Dynamic Quantization for Belief Propagation in Sparse Spaces

James Coughlan and Huiying Shen

Smith-Kettlewell Institute
San Francisco, CA 94115
coughlan@ski.org, hshen@ski.org

Abstract

Graphical models provide an attractive framework for modeling a variety of problems in computer vision. The advent of powerful inference techniques such as belief propagation (BP) has recently made inference with many of these models tractable. Even so, the enormous size of the state spaces required for some applications can create a heavy computational burden. Pruning is a standard technique for reducing this burden, but since pruning is irreversible it carries the risk of greedily deleting important states, which can subsequently result in gross errors in BP.

To address this problem, we propose a novel extension of pruning, which we call dynamic quantization (DQ), that allows BP to adaptively add as well as subtract states as needed. We examine DQ in the context of graphical-model based deformable template matching, in which the state space size is on the order of the number of pixels in an image. The combination of BP and DQ yields deformable templates that are both fast and robust to significant occlusions, without requiring any user initialization. Experimental results are shown on deformable templates of planar shapes. Finally, we argue that DQ is applicable to a variety of graphical models in which the state spaces are sparsely populated.

Key words:
belief propagation, graphical models, pruning, deformable templates

1 Introduction

Graphical models (also known as Markov random fields) have become a standard modeling and inference tool for use in computer vision, pattern recognition and AI. They provide an attractive framework for representing large problems with many variables and permit the interactions among the variables to be modeled explicitly. Moreover, this framework allows these interactions to be learned from training data. Graphical models have been applied to a wide variety of computer vision problems, including image labeling and structure detection [18,20], image de-noising [29], texture analysis and synthesis [37], super-resolution [14], stereo [33,5], optical flow [11], tracking [32] and shape matching [28,8]. A key property of such problems that is exploited by graphical models is the Markov interaction structure that typically holds: each variable only interacts directly with only a small neighborhood of other variables, which greatly constrains the types of interactions that are possible.

The main drawback of graphical models is the heavy computational burden they require for performing inference and learning. Methods such as MCMC sampling [16] and thin junction trees [3] provide useful approximations and even exact results in some cases, but are not powerful enough to attain the speed and/or accuracy needed in many applications.

The belief propagation (BP) algorithm, invented by Pearl [26], has demonstrated great effectiveness on a variety of graphical models. Although it was originally intended only for graphical models without loops (e.g. chains or trees), for which it is guaranteed to provide an exact solution, empirically it has been found to be effective on a number of loopy graphical models [24]. It has greatly expanded the range of graphical models for which tractable inference is possible. More recently, work by [31,19] has extended BP to the use of continuous variables, which is a natural domain for many problems.

Despite these advances, the enormous size of the state spaces required for some applications still carries an exorbitant computational burden for many problems. Augmenting BP with pruning – i.e. removing states that are deemed sufficiently unlikely during the course of BP – is a standard technique for lessening this burden, but since pruning is irreversible it carries the risk of greedily deleting important states, which can subsequently result in gross errors in BP.

As an alternative, we propose a novel technique for speeding up BP, called dynamic quantization (DQ), which is a compromise between pre-pruning unlikely candidates (and removing them from subsequent consideration) and representing all possible states. The main idea of DQ (see Figure (2)) is to combine standard pruning techniques with a technique for augmenting state spaces by adding sufficiently promising states. In this way, the number of allowed states increases or decreases over time as needed, allowing BP to focus on the more probable regions of state space.

We describe and demonstrate DQ in a particular experimental domain, graphical-model based deformable template matching, which has previously been implemented using BP with pruning [8]. The application of DQ is well suited to this problem, since the number of states (roughly, the number of pixels) relevant to BP varies over the course of the algorithm, and since the set of important states is sparsely distributed (see Section 4). Initially, every pixel with sufficient edge magnitude is worth considering (since the template is edge contour-based). However, this set of pixels includes significant amounts of clutter, and may lack pixels located on occluded parts of the object. Over time, the contextual information propagated by BP will rule out the clutter pixels and suggest the need to admit states corresponding to occluded regions of the object.

The rest of this paper focuses on the application of DQ to the domain of graphical-model based deformable template matching. First we provide background on deformable templates, particularly those based on graphical models. Next we describe and define the graphical model used to define our deformable template. Then we define DQ in detail and show experimental results. These results demonstrate that DQ enables our deformable templates to run quickly and robustly in the presence of significant partial occlusions and scene clutter, without requiring any user initialization. Finally, we argue that DQ is applicable to a variety of graphical models in which the state spaces are sparsely populated.

2 Background on Deformable Templates

A variety of problems relating to the detection and matching of shapes have been formulated in terms of deformable template models [36], which explicitly model the shape and appearance of flexible objects. A common property of many deformable template models is the specification of a global shape in terms of parts, with geometric relationships among the parts enforcing constraints on the global shape. To find the best match of such a deformable template model with an image, candidates for the model parts are extracted from the image by a feature selection process (or in some cases drawn from a quantized space of values aligned to the image pixel grid), and the candidates that best satisfy the desired geometric relationships among the parts are chosen as the best-fitting solution.

An early example of this paradigm is the “pictorial structures” framework of [13], which describes a face template composed of elementary parts (eyes, nose, etc.) with spring-like spatial interactions between them, and a technique similar to dynamic programming (DP) used to find a nearly globally optimal solution. Similar matching procedures based on DP include landmark matching [2] and work addressing the related problem of finding generic smooth image contours [4]. Recently, [10] has introduced a graphical-model based deformable template which exploits the decomposition of a large class of shapes into triangulated polygons, giving rise to tree-shaped graphical models (without loops) that can be solved exactly with DP. One of the major advantages of deformable templates which can be matched by DP is that the globally optimal match is found without the need for an initial approximate guess of where the target is located in the image. By contrast, local gradient-based optimization procedures, which need a suitable initial condition to converge to the correct solution, are commonly used to match many other types of deformable templates (e.g. [6]).

Dynamic programming is a useful optimization tool for these types of deformable templates, but it is subsumed by belief propagation, which applies to a much larger class of graphical models. BP is exact on graphical models without loops (i.e. chains or trees) – the same models for which DP is guaranteed to work – but has also been shown empirically to provide good approximate solutions on a variety of loopy graphs [23]. Moreover, the graphical model formulation is attractive because a graphical model specifies a statistical model of the shape and appearance variability, rather than simply specifying a global fitness function to evaluate the quality of a match between a template and an image. The shape (prior) and appearance (likelihood) models are specified separately, which makes it easier to understand the behavior of the deformable template, and both the prior and likelihood models can be learned from training data.

Graphical models used to represent shape deformations and matching processes have recently been applied to deformable templates in tracking applications [27,15] and to dense stereo matching [33], all of which use BP to perform inference on the models. Work by [31,19] to extend the use of BP to the continuous variable domain demonstrates simple graphical models for recovering facial appearance under partial occlusion and for detecting articulated objects in clutter. These techniques are used in [32] to perform tracking of an articulated object under partial occlusion. An optimization technique related to BP is also used in the graphical-model based shape matching work by [28]. All of these algorithms require the entire target to be visible, except for [31,32,19,30], which are too computationally demanding for real-time use. Finally, current research on 3-D registration by [1] using BP to match 3-D non-rigid surfaces defined by range data can handle significant amounts of missing data (range occlusions), but requires that there be no clutter.

In previous work [8] we devised the first deformable template we are aware of based on BP. This template used a graphical model to represent the contour of a deformable planar shape such as a hand or a letter. The algorithm used a pre-pruning step to eliminate unlikely pixel locations from subsequent processing by BP. Although this pre-pruning stage greatly sped up the algorithm, it did so at the expense of requiring the entire target to be visible. Earlier work with a DP-based deformable template [7] removed this restriction by eliminating the pre-pruning step and allowing the possibility of features to be visible or occluded at *every* pixel. This technique was robust to occlusions but computationally very expensive.

One way to speed up graphical model template-matching is to apply the distance transform to DP or BP (max-sum), so that all possible pixel locations can be rapidly processed. This was demonstrated by [12], which enabled a simple deformable template of faces (with just five nodes using feature detectors such as eyes) to find frontally-viewed faces in an image in under 1 second. However, the distance transform only applies to a limited class of graphical model shape priors (e.g. with Gaussian interactions that are functions of (x,y) pixel coordinates), which may be unsuitable for some applications, e.g. those requiring full rotation invariance.

The alternative method we use to speed up graphical model template-matching is DQ, which allows BP to perform template matching even with partial occlusions in the presence of considerable clutter, while remaining computationally tractable. We illustrate the DQ modification of BP with experimental results on planar deformable templates, demonstrating robustness to partial occlusions.

3 Generative Model

In this section we describe in detail the graphical models we use to define our deformable templates, including the shape prior, appearance likelihood model and a discussion of the resulting posterior.

3.1 Graphical Model Shape Prior

This subsection summarizes the basic graphical model used in our previous work [8], which models the shape of the boundary of a planar object as a sequence of points with associated orientations. The graphical model enforces spring-like geometric relationships between neighboring points to ensure that the overall shape is similar to a reference shape used to define the template, assigning high probabilities to configurations of points that are most similar to the reference. We will begin this section by specializing to the case of the letter A; generalization to models of other shapes follows naturally, which we discuss in the next subsection and demonstrate in our experimental results.

The variability of the template shape is modeled by the shape prior, which assigns a probability to each possible deformation of the shape. The shape is represented by a set of points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ in the plane which trace the contours of the shape, and by an associated chain $\theta_1, \theta_2, \cdots, \theta_N$ of tangent directions which describe the orientation of the tangent to the boundary at each point. In the case of the model of the letter A, $N = 20$ (see Figure 1). (Our notation is different from that used in [8], in which θ_i represented the orientation of the normal to the boundary rather than the tangent.) Each point $\mathbf{x}_i$ has two components x_i and y_i. For brevity we define variable $\mathbf{q}_i = (\mathbf{x}_i, \theta_i)$, which we also refer to as "node" i. The *configuration* $\mathbf{Q}$, defined as $\mathbf{Q} = (\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_N)$, completely defines the shape.

The shape prior is defined relative to a *reference shape* so as to assign high probability to configurations $\mathbf{Q}$ which are similar to the reference configuration $\tilde{\mathbf{Q}} = (\tilde{\mathbf{q}}_1, \tilde{\mathbf{q}}_2, \cdots, \tilde{\mathbf{q}}_N)$ and low probability to configurations that are not. This is achieved using a graphical model which penalizes the amount of deviation in shape between $\mathbf{Q}$ and $\tilde{\mathbf{Q}}$ in a way that is invariant to global rotation and translation. (The scale of the shape prior is fixed and we assume knowledge of this scale when we execute our algorithm.)

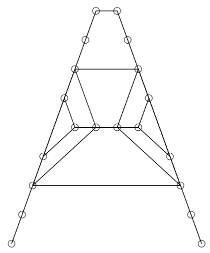

Fig. 1. Letter A template. Nodes drawn as circles, with lines indicating graph connectivity. All nodes are shown in their reference shape positions $(\tilde{x}_i, \tilde{y}_i)$.

Deviations in shape are measured by the geometric relationships of connected pairs of points $\mathbf{q}_i$ and $\mathbf{q}_j$ on the template (see Figure 1 for the connectivity), and are expressed in terms of *pair-wise interaction potentials* $\psi_{ij}(\mathbf{q}_i, \mathbf{q}_j)$. High potential values occur for highly probable shape configurations, for which the geometric relationships of pairs of points are most faithful to the reference shape $\tilde{\mathbf{Q}}$. We first outline the main properties of the potentials before defining them precisely.

Two kinds of geometric relationships, both of which are invariant to global translations and translations, are used to define the potentials. First, we expect that the relative orientations of tangent directions at nearby points on the boundary should be roughly invariant to local deformations. In other words, we expect $\theta_j - \theta_i \approx \tilde{\theta}_j - \tilde{\theta}_i$ for connected nodes i and j. Second, we note that the location of point $\mathbf{x}_j$ can be expressed relative to the location $\mathbf{x}_i$ and tangent θ_i; this relationship should also be roughly invariant to local deformations. Just as θ_j must be roughly consistent with θ_i, the location $\mathbf{x}_j$ must also be roughly consistent with the location and tangent direction of $\mathbf{q}_i$. (The reciprocal relationship between the location $\mathbf{x}_i$ and the location and tangent direction of $\mathbf{q}_j$ also holds.)

More precisely, we can express these geometric relationships in terms of *interaction energies* $U_{ij}(\mathbf{q}_i, \mathbf{q}_j)$. Low interaction energies occur for highly probable shape configurations, for which the geometric relationships of pairs of points tend to be faithful to the reference shape $\tilde{\mathbf{Q}}$, and high interaction energies are obtained for improbable configurations; the precise connection to probabilities is formulated in Equation (5).

The soft constraint that $\theta_j - \theta_i \approx \tilde{\theta}_j - \tilde{\theta}_i$ is expressed in the following interaction energy $U_{ij}(\mathbf{q}_i, \mathbf{q}_j)$:

$$U_{ij}^C(\mathbf{q}_i, \mathbf{q}_j) = \sin^2\left(\frac{\theta_j - \theta_i - C_{ij}}{2}\right) \tag{1}$$

where $C_{ij} = \tilde{\theta}_j - \tilde{\theta}_i$. This energy attains a minimum when $\theta_j - \theta_i = C_{ij}$ (and a maximum when $\theta_j - \theta_i = C_{ij} + \pi$).

Next we define the relationship between the location of point $\mathbf{x}_j$ relative to $\mathbf{q}_i$. $\tilde{\mathbf{x}}_i$ and $\tilde{\theta}_i$ define a local coordinate system, and the coordinates of $\tilde{\mathbf{x}}_j$ in that coordinate system are invariant to global translation and rotation. If we define the unit tangent vectors $\mathbf{t}_i = (\cos\theta_i, \sin\theta_i)$ and $\tilde{\mathbf{t}}_i = (\cos\tilde{\theta}_i, \sin\tilde{\theta}_i)$ and vectors perpendicular to them $\mathbf{t}_i^{\perp} = (-\sin\theta_i, \cos\theta_i)$ and $\tilde{\mathbf{t}}_i^{\perp} = (-\sin\tilde{\theta}_i, \cos\tilde{\theta}_i)$, then the dot product of $\mathbf{x}_j - \mathbf{x}_i$ with $\mathbf{t}_i$ and $\mathbf{t}_i^{\perp}$ should have values similar to the corresponding values for the reference shape: $(\mathbf{x}_j - \mathbf{x}_i) \cdot \mathbf{t}_i \approx (\tilde{\mathbf{x}}_j - \tilde{\mathbf{x}}_i) \cdot \tilde{\mathbf{t}}_i$ and $(\mathbf{x}_j - \mathbf{x}_i) \cdot \mathbf{t}_i^{\perp} \approx (\tilde{\mathbf{x}}_j - \tilde{\mathbf{x}}_i) \cdot \tilde{\mathbf{t}}_i^{\perp}$. Now we can define the remaining two terms in $U_{ij}(\mathbf{q}_i, \mathbf{q}_j)$, the energies

$$U_{ij}^A(\mathbf{q}_i, \mathbf{q}_j) = [(\mathbf{x}_j - \mathbf{x}_i) \cdot \mathbf{t}_i - A_{ij}]^2 \tag{2}$$

and

$$U_{ij}^B(\mathbf{q}_i, \mathbf{q}_j) = [(\mathbf{x}_j - \mathbf{x}_i) \cdot \mathbf{t}_i^{\perp} - B_{ij}]^2 \tag{3}$$

where $A_{ij} = (\tilde{\mathbf{x}}_j - \tilde{\mathbf{x}}_i) \cdot \tilde{\mathbf{t}}_i$ and $B_{ij} = (\tilde{\mathbf{x}}_j - \tilde{\mathbf{x}}_i) \cdot \tilde{\mathbf{t}}_i^{\perp}$. The full interaction energy is then given as (omitting arguments $(\mathbf{q}_i, \mathbf{q}_j)$ for brevity):

$$U_{ij} = \frac{1}{2}\{K_{ij}^A U_{ij}^A + K_{ij}^B U_{ij}^B + K_{ij}^C U_{ij}^C\} \tag{4}$$

where the non-negative coefficients K_{ij}^A, K_{ij}^B and K_{ij}^C define the strengths of the interactions and are set to 0 for those pairs i and j with no direct interactions (the majority of pairs). Higher values of K_{ij}^A, K_{ij}^B and K_{ij}^C produce a stiffer (less deformable) template.

Noting that in general $U_{ij}(\mathbf{q}_i, \mathbf{q}_j) \neq U_{ji}(\mathbf{q}_j, \mathbf{q}_i)$, we symmetrize the interaction energy as follows: $U_{ij}^{sym}(\mathbf{q}_i, \mathbf{q}_j) = U_{ij}(\mathbf{q}_i, \mathbf{q}_j) + U_{ji}(\mathbf{q}_j, \mathbf{q}_i)$. We use the symmetrized energy to define the shape prior:

$$P(\mathbf{Q}) = \frac{1}{Z} \prod_{i<j} \exp(-U_{ij}^{sym}(\mathbf{q}_i, \mathbf{q}_j)) \tag{5}$$

where Z is a normalization constant and the product is over all pairs i and j, with the restriction $i < j$ to eliminate double-counting and self-interactions. Note that the prior is a Markov random field or graphical model that has pair-wise connections between all pairs i and j whose coefficients are non-zero. The graph connectivity and the values of the coefficients were chosen experimentally by stochastically sampling the prior using a Metropolis MCMC sampler, i.e. generating samples from the prior distribution to illustrate what shapes have high probability (as in [8]).

3.1.1 Constructing Shape Priors

Once we constructed and tested our letter A template, we used a simple procedure to construct other deformable template prior models. A clear, representative image of the shape was taken to define the reference shape. The points used to define the reference shape contour were chosen manually by clicking on the image with a mouse at roughly equal intervals along the contour. (The associated tangent orientations were also extracted by defining them to be perpendicular to the image gradient direction at each point, assuming an idealized version of the appearance model described in the next section.)

In our experiments it sufficed to use the same coefficient values for all the shapes we tried. The connectivity was initially chosen so most nodes had no more than two nearest neighbors (as in a Markov chain), with more nearest neighbors chosen near junctions and high-curvature points. Longer-range connections were added as needed to speed up the convergence of BP, by preventing severe shape distortions – e.g. adjacent pieces of the template being torn far apart – earlier rather than having to wait for the local connections to propagate the information necessary to rule out these distortions. Learning techniques such as maximum likelihood estimation could be employed to determine more optimal coefficient values and graph connectivities.

3.2 Appearance Model

The appearance (i.e. imaging, or likelihood) model explains what image data may be expected given a specific shape configuration. Rather than model raw image pixel data, we extract information from the image gradient.

The first type of image data derived from the raw grayscale image $I(\mathbf{x})$ is an edge map $I_e(\mathbf{x})$, and an associated edge orientation map $\phi(\mathbf{x})$ to provide estimates of the orientation of edges throughout the image. $I_e(\mathbf{x})$ is defined as the magnitude of the image gradient: $I_e(\mathbf{x}) = |G \star \nabla I(\mathbf{x})|$ where $G(.)$ is a smoothing Gaussian. The orientation map $\phi(\mathbf{x})$ is calculated as $\arctan(g_y/g_x)$ where $(g_x, g_y) = G \star \nabla I(\mathbf{x})$.

The edge strength part of the likelihood model quantifies the tendency for edge strength values to be high *on* edge and low *off* edge. As in our previous work [8], we use the approach of Geman and Jedynak [17] and define two conditional distributions of edge strength that are empirically measured: $P_{\text{on}}(I_e(\mathbf{x})) = P(I_e(\mathbf{x})|\mathbf{x}ON edge)$ and $P_{\text{off}}(I_e(\mathbf{x})) = P(I_e(\mathbf{x})|\mathbf{x}OFF edge)$. The most important property of these two distributions is that the log likelihood ratio $\log P_{\text{on}}(I_e)/P_{\text{off}}(I_e)$ increases monotonically with edge strength, meaning that higher edge strengths correspond to greater evidence for edges.

As shown in [8], *the overall likelihood of the entire template is proportional to terms depending only on these log likelihood ratios,* rather than the individual distributions $P_{\text{on}}(I_e)$ and $P_{\text{off}}(I_e)$. Rather than learning these individual distributions, we will propose simple models of the likelihood ratios themselves. More specifically, we will approximate the likelihood ratio $P_{\text{on}}(I_e)/P_{\text{off}}(I_e)$ as a step function having a value of 1 for edge strengths above a certain threshold and a value of 0.1 below threshold. In other words, *we are binarizing the edge strength map.* However, this thresholding procedure is done just for simplicity and is not a necessary step of our algorithm. The likelihood models could be extended to continuous (or more finely quantized) values, but we found the thresholding procedure to suffice for our deformable template model.

Next we turn to the orientation map. We expect that on a true object boundary the direction of ∇I should point roughly *perpendicular* to the tangent of the boundary. Denoting the true normal tangent direction of the boundary as θ, we then expect that $\phi(\mathbf{x})$ is approximately equal to either $\theta + \pi/2$ or $\theta - \pi/2$ (corresponding to the two possible edge polarities). This relationship between θ and $\phi(\mathbf{x})$ may also be quantified as a conditional distribution $P_{ang}(\phi|\theta)$, which was assumed to be of the form $P_{ang}(\phi - \theta)$ and measured [8] to have sharp peaks at $\pm\pi/2$. If $\mathbf{x}$ is not on an edge then we may assume that the distribution of $\phi(\mathbf{x})$ is uniform in all directions: $U(\phi) = 1/2\pi$. Again, we approximate the likelihood ratio $P_{ang}(\phi - \theta)/U(\phi - \theta)$ as a step function having a value of 1 when ϕ and θ are aligned within $10°$ of perpendicular and a value of 0.1 otherwise.

The complete imaging model is a distribution of all the image gradient data across the entire image, conditioned on the template shape configuration $\mathbf{Q}$.

(We adopt the approximation that the likelihoods at different pixels are independent conditioned on the template shape configuration.) We can express the likelihood model as a distribution that factors over every pixel. We define $\mathbf{d}(\mathbf{x}) = (I_e(\mathbf{x}), \phi(\mathbf{x}))$ and let $\mathbf{D}$ denote the values of $\mathbf{d}(\mathbf{x})$ across the entire image. Defining the likelihood ratio

$$R_i(\mathbf{q}_i) = \frac{P_{\text{on}}(I_e(\mathbf{x}_i)}{P_{\text{off}}(I_e(\mathbf{x}_i)} \frac{P_{ang}(\phi(\mathbf{x}_i) - \theta_i)}{U(\phi(\mathbf{x}_i) - \theta_i)} \tag{6}$$

we obtain (after some manipulation [8]):

$$P(\mathbf{D}|\mathbf{Q}) \propto [\prod_{i=1}^{N} R_i(\mathbf{q}_i)] \tag{7}$$

where the constant of proportionality is a function of $\mathbf{D}$ only and does not depend on $\mathbf{Q}$. (This dependence on $\mathbf{D}$ will not matter for estimating the template configuration, described in the next subsection.)

3.3 Posterior Distribution

The shape configuration $\mathbf{Q}$ is determined by the posterior distribution $P(\mathbf{Q}|\mathbf{D}) = P(\mathbf{Q})P(\mathbf{D}|\mathbf{Q})/P(\mathbf{D})$. Multiplying the likelihood Equation (7) by the prior yields an expression for the posterior of the following form:

$$P(\mathbf{q}_1, \cdots, \mathbf{q}_N|\mathbf{D}) = \frac{1}{Z} \prod_i \psi_i(\mathbf{q}_i) \prod_{i<j} \psi_{ij}(\mathbf{q}_i, \mathbf{q}_j) \tag{8}$$

where $\psi_i(\mathbf{q}_i) = R_i(\mathbf{q}_i)$ is the local evidence for $\mathbf{q}_i$ (i.e. the likelihood ratio in Equation (6)) and $\psi_{ij}(\mathbf{q}_i, \mathbf{q}_j) = e^{-U_{ij}^{sym}(\mathbf{q}_i,\mathbf{q}_j)}$ is the compatibility (or pair-wise potential) between $\mathbf{q}_i$ and $\mathbf{q}_j$ (from the shape prior, Equation (5)).

In the next section we discuss how to perform inference using the posterior distribution.

4 Methods: Dynamic Quantization

Now that we have defined our graphical model of a deformable template, we will use the posterior distribution to decide the most likely shape configuration given the image data. A natural decision rule for choosing the most representative configuration is the MAP (maximum a posterior) estimate. Instead we use the output of BP – i.e. estimates of marginals for each variable – to calculate the MPM (Maximizer of the Posterior Marginals [22]), i.e. the MAP estimate applied separately to the marginal posterior of each variable $\mathbf{q}_i$. More precisely, the MPM is given by $\mathbf{q}_i^* = \arg\max_{\mathbf{q}_i} P(\mathbf{q}_i|\mathbf{D})$ for all i. If the posterior is strongly peaked about its mode, as it should be when there is sufficient evidence for one correct match in the image, we expect the MPM and MAP to be similar. (Techniques similar to those used to find multiple targets in [7] may be used in the case of multiple matches, i.e. multiple targets in one image.) Empirically we find the MPM to be a satisfactory estimator, as shown in our experimental results.

Standard BP could be straightforwardly applied to estimate the marginal posteriors by quantizing the variables $\mathbf{q}_i$ to a sufficiently fine lattice (e.g. (x_i, y_i) lying on the input image pixel lattice and θ_i drawn from a discrete set of equally spaced angles from 0 to 2π), but a huge number of allowed states would result, making BP prohibitively slow. Instead, we propose a dynamic quantization (DQ) technique for efficiently quantizing the states of the variables in the graphical model. Although we demonstrate this technique in the context of our deformable template, it is a general technique that applies to a variety of graphical models. (A technique similar to DQ that uses Monte Carlo proposals to dynamically construct state space discretizations for hidden Markov Models is proposed and demonstrated in [25].)

4.1 Motivation

BP was originally formulated for use with graphical models having discrete variables, whereas our graphical models are most naturally represented using continuous variables. Recent work [31,19] building on ideas from particle filtering has made it possible to perform BP on graphs with continuous variables, using stochastic particles to represent messages. However, a major drawback of the particle representation is that the multiplication of these particle-based messages – a fundamental component of each BP iteration – requires a computationally intensive Gibbs sampling procedure. As mentioned above, it is also possible to finely quantize the continuous variables to obviate the need for the particle representation, but only at the expense of performing BP on very large (discrete) state spaces.

We choose instead to selectively quantize the variables in our graphical models, based on our intuition that only a fairly small number of "hot spots" in the variable state spaces are important. Therefore, it should suffice to quantize the space only in the neighborhood of these hot spots.

Our new dynamic quantization technique builds on our previous work [8], which used two forms of state pruning. The first form of pruning, which we called pre-pruning or adaptive quantization, initializes the allowed states for each variable $\mathbf{q}_i = (\mathbf{x}_i, \theta_i)$ to encompass those values most consistent with image evidence. More specifically, only those locations $\mathbf{x}_i$ are allowed that correspond to pixels with edge strengths above a certain threshold. At these locations only two possible orientation values of θ_i are allowed, $\phi \pm \pi/2$, where ϕ denotes the image gradient direction. (These two values correspond to assuming that the image gradient direction is exactly equal to the normal orientation of the edge boundary, and allowing for two possible polarities of the edge.) We use the same procedure to initialize the state spaces in BP.

The second form of pruning, called belief pruning, consists of monitoring the beliefs of each variable at each iteration of BP and discarding any states whose beliefs dropped below a certain threshold. (This is very similar to the "beam search" technique used to prune states in hidden Markov models (HMM's) in speech recognition [21].) While we have also retained this second form of pruning in our current work, we have added a modification that allows for new states to be created as well as old states to be destroyed.

The new ingredient that DQ adds to the two existing pruning techniques is a procedure for deciding when there is a possible deficiency of important states in a variable's state space, and a method for determining which states to add to correct such deficiencies.

The intuition for this new procedure can be illustrated by considering the case of a simple graphical model deformable template representing a generic smooth curve (see Figure (2)). For simplicity no orientation variables are used in this toy model, only location variables. If the variable state spaces are initialized very conservatively, so that no pixels with true edges are omitted, then BP will detect the correct target curve. The cost of such a conservative initialization will be to slow down BP with a lot of superfluous states corresponding to false positives and background clutter in the edge detection. If the variable state spaces are initialized to include only pixels with edge strengths above a moderate threshold, then a small fraction of the true edge pixels will be omitted from the state spaces, and BP will find an incorrect solution because of their absence. However, a simple criterion will allow BP to consider previously disallowed states: if an edge pixel candidate lacks any suitable continuations, then all pixel locations that would make reasonable continuations should be added to the appropriate state space for consideration.

In the next subsections we formalize these procedures for pruning unpromising states and "resurrecting" previously disallowed states.

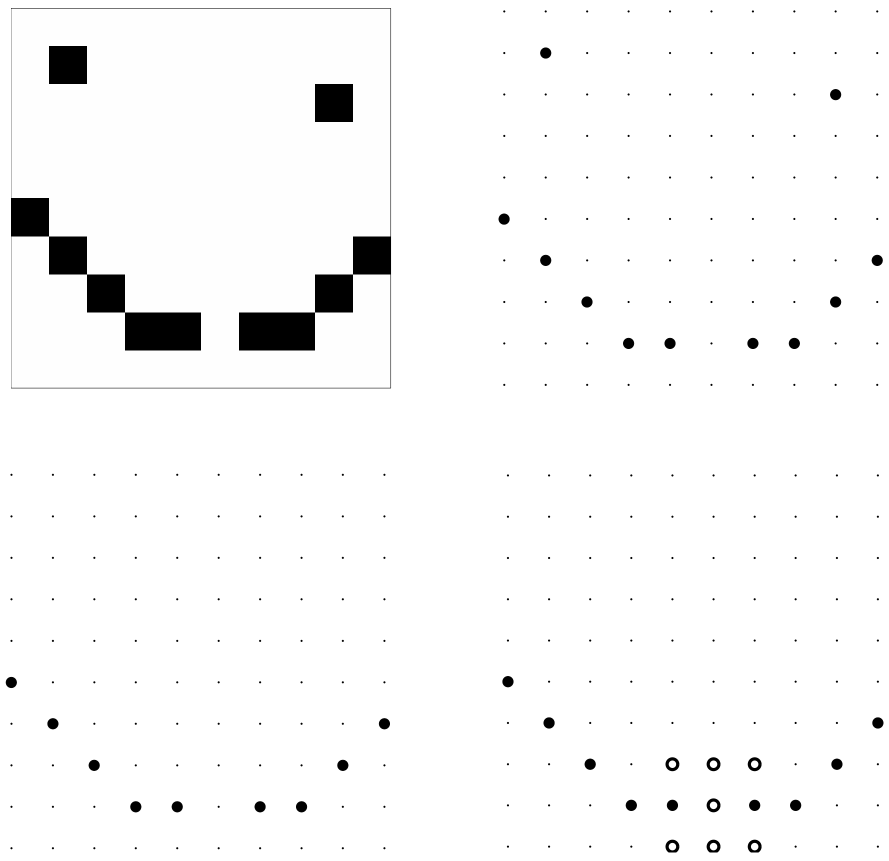

Fig. 2. DQ schematic showing evolution of state space for a toy model deformable template for generic smooth curves. Upper left: noisy image of smooth contour. Note gap in contour and clutter near top. Upper right: initial state space depicted as dark circles chosen from pixel lattice (indicated by dots). Lower left: after running standard BP with pruning, clutter is pruned out. Lower right: running DQ adds new states, shown as open circles, that allow gap to be filled in contour.

4.2 Notation

The BP message from node i to node j is denoted by $m_{ij}(\mathbf{q}_j)$ and is updated according to the following equation:

$$m_{ij}(\mathbf{q}_j) \leftarrow \frac{1}{Z_{ij}} \sum_{\mathbf{q}_i} \psi_{ij}(\mathbf{q}_i, \mathbf{q}_j)\psi_i(\mathbf{q}_i) \prod_{k \in N(i) \setminus j} m_{ki}(\mathbf{q}_i) \tag{9}$$

where Z_{ij} is a normalization factor and the neighborhood $N(i)$ denotes the set of nodes directly coupled to i (excluding i itself).

The belief of a node i is an estimate of the marginal (posterior) probability of $\mathbf{q}_i$ and is computed in terms of messages as follows:

$$b_i(\mathbf{q}_i) = \frac{1}{Z'_i}\psi_i(\mathbf{q}_i) \prod_{k \in N(i)} m_{ki}(\mathbf{q}_i) \tag{10}$$

where Z'_i is a normalization factor.

4.3 Pruning

We assume that, for each variable $\mathbf{q}_i$, the set of all possible states S_i is a subset of a lattice L_i (e.g. the cross product of the pixel lattice and a set of regularly spaced orientations from 0 to 2π), i.e. $S_i \subset L_i$. S_i is the set of all states in L_i that have not been pruned out; typically it is initialized to the set of all states corresponding to features extracted from the image (e.g. edge pixels in our deformable template application, as explained in Section 4.1).

A message from a particular node i to another node j, $m_{ij}(\mathbf{q}_j)$, may be thought of as a weighted list of states belonging to lattice L_j: for each state $\mathbf{q}_j \in S_j$, $m_{ij}(\mathbf{q}_j)$ is the message weight assigned to it. The message weight $m_{ij}(\mathbf{q}_j)$ of a pruned-out state $\mathbf{q}_j \in L_j - S_j$ is taken to be zero. (The zero value may be thought of as an approximation to the true value, which should be small.)

We initialize BP by setting the messages to uniform values: $m_{ij}(\mathbf{q}_j) = 1/|S_j|$, which corresponds to all of the beliefs being initialized to uniform distributions across S_j. We then perform message updates followed by calculations of the beliefs, and iterate this process. If at any point the (normalized) belief of a state $\mathbf{q}_i$ associated with node i falls below a small threshold value ϵ, we *remove (prune) this state from further consideration* from set S_i.

While pruning is very effective for removing many unnecessary states, which can greatly speed BP, it does so at the risk of inadvertently removing important states in the process. This is the motivation for "resurrecting" (adding) states in DQ.

4.4 Pruning and Adding States with DQ

In order to describe how states are added with DQ, we first define the *fan-out* of a state $\mathbf{q}_i$ to node j to be all states $\mathbf{q}_j$ (on the lattice) that are consistent according to the pair-wise potential ψ_{ij}:

$$F_{ij}(\mathbf{q}_i) = \{\mathbf{q}_j \in L_j | \psi_{ij}(\mathbf{q}_i, \mathbf{q}_j) > \epsilon\}$$

where ϵ is a small constant.

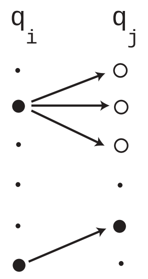

Fig. 3. Illustration of DQ in the update of the message $m_{ij}(.)$ from node i to node j. Dots, filled circles and empty circles represent the states of the lattices L_i and L_j (six states in each). Filled circles represent allowed states, i.e. members of S_i and S_j (which initially correspond to edge locations). Given the value of $\mathbf{q}_i$ at the lower left, at least one allowed state of $\mathbf{q}_j$ exists that is compatible with it (indicated by lowest arrow). However, given the value of $\mathbf{q}_i$ near the top left, no states $\mathbf{q}_j$ are compatible. DQ will therefore add the states in the fan-out $F_{ij}(\mathbf{q}_i)$, represented by empty circles, to the state space S_j.

The key idea of DQ (see 2) is to check that for each allowed state $\mathbf{q}_i$, there is at least one allowed state $\mathbf{q}_j$ in its fan-out (a continuation of the edge at $\mathbf{q}_i$). If not, then the fan-out states $F_{ij}(\mathbf{q}_i)$ are added to the state space S_j. After this procedure has been applied to every possible allowed state $\mathbf{q}_i$, the message update proceeds as usual (except that the state space S_j may now have been modified). The usual belief pruning procedure is also used to discard very improbable states, which offsets the growth of states from the adaptive quantization.

Note that, in order for DQ to be practical, the size of the fan-out must be relatively small compared to the size of the entire state space. This condition is guaranteed if the binary potential $\psi_{ij}(\mathbf{q}_i, \mathbf{q}_j)$ has a non-negligible value for only a *sparse* set of pairs $(\mathbf{q}_i, \mathbf{q}_j)$. In practice it is difficult to satisfy this condition unless the state spaces of the nodes are fairly low in dimension (perhaps three or lower). Appropriate choices of the sparseness threshold ϵ, which trades off speed and accuracy, are problem-dependent and a matter of experimentation.

DQ allows BP to begin with modest-sized state spaces, corresponding to strong edges in the image. It adds new states as needed in order to "fill in" features which are either faint or entirely missing because of occlusions. Much like the DDMCMC (data-driven Monte Carlo Markov chain) technique [34] for searching posterior distributions with the help of simpler data-driven distributions, DQ draws on candidate states suggested by the image data, but is not limited by the choice of candidates. We feel that this technique preserves one of the most attractive properties of continuous BP – the ability to dynamically allocate resources (in the form of particles) only to the more important regions of state space – without requiring costly sampling procedures to perform message multiplications.

4.5 Implementation Details

In our preliminary experiments we found that the straightforward application of DQ to each step of BP made the algorithm resurrect too many states and severely slowed down the algorithm. Rather than wait for pruning to eventually delete most of these unnecessary resurrected states, we decided to speed up the algorithm by applying DQ only at certain times in the course of BP. Specifically, we alternated between running ordinary BP for several sweeps (one sweep corresponds to performing one message update for each message in the entire graph) and applying DQ to all the nodes to allow the state spaces to enlarge. (See DQ Algorithm Box.) Note that each time BP was begun (on a new image or after each application of DQ), all the messages were initialized to 1; the only contribution of DQ was to allow some states to be added to the state spaces before resuming BP. The intuition behind this procedure is that the first several sweeps of ordinary BP strengthen many of the desired states and prune out many of the undesired ones (albeit with some errors). DQ allows the errors to be corrected (e.g. including states corresponding to missing features), so that ordinary BP can then work properly.

```
The Dynamic Quantization (DQ) Algorithm

1.) Initialize: pre-prune state spaces
        • select likely states using features extracted from data
2.) Run BP with pruning for a few sweeps
        • prune states whose beliefs are sufficiently small
3.) Run BP with DQ for one sweep
        • continue pruning unlikely states
        • also add states as necessary
          (those lacking compatible states in neighboring nodes)
4.) Alternate steps 2) and 3) as needed.
```

The BP algorithm (run either with or without DQ) was run using a simple serial (asynchronous) message update schedule. This schedule is based on the use of message update “chunks.” Each update chunk, labeled by node i, consists of an update of all messages from each node i to a subset of all its neighbors j (with the restriction of either $i < j$ or $i > j$, depending on which half of the sweep the update chunk occurs). One sweep of the schedule consists of performing each possible update chunk i, one at a time, in the following order: $1, 2, \ldots, N, N-1, \ldots, 2, 1$. In the first half of the sweep (increasing i), the neighbors j in each update chunk i are restricted to those for which $j > i$; in the second half (decreasing i), the neighbors are restricted to $j < i$.

The intuition for using a serial schedule rather than the simpler parallel (synchronous) one comes from considering the fact that, for a 1-D Markov chain, BP requires less CPU time to converge using serial updates (much like dynamic programming) rather than parallel updates. Although our graphs are loopy, there is significant chain-like structure in them, which led us to apply this intuition to our algorithm. However, determining an efficient message update schedule is a difficult problem and a topic of ongoing research.

5 Results

We tested the DQ modification to BP on our graphical deformable template models applied to real images. Four template shapes were tested: the letters A and B, a car shape and a cat shape. For the first two templates the images were grayscale images of a whiteboard with handwritten characters; street scene images were used for the car template and close-up images were used for the cat template. The original images, which ranged from about 400 x 300 to 2000 x 1500, were downsampled by a factor of 3 in both dimensions before the image gradient information was computed. Aside from the scale of the target shape, which was chosen manually for each image, no other form of user initialization was required.

Figure (4) shows typical detection results obtainable for the letter A template *without* DQ, demonstrating the deformable template's ability to automatically find a correct match even in the presence of substantial clutter, as well as its rotation invariance and robustness to local shape deformations.

We tested the ability of DQ to allow the algorithm to recover from gaps in the target shapes. An example of this capability is shown in Figure (5). Pixel locations in the gap are not represented in the state spaces $\{S_i\}$ at the beginning of BP, but the DQ procedure is able to "fill" the gap using the prior knowledge of the shape prior. Although the locations in the gap have substantially weaker edge evidence than locations along the existing target edges, these locations are favorable as continuations of the existing edges near them. Once states corresponding to these gap locations are resurrected in the course of BP, subsequent message updates establish that they are probable locations given the context of the entire target shape.

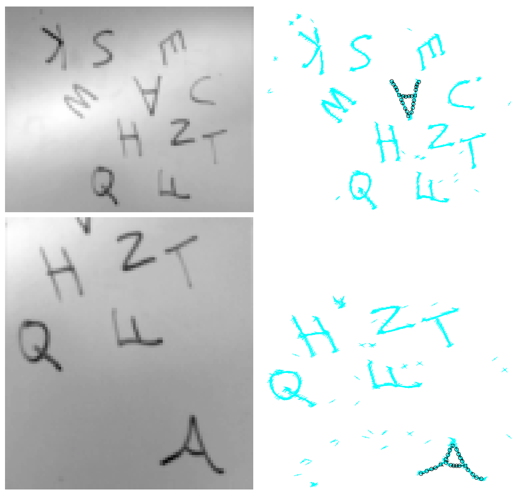
Fig. 4. Typical detection results without DQ. Original images on left, solutions superimposed in black on right. Top row example demonstrates rotation invariance of deformable template, and bottom row example shows robustness to local shape deformations.

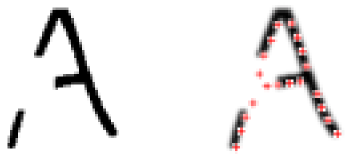
Fig. 5. Robustness of template to partial occlusion. (a) Image contains an A missing its middle left corner and portions of adjoining segments. (b) DQ is able to recover the missing points (red and white crosses), enabling a successful detection.

We also demonstrate that DQ is able to fill in gaps on targets surrounded by clutter, as shown in Figure (6). When clutter is present, many spurious states corresponding to non-target locations are added by DQ in attempts to explore possible continuations of features resembling letter A parts. (E.g. parts of the letter X that resemble the sides of the letter A are likely to spawn states corresponding to a non-existent centerline between the sides of the A.) However, subsequent BP message updates rule out such states as improbable, since they correspond to matches with even less edge support than the true target.

Additional results are demonstrated in Figures (7), (8), (9), and (10). The letter B example in Figure (7b) demonstrates a successful match for a partially occluded target in clutter; note that the solution deviates from some of the visible edges of the target, reflecting the influence of the prior (reference) shape. The car example in Figure (8) is based on a simple contour model of the tops of both wheels and the chassis between them. The algorithm finds a satisfactory match in the presence of considerable clutter (Figure (8), bot-

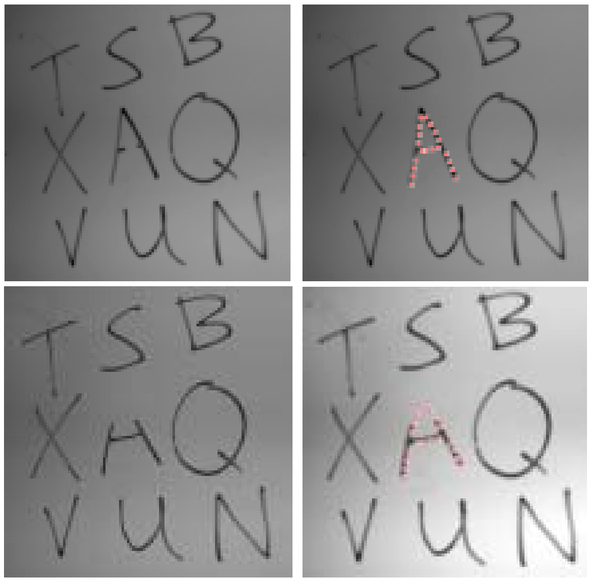

Fig. 6. Robustness of template to partial occlusions in presence of clutter. Top row: (a) Image contains an A missing part of its centerline, with solution in (b). Bottom row: (a) Image contains an A missing part of its top, with solution in (b).

tom) show the edges selected by pre-pruning) and occlusions (see Figure (9)). Finally, we demonstrate a simple head-on cat head template based on the contour of the two ears and the portion of the head between them. An edge-based contour representation is less appropriate for this object than for the others we modeled, since it is difficult to extract clean edges from the cat's furry silhouette (see Figure (11)). However, Figure (10) shows some successful matches, including one with an occlusion.

We illustrate the evolution of the total number of states during the course of BP/DQ with another deformable template that represents a crosswalk by the ends of two adjacent stripes. Figure (12) shows the result of matching this template to an image of a partially occluded crosswalk, and plots the total number of states (summed over all the nodes) over time. In this example, DQ is only applied twice – the only two points at which the total number of states increases. Note the dramatic drop-off in the number of states in the first half of the time sequence. In this case some important states are incorrectly pruned out, and DQ is necessary to resurrect them so that the template can be matched correctly.

Execution times were on the order of tens of seconds on a standard desktop PC running a C++ implementation of the algorithm. We emphasize that there is no need for the user to initialize the template near the target shape, since BP considers all edge pixels across the entire image to be equally likely a priori, and the template is invariant to global rotation and translation.

The scale of the target is assumed to be known; however, it would be straightforward (but slow) to apply the algorithm to a range of possible scales and choose the scale that results in the maximum posterior probability. (A faster method might entail the use of features richer than local edges, such as straight line segments, which would individually provide approximate evidence for a particular scale, and which would collectively determine the correct scale.) Also, our algorithm currently assumes there that is exactly one instance of the target in the image. Techniques for finding the top N local maxima of the posterior, such as those used in [35] and [7], may be used to handle multiple instances of targets. If the maximum posterior probability determined by DQ is sufficiently low, this indicates that no targets are present in the image. The choice of threshold requires experimentation, in order to cope with inaccuracies in the model and approximations introduced by BP and DQ, so this is a topic for future research.

A simple experiment was performed to compare DQ with the "brute-force" alternative of quantizing the state spaces on regular grids (24 possible orientations at every pixel) and running ordinary BP with pruning; for a very small image, DQ was about two orders of magnitude faster and produced results comparable to BP. Moreover, in contrast with an earlier DP-based deformable template that handles occlusions [7], the DQ-based algorithm does not rely on the existence of a corner or T-junction to signal the absence of an edge (i.e. due to the presence of an occluding boundary), as shown in many of the results in which missing edges are not signaled in this way (Figures 5,6,7).

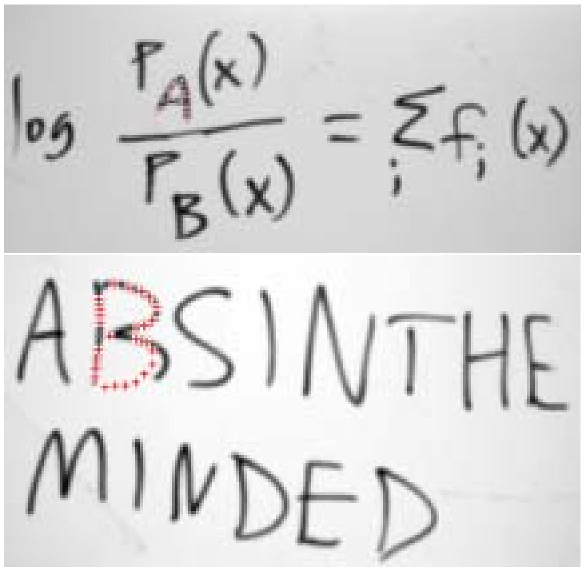

Fig. 7. Additional A result, and sample result for B template.

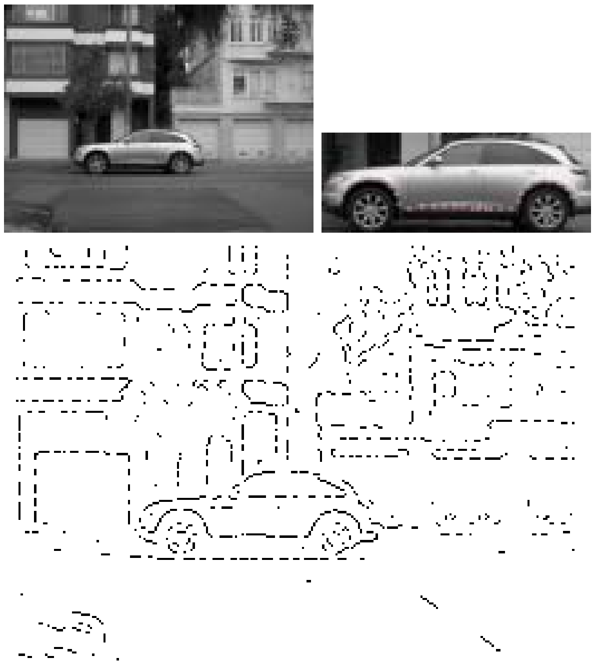

Fig. 8. Car template. Top: solution shown superimposed on original image on left, and zoomed in on right. Bottom: edge pixels.

6 Conclusions

DQ is a promising new enhancement of BP. It extends standard pruning techniques, allowing BP to adaptively add as well as subtract states as needed. In the case of deformable template matching, DQ allows BP to focus on the more probable regions of the image, so that state spaces can be adaptively enlarged to include locations where features are occluded, without the computational burden of representing all possible pixel locations. As a result, deformable template matching by BP is able to fill in gaps in target shapes in reasonable amounts of time. Although DQ is presented in the context of deformable template matching, the technique should apply to inference on any graphical model in which the pair-wise potentials are sparse (i.e. the state of one node strongly constrains the possible states of neighboring nodes) and the state spaces are fairly low in dimension (perhaps three or lower).

Acknowledgements

We would like to thank Drago Anguelov for his useful comments and suggestions, and Fusilli Campbell for graciously agreeing to pose for us. Both authors were supported by the National Institute on Disability and Rehabilitation Research grant number H133G030080, NSF grant number IIS0415310 and the Smith-Kettlewell Eye Research Institute RERC grant.

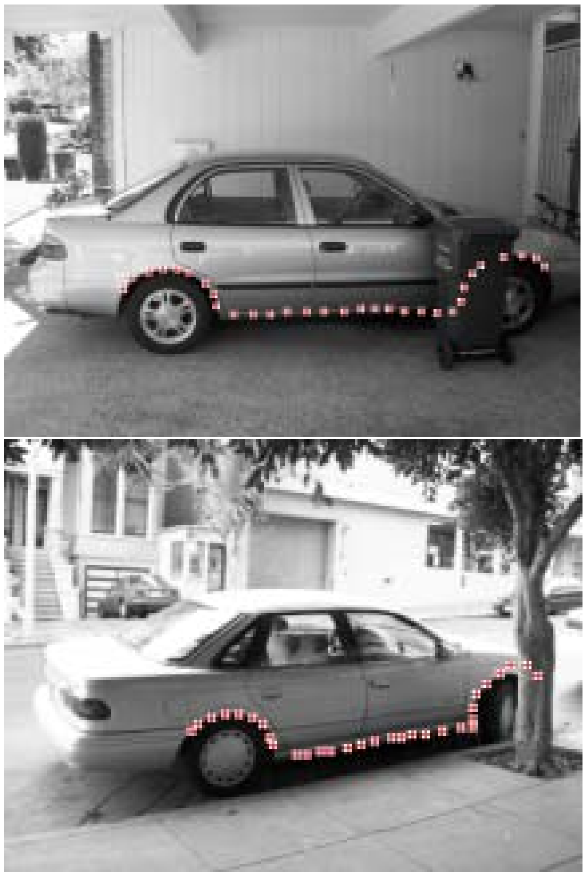

Fig. 9. Car template results with occlusions.

References

[1] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, H. Pang and J. Davis. "The Correlated Correspondence Algorithm for Unsupervised Registration of Nonrigid Surfaces." Technical Report TR-SAIL-2004-100. Stanford University, 2004.

[2] Y. Amit and A. Kong. "Graphical Templates for Image Matching." PAMI, Vol 18, No. 3, pp. 225-236. 1996.

[3] F.R. Bach and M.I. Jordan. "Thin Junction Trees." In Advances in Neural Information Processing Systems 14, NIPS 2001.

[4] M. Barzohar and D. B. Cooper, "Automatic Finding of Main Roads in Aerial Images by Using Geometric-Stochastic Models and Estimation," CVPR 1993. pp. 459-464.

[5] L. Cheng and T. Caelli. "Bayesian Stereo Matching." Computer Vision and Image Understanding. Special issue on Generative-Model Based Vision. This issue.

[6] T. F. Cootes and C. J. Taylor, "Active Shape Models - 'Smart Snakes'," *British Machine Vision Conference,* pp. 266-275, Leeds, UK, September 1992.

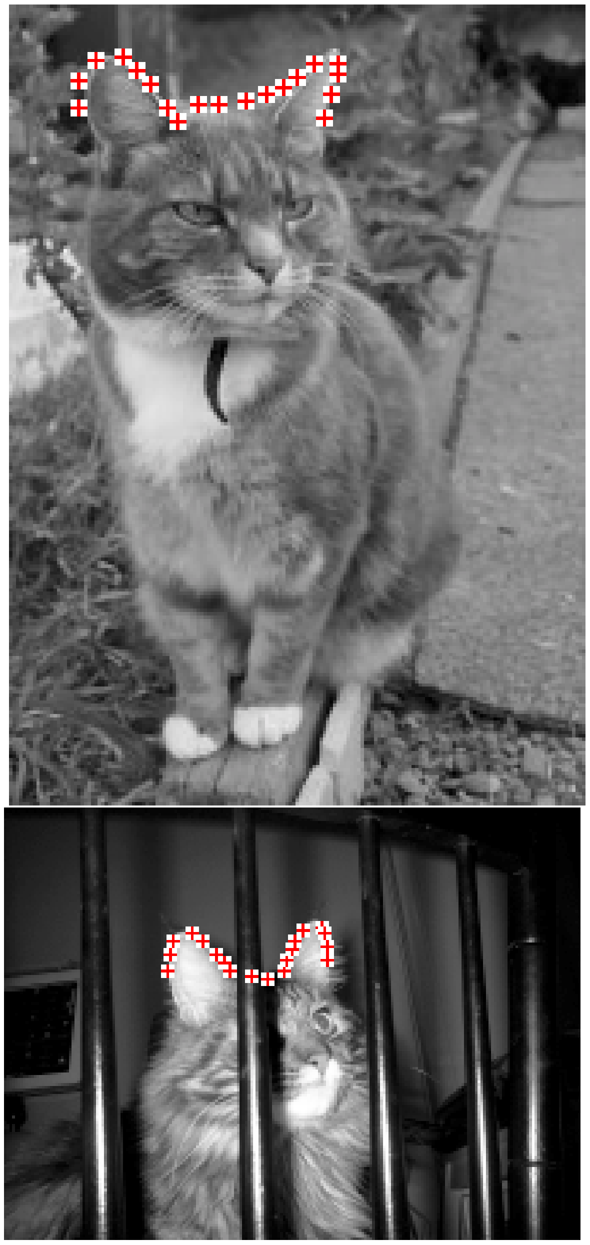

Fig. 10. Cat results. Note occlusion on bottom panel.

[7] J. Coughlan, A.L. Yuille, C. English, D. Snow. ”Efficient Deformable Template Detection and Localization without User Initialization.” Computer Vision and Image Understanding, Vol. 78, No. 3, pp. 303-319. June 2000.

[8] J. Coughlan and S. Ferreira. ”Finding Deformable Shapes using Loopy Belief Propagation.” The Seventh European Conference on Computer Vision (ECCV ’02). pp. 453-468. Copenhagen, Denmark. May 2002.

[9] P. Felzenszwalb and D. Huttenlocher. “Efficient Matching of Pictorial Structures.” CVPR 2000. pp. 66-73.

[10] P. Felzenszwalb. “Representation and Detection of Deformable Shapes.” CVPR

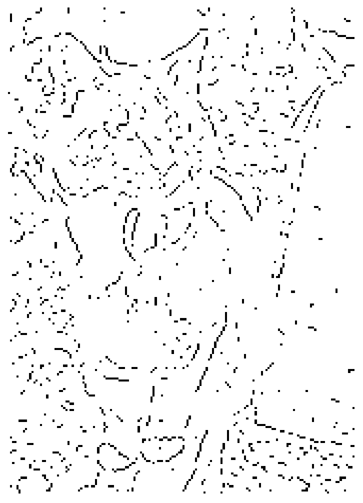

Fig. 11. Edges for image in top panel of previous figure.

2003. pp. 102-108.

[11] P. Felzenszwalb and D. Huttenlocher. “Efficient Belief Propagation for Early Vision.” IEEE Conference on Computer Vision and Pattern Recognition, 2004.

[12] P. Felzenszwalb and D. Huttenlocher. “Pictorial Structures for Object Recognition, Intl. Journal of Computer Vision, 61(1), pp. 55-79, January 2005.

[13] M.A. Fischler and R.A. Erschlager. ”The Representation and Matching of Pictorial Structures.” IEEE Trans. Computers. C-22. 1973.

[14] W.T. Freeman, T.R. Jones, and E.C. Pasztor. “ Example-based super-resolution.” IEEE Computer Graphics and Applications, March/April, 2002.

[15] J. Gao and J. Shi. “Inferring Human Upper Body Motion Using Belief Propagation.” Tech. report CMU-RI-TR-03-06, Robotics Institute, Carnegie Mellon University, June, 2003.

[16] W.R. Gilks, S. Richardson and D.J. Spiegelhalter. “Markov Chain Monte Carlo in Practice.” Chapman and Hall/CRC. 1995.

[17] D. Geman and B. Jedynak. “An active testing model for tracking roads in satellite images”. PAMI. Vol. 18. No. 1, pp. 1-14. January 1996.

[18] X. He, R.S. Zemel, M.A. Carreira-Perpinan. “Multiscale Conditional Random Fields for Image Labeling.” In CVPR, 2004. Vol. II, pp. 695-702.

[19] M. Isard. “Pampas: Real-Valued Graphical Models for Computer Vision.” In CVPR, 2003. pp. 613-620.

[20] S. Kumar and M. Hebert. “ Man-Made Structure Detection in Natural Images using a Causal Multiscale Random Field.” CVPR 2003.

[21] B. Lowerre and R. Reddy, “The Harpy Speech Understanding System.” Trends in Speech Recognition. Ed. W. Lea. Prentice Hall. 1980.

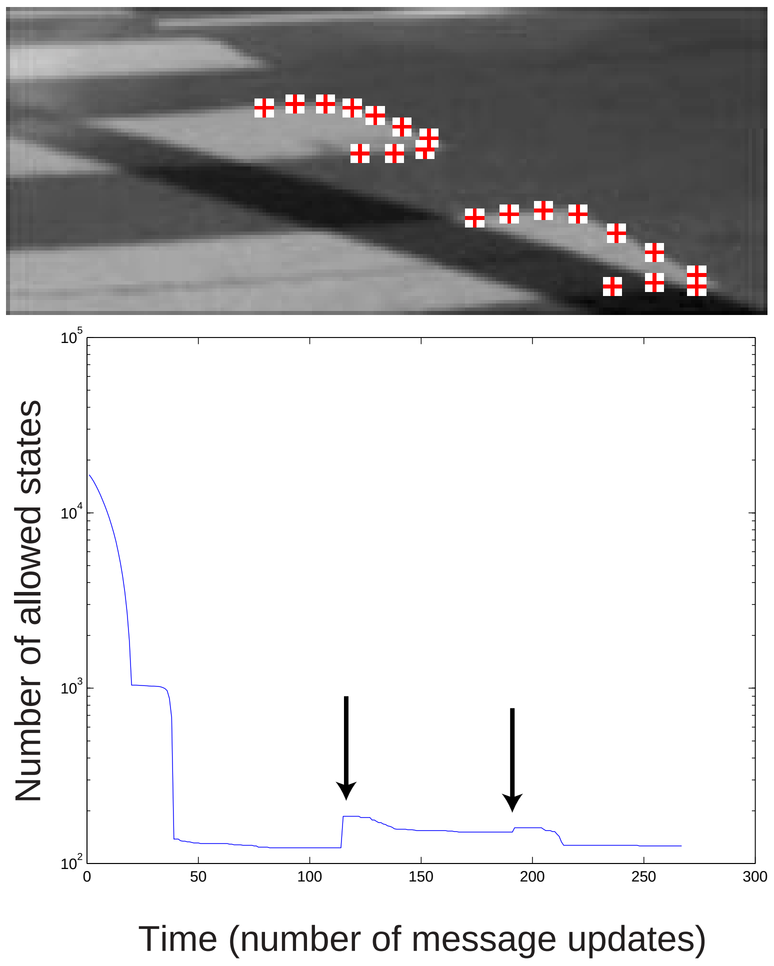

Fig. 12. Number of allowed states over time. Top, double-stripe crosswalk template with result shown on crosswalk partially occluded by shadow. Bottom, log-plot of number of states (total across all nodes) over time (one unit corresponds to one message update). DQ is applied twice, corresponding to the times when the number of states increases (indicated by black arrows).

[22] J. Marroquin, S. Mitter, and T. Poggio, "Probabilistic Solution of Ill–Posed Problems in Computational Vision." J. Amer. Stat. Assoc." Vol. 82, num. 397. March 1987, pp. 76-89.

[23] K.P. Murphy,Y. Weiss and M.I. Jordan. "Loopy belief propagation for approximate inference: an empirical study". In Proceedings of Uncertainty in AI. 1999.

[24] K. Murphy, A. Torralba, W. Freeman. "Using the Forest to See the Trees: A Graphical Model Relating Features, Objects and Scenes." NIPS'03.

[25] R. M. Neal, M. J. Beal and S. T. Roweis. (2004) "Inferring state sequences for non-linear systems with embedded hidden Markov models." NIPS'03.

[26] J. Pearl. Probabilistic Reasoning in Intelligent Systems. Morgan Kaufman. 1988.

[27] D. Ramanan and D. Forsyth. ”Finding and Tracking People from the Bottom Up.” CVPR 2003. pp. 467-474.

[28] A. Rangarajan, J. Coughlan and A. L. Yuille. A Bayesian network framework for relational shape matching. ICCV 2003. Nice, France. pp. 671-678. October 2003.

[29] S. Roth and M.J. Black. “Fields of Experts: A Framework for Learning Image Priors.” In CVPR 2005.

[30] L. Sigal, M. Isard, B. H. Sigelman, M. J. Black. “Attractive People: Assembling Loose-Limbed Models using Non-parametric Belief Propagation.” Advances in Neural Information Processing Systems 16, NIPS 2003.

[31] E.B. Sudderth, A.T. Ihler, W.T. Freeman, and A.S. Willsky. Nonparametric belief propagation. In CVPR 2003. pp. 605-612.

[32] E.B. Sudderth, M.I. Mandel, W.T. Freeman and A.S. Willsky. “Visual Hand Tracking Using Occlusion Compensated Message Passing.” Computer Vision and Image Understanding. Special issue on Generative-Model Based Vision. This issue.

[33] J. Sun, H. Shum, and N. Zheng. Stereo matching using belief propagation. ECCV 2002. pp. 510-524, 2002.

[34] Z.W. Tu and S.C. Zhu. ”Image segmentation by Data-driven Markov chain Monte Carlo.” PAMI, May 2002.

[35] C. Yanover and Y. Weiss. “Finding the M Most Probable Configurations using Loopy Belief Propagation.” Advances in Neural Information Processing Systems 16, NIPS 2003.

[36] A. L. Yuille, “Deformable Templates for Face Recognition”. *Journal of Cognitive Neuroscience.* Vol 3, Number 1. 1991.

[37] S. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. IJCV, 27(2):107126, 1998.