelihood of an image is the same if the two parts are arranged to explain different areas of the image, or if the two parts are on top of each other and explain the same area twice. Therefore, with this measure the MAP estimate of an object configuration can be a bad guess for its true position. This is not because the posterior probability of the true configuration is low, but because there are configurations which have high posterior and are wrong. In our experiments, we obtain a number of configurations which have high posterior probability by sampling from that distribution. We then select one of the samples by computing a quality measure that does not over-count evidence.

There is one more thing we have to take into account for sampling to work. When $p(I \mid L, u)$ over-counts evidence, it tends to create high peaks. This in turn creates high peaks in the posterior. When a distribution has a very strong peak, sampling from the distribution will almost always obtain the location of the peak. To ensure that we get a number of different hypotheses from sampling we use a smoothed version of the likelihood function, defined as

$$p'(I \mid L, u) \propto p(I \mid L, u)^{1/T} \propto \prod_{i=1}^{n} p(I \mid l_i, u_i)^{1/T},$$

where T controls the degree of smoothing. This is a standard technique, borrowed from the principle of annealing (see Geman and Geman, 1984). In all our experiments we used $T = 10$.

6.2. *Spatial Relations*

For the articulated objects, pairs of parts are connected by flexible joints. A pair of connected parts is illustrated in Fig. 10. The location of the joint is specified by two points (x_{ij}, y_{ij}) and (x_{ji}, y_{ji}), one in the coordinate frame of each part, as indicated by circles in Fig. 10(a). In an ideal configuration these points coincide, as illustrated in Fig. 10(b). The ideal relative orientation is given by θ_{ij}, the difference between the orientation of the two parts.

Suppose $l_i = (x_i, y_i, s_i, \theta_i)$ and $l_j = (x_j, y_j, s_j, \theta_j)$ are the locations of two connected parts. The joint probability for the two locations is based on the deviation between their ideal relative values and the observed

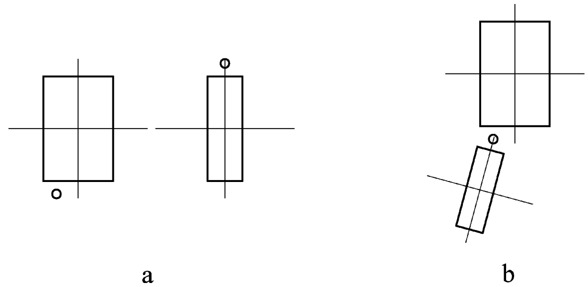

Figure 10. Two parts of an articulated object, (a) in their own coordinate system and (b) the ideal configuration of the pair.

ones,

$$\begin{aligned} p(l_i, l_j \mid c_{ij}) = {} & \mathcal{N}\left(x_i' - x_j', 0, \sigma_x^2\right) \\ & \mathcal{N}\left(y_i' - y_j', 0, \sigma_y^2\right) \\ & \mathcal{N}\left(s_i - s_j, 0, \sigma_s^2\right) \\ & \mathcal{M}(\theta_i - \theta_j, \theta_{ij}, k), \end{aligned} \qquad (16)$$

where (x_i', y_i') and (x_j', y_j') are the positions of the joints in image coordinates. Let R_θ be the matrix that performs a rotation of θ radians about the origin. Then,

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + s_i R_{\theta_i} \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} x_j' \\ y_j' \end{bmatrix} = \begin{bmatrix} x_j \\ y_j \end{bmatrix} + s_j R_{\theta_j} \begin{bmatrix} x_{ji} \\ y_{ji} \end{bmatrix}.$$

The distribution over angles, $\mathcal{M}$, is the von Mises distribution (Gumbel et al., 1953),

$$\mathcal{M}(\theta, \mu, k) \propto e^{k \cos(\theta - \mu)}.$$

The first two terms in the joint distribution measure the horizontal and vertical distances between the observed joint positions in the image. The third term measures the difference in foreshortening between the two parts. The last term measures the difference between the relative angle of the two parts and the ideal relative angle. Usually σ_x and σ_y will be small so parts tend to be aligned at their joint. And if k is small the angle between the two parts is fairly unconstrained, modeling a revolute joint. The connection parameters under this model are,

$$c_{ij} = \left(x_{ij}, y_{ij}, x_{ji}, y_{ji}, \sigma_x^2, \sigma_y^2, \sigma_s^2, \theta_{ij}, k\right).$$

Finding the maximum likelihood estimate of σ_s^2 is easy since we just have a Gaussian distribution over $s_i - s_j$. Similarly, there are known methods for finding the ML parameters (θ_{ij}, k) of a von Mises distribution (see Gumbel et al., 1953). The ML estimate of the joint location in each part are the values $(x_{ij}, y_{ij}, x_{ji}, y_{ji})$ which minimize the sum of square distances between (x'_i, y'_i) and (x'_j, y'_j) over the examples. We can compute this as a linear least squares problem.

We need to write the joint distribution of l_i and l_j in the specific form required by our algorithms. It must be a Gaussian distribution with zero mean and diagonal covariance in a transformed space, as shown in Eq. (6). First note that a von Mises distribution over angular parameters can be specified in terms of a Gaussian over the unit vector representation of the angles. Let $\vec{\alpha}$ and $\vec{\beta}$ be the unit vectors corresponding to two angles α and β. That is, $\vec{\alpha} = [cos(\alpha), \sin(\alpha)]^T$, and similarly for $\vec{\beta}$. Then,

$$\cos(\alpha - \beta) = \vec{\alpha} \cdot \vec{\beta} = -\frac{\|\vec{\alpha} - \vec{\beta}\|^2 - 2}{2}.$$

Now let

$$\begin{aligned} T_{ij}(l_i) &= (x'_i, y'_i, s_i, \cos(\theta_i + \theta_{ij}), \sin(\theta_i + \theta_{ij})), \\ T_{ji}(l_j) &= (x'_j, y'_j, s_j, \cos(\theta_j), \sin(\theta_j)), \\ D_{ij} &= \operatorname{diag}(\sigma_x^2, \sigma_y^2, \sigma_s^2, 1/k, 1/k), \end{aligned}$$

which allow us to write Eq. (16) in the right form,

$$p(l_i, l_j \mid c_{ij}) \propto \mathcal{N}(T_{ji}(l_j) - T_{ij}(l_i), 0, D_{ij}).$$

For these models, the number of discrete locations h' in the transformed space is a little larger than the number of locations h for each part. This is because we represent the orientation of a part as a unit vector which lives in a two-dimensional grid. In practice, we use 32 possible angles for each part, and represent them as points in a 11×11 grid, which makes h' about four times h.

6.3. *Experiments*

We use a coarse articulated model to represent the human body. Our model has ten parts, corresponding to the torso, head, two parts per arm and two parts per leg. To generate training examples we labeled the location of each part in ten different images (without too much precision). The learned model is illustrated in Fig. 11. The crosses indicate joints between parts. We never told the system which parts should be connected together, this is automatically learned during the ML learning procedure. Note that the correct structure was learned, and the joint locations agree with the human body anatomy (the joint in the middle of the torso connects to the head). The configuration of parts shown in Fig. 11 was obtained by fixing the position of the torso and placing all other parts in their optimal location with respect to each other.

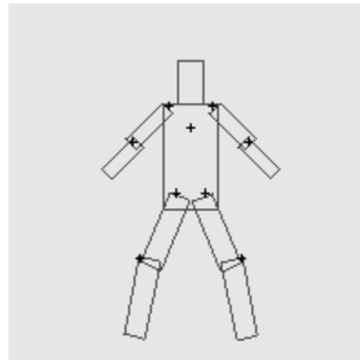

Figure 11. Human body model learned from example configurations.

We tested the model by matching it to novel images. As described in Section 6.1, we sample configurations from the posterior distribution to obtain multiple hypotheses and rate each sample using a separate measure. For each sample we compute the Chamfer distance between the shape of the object under the hypothesized configuration and the binary image obtained from the input. The Chamfer distance is a robust measure of binary correlation (Borgefors, 1988). The matching process is illustrated in Fig. 12. First, a binary image is obtained from the original image using background subtraction. We use this binary image as input to the sampling algorithm to obtain a number of different pose hypotheses. The best pose is then selected using the Chamfer measure.

More matching results are shown in Fig. 13. For each image, we sampled two-hundred object configurations from the posterior distribution and picked the best one under the Chamfer distance. Using a desktop computer it took about one minute to process each image. The space of possible locations for each part was discretized into a $70 \times 70 \times 10 \times 32$ grid, corresponding to (x, y, s, θ) parameters. There are over 1.5 million locations for each part, making any algorithm that considers locations for pairs of parts at a time impractical.

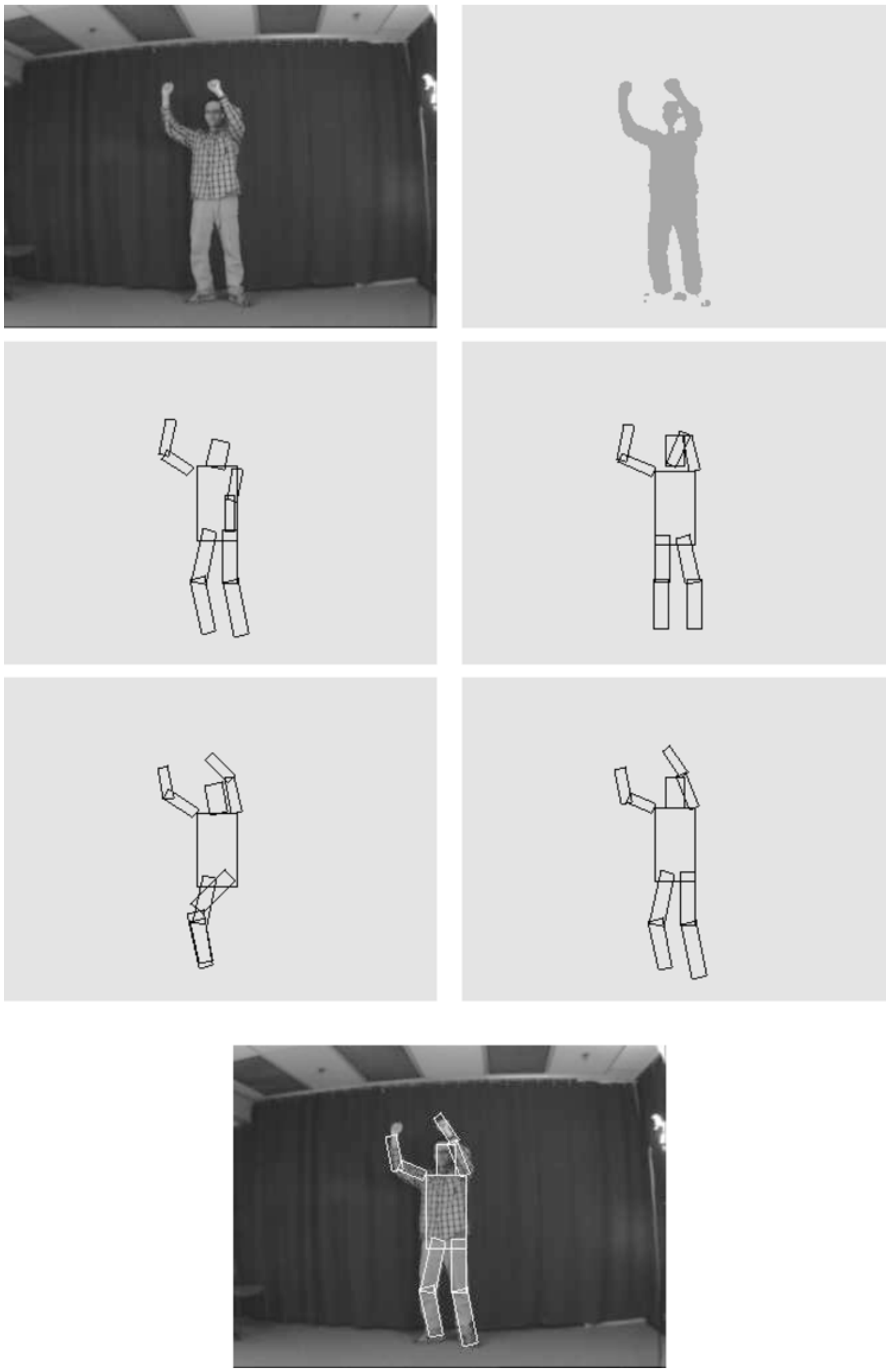

Figure 12. Input image, binary image, random samples from the posterior distribution of configurations, and best result selected using the Chamfer distance.

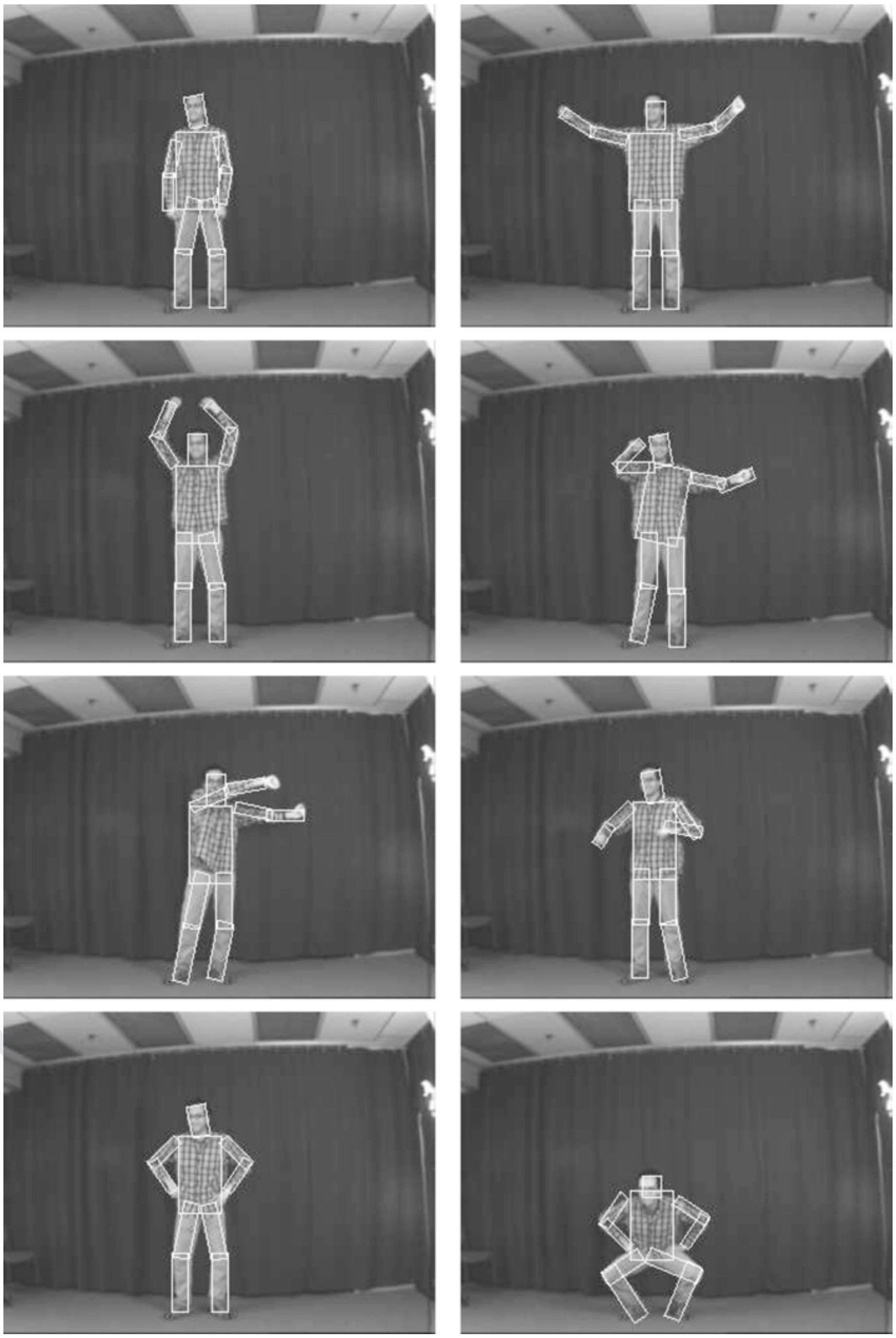

Figure 13. Matching results (sampling 200 times).

Of course, sometimes the estimated pose is not correct. The most common source of error comes from ambiguities in the binary images. Figure 14 shows an example where the image does not provide enough information to estimate the position of one arm. Even in that case we get a fairly good estimate. We can detect when ambiguities happen because we obtain many different poses with equally good Chamfer score. Thus

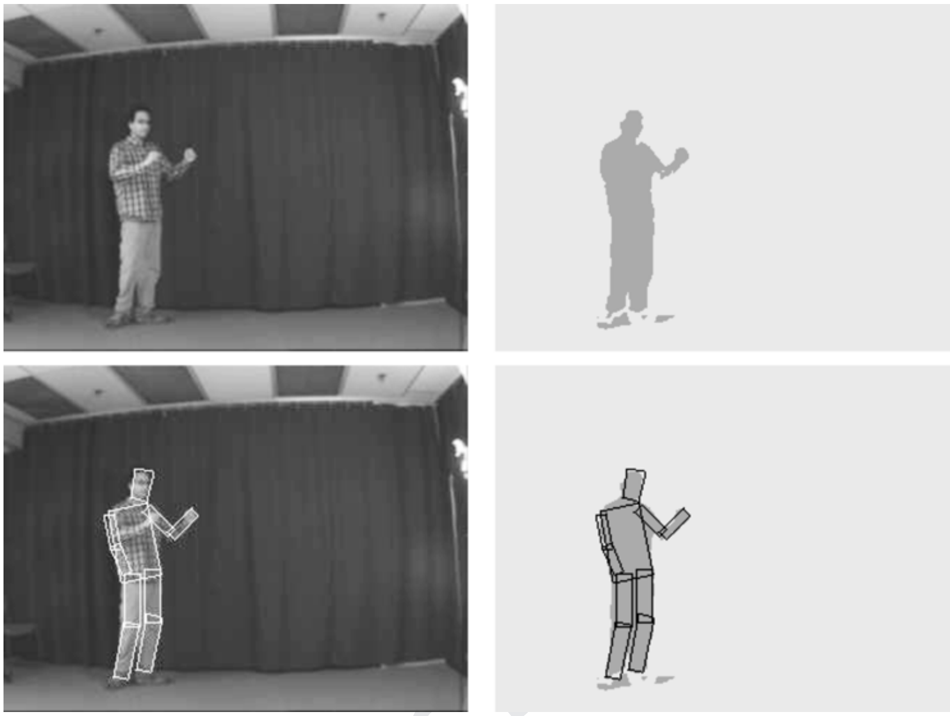

Figure 14. In this case, the binary image doesn't provide enough information to estimate the position of one arm.

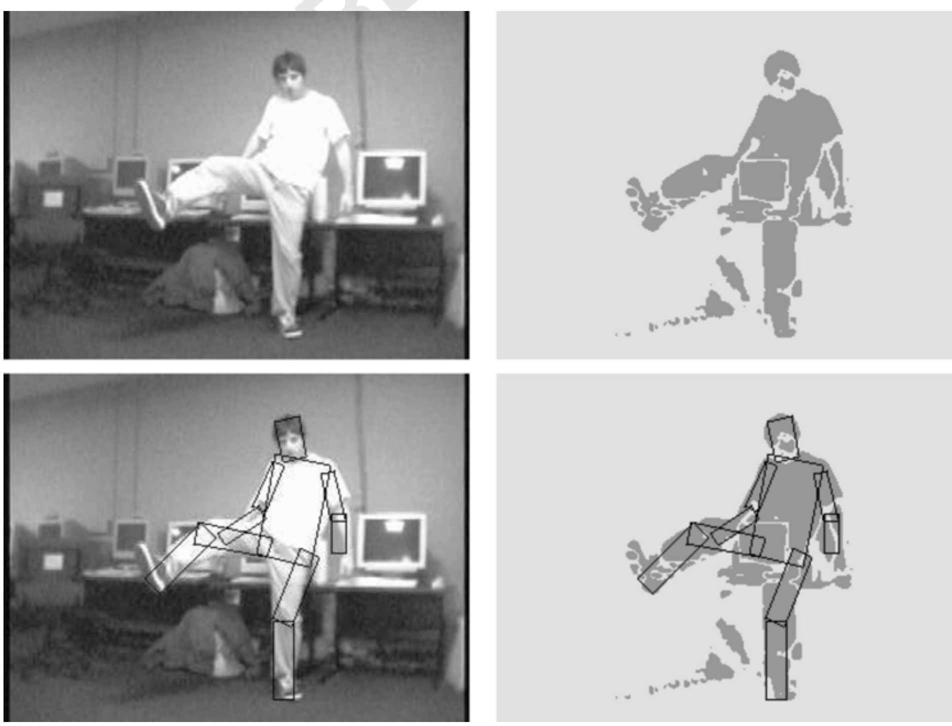

Figure 15. This example illustrates how our method works well with noisy images.

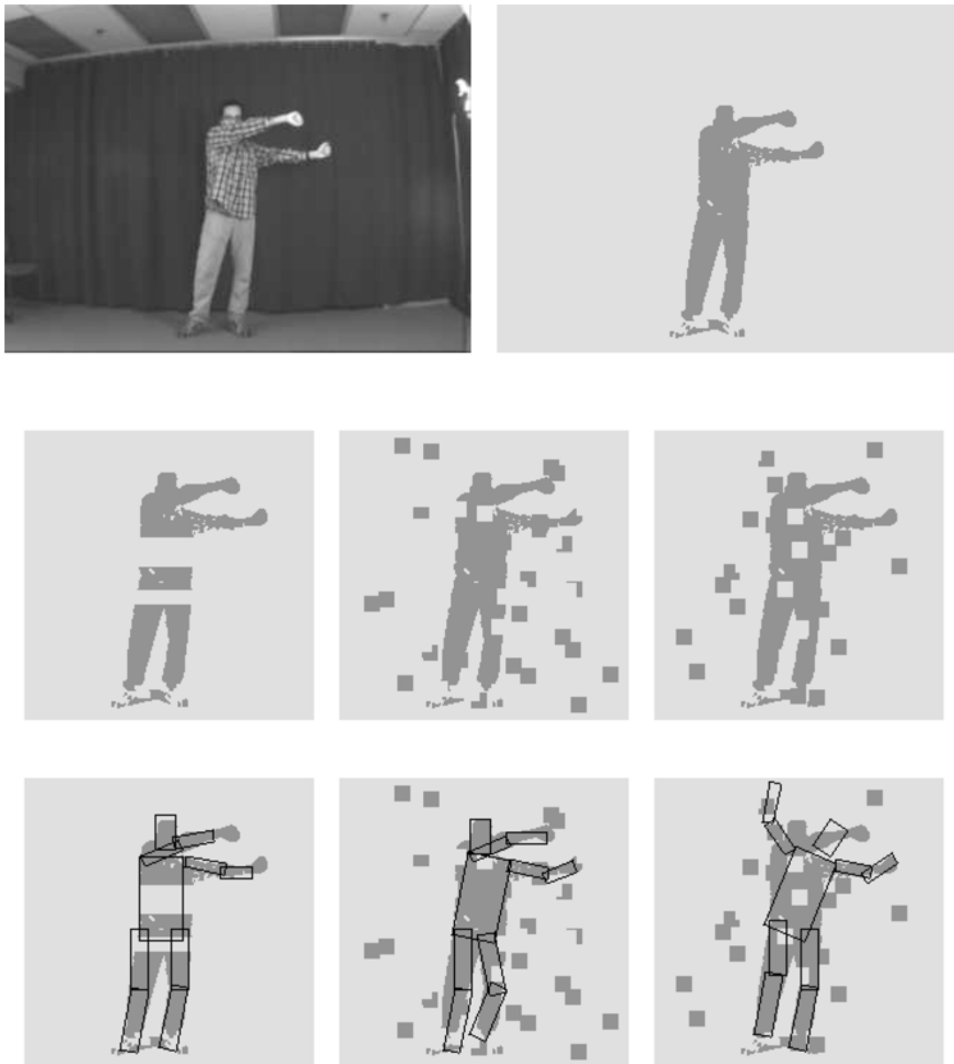

Figure 16. Matching results on corrupted images. The top row shows the original input, the middle row shows corrupted versions of the binary input and the last row shows the matching results. The first two cases demonstrate how the algorithm can handle good amounts of noise and occlusion. The third case shows an incorrect matching result.

we know that there are different configurations that are equally good interpretations of the image.

Figure 15 shows how our method works well on a noisy input. More examples of matching to noisy inputs are shown in Fig. 16, using corrupted binary images, including a case where large portions of the foreground are missing. These examples illustrate two of the main advantages of our approach. It would be difficult to detect body parts individually on inputs such as these, but the dependencies between parts provide sufficient context to detect the human body as a whole. Moreover, the presence of clutter and occlusion create difficulties for heuristics or local search techniques, while our global method can find the correct configuration in these cases.

7. Summary

This paper describes a statistical framework for representing the visual appearance of objects composed of rigid parts arranged in a deformable configuration. The models are based on the pictorial structure representation introduced in Fischler and Elschlager (1973), which allows for qualitative descriptions of appearance

P1: xxx
International Journal of Computer Vision KL3162-04/5277531 September 6, 2004 22:24

and is suitable for generic recognition problems. There are three main contributions in the paper. First, we introduce efficient algorithms for finding the best *global* match of a large class of pictorial structure models to an image. In contrast, prior work use heuristics or local search techniques that must be somehow initialized near the right answer. Second, we introduce the use of statistical sampling techniques to identify multiple good matches of a model to an image. Third, our use of a statistical formulation provides a natural way of learning pictorial structure models from labeled example images. Most of the prior work uses manually constructed models, which are difficult to create and to validate.

One of the difficulties in representing generic objects is the large variation in shape and photometric information in each object class. Pictorial structure models represent the appearance of each part separately and explicitly capture the spatial configuration of the parts independently of their appearances. This framework is general, in the sense that it is independent of the specific method used to represent the appearance of parts, and the type of the geometric relationships between the parts. By using a general framework we have provided a set of computational mechanisms that can be used for many different modeling schemes. We have described two quite different modeling schemes, one was used to model faces and the other to model articulated bodies.

Acknowledgments

We would like to thank Romer Rosales for providing a database of human body images with a variety of poses.

References

Amini, A.A., Weymouth, T.E., and Jain, R.C. 1990. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867.

Amit, Y. and Geman, D. 1999. A computational model for visual selection. *Neural Computation*, 11(7):1691–1715.

Ayache, N.J. and Faugeras, O.D. 1986. Hyper: A new approach for the recognition and positioning of two-dimensional objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44–54.

Berger, J.O. 1985. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag.

Borgefors, G. 1986. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34(3):344–371.

Borgefors, G. 1988. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):849–865.

Boykov, Y., Veksler, O., and Zabih, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239.

Bregler, C. and Malik, J. 1998. Tracking people with twists and exponential maps. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8–15.

Burl, M.C. and Perona, P. 1996. Recognition of planar object classes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 223–230.

Burl, M.C., Weber, M., and Perona, P. 1998. A probabilistic approach to object recognition using local photometry and global geometry. In *European Conference on Computer Vision*, pp. II:628–641.

Chow, C.K. and Liu, C.N. 1968. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467.

Cormen, T.H., Leiserson, C.E., and Rivest, R.L. 1996. *Introduction to Algorithms*. MIT Press and McGraw-Hill.

Dickinson, S.J., Biederman, I., Pentland, A.P., Eklundh, J.O., Bergevin, R., and Munck-Fairwood, R.C. 1993. The use of geons for generic 3-d object recognition. In *International Joint Conference on Artificial Intelligence*, pp. 1693–1699.

Felzenszwalb, P.F. and Huttenlocher, D.P. 2000. Efficient matching of pictorial structures. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. II:66–73.

Fischler, M.A. and Bolles, R.C. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Fischler, M.A. and Elschlager, R.A. 1973. The representation and matching of pictorial structures. *IEEE Transactions on Computer*, 22(1):67–92.

Freeman, W.T. and Adelson, E.H. 1991. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906.

Gdalyahu, Y. and Weinshall, D. 1999. Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1312–1328.

Geman, S. and Geman, D. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741.

Grimson, W.E.L. and Lozano-Perez, T. 1987. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469–482.

Gumbel, E.J., Greenwood, J.A., and Durand, D. 1953. The circular normal distribution: Theory and tables. *Journal of the American Statistical Association*, 48:131–152.

Huttenlocher, D.P., Klanderman, G.A., and Rucklidge, W.J. 1993. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863.

Huttenlocher, D.P. and Ullman, S. 1990. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212.

Ioffe, S. and Forsyth, D.A. 2001. Probabilistic methods for finding people. *International Journal of Computer Vision*, 43(1):45–68.

Ishikawa, H. and Geiger, D. 1998. Segmentation by grouping junctions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 125–131.

Ju, S.X., Black, M.J., and Yacoob, Y. 1996. Cardboard people: A parameterized model of articulated motion. In *International Conference on Automatic Face and Gesture Recognition*, pp. 38–44.

Karzanov, A.V. 1992. Quick algorithm for determining the distances from the points of the given subset of an integer lattice to the points of its complement. *Cybernetics and System Analysis*, pp. 177–181. Translation from the Russian by Julia Komissarchik.

Lamdan, Y., Schwartz, J.T., and Wolfson, H.J. 1990. Affine invariant model-based object recognition. *IEEE Transactions on Robotics and Automation*, 6(5):578–589.

Moghaddam, B. and Pentland, A.P. 1997. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710.

Murase, H. and Nayar, S.K. 1995. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14(1):5–24.

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

Pentland, A.P. 1987. Recognition by parts. In *IEEE International Conference on Computer Vision*, pp. 612–620.

Rabiner, L. and Juang, B. 1993. *Fundamentals of Speech Recognition*. Prentice Hall.

Ramanan, D. and Forsyth, D.A. 2003. Finding and tracking people from the bottom up. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. II: 467–474.

Rao, R.P.N. and Ballard, D.H. 1995. An active vision architecture based on iconic representations. *Artificial Intelligence*, 78(1/2):461–505.

Rivlin, E., Dickinson, S.J., and Rosenfeld, A. Recognition by functional parts. *Computer Vision and Image Understanding*, 62(2):164–176, September 1995.

Roberts, L.G. 1965. Machine perception of 3-d solids. In *Optical and Electro-optical Information Processing*, pp. 159–197.

Rucklidge, W. 1996. *Efficient Visual Recognition Using the Hausdorff Distance*. Springer-Verlag, LNCS 1173.

Sebastian, T.B., Klein, P.N., and Kimia, B.B. 2001. Recognition of shapes by editing shock graphs. In *IEEE International Conference on Computer Vision*, pp. I:755–762.

Turk, M. and Pentland, A.P. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–96.

Wells, W.M. III 1986. Efficient synthesis of Gaussian filters by cascaded uniform filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):234–239.