Dynamic Positional Trees for Structural Image Analysis

Amos J Storkey **Christopher K I Williams**
Institute of Adaptive and Neural Computation
Division of Informatics, University of Edinburgh
5 Forrest Hill, Edinburgh UK
a.storkey@ed.ac.uk *c.k.i.williams@ed.ac.uk*

Abstract

Dynamic positional trees are a significant extension of dynamic trees, incorporating movable nodes. This addition makes sequence tracking viable within the model, but requires a new formulation to incorporate the prior over positions. The model is implemented using a structured variational procedure, and is illustrated on synthetic raytraced images and image sequences.

We consider the problem of structural image analysis and in particular the inference of scene properties from image data. We are especially concerned with image decomposition, that is obtaining the characteristic parts of an image and the relationships between them. The components of an image are not independent of each other; certain objects are expected to occur together, and objects are made up of different subcomponents. One way of thinking of this problem is by analogy with parsing a language; we are interested in parsing images. However, the important characteristics and structure in an image is significantly different from linguistic data.

Those familiar with work on dynamic trees will be aware that they have been developed in the context of single static images [15, 1, 13]. It would be desirable if the benefits of the dynamic tree approach could also be made available for image sequences. Introducing a sequence model into the basic dynamic tree formalism is not straightforward as a change in the position of an object is reflected in a change in the connectivity structure of the dynamic tree. This change would be hard to predict from the previous time slice and would be an inelegant representation of the dynamics: the connectivity structure is supposed to represent the structural characteristics of an object, most of which will be preserved during movement. Here the dynamic tree is modified to incorporate position variables, resulting in a model where object movement can be represented in terms of a change in position components of the nodes representing that object.

The structure of the remainder of the paper is as follows. The first section of this paper develops some of the issues surrounding image analysis in general and then outlines the form of the dynamic positional tree, and the rationale behind its design. This leads in to a more formal definition of the dynamic tree model in section 2, and we discuss related models in section 3. Defining a Bayesian model is one thing, being able to implement it is another. In section 4, we take a variational approach to the implementation problem and give a set of structured variational approximations which can be calculated efficiently, and which have the structural information we need. The resulting set of update equations are given in section 5. Illustrations of the approach appear in section 7, after a brief discussion of the issue of sequences (section 6).

1 Dynamic Positional Trees

1.1 Pixel Models

When developing models for images, it is advisable to separate the concept of an image model from that of a pixel model. The former should develop a model of the scene, and the latter gives the method by which the scene is represented in terms of pixels. The latter depends on the characteristics of the camera or measurement process, the nature of the measurement noise etc. The scene description should not depend on a particular choice of pixellation. Having the measurement process dealt with by a separate model from that which deals with the structure of the scene ensures that this is the case. The above approach will be followed here. For now we will concentrate on the scene model, and will make the assumption that there is some pixel model which will relate the scene to the actual pixel image which the observer is presented with. Details of the pixel model are given in §1.6

1.2 Scene Structure

Dynamic positional trees are used to build representative tree structured belief networks for images. Like dynamic trees, dynamic positional tree structures are designed to represent the inherent relationships between and within objects in a scene. The following are some of the features which would be appropriate for image related structures:

• Locality: In general, child nodes will be spatially closer to the parent rather than far from the parent.

• Spatial coherence: Two nearby parts of an image are more likely to be related to one another than parts separated by a larger distance.

• Multiscale representation: Objects further up the hierarchy will have greater spatial extent than those further down the hierarchy.

Spatial coherence is a key feature, and is illustrated by figure 1. Like its predecessor [15] the dynamic positional tree can be seen as a mixture of tree structured belief networks. Each tree structure represents one set of relationships which might be useful in describing an image.

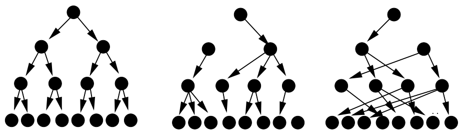

Figure 1: Various trees. Left: highest prior probability. Middle: high probability. Right: low probability. Think of the leaf node positions as representing pixels.

1.3 Node Properties

Before looking at the structure model in detail, it is important to describe what each node of the belief network represents. It is these nodes which denote the characteristic variables needed to generate the image. In this implementation they must represent the following random variables:

• Class labels. These label the type of object or object characteristic which the node is representing.

• Positions. Two dimensional real variables representing the position of the node in image space.

In addition variables/parameters representing spatial extent would be useful, and for sequences, momenta would also be necessary. These momenta would describe what movement would be expected between time frames.

1.4 Structure and Position

The prior over possible network structures and the prior over node positions must be defined together. The belief networks with the highest prior probabilities will have spatial coherence, but positions are spatial variables, and so the positions cannot be ignored when defining the structure. Likewise the prior on positions will depend on the network structure. Different structures will produce different localisation requirements when it comes to defining the node positions. We cannot escape the need for these concepts to be dealt with together.

One way of defining a prior over structure and position involves using a distribution similar to that introduced in [14] as a hierarchical Gaussian mixture model.

Suppose we have a network with a given number of layers, and that the number of nodes in each layer is fixed. In the hierarchical Gaussian mixture model each node is allowed to choose its parent uniformly from the nodes in the layer above. This defines a tree structure. Given this tree structure, the probability of the position $\mathbf{r}_i$ of each node i is given by a Gaussian distribution centred at the position of the parent node (see figure 2). The result of this model is far from uniform or regular at the leaf nodes. In fact this approach was developed specifically to cater for clustering of the leaf node values. This is a problem for images as it means the model would not describe large parts of the image space, whereas we expect every part of an image to relate to some object. However, the standard hierarchical Gaussian mixture model can be modified to give a distribution with the required structure. The approach used here is to condition the hierarchical Gaussian mixture model on the fact that the leaf nodes are on a regular grid.

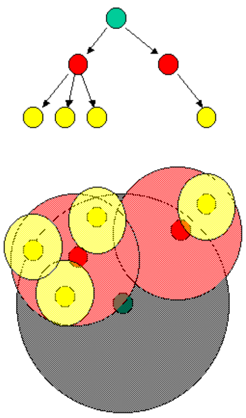

Figure 2: The first few layers of the hierarchical Gaussian mixture. The top picture give the parent/child structure, and the bottom picture gives the node centres and variances.

1.5 Class Label Model

The main remaining component of the scene model involves the relationship between the class labels of the nodes. A sample from the structure-position model defines a tree structured belief network. Hence conditioned on the structure-position, each node has a known parent. This parent-child relationship represents the component-subcomponent relationship between objects in an image. Hence we need some conditional probability which says how likely the child node is to have a certain label given that the parent has a particular label. The only constraint we make here is that this conditional probability is always the same for this child/parent pair whatever the state of the other connections.

1.6 Pixel Model

We can now return to the pixel model, and relate the leaf nodes of the dynamic positional tree with the pixel RGB values. The simplest form of pixel model generates the pixels from the state of the leaf nodes whose position lies within the pixel area. Then what is needed is a model of pixel colour intensity conditioned on node class label. The resulting colour intensities can be averaged over the leaf nodes within a pixel area to give the intensity for that pixel. The simplest case is where we arrange for there to be one leaf node per pixel. Sometimes more textural information can be utilised if multi-pixel regions are used instead of pixels.

There are many ways that the class conditional model of pixel colour intensity can be obtained. We have examined a number of approaches including modelling scaled likelihoods for multi-pixel regions using neural network methods [16], and simple approaches, such as using empirical (histogram) class conditional distributions.

2 Theory

In this section we define the dynamic positional tree model explicitly. We denote the set of network nodes by $N = \{1, 2, \dots, n\}$. The nodes are organised into layers, and the bottom layer of nodes are the leaf nodes, denoted by L. The remainder are denoted by L'.

We denote the position value of node i by the random variable R_i which takes vector values $\mathbf{r}_i$, one component for each dimension of the system (2 for an image). We denote the class state of the node i by random variable X_i which takes vector values $\mathbf{x}_i$, one component for each of C possible object labels. The indicator x_i^k is zero for all k except if node i is of class k, when $x_i^k = 1$. We generally use the superscript notation to represent a set of random variables. For example X^B represents the set $\{X_i | i \in B\}$, and R^B represents $\{R_i | i \in B\}$. For the state of all nodes we drop the N: $X^N = X$.

The tree connectivity is given by $Z = \{Z_{ij} | i, j \in N\}$, where Z_{ij} is a random variable which takes indicator values $z_{ij} = 1$ if node j is the parent of node i, and $z_{ij} = 0$ otherwise. We can allow node i to disconnect from all parents; this disconnection state is denoted by $z_{i0} = 1$; $z_{ij} = 0\ \forall j \neq 0$. Finally, the set of pixels (which will be our data) is denoted by Y.

2.1 The Node Position Model

In section 1.4 it was argued that the distribution $P(X, R)$ of node position and connectivity need to be defined together. We do that using a conditional hierarchical Gaussian mixture model.

2.1.1 Hierarchical Gaussian Mixture Models

We define the distributions over structure, $P(Z)$, and position, $P(R|Z)$, as follows. First the distribution over trees $P(Z)$ is given by $P(Z) = \prod_{ij} \gamma_{ij}^{z_{ij}}$ where γ_{ij} is a prior probability of i connecting to j. This simply says that each node in can choose its parent from one of the nodes in the layer above. Usually the probability γ_{ij} is taken to be uniform over all parents j, with an additional low probability of the node choosing to be a root.

The other term $P(R|Z)$ is given by $P(R|Z) = \prod_{i,j|z_{ij}=1} P(\mathbf{r}_i|\mathbf{r}_j)$ with

$$P(\mathbf{r}_i|\mathbf{r}_j) = \frac{1}{\zeta} \exp\left(-\frac{1}{2}(\mathbf{r}_i - \mathbf{r}_j - \boldsymbol{\eta}_{ij})^T \Sigma_{ij}^{-1} (\mathbf{r}_i - \mathbf{r}_j - \boldsymbol{\eta}_{ij})\right) \tag{1}$$

and where the normalisation constant $\zeta = (2\pi)|\Sigma_{ij}|^{1/2}$. Here Σ_{ij} is a given covariance matrix, and $\boldsymbol{\eta}_{ij}$ is an offset, usually set to zero. Note that z_{ij} is non-zero for only one value of j, which must be in the layer above that containing node i. More informally $P(R|Z)$ is formed by generating the positions of the next layer from a Gaussian centred at the position of the parent.

The position of a root node i is chosen independently from a Gaussian mean $\mathbf{r}_{i0}$ and variance σ_{i0}. We generally take $\mathbf{r}_{i0} \equiv \mathbf{r}_0$ to be zero, and σ_{i0} to be large. In other words root positions are chosen from a relatively broad Gaussian.

This hierarchical Gaussian mixture model has many of the components which we want. It has component/subcomponent structure, spatial coherence and hierarchical form. It remains to ensure that every part of the image is properly described by some component of the model. To enforce this, the leaf nodes R^L are taken to be fixed in a suitable grid. The distribution of the remaining variables, $P(Z, R^{L'}|R^L)$, is given by the conditional distribution

$$P(Z, R^{L'}|R^L) = P(Z|R^L) P(R^{L'}|Z, R^L). \tag{2}$$

This gives the final prior in equation (2).

2.2 Node Labels and Pixel Model

The hierarchical Gaussian mixture model of the previous section needs to be combined with some distribution over X to get the full dynamic positional tree prior.

Including these positions, and a pixel model, the overall prior model (again given fixed R^L) can be written as

$$P(Z, X, R^{L'}, Y|R^L) = P(Z, R^{L'}|R^L)P(X|Z)P(Y|X, R). \quad (3)$$

$P(Y|X, R)$ is the pixel model and determines how the object structure is represented in terms of pixels Y. The original dynamic tree model appears here through the distribution of the node states $P(X|Z)$ which is to be

$$P(X|Z) = \prod_{i,j} (P_{ij}^{kl})^{x_i^k x_j^l z_{ij}}$$

where P_{ij}^{kl} is the probability that node i is in state k given that j is the parent of i and node j is in state l. Note again the power of z_{ij} only picks out one of the j elements (the parent) for each i in the product.

Finally we want to choose some form for the pixel model. The simplest form assumes a one to one relationship between pixels Y_i and leaf nodes X_i $i \in L$, and takes $P(Y|X, R) = P(Y|X^L) = \prod_{i \in L} P(Y_i|X_i)$, meaning that the pixel representation comes directly from the lowest level object representation.

In generative terms we choose a structure Z and positions $R^{L'}$ according to the model $P(Z, R^{L'}|R^L)$. Then the object class labels are generated according to $P(X|Z)$. Finally the pixel values are obtained from $P(Y|X, R)$. This gives the full prior model of the image in terms of a position encoding dynamic tree. This prior model is a rather complicated mixture of trees, and so we would not expect to be able to calculate the posterior exactly. In section 4 we develop a structured variational approach.

3 Related Work

The general aim of our work is to provide a prior distribution which is spatially coherent, giving rise to object-like groupings of pixels. There are a number of other approaches to this problem; two of the best known are tree-structured belief networks (TSBNs) and Markov random fields (MRFs). Below we discuss these methods, and their relative strengths and weaknesses.

In tree-structured belief networks, the leaves will be taken as pixels; the higher levels of the tree induce correlations between the leaf nodes. TSBNs using discrete-valued nodes [4, 10] and Gaussian nodes [2, 11] have been investigated. These architectures are tree-structured analogues of the linear hidden Markov model and Kalman filter respectively. TSBNs have the advantage that inference calculations can be carried out efficiently (using upward-downward algorithms analogous to forward-backward algorithms on chains). However, they have a rigid architecture (often of quad-tree type) that is unresponsive to the image content. This can give rise to "blocky" artefacts in image generation/analysis. We also note that DeBonet and Viola [6] have used an interesting tree-structured network for image synthesis using non-Gaussian densities. In this work the higher levels correspond to wavelet coefficients and are observable rather than hidden variables.

Markov random field models [3, 8] are undirected graphical models that define a stationary process (thereby overcoming problems of blockiness). They have two disadvantages (i) they are non-hierarchical and (ii) inference in such a MRF is NP-hard in general.

The Dynamic Tree (DT) architecture seeks to gain the advantages of the hierarchical structure of the TSBN whilst overcoming the disadvantage of its rigid structure. It does this by defining a prior distribution over trees; conditional on the tree structure (denoted by Z), the network is a TSBN. Typically there are a very large number of possible trees in the prior; in response to data the posterior distribution will be re-weighted to favour those architectures most consistent with the image data. Dynamic tree architectures were introduced in [15] and in [9].

One attractive feature of the DT is that disconnections can occur, giving rise to a 'forest' of more than one tree. The roots can be interpreted as identifying individual objects; an object is defined by all of those nodes which are children of a root. Notice that this interpretation is not possible in a single TSBN. In [8] an edge process was introduced to the MRF allowing explicit disconnections. However, in contrast to the dynamic tree architecture there is no guarantee that the prior over the edge process will produce a nice set of regions.

The rich variety of trees generated in the DT architecture is reminiscent of the parse trees in context free grammars (CFGs), although the DT models are constrained to have a fixed number of layers. There is a $O(n^3)$ algorithm for evaluating the MAP parse etc in CFGs; however, this algorithm depends crucially on a one-dimensional ordering of inputs and thus cannot be applied to 2-d analyses.

The construction of a number of belief networks dependent on a variable Z has been used in the work of Geiger and Heckerman on multinets [7]. In that work multinets were used as a way of speeding inference, as conditional on Z the networks will typically be much simpler than an equivalent network which ignores this conditioning. In our work, integrating out Z leads to a network which is layerwise fully connected. Such fully-connected models have been used before e.g. the sigmoidal belief network model used in the Helmoltz machine [5], but we believe that the 'clean' semantics of the DT model (where we expect that each pixel should belong to one object) should aid the interpretability and utility of the model.

4 Variational Approach

Exact inference using propagation methods is not feasible in this network, and so a variational approach is used. This develops and extends the approach used in [13] to the new case of the positional dynamic tree. This approach involves approximating the posterior distribution with a factorising distribution of the form $Q(Z)Q(X|Z)Q(R^{L'})$, where $Q(Z)$ is the approximating distribution over the Z variables, $Q(X|Z)$ is the approximating distribution over the states, and $Q(R^{L'})$ is an approximating distribution over the non-leaf node positions.

To choose good forms for the Q's the Kullback-Liebler divergence between the $Q(Z)Q(X|Z)Q(R^{L'}|Z)$ distribution and the true posterior should be minimised. In fact the approximate distributions which are used take the form of a dynamic tree model, and give propagation rules which are efficient and local. Similar approximations used for the basic dynamic tree can be seen in [13].

The KL divergence between the approximation and the true posterior is of the form

$$\int_{R^{L'}} dR \sum\nolimits_L' Z, X Q(Z)Q(X|Z)Q(R^{L'})$$
$$\log\left(\frac{Q(Z)Q(X|Z)Q(R^{L'})}{P(Z,R^{L'}|Y,R^L)P(X|Z,Y,R^L)}\right). \quad (4)$$

We now need to discuss the forms of each of the approximating distributions. We use a $Q(Z)$ of the form

$$Q(Z) = \prod_{ij} \alpha_{ij}^{z_{ij}}$$

with parameters α. $Q(R^{L'})$ takes the form $Q(R^{L'}) =$

$$\prod_{i \in L'} \frac{1}{\sqrt{(2\pi)|\Omega_i|}} \exp\left(-\frac{1}{2}(\mathbf{r}_i^T - \boldsymbol{\mu}_i^T)\Omega_i^{-1}(\mathbf{r}_i - \boldsymbol{\mu}_i)\right)$$

where $\boldsymbol{\mu}_i$ and Ω_i are position and covariance parameters respectively. In this paper Ω_i is assumed to be diagonal. Lastly the $Q(X|Z)$ is a dynamic tree approximation of a form identical to that used in [13]:

$$Q(X|Z) = \prod_{ijkl} (Q_{ij}^{kl})^{x_i^k x_j^l z_{ij}}.$$

Again Q_{ij}^{kl} are parameters to be optimised.

5 Update Equations

We want to minimize the KL divergence (4), with the forms of approximate distribution given in the last section. We need to do this subject to contstraint that $\sum_k Q_{ij}^{kl} = 1$ (probabilities sum to 1). We add to (4) a set of Lagrange multiplier terms corresponding to these constraints, and set the derivatives to zero. Solving this gives the following set of update equations.

5.1 Class Labels

Minimizing the KL divergence gives us a set of update equations. Given all the α's, let m_i^k be given recursively from the top down by

$$m_i^k = \sum_{jl} \alpha_{ij} Q_{ij}^{kl} m_j^l.$$

Then m_i^k is the marginal probability of node i being in class k under the variational distribution. Again given the α's we find that minimization of the KL divergence gives

$$Q_{ij}^{kl} = \frac{P_{ij}^{kl}\lambda_i^k}{\sum_{k'} P_{ij}^{k'l}\lambda_i^{k'}} \text{ where } \lambda_i^k = \prod_{c \in c(i)} \left[\sum_g P_{ci}^{gk}\lambda_c^g\right]^{\alpha_{ci}}.$$

In the last equation $c(i)$ is used to denote the possible children of i, in other words the nodes in the layer below that containing node i.

Hence given α all the Q can be updated by making a single pass up the tree to calculate the λ values, and then calculating the Q. In fact calculating the Q values themselves can be avoided completely as the marginal m values can be obtained directly from the λ and the prior P values.

5.2 Positions

The update equations for the positions (again given the α's) take the following forms

$$\boldsymbol{\mu}_i = \sum_j (\alpha_{ji}(\Sigma_{ji})^{-1} + \alpha_{ij}(\Sigma_{ij})^{-1})\boldsymbol{\mu}_j,$$
$$(\Omega_i)_{pp} = \frac{1}{\sum_j \left(\alpha_{ij}(\Sigma_{ij})_{pp}^{-1} + \alpha_{ji}(\Sigma_{ji})_{pp}^{-1}\right)}$$

where we have assumed that both Σ and Ω are diagonal. The equations for μ need to be iterated until suitably converged.

5.3 Connectivity

Lastly the connectivity needs to be considered. For fixed parameters in $Q(X|Z)$ and $Q(R)$ of the forms given above, we obtain

$$\alpha_{ij} \propto \gamma_{ij} \exp(\Xi_{ij}) \exp(\Phi_{ij})$$

where

$$
\begin{aligned}
\Xi_{ij} &= \sum_l m_j^l [\log \sum_k P_{ij}^{kl} \lambda_i^k] \qquad \text{and} \\
\Phi_{ij} &= \frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j - \rho_{ij})^T \Sigma_{ij}^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j - \rho_{ij}) \\
&+ \frac{1}{2}\mathrm{Tr}(\Sigma_{ij}^{-1}\Omega_i) + \frac{1}{2}\mathrm{Tr}(\Sigma_{ij}^{-1}\Omega_j).
\end{aligned}
$$

In the above the constant of proportionality is found by normalisation.

5.4 Optimization Process

The above equations give all the necessary update rules. The whole optimization process involves an outer loop optimizing the $Q(Z)$ values and an inner loop containing up and down passes of the $Q(X|Z)$ optimzation and a number of passes of the $Q(R)$ optimization. The KL divergence can be calculated up to an additive constant, and so can be used as an explicit objective function and be monitored accordingly.

There are a few hidden problems in the optimzation. Most of the updates are inexpensive. However there is the issue of summing over all possible children/parents. Most of these will give negligible contributions to the relevant sums because their contribution contains a probabilistic factor from the tail of a Gaussian. Hence grid based methods are used to index positions and thereby reduce the number of references to z_{ij} components which are irrelevant. This keeps computations down to something near the order of the number of nodes.

Learning This variational method gives a lower bound to the log likelihood. The lower bound can be used in the way described in [12] to optimise the parameters of the actual distributions using a form of EM algorithm. This approach was used here for learning the conditional probabilities P_{ij}^{kl}. It was assumed that these conditional probabilities were the same for all nodes in the same layer.

6 Sequence Model

Given the dynamic positional tree formalism, we are able to develop a model of sequences. We consider a Markovian model, where the dynamic positional tree posterior at the previous time step influences the prior at the next step. There is not space to give full details of the model here. However, the form chosen sets all of the approximating distributions at time $t+1$ to be similar posterior at time t, but at the same time allowing for some change in the structure, some movement of the position values, or some change in the class label. This form also allows the information from the approximate distribution to then be filtered through the Markovian dynamics, and obtain a model of the similar form to the prior of the single image model. Hence this process can be repeated for as many images as there are in the sequence.

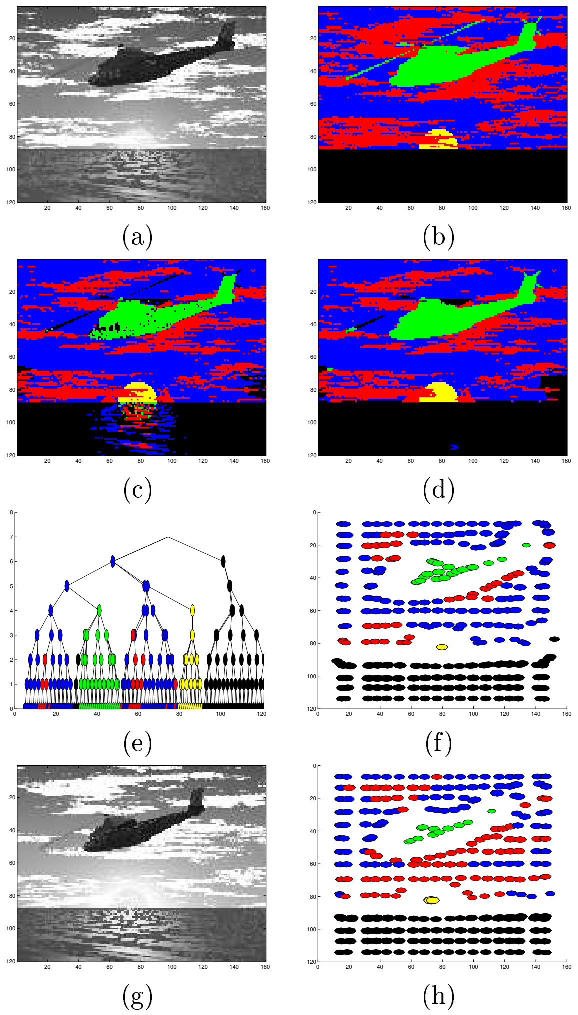

Figure 3: A test example. (a) The image, and (b) the ground truth labelling. (c) The pixelwise labelling and (d) the positional dynamic tree labelling. (e) A slice projection of the highest posterior dynamic tree (from down the middle of the image) and (f) the positions and labels of the sixth layer of the tree. (g) gives the next image in the sequence, while (h) gives shows where the nodes in the sixth layer move to.

7 Illustrations

The dynamic positional tree model was tested on artificial ray traced images. There were 6 training images, each with a ground truth label set of 5 class labels, one for each of sky, cloud, helicopter, sea, sun. The images were 160 by 120 pixels and were used to learn the conditional probabilities (CPTs) $P(X_i|X_j)$. The CPTs were assumed to be the same for all nodes in a given layer. The training images were also used to build a simple empirical pixel model: the RGB colourcube was partitioned into 64 subcubes, and the histogram of the pixel values was used as the class conditional probabilities for each label class. The standard deviations of the Gaussian distributions in the Gaussian affinity model were set by hand to be of a suitable width: one which generally gave a few (10 to 20) possible choices of parent for a node.

One image can be seen in figure 3a along with the ground truth segmentation (3b). Below that, the pixelwise segmentation without the use of the dynamic positional tree can be seen in figure 3c. The picture in 3d gives the maximum posterior segmentation obtained using the variational approach on the dynamic positional tree. This picture only gives a crude picture of the overall posterior distribution. Note the difference in form between the solid objects and more etherial ones such as clouds. Figure 3e gives a projection of a slice of the maximum posterior tree structure obtained (we actually have a distribution over trees), while figure 3f gives a picture of the positions and labels of nodes in the sixth layer from the root (out of nine) of the posterior dynamic positional tree. Given a second image (figure 3g) in sequence with the first, we can see what happens to the node positions after passing through the sequence model in figure 3h.

8 Discussion

The dynamic positional tree model enables the possibility of using dynamic tree like structures for image sequences. However it also has the benefit that the structures obtained can be interpreted in terms of objects, where the position labels relate to the position of the object and each of the parts of the object. Dynamic positional trees go beyond simple segmentation methods, and move towards structural scene decomposition.

Acknowledgements

This work is supported through EPSRC grant GR/L78161 *Probabilistic Models for Sequences.*

References

[1] N. J. Adams, A. J. Storkey, Z. Ghahramani, and C. K. I. Williams. MFDTs: Mean field dynamic trees. Proceedings of International Conference on Pattern Recognition 2000, 2000.

[2] M. Basseville, A. Benveniste, K. C. Chou, S. A. Golden, R. Nikoukhah, and A. S. Willsky. Modelling and estimation of multiresolution stochastic processes. *IEEE Transactions on Information Theory*, 38:766–784, 1992.

[3] J. Besag. On the statistical analysis of dirty pirtures. *Journal of Royal Statistics, Soc. B*, 48(3):259–302, 1974.

[4] C. A. Bouman and M. Shapiro. A Multiscale Random Field Model for Bayesian Image Segmentation. *IEEE Transactions on Image Processing*, 3(2):162–177, 1994.

[5] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel. The Helmholtz Machine. *Neural Computation*, 7(5):889–904, 1995.

[6] J. S. de Bonet and P. A. Viola. A Non-Parametric Multi-Scale Statistical Model for Natural Images. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 773–779. MIT Press, Cambridge, MA, 1998.

[7] D. Geiger and D. Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82:45–74, 1996.

[8] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Analysis and Machine Intellligence*, 6(6):721–741, 1984.

[9] G. E. Hinton, Z. Ghahramani, and Y. W. Teh. Learning to Parse Images. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 463–469. MIT Press, Cambridge, MA, 2000.

[10] J-M. Laferté, P. Pérez, and F. Heitz. Discrete Markov Image Modelling and Inference on the Quadtree. *IEEE Transactions on Image Processing*, 9(3):390–404, 2000.

[11] M. R. Luettgen, W. Karl, and A. S. Willsky. Efficient multiscale regularization with applications to the computation of optical flow. *IEEE Trans. Image Processing*, 3:41–64, 1994.

[12] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 353–358. Kluwer Academic Publishers, 1998.

[13] A. J. Storkey. Dynamic trees: A structured variational method giving efficient propagation rules. In C. Boutilier and M. Goldszmidt, editors, *Uncertainty in Artificial Intelligence*, pages 566–573. Morgan Kauffmann, 2000.

[14] C. K. I. Williams. A MCMC approach to hierarchical mixture modelling. In *Advances in Neural Information Processing Systems 12*, pages 680–686. MIT Press, 2000.

[15] C. K. I. Williams and N. J. Adams. DTs: Dynamic trees. In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11.* MIT Press, 1999.

[16] C. K. I. Williams and X. Feng. Combining neural networks and belief networks for image segmentation. In T. Constantinides, S-Y. Kung, M. Niranjan, and E. Wilson, editors, *Neural Networks for Signal Processing VIII.* IEEE, 1998.