Position Encoding Dynamic Trees for Image Sequence Analysis

Stephen N. Felderhof, Amos J. Storkey, Christopher K.I. Williams
Division of Informatics,
University of Edinburgh.
s.felderhof@ed.ac.uk

21st December 2001, revised 9th January 2002

Abstract

Position Encoding Dynamic Trees have been demonstrated to be useful models for individual images. In the theory and experiments presented here, we show that the PEDT models can be used for the analysis of ordered sequences of images. We provide a theoretical framework in which to update the current prior model sequentially using the variational approximation to the posterior probability distribution from the previous time frame. We present experimental results demonstrating improved segmentation performance using the new approach.

1 Introduction

Position Encoding Dynamic Trees (PEDTs) are a form of hierarchical graphical model with adaptable structure in which nodes of the graph have both class and position variables. They are described in [14] and [13]. In this paper we describe an extension of the PEDT model to deal with the analysis of image sequences. Standard probabilistic models for sequences include Hidden Markov models (see, e.g. [10]) and Linear Dynamical systems or Kalman Filters (see, e.g.[17]). These two models have attractive polynomial-time algorithms but have a relatively simple latent variable structure. Our aim is to develop more powerful multiple-cause latent variable graphical models for sequential problems, particularly image sequence analysis.

The key idea in sequential modelling is that information at time $t-1$ should be predictive of the information at time t. Within a probabilistic framework, this means that one's model for an image at time t should be influenced by previously observed evidence. As with the HMM and LDS models we use a Markov-type model for the evolution of the state variables. Within the PEDT framework it is not possible to compute exactly the predictive distribution of the model at time t given data up to time $t-1$, but a variational approximation can be used.

Our experiments are concerned with the labelling of pixels in image sequences into one of a finite number of classes. For example in images of outdoor scenes, the labels could be "sky", "vegetation", "buildings" etc. We compare the performance on this task of static and temporal PEDT models, and also simpler pixelwise classifiers.

The structure of the remainder of this paper is as follows: In section 2 we describe static PEDT models, and then go on to discuss the PEDT-sequence model in section 3. Experiments on ray-traced image data are described in section 4, and these results are discussed in section 4.6.

2 The Static PEDT Model

In this section, we will introduce the static PEDT in 2.1. In 2.2, we outline a variational approximation which we will use to perform inference within the framework and section 2.3 describes the KL-divergence minimisation update equations for the variational procedure. This is a review of the technique and full details can be found in [12, 13].

2.1 The Position-encoding dynamic tree model

Firstly, we look at the overall generative model for the position-encoding dynamic tree. There are four fundamental parts to it, the network nodes, the dynamic structure, the node classes and the node positions.

The model will be described using the belief network formalism. The graphical model will have a number of layers of nodes which will be used to describe objects or object parts, each layer representing object parts of similar sizes. The lower nodes will represent smaller components and the higher nodes larger ones.

The set N of n network nodes is denoted by $\{1, 2, \dots, n\}$. The nodes are organised into layers $\{1, 2, \dots, H\}$, where 1 denotes the top layer. We use L^h to denote the set of nodes in layer h. The network nodes are connected together in a way that represents the structural relationships in the image. We generally use the superscript notation to represent a set of random variables. For example X^B represents the set $\{X_i | i \in B\}$. For the state of all nodes we drop the N: $X^N = X$. We will also use the shorthand X' to denote X^{L^B} where $B = \{1, 2, \dots, H-1\}$. In other words X' represents the set of X_i corresponding to all nodes i not in the bottom layer.

The network connectivity is denoted by a matrix Z of indicator variables z_{ij}. Setting $z_{ij} = 1$ represents the existence of the connection between child i and parent j. We also include the variable z_{i0}, which is set to 1 if and only if i is a root node (i.e. it has no parent). Because Z must be a tree structure, i can have at most one parent, and so given i, $z_{ij} = 1$ for one and only one value of j and is zero otherwise. Furthermore, the constraint that all nodes can only have parents in the layer above means z_{ij} can only equal 1 if node j is in a layer one higher than node i (or $j = 0$).

The variable Z can represent all the possible tree structures which are of interest. It is now possible to define a prior distribution over tree connectivity, which is taken to be of the form

$$P(Z) = \prod_{i=1,j=1}^{n} \gamma_{ij}^{z_{ij}}, \tag{1}$$

where γ_{ij} is the probability of i being the child of j (we will say i chooses parent j with probability γ_{ij}). Generally it is assumed that this probability is uniform across all parents $j \neq 0$, expressing node interchangeability. Interchangeability is not a theoretical necessity (and so a more general form of γ_{ij} could be used). However, because we will allow each node to take any position, interchangeability does make the later computational optimisations easier as it introduces a symmetry which means that any one of the symmetric optima can be found, rather than one particular optimum. The disconnection probability γ_{i0} needs to be specified *a priori.*

2.1.1 Positions

Each node has a value which represents the position of the centre of the object. The position of each node is given relative to the position of the parent node, or if the node has no parent the position is taken to be absolute. The prior distribution over each of these nodes needs to be specified.

For the root nodes, we suppose the position is taken from a broad Gaussian distribution. Gaussian distributions are used to specify the centres of the child objects given the parent. Formally[1]

$$P(\mathbf{r}_i|\mathbf{r}_j, z_{ij} = 1) = \frac{1}{(2\pi)^{d/2}|\Sigma_{ij}|^{1/2}} \exp(-\frac{1}{2}(\mathbf{r}_i - \mathbf{r}_j)^T(\Sigma_{ij})^{-1}(\mathbf{r}_i - \mathbf{r}_j)), \tag{2}$$

where $d = 2$ is the dimensionality of the space, and Σ_{ij} is a covariance used to represent the order of magnitude of the object size.

The overall distribution is given by

$$P(R|Z) = \prod_{ij|z_{ij}=1} \frac{1}{(2\pi)^{d/2}|\Sigma_{ij}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{r}_i - \mathbf{r}_j)^T(\Sigma_{ij})^{-1}(\mathbf{r}_i - \mathbf{r}_j)\right). \tag{3}$$

At the lowest level (layer H) we will need to connect the nodes up to the pixels. For this reason we set the positions of the bottom layer of node to be given by the pixel positions, and then use the distribution $P(Z, R'|R^{L^H})$ as the prior over positions and connectivity.

2.1.2 The node labels

The other remaining important concept is the class label of each of the nodes. This represents the type of object or texture which the node is representing.

Suppose there are C possible classes which nodes can belong to. We denote the class of a node i by X_i and use the indicator $x_i^k = 1$ to represent the state that X_i is in class k. We represent the probability of a node i being in state k given its parent j is in state l by a conditional probability table P_{ij}^{kl}. We make a simplifying assumption that this conditional probability is independent of the positions of the parent and child nodes. This is not entirely realistic, but is a useful first approximation. Hence $P(X|Z)$ can be written as

$$P(X|Z) = \prod_{ijkl} (P_{ij}^{kl})^{x_i^k x_j^l z_{ij}}. \tag{4}$$

2.1.3 Pixel intensities

The intensity of each pixel will depend on the class label of the node it is connected to. For each bottom layer node i there is a pixel in position $\mathbf{r}_i$. Let this pixel also be enumerated by i, and assume its value is entirely dependent on the node state X_i through the distribution $P(Y_i|X_i)$, where Y_i denotes the pixel RGB intensity. The overall pixel model is

$$P(Y|X, R) = \prod_{i \in L^H} P(Y_i|X_i). \tag{5}$$

2.1.4 Complete Model

The full definition of the prior distribution is given by the following. First $P(Z)$ is defined using (1) and $P(R|Z)$ using (3) to give us $P(R, Z)$. We then impose the condition fixing

[1] It is also possible to include offests ρ_{ij} within this formalism, so that $P(\mathbf{r}_i|\mathbf{r}_j, z_{ij} = 1)$ takes the form $(2\pi)^{-d/2}|\Sigma_{ij}|^{-1/2} \exp(-\frac{1}{2}(\mathbf{r}_i - \mathbf{r}_j - \boldsymbol{\rho}_{ij})^T(\Sigma_{ij})^{-1}(\mathbf{r}_i - \mathbf{r}_j - \boldsymbol{\rho}_{ij}))$. They are left out here for the sake of simplicity.

the bottom layer nodes to the pixel positions to get $P(Z, R'|R^{L^H})$. Using equation (4) gives $P(X, R', Z|R^{L^H}) = P(Z, R'|R^{L^H})P(X|Z)$. Lastly $P(Y|X)$ is of the form (5), giving the final joint distribution $P(Y, X, R', Z|R^{L^H})$. This fully specifies a position encoding dynamic tree model.

2.1.5 Using the position encoding dynamic tree model

The position-encoding dynamic tree model gives the prior probability distribution for the image through the use of a large hierarchical latent variable model. To use it, we need to find out what the posterior probability distribution of the latent variables is given a specific image. In other words we condition on the fact that we have pixels Y, and try to discover the distribution over Z, X and R' that results. This distribution will give us a set of possible good interpretations of the connectivity, content and positioning of objects in the image.

2.2 Variational Inference in the PEDT

To use the PEDT model, some approach for doing Bayesian inference is needed. Were this model a tree structured belief network, we could use techniques such as belief propagation [9] to exactly calculate the required posterior distribution $P(X, R', Z|Y, R^{L^h})$. However because the PEDT is not of this form, and because other approaches such as the junction tree method are inappropriate because of the large clique size of the triangulated network, exact calculations are not feasible. Instead the graphical structure of the distribution is used to develop variational approximation methods to the posterior.

Variational methods are one of a number of approaches for approximate inference in networks. Other approaches include Markov chain Monte-Carlo methods [3], where a Markov chain is constructed which has the desired posterior as the limit distribution. Samples from this Markov chain are used as an approximation for the posterior. Monte-Carlo and annealing approaches for dynamic trees were investigated in [18]. Unfortunately obtaining Monte-Carlo samples for a network of this size would take longer than is affordable. The variational approach, on the other hand, fits an approximating distribution to the true posterior. The variational distribution must be chosen to be tractable to calculate with and must allow a variational fit to the posterior to be obtained in a reasonable time. At the same time the fit must be as good as possible so that the approximation is not a poor one. This makes choosing the variational distribution a non-trivial exercise. Further details of the variational method can be found in [11].

Below we give an outline of the variational method and the form of variational distribution which is used, the resulting update equations used to fit the distribution to the posterior are given in section 2.3.

The variational approach of this section develops and extends the approach used in [12] to the new case of the position-encoding dynamic tree. This approach involves approximating the posterior distribution with a factorising distribution of the form $Q(Z)Q(R)\, Q(X|Z)$, where $Q(Z)$ is the approximating distribution over the Z variables, $Q(X|Z)$ is the approximating distribution over the states, and $Q(R)$ is an approximating distribution over the node positions.

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

$$KL(Q||P) = \int dR' \sum_{Z,X} Q(Z)Q(X|Z)Q(R') \log\left(\frac{Q(Z)Q(X|Z)Q(R')}{P(Z, R', X|R^{L^H}, Y)}\right). \qquad (6)$$

To choose good forms for the Q's the Kullback-Leibler divergence $KL(Q||P)$ between the $Q(Z)$ $Q(R)Q(X|Z)$ 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.

Introducing Θ to represent the conditional probability table parameters, we define

$$\begin{aligned} L_{var}(Y|\Theta) &\stackrel{def}{=} -\int dR' \sum_{Z,X} Q(Z)Q(R')Q(X|Z) \log \frac{Q(Z)Q(R')Q(X|Z)}{P(Z,R,X,Y|\Theta)} \\ &= -KL_\theta(Q||P) + \log P(Y|R^{L^H},\Theta) + \log P(R^{L^H}) \end{aligned} \tag{7}$$

and use the fact $P(Z,R,X,Y|\Theta) = P(Z|\Theta)P(R|Z,\Theta)P(X|Z,\Theta)P(Y|X,\Theta)$. $L_{var}(Y|\Theta)$ is called the variational log likelihood. The fact that $KL_\Theta(Q||P) \geq 0$ implies the log likelihood $\log P(Y|R^{L^H},\Theta)$ is lower bounded by the variational log likelihood minus the additive constant $\log P(R^{L^H})$. Minimising the KL divergence maximises the variational log likelihood.

The forms of each of the approximating distributions needs to be finalised. We use a $Q(Z)$ and $Q(R')$ of the form

$$Q(Z) = \prod_{ij} \alpha_{ij}^{z_{ij}} \text{ and } Q(R') = \prod_{i \in N \backslash L^H} \frac{1}{(2\pi)^{d/2}|\Omega_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{r}_i^T - \boldsymbol{\mu}_i^T)\Omega_i^{-1}(\mathbf{r}_i - \boldsymbol{\mu}_i)\right) \tag{8}$$

where the α's are probabilities and where $\boldsymbol{\mu}_i$ and Ω_i are position and covariance parameters respectively, all of which need to be optimised. In this paper Ω_i is assumed to be diagonal. Lastly the $Q(X|Z)$ is a dynamic tree distribution of a form identical to that used in [12]:

$$Q(X|Z) = \prod_{ijkl} (Q_{ij}^{kl})^{x_i^k x_j^l z_{ij}}. \tag{9}$$

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

This distribution is chosen because local update propagations can be obtained through KL divergence minimisation. Hence the model can be tuned to the posterior efficiently. Also the marginal values of this distribution can be obtained straightforwardly. In addition this variational form is guaranteed to give better approximations than simpler ones such as a mean field approximation [12]. The difference between this approach and the mean field approach stems from the fact that we allow dependence on Z in (9), and that we also allow dependence of a node i on parent nodes rather than forcing $Q(X|Z)$ to take a factorised form. Because there are significant dependencies between child and parent in the prior we would expect some dependence to remain in the posterior. The mean field approach ignores that dependence, whereas this more structured approximating distribution allows some of those dependencies to be captured.

2.3 Update Equations

We want to minimise the KL divergence (6), with the forms of approximate distribution given in the last section. This must be done subject to the constraints that $\sum_k Q_{ij}^{kl} = 1$ and $\sum_j \alpha_{ij} = 1$ (probabilities sum to 1). We add to (6) a set of Lagrange multiplier terms corresponding to these constraints, and set the derivatives to zero. Solving this gives a set of update equations. The full derivations are given in [14]. Below, the update equations for the conditional probability tables, position and tree connectivity of the variational distribution are given.

2.3.1 Class Labels

Minimising 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

$$m_i^k = \sum_{jl} \alpha_{ij} Q_{ij}^{kl} m_j^l. \tag{10}$$

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 minimisation of the KL divergence gives

$$Q_{ij}^{kl} = \frac{P_{ij}^{kl} \lambda_i^k}{\sum_{k'} P_{ij}^{k'l} \lambda_i^{k'}} \text{ where } \lambda_i^k = \prod_{c \in c(i)} \left[\sum_g P_{ci}^{gk} \lambda_c^g \right]^{\alpha_{ci}}, \tag{11}$$

where $c(i)$ 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's can be updated by making a single pass up the tree to calculate the λ values, and then calculating the Q's.

2.3.2 Positions

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

$$\boldsymbol{\mu}_i = \sum_k \alpha_{ki} (\Sigma_{ki})^{-1} \boldsymbol{\mu}_k + \sum_j \alpha_{ij} (\Sigma_{ij})^{-1} \boldsymbol{\mu}_j, \tag{12}$$

$$(\Omega_i)_{pp} = \frac{1}{\sum_j \left(\alpha_{ij} (\Sigma_{ij})_{pp}^{-1} + \alpha_{ji} (\Sigma_{ji})_{pp}^{-1} \right)} \tag{13}$$

where we have assumed that both Σ and Ω are diagonal. The equations for $\boldsymbol{\mu}$ involve a sum across all possible parents and a sum across all possible children, and hence need to be iterated until suitably converged.

2.3.3 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

$$\begin{aligned} \alpha_{ij} &\propto \gamma_{ij} \exp(-\Psi_{ij}) \exp(-\Phi_{ij}) \\ \text{where } \Psi_{ij} &= \sum_l m_j^l [\log \sum_k P_{ij}^{kl} \lambda_i^k] \\ \text{and } \Phi_{ij} &= \frac{1}{2} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \Sigma_{ij}^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) + \frac{1}{2} \mathrm{Tr}(\Sigma_{ij}^{-1} \Omega_i) + \frac{1}{2} \mathrm{Tr}(\Sigma_{ij}^{-1} \Omega_j). \end{aligned} \tag{14}$$

In the above the constant of proportionality is found by normalisation as $\sum_j \alpha_{ij} = 1$.

Highest Variational Posterior Probability The tree Z which maximises $Q(Z)$ after the variational distributions have been optimised is called the tree with the highest variational posterior probability (HVPP). It is effectively the maximum a posteriori (MAP) tree under the variational approximation. Likewise we can pick the HVPP solution for any set of random variables, such as the classification labels or positions. This is useful for visualisation.

2.3.4 Optimisation Process

The above equations give all the necessary update rules. The whole optimisation process involves an outer loop optimising the $Q(Z)$ values and an inner loop containing up and down passes of the $Q(X|Z)$ optimisation and a number of passes of the $Q(R)$ optimisation. 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. A termination criterion based on the change in the KL divergence can be set. However the calculation of the KL divergence itself is one of the most costly parts of the optimisation, so in the experiments we used a fixed number of iterations, and only calculated the KL divergence at the end of each inferential run. The whole optimisation takes the following form:

```
Initialise Q(R)
Optimise Q(Z) ignoring Q(X|Z) contribution
for outerloop=1 to numouterloop
  Optimise Q(X|Z)
  for innerloop=1 to numinnerloop
    for qroptim=1 to numqroptim
      Do one pass of the Q(R) optimisation
    end
    Optimise Q(Z)
  end
end
```

3 The PEDT Sequence Model

Position-Encoding Dynamic Trees (PEDTs) have been designed to provide the type of positional and structural information needed for modelling sequences, but there are still a number of forms that sequence models can take and a number of sequence-specific issues, such as occlusion, which will impact on the choice of model.

In section 3.1 the requirements of a PEDT sequence model are elaborated, and the form the model should take is described along with some model related issues and difficulties. This eventually leads to the final form of the model described and defined in section 3.1.1.

3.1 Requirements of a Sequence Model

Because of the large size of any model of general image forms, most methods for image sequence modelling focus on the filtering problem. That is they are interested in developing and working with a conditional latent model of the next time frame given the previous one, and given the current image information. We will also be focussing on the filtering problem here, and will need to specify a predictive model of the PEDT at a new time t given we know what the state is at time $t-1$. We also need to specify how an image at time t would be generated from the PEDT, and hence how the PEDT at time t is updated when we discover what the image at time t actually is.

The basic characteristics of the PEDT are node labels, positions and tree structures, and these are used to describe the image contents. As we progress through the sequence, the positions of objects will change. Some objects might start to become occluded by other objects, parts of some objects might appear from behind others etc. We can be sure of a significant continuity of the images and so nodes in certain positions in one frame are likely to have the same class labels as nodes in similar positions in the next time frame. What we need is some method of making a loose correspondence between nodes in the two time frames, which will then allow information to be shared through time. This correspondence will map a node which represents a certain object or part object to a node in the next time frame which corresponds to the same object or part object. The correspondence needs to be loose and flexible because certain parts of objects might be occluded or have disappeared from the field of view, or might have got larger or smaller. There is no requirement for a one to one mapping. Also it is important that the model allows all areas of the image to be represented. Variational methods for the PEDT can result in only a small number of all the possible nodes in a layer being distinct from one another if no other objects need to be represented. It is important that such a contraction in representation does not limit the representation at the next time step. The distribution over maps used here and given in section 3.1.1 satisfies these requirements.

Once a map between nodes at two adjacent time frames is obtained, a predictive distribution for the PEDT at the later time frame can be defined taking into account not only the usual static PEDT prior information but also the information from the previous frame. As nodes represent object parts, we would expect class label information to be passed from the past node to its corresponding node. For example, the node at the second time frame will generally have the same class label, and its position (relative to its parent if it has one), should be similar in both time frames, representing the fact that typically inter-frame movements should be small.

3.1.1 The basic Sequence Model

The basic sequence model would therefore include a method of generating a correspondence between nodes at two time frames and then generating the predictive distribution at time t given the information at time $t-1$. We define that form of model here. First we define a correspondence model. Then we define a temporal model for the node states and connectivity. It turns out that this temporal model produces a predictive model at time t which has the same PEDT prior but with modified parameters. Hence all the variational machinery already derived is directly applicable for inference in the model at this new time frame t.

Let S^t denote a set of connections from nodes at time $t-1$ to nodes at time t. S^t consists of indicator variables s^t_{ij} where $s^t_{ij}=1$ denotes that node i of the PEDT at time t corresponds to node j from the PEDT at time $t-1$. We also allow a connection to a null node 0 denoted $s^t_{i0}=1$ representing the fact that i has no corresponding node at time $t-1$. $s^t_{ij}=1$ for one and only one value of j, and is zero otherwise.

In order to define the node correspondence, we will define a set of reference positions for each node i, denoted by $\boldsymbol{\nu}_i$.

We can define a probability distribution

$$P(s^t_{ia}|\boldsymbol{\nu}_i, \mathbf{r}^{t-1}_a) \tag{15}$$

which depends on the positions of nodes at the past time frame and the chosen reference positions $\boldsymbol{\nu}_i$ of the nodes at the new time frame. This distribution gives the through time connection probabilities. The initial node centres $\boldsymbol{\nu}_i$ are set *a priori*. In order to ensure that, at each time step, reasonable coverage of the space is obtained, we set the initial node centres to take either a uniform or a quadtree-like distribution[2]. Thus we use

$$P(s^t_{ia}|\boldsymbol{\nu}^t_i, \mathbf{r}^{t-1}_a) \propto \exp(-\frac{1}{2}(\mathbf{r}^{t-1}_a - \boldsymbol{\nu}_i)\Lambda_l^{-1}(\mathbf{r}^{t-1}_a - \boldsymbol{\nu}_i)). \tag{16}$$

[2]Each layer is a grid suitably positioned so that, where possible, four nodes of a layer are equidistant from the closest node in the layer above.

which can alternatively be seen as the responsibilities of each node a for each point i under a mixture of Gaussian distributions centred at nodes a with covariance Λ_l dependent on layer l. The probability of a node being a new node $P(s_{i0}^t)$ is set *a priori*.

The actual map S^t from nodes at time t to nodes at time $t-1$ are drawn from the distribution

$$P(S^t) = \prod_{ij} P(s_{ij}^t|\boldsymbol{\nu}_i, \mathbf{r}_j^{t-1}). \tag{17}$$

Having obtained the connection probabilities, the predictive distribution of the nodes at time t must be specified. The simplest approach is to allow the class variables and connectivity to be passed forward. If a node is connected through the past we would expect the class variable and parent to be identical to its state in the previous time frame. We write this as

$$P(z_{ij}^t = 1|s_{ia}^t = 1, s_{jb}^t = 1, z_{ab}^{t-1} = 1) = 1 \text{ and} \tag{18}$$

$$P(X_i^t = k|X_i^{t-1} = l, s_{ij}^t = 1) = \delta_{kl}. \tag{19}$$

If a node i is considered to be a new node (i.e. $s_{i0} = 1$), then its class is entirely dependent on its parent at time t through the usual prior CPTs, and if the parent of a node does not propagate through to time t, or a node is a new node, then the node has the usual uniform connection probabilities with the parents in the layer above.

This is the simplest form of dynamic model we can use. It encodes the fact that at the next time step we expect to see similar sorts of objects and structures in our image. The uncertainty in the dynamic model comes from the uncertainty in the position that a node moves to through time, and in whether or not the node is still represented at the next time step.

3.2 Working with the Sequence Model

The PEDT uses approximate variational inference in order to obtain a representation of the posterior distribution. When a PEDT model is used in a sequence it would be correct to do variational updates on the whole sequence. However the computational cost of this is prohibitive (even on smaller images there are tens of thousands of nodes per frame). In almost all practical circumstances models of the filtering process make an assumption that the approximate representation of the posterior at time $t-1$ could be taken as suitably close to the true posterior for practical purposes. For example in [5] the Monte-Carlo sample from time $t-1$ is presumed practically exact when propagation to the next time frame is done. We make the same assumptions here, assuming that the variational distribution at time $t-1$ can be presumed to be good enough without repeated iteration of the variational method through time.

The first step, then is to find the predictive PEDT at time t given the variational posterior at time $t-1$. It turns out that this propagation can be done exactly. This relies on the fact that the variational posterior $Q(X_i^t|X_j^t, z_{ij}^{t-1} = 1)$ at time $t-1$ takes the same values whatever parent node j is being considered. The implication of equations (18) and (19) for the class labels is considered first, followed by the implication for the Z prediction.

3.2.1 Class labels

We have a predictive distribution taking the PEDT form at time t with the CPTs given by

$$\begin{aligned} P(X_i^t = k|X_j^t = l, z_{ij}^t = 1) = P(s_{i0}^t)P_s(X_i^t = k|X_j^t = l, z_{ij}^t = 1) \; + \\ \sum_a P(s_{ia}^t)P(s_{j0}^t)P(X_a^{t-1} = k) + \sum_{ab} P(s_{ia}^t)P(s_{jb}^t)P(X_a^{t-1} = k|X_b^{t-1} = l, z_{ab}^{t-1} = 1), \end{aligned} \tag{20}$$

where P_s denotes the standard 'static' PEDT prior conditional probabilities. The three contributions to this term correspond to the following three situations:

1. Node i is a new node at time t
2. Node i at time t corresponds to node a at time $t-1$ but node j is a new node at time t
3. Node i corresponds to node a and node j corresponds to node b

If we make $P(s_{j0}^t) = (1 - \Gamma^h)$ identical for all nodes j in the same layer h and at every time t, and use the fact that $P(X_a^{t-1} = k | X_b^{t-1} = l, z_{ab}^{t-1} = 1) = Q_a^{kl,t-1}$, the variational posterior CPTs from the previous time frame we obtain

$$P(X_i^t = k | X_j^t = l, z_{ij}^t = 1) = P(s_{i0}^t) P_s(X_i^k | X_j^k, z_{ij}^t) \\ + \sum_a P(s_{ia}^t)(1 - \Gamma^h) m_a^{k,t-1} + \sum_a P(s_{ia}^t) \Gamma^h Q_a^{kl,t-1} \quad (21)$$

where $m_i^{k,t-1}$ denotes the marginal variational posterior probability of node i being in state k at time $t-1$, and h denotes the layer that node j is in.

3.2.2 Connectivity

Looking now at the predictive prior over the Z variables we have the distribution

$$P(z_{ij}^t = 1) \propto \left(P(s_{i0}^t) + \sum_a P(s_{ia}^t) P(s_{j0}^t) \right) P_s(z_{ij}^t = 1) + \sum_{ab} P(s_{ia}^t) P(s_{jb}^t) Q(z_{ab}^{t-1} = 1) \quad (22)$$

for $j \neq 0$ and

$$P(z_{i0}^t = 1) \propto P(s_{i0}^t) P_s(z_{i0}^t = 1) + \sum_a P(s_{ia}^t) Q(z_{a0}^{t-1} = 1). \quad (23)$$

This gives us the predictive prior $P(Z^t | Q^{t-1})$ and $P(X^t | Z^t, Q^{t-1})$ at time t given the variational posterior Q^{t-1} at time $t-1$. Again P_s denotes the prior under the static PEDT model.

3.2.3 The Resulting Predictive Model

These equations for the predictive model take the form of another PEDT, with modified CPTs and modified connection probabilities. Because it is a PEDT, the variational posterior at time t can be calculated using the usual PEDT variational optimisation. Then this can be fed forward to time $t+1$ etc. Hence we have a way of implementing filtering in a chain of PEDTs. The implementation of this method involves running a standard static PEDT for the first image of the sequence. Then equations (21) and (22) are used to calculate the effective CPTs and predictive prior over Z for time t given what we know at time $t-1$. (In fact appendix A shows that the predictive prior over Z can be well approximated by a uniform distribution.) Once these calculations are done we effectively have a static PEDT again (but with modified CPTs etc.).

The overall form of the optimisation involves initialising the positions of the node's variational centres to the reference positions $\boldsymbol{\mu}_i = \boldsymbol{\nu}_i$, calculating the responsibilities $P(s_{ia})$ of each node a at time $t-1$ for each node i at time t, propagating forward the conditional probabilities, and then conditioning on the positions of the root nodes and the pixel data to get a final posterior at time t. This posterior is calculated using the usual variational optimisation given the initialisation described above.

Benefits of node map model The model for mapping nodes in this time frame to those in the past appears to depend on a characteristic of the variational method. However the precise initialisation can be set *a priori*, and so is not dependent on the variational optimisation. Furthermore correspondence between nodes is somewhat arbitrary due to node exchangeability (relabelling the nodes makes no difference to the predictive distribution we define for the Z and X variables). The reason we use the variational centres to set the correspondence is in order to avoid local minima problems which can be encountered if the initial positions of nodes are far from their positions at the previous time frame, and to avoid node clumping problems (another form of local minimum), where a large number of the nodes are used to represent a small part of the overall space. This is achieved because at each stage the past nodes are associated with new nodes which are evenly or uniformly distributed through space. Other model forms which say generate node positions dependent on the positions in the past can easily suffer from node clumping at later time stages.

3.3 Related work

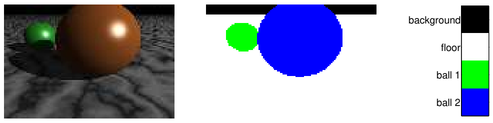
There is of course a huge literature on the analysis of image sequences in computer vision. Our work has some similarities to work on tracking (see, e.g. [2]). The work of Blake and Isard and others has essentially developed non-linear Kalman filtering methods for the tracking problem. Usually this is carried out using a contour-based shape representation. In contrast we use a region-based approach which should be robust to objects whose shape changes continually and those which merge and split into discontinuous components, for instance, clouds. Note also that contour-based tracking needs to represent explicitly the number of objects moving; again, the PEDT framework does not require this.

The static PEDT was developed as an alternative to Markov Random Field (MRF) image model. The generalization of the PEDT to sequential data mirrors the similar generalization of MRFs, as carried out e.g. by [8, 4, 6].

The is also a considerable literature on motion estimation, see e.g. [16]. This differs from our work in that it is involved with grouping local motion estimates derived from optic-flow computations. In order to incorporate optic-flow observations into our model it would be necessary to have velocity information in the hidden state. We also note that both motion and intensity cues can be used for segementation, see e.g. [15].

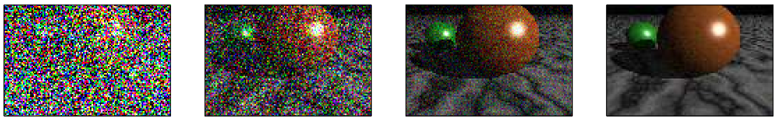
4 Experiments

Here we demonstrate the application of PEDT models to image sequence segmentation. We evaluate how the incorporation of sequential information, as described above, into our PEDT model affects the performance of the model on the segmentation task. Section 4.1 discusses the data that we use for the investigation. The local pixel-wise image model that we employ is described in 4.2 and the PEDT training set-up is briefly reported in section 4.3. Section 4.4 describes the investigations that were carried out into the behaviour of the algorithm with respect to the various parameter settings of PEDT. In section 4.5, we discuss the various ways in which we can measure the performance of the approach. Finally, the results are presented and discussed in section 4.6.



(a) (b) (c)

Figure 1: A training image. Subfigure (a) shows one image from the training dataset, subfigure (b) shows the corresponding image labels, (c) shows the key for the image label colours.



(a) (b) (c) (d)

Figure 2: A training image with added Gaussian noise. In each of the subfigures respectively, the added Gaussian noise has a variance of: (a) 1; (b) 10^{-1}; (c) 10^{-2}; (d) 10^{-3}.

4.1 Data

We use sequences of ray-traced images in the following experiments. It is a difficult and time-consuming procedure to produce reliably labelled real world images. This is true of single individual images, the task is even more cumbersome when dealing with large sequences of images.

The use of artifically generated image sequences gives us access to an unlimited supply of training and test data for which it is straightforward to generate precisely the ground truth image labels.

We have used a set of 3 training and 50 test images showing two balls rolling around on a flat plane. Each 8-bit colour image is 80×120 pixels in size. Figure 1 shows an example image and its corresponding ground truth labelling. The labels are "background", "floor", "ball 1" and "ball 2". The image is one frame of the training sequence. (Animations of the full training and test sequences and their corresponding ground truth labels can be found online at `http://www.dai.ed.ac.uk/homes/stephenf/pedt`).

A disadvantage of using ray-traced images is that with simple examples the pictures lack the complexity and detail of real world images. Moreover, when we are trying to ascertain the benefits of using the PEDT framework, if the problem is too simple then a purely local image model may perform very well on pixel-wise classification and it will be hard to assess the benefits of a more sophisticated approach. This is even more pronounced when we come to deal with assessing the fusion of temporal information within the PEDT framework in comparison with both an ordinary PEDT model and simple local models. It is difficult to determine the effects of the extra work if the simpler models already perform well. In order to address these issues, we perform all the experiments with a number of levels of independent Gaussian noise added to each pixel. The performance of each of models can thus be assessed in situations of varying degrees of difficulty.

Figure 2 shows the same training image as illustrated above but to which various amount of noise have been added. Each image has RGB values on a scale from 0 to 1. The independent Gaussian noise is added to each pixel with variances of 1, 10^{-1}, 10^{-2}, 10^{-3}. This operation is performed using the `imnoise` function from the image processing toolbox in Matlab. Noise

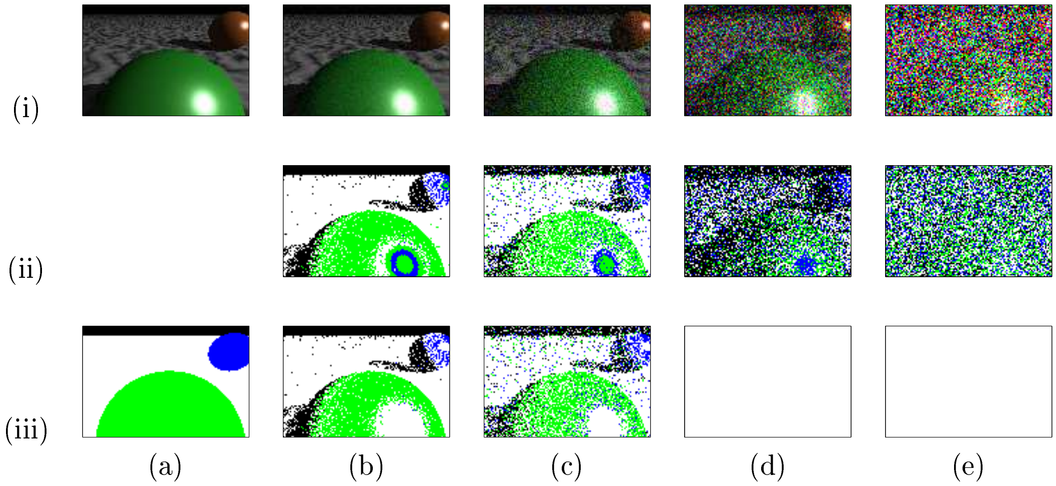


Figure 3: The local segmentations of a test image. Subfigure (a-i) shows the example test image and subfigure (a-iii) shows its corresponding ground truth labelling. Excluding column (a), row (i) shows the test image with various amounts of added noise, row (ii) shows the maximum likelihood segmentation under the local image model and row (iii) shows the MAP segmentation under the local image model. The variance of the Gaussian noise added is 10^{-3} in column (b), 10^{-2} in column (c), 10^{-1} in column (d) and 1 in column (e). The key for the colour of image labels is the same as that found in figure 1.

added to a pixel value that takes it outside the RGB colour cube is truncated.

4.2 Local Image Model

To make inferences about a particular previously unseen test image, we need to make an estimate of the likelihood, $p(y_i|x_i)$, that pixel i has value y_i given it has class label x_i for each of the pixels in that frame. For the purposes of these experiments we use a simple local image model which we derive from a histogram of the training set class frequencies.

To construct the histogram, the RGB colour cube is divided into $3 \times 3 \times 3 = 27$ bins of equal volume and the frequency with which the training set pixels fall into each of these bins is counted for all the pixels of each class. For a particular test pixel whose pixel value falls into bin b of the colourcube, our estimate for the likelihood is given by the measured frequency of bin b for each of the four possible class labels.

The local image model depends on the level of the noise added to the training set. Figure 3 shows an example image from the test image database to which the various amounts of noise have been added and then the appropriate local model used to construct both the maximum likelihood segmentation under the local pixelwise model and the MAP segmentation.

For the highest two levels of noise, the MAP local segmentation just picks the class with the highest prior probability for all the pixels. Moreover, although the PEDT experiments were carried out with the four noise levels described above, the noisiest images (with noise variance 1) were classified as the highest prior class for all pixels under all the model parameter settings tried. As these results are rather uninteresting, they are omitted from the subsequent discussion.

4.3 Training the PEDT

We learn the PEDT parameters by carrying out 40 iterations of the variational optimisation as described in section 2.3.4. During training the probability of backconnection Γ is set to zero.

4.4 Investigating the Parameters of the PEDT

We perform three sets of experiments, in each case we update the PEDT prior model as described in equation 21. We fix the probability that a node connects to any node in its level at the previous time step to be same for every node in all the layers down to a specified depth, and set $\Gamma^k = 0$ for layers below the specified depth. (The implications of this assumption are discussed in section 4.6.2.) We examine how the algorithm performs on the segmentation task for a range of values for Γ (0, 0.10, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80, 0.90, 1.00).

In the first experiment, we vary the levels of noise added to the image. In the second, we examine how varying the depth in the tree to which the updates are performed affects the performance and in the third we vary the widths of the Gaussian weights which determine relative strengths of the connections to the nodes in the previous frame.

4.4.1 Noise Levels

We add independent Gaussian noise to each pixel as described in the Data section with a variance of $0.1, 0.01, 0.001$ and measure the segmentation accuracy. (We also used noise with variance 1, however this is too much noise, all segmentations predict the most probable class in the prior for all pixels. This is uninteresting and this setting is ignored for the rest of the discussion.) The prior updates were performed down to, and including, the fifth layer from the root.

4.4.2 Update Depth

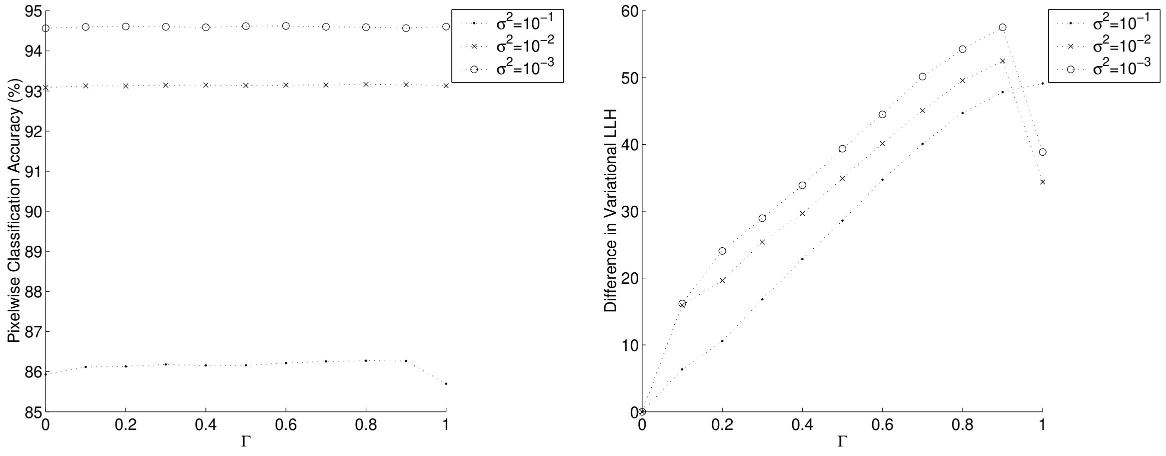
For a fixed noise level (variance 0.1), we experiment with varying the depth in the tree to which the updates are performed. (Note that updating the prior to depth n is equivalent to setting Γ to zero for all the nodes in the layers $n+1$ to 8.)

We look at all possible update depths. However, in the results section, the results for the updates down to only layers 1 and 2 are omitted for convenience because with this setting little information is propagated forward between time frames and the results show little difference from having Γ set to zero and are thus uninteresting.

4.4.3 Update Width Parameter, ω

From equation 21, we can see that the probability that node i connects to node a in the previous time frame is a Gaussian function of the relative distance between the two nodes. The distribution has a covariance matrix, Λ_l, which is specified for each layer. If we replace this covariance term with $\omega\Lambda_l$, we can then use ω as scale-factor which controls the width of the support for node i in the previous frame.

In this experiment, the image noise is fixed with a variance of 10^{-1}, the update depth is fixed at 5 and we measure the performance for all the above values of Γ and for a range of values of the update width, $\omega = \{2.0, 1.5, 1.0, 0.8, 0.66, 0.57, 0.5, 0.33\}$.



(a) (b)

Figure 4: Noise levels: subfigure (a) shows the pixelwise classification accuracy for each of the levels of image noise over a range of values for Γ, subfigure (b) shows the improvement in the average variational log likelihood of all the images in the test set for a range of values of Γ and for each of the noise levels, over the average variational log likelihood of all the test images under the static PEDT model ($\Gamma = 0$). The average values of L_{var} for the static PEDT model are 2706.1, 2531.2 and 2334.4 for noise levels of 10^{-1}, 10^{-2} and 10^{-3} respectively.

4.5 Performance Measures

As mentioned in the theory section above, in all the following experiments, it is possible to inspect the highest variational posterior probability tree and the segmentations that it generates, for a particular combination of parameters. See for example the segmentation shown in figure 6. Although this is just a qualitative assessment, it can be very useful as an aid to understanding what the algorithm is doing.

As well as this, we use two quantitative assessments of the performance. Firstly we measure the pixelwise classification accuracy, that is to say, the percentage of pixels correctly classified by the variational approximation to the true posterior probability distribution. Secondly, we compare the average variational log likelihood of the test images under the static model and under the temporal model. The variational log likelihood for the temporal model is the same as for the static model, but using the modified CPTs given in equation 20 instead of the static CPTs. The additive constant $\log P(R^{L^H})$ is the same in both cases, and so this term disappears when one term is subtracted from the other.

4.6 Results

In this section we will discuss the quantitative results obtained from each experiment. The tables of results for the above experiments can be in Appendix B. However, before looking at the details, as an illustrative example refer to Figure 6 which displays the PEDT that produces the MAP segmentation for one of the test images. It allows a comparison between the segmention produced by a static PEDT with $\Gamma = 0$, (see figure 6m) and that produced by a PEDT using temporal updates, $\Gamma = 0.15$, (see figure 6k). Figure 6o highlights where the use of the temporal updates has improved the segmentation (white) and where it has made matters worse (black). The areas which are either correctly or incorrectly classified in both segmentations are coloured

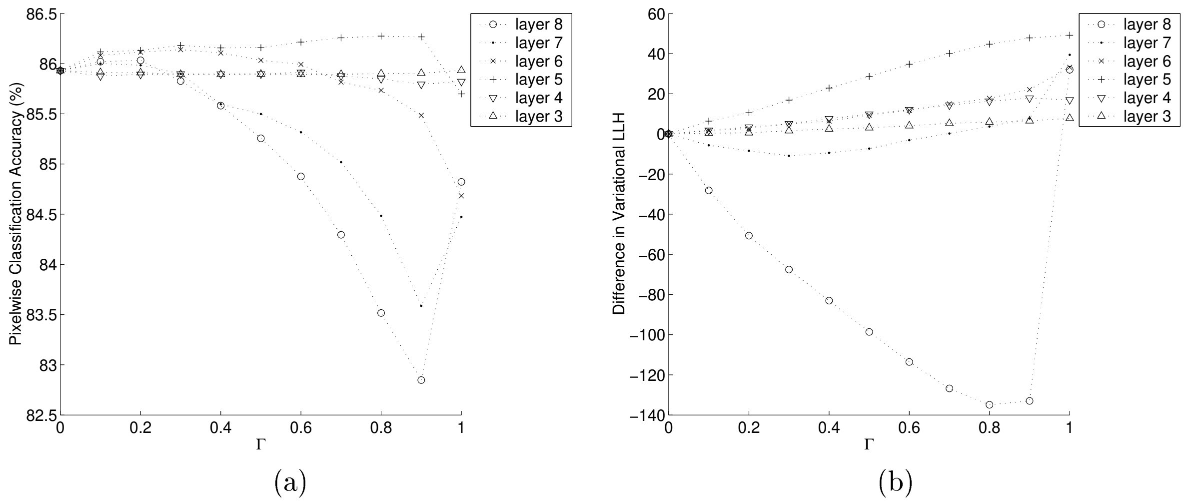


Figure 5: Update Depths: subfigure (a) shows the pixelwise classification accuracy for each of the update depths over a range of values for Γ, subfigure (b) shows the improvement in the average variational log likelihood of all the images in the test set for a range of values of Γ and for each of the update depths, over the average variational log likelihood of all the test images under the static PEDT model ($\Gamma = 0$).

grey.

4.6.1 Noise levels

Table 1a shows the classification accuracy of the PEDT model with updates using varying amounts of past information. Figure 4 illustrates these results graphically. The points to note are:

- The use of the PEDT model gives rise to improvements in classification accuracy of 9.2, 13.2 and 5.3 percent over the simple pixelwise local model for noise levels of 10^{-1}, 10^{-2} and 10^{-3} respectively.
- The less noise added, the higher the classification accuracy.
- When the added noise has variance 10^{-1}, there is improved classification accuracy over a broad range of Γ with a maximum at around 0.8 to 0.9.
- For the test images with less noise, there is no clear trend in the classification results as Γ varies. It could be that in these low noise regimes, it is hard to improve the performance of the static PEDT so the temporal information is redundant.
- The average variational log likelihood is improved for all values of $\Gamma > 0$ for all three noise levels.

4.6.2 Update Depths

Table 2 and Figure 5 show how the pixel-wise classification performance depends on Γ and the update depth for a fixed amount of noise and fixed $\omega = 1.00$. The points to note are:

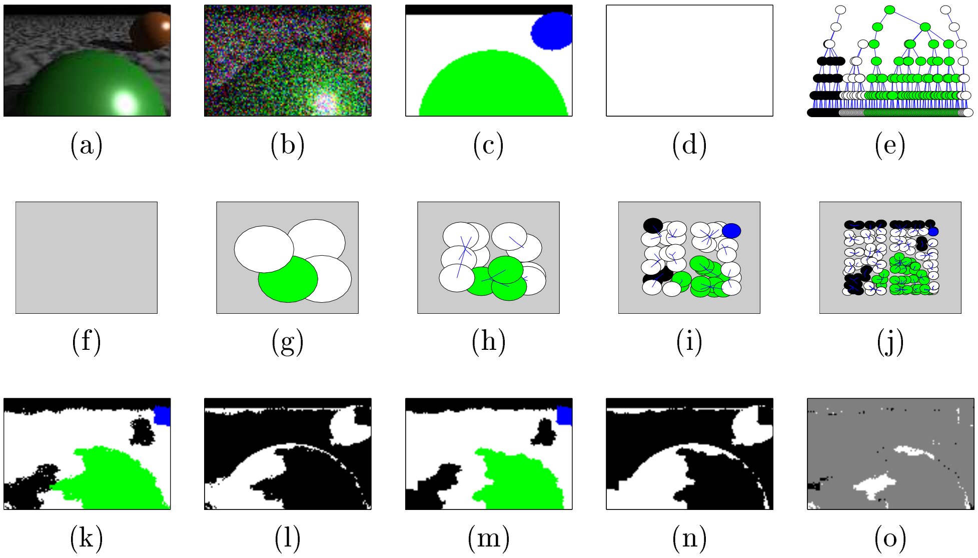


Figure 6: The MAP position encoding dynamic tree for an image in the test sequence. The Subfigures display: (a) the test image; (b) the noisy image on which the inference is performed; (c) the ground truth target image; (d) the MAP segmentation under the local image model; (e) a cross-section through the MAP PEDT (the leaf nodes in the 70th row of the image are shown along with their hierarchy of ancestors); (f)-(j) the highest five layers of nodes (the nodes in subfigure (g). form root nodes so there are no nodes present in the top layer), the bottom three layers of nodes have been omitted as they are difficult to read; (k) the MAP PEDT segmentation with $\Gamma = 0.15$; (l) the pixels misclassified by the PEDT with $\Gamma = 0.15$ (white); (m) the MAP segmentation with the static PEDT ($\Gamma = 0$); (n) the pixels misclassified by the static PEDT; (o) the gain between using the static PEDT and the PEDT with $\Gamma = 0.15$, (the pixels wrongly classified by the former and correctly by the latter are white, the pixels correctly classified by the former and wrongly by the latter are black, the rest are grey).

- For shallow updates (down to layers 1-4), the sequential information has little effect.
- When the updates go down close to the leaf layers, the performance can be heavily penalised especially at high Γ.
- The best performance occurs for 5 layer updates, as seen for both the classification and variational log likelihood measures.

To understand the poor performance when the Γ updates do down to near leaf level, bear in mind that each level has an associated scale over which correspondences are made, and this decreases as we approach the leaves. To yield good motion predictions, this scale should match the scale of inter-frame motion that we wish to capture, and enforcing Γ updates at low levels makes this scale too small.

The strong effect of the update depth parameter suggests that fixing Γ^k to be the same for all layers is perhaps not a good approach. Varying Γ from layer to layer would allow the influence of the previous timeframe posterior to depend on length scale. Some more investigation of this parameter space is required.

4.6.3 Update Width

The results for the experiment with different update widths show that the width of the update has very little effect on the performance of the approach, consequently the detailed results are omitted.

4.6.4 Demonstration

As well as the quantitative evaluation of the performance of the algorithm presented above, it may be instructive to look at an example image sequence segmentation for a short test image set. An mpeg movie clip can be found on the technical report website[3]. It shows the PEDT segmentation and an inferred image structure which illustrates the way in which the probability distribution over image segmentations is constructed.

Figure 7 shows a sample of frames from the sequence. We can see how the inferred tree structure changes as the balls in the test image sequence move and as ball 2 starts to occlude ball 1. The PEDT tree provides a plausible hierarchical parse structure for the image. (It is worth bearing in mind that the cross-section drawn is of the HVPP tree and that in fact the PEDT framework generates a full probability distribution over node positions, tree structures and node labels in the form of the variational approximation to the true posterior probability distribution).

5 Discussion

Above we have developed the Temporal PEDT architecture and evaluated its performance on a test image sequence. We have shown that adding temporal constraints can lead to increases in performance, although these benefits may not always be very large as evaluated on the image labelling task. One reason for this may be that there are a relatively small number of labels, and thus the predictive gain one can achieve through the use of temporal context (in bits/pixel) is relatively small. A possible alternative evaluation would involve using temporal context in image coding, where the potential gains would be larger. To do this the discrete-state structure

[3] See: `http://www.dai.ed.ac.uk/homes/stephenf/pedt/`

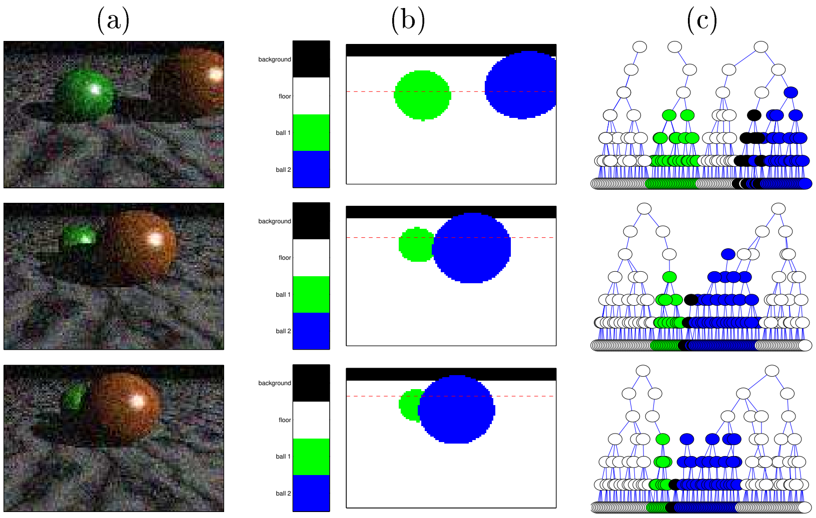


Figure 7: Frames 15, 19 and 20 from the test image sequence. Column (a) shows the test images (with variance of added noise 10^{-1}); (b) shows their ground truth segmentation (key: black = background, white = floor, green = ball 1, blue = ball 2). The red dotted line on the ground truth images indicates the line on which lie the leaf-nodes of the cross-section through the highest variational posterior probability (HVPP) tree that is shown in column (c). The line of the cross-section is chosen such that it is the horizontal line that passes through the widest part of ball 2 in the ground truth display. (Note that although this line is chosen based on the ground truth image in order hopefully to show a part of the tree with interesting features, the tree itself and the probability distribution from which it is drawn was generated with no reference to the ground truth labels.)

of the X variables would need to be replaced by real-valued variables; for example one could use a Gaussian distribution. This would extend the fixed-tree Gaussian TSBN models of Willsky *et al* [1, 7] to temporal dynamic tree architectures.

Acknowledgements

This work is supported by EPSRC grant GR/L78161 *Probabilistic Models for Sequences.* SNF gratefully acknowledges additional support from BAE Systems. We would like to thank the members of the Advanced Information Processing Department of the BAE Sowerby Research Centre, and in particular Dr. Gareth Rees, for helpful discussions.

References

[1] 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.

[2] A. Blake and M. Isard. *Active Contours.* Springer-Verlag, 1998.

[3] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. *Markov Chain Monte Carlo in Practice.* Chapman and Hall, London, 1996.

[4] T.J. Hainsworth and K.V. Mardia. A markov random field restoration of image sequences. In J. Chellappard, editor, *Markov Random Fields - Theory and Applications.* Academic Press, 1993.

[5] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditinal density. In *Proceedings European Conf. on Computer Vision*, pages 343–356, 1996.

[6] A.C. Kokaram and S.J. Godsill. A system for reconstruction of missing data in image sequences using sampled 3D AR models and MRF motion priors. In *Springer Lecture Notes in Computer Science: Computer Vision - ECCV '96*, volume II, pages 613–624. Springer-Verlag, April 1996.

[7] M.R. Luettgen and A.S. Willsky. Likelihood calculation for a class of multiscale stochastic models, with application to texture discrimination. *IEEE Trans. Image Processing*, 4(2):194–207, 1995.

[8] D.W. Murray and B.F. Buxton. Scene segmentation from visual motion using global optimisation. *IEEE Trans. PAMI*, 9(2):220–228, 1987.

[9] J. Pearl. *Probabilistic Reasoning in Intelligence Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, CA, 1988.

[10] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of IEEE*, 77(2):257–285, 1989.

[11] L. K. Saul and M .I. Jordan. Exploiting tractable substructures in intractable networks. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8.* MIT Press, Cambridge, MA, 1996.

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

[13] A. J. Storkey and C.K.I. Williams. Dynamic positional trees for structural image analysis. In T. Jaakkola and T. Richardson, editors, *Proceedings of the Eighth Internation Workshop on Artificial Intelligence and Statistics*, pages 298–304. Morgan Kaufmann, 2001.

[14] A.J. Storkey and C.K.I. Williams. Image modelling with position-encoding dynamic trees. *Submitted to IEEE PAMI*, 2001.

[15] W.B. Thompson. Combining motion and contrast for segmentation. *IEEE Trans. PAMI*, 2(6):549–543, 1980.

[16] Y. Weiss and E.H. Adelson. A unified mixture framework for motion segmentation: incorporating spatial coherence and estimating the number of models. *CVPR*, 1996.

[17] M. West and J. Harrison. *Forecasting and Dynamic Models*. Springer, 2nd edition, 1997.

[18] 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*, volume 11, Cambridge, MA, 1999. MIT Press.

A Approximating the Predictive Prior over Z

Originally in equation (14), the prior γ_{ij} was uniform. When adapting this for sequence use we get a non-uniform γ_{ij}. However the last part of (22) is a broad distribution over possible parents. In fact even if $Q(z_{ab})$ all have the same parent with certainty we have very approximately

$$\sum_{ab} P(s^t_{ia})P(s^t_{jb}))Q(z^t_{ab}=1) \propto \exp(-\frac{1}{2}(\boldsymbol{\mu}^t_i - \boldsymbol{\nu}^t_j)(\Lambda_{l_i}+\Lambda_{l_j})^{-1}(\boldsymbol{\mu}^t_j - \boldsymbol{\nu}^t_i)) \tag{24}$$

where the approximation consists in replacing the finite sum over node positions at time $t-1$ with a uniform integral. Despite this approximation, this serves to show how broad the distribution is. For $Q(Z)$ which are not of this worst case contrived form, this distribution will be even broader. We will generally be using values of Λ_l and Σ_{ij} which are diagonal and which make the components of $(\Lambda_{l_i}+\Lambda_{l_j})$ significantly greater than those of Σ_{ij}. Furthermore the values of $\boldsymbol{\mu}_i$ at time t will be initialised to $\boldsymbol{\nu}^t_i$, and will not stray great distances from this initialisation. Hence the contribution of γ_{ij} in (14) will not be significantly different from a uniform distribution as the more peaked Φ_{ij} term will dominate. In all experiments we have simplified computations by using a uniform distribution for $P(z_{ij})$ instead of (22).

B Tables of Results

(a) Pixelwise classification accuracy

Variance of added Gaussian noise	local	PEDT										
		$\Gamma = 0$	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
$\sigma^2 = 10^{-1}$	76.66	85.93	86.12	86.13	86.18	86.16	86.16	86.21	86.26	86.27	86.27	85.7
$\sigma^2 = 10^{-2}$	79.86	93.08	93.13	93.12	93.14	93.15	93.14	93.15	93.15	93.16	93.16	93.13
$\sigma^2 = 10^{-3}$	89.21	94.56	94.6	94.61	94.6	94.59	94.62	94.62	94.6	94.59	94.57	94.6

(b) Difference in the average variational loglikelihood
of the test images between a static PEDT and a PEDT with Γ updates

Variance of added Gaussian noise	Γ									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
$\sigma^2 = 10^{-1}$	6.36	10.6	16.8	22.8	28.6	34.7	40.1	44.7	47.8	49.1
$\sigma^2 = 10^{-2}$	16	19.6	25.4	29.7	34.9	40.1	45.1	49.6	52.5	34.4
$\sigma^2 = 10^{-3}$	16.2	24.1	29	33.9	39.4	44.5	50.2	54.3	57.5	38.9

(c) Percentage of images with an improved variational log likelihood over the static PEDT

Variance of added Gaussian noise	Γ									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
$\sigma^2 = 10^{-1}$	93.88	97.96	97.96	100	100	100	100	100	100	95.92
$\sigma^2 = 10^{-2}$	97.96	95.92	100	100	100	100	100	100	100	77.55
$\sigma^2 = 10^{-3}$	87.76	97.96	97.96	97.96	97.96	97.96	97.96	95.92	100	77.55

Table 1: Varying the image noise levels. (Update depth = 5, width parameter = 1.)

(a) Pixelwise classification accuracy

Update Depth to:	local	PEDT										
		$\Gamma = 0$	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
layer 8 (leaf)	76.66	85.93	86.02	86.03	85.83	85.58	85.26	84.88	84.29	83.52	82.85	84.82
layer 7	76.66	85.93	86	85.98	85.9	85.6	85.5	85.32	85.02	84.48	83.59	84.47
layer 6	76.66	85.93	86.08	86.11	86.14	86.11	86.03	85.99	85.81	85.73	85.49	84.68
layer 5	76.66	85.93	86.12	86.13	86.18	86.16	86.16	86.21	86.26	86.27	86.27	85.7
layer 4	76.66	85.93	85.88	85.89	85.9	85.9	85.9	85.91	85.88	85.85	85.79	85.82
layer 3	76.66	85.93	85.91	85.91	85.89	85.89	85.9	85.89	85.89	85.9	85.9	85.93

(b) Difference in the average variational loglikelihood
of the test images between a static PEDT and a PEDT with Γ updates

Update Depth to:	PEDT									
	$\Gamma = 0.10$	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
layer 8 (leaf)	-28.1	-50.7	-67.6	-83.1	-98.6	-114	-127	-135	-133	32
layer 7	-5.64	-8.41	-10.9	-9.47	-7.31	-3.13	0.188	3.65	7.69	39.4
layer 6	1.58	2.7	4.97	6.41	9.39	11.7	15	17.7	22.1	33.2
layer 5	6.36	10.6	16.8	22.8	28.6	34.7	40.1	44.7	47.8	49.1
layer 4	1.78	3.35	5.07	7.59	9.82	12.1	14.3	16.2	17.8	16.9
layer 3	0.338	0.532	1.58	2.42	3.02	4.02	5.25	5.9	6.48	7.77

(c) Percentage of images with an improved variational log likelihood over the static PEDT

Update Depth to:	Γ									
	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
layer 8 (leaf)	2.041	0	0	0	0	0	0	2.041	4.082	87.76
layer 7	28.57	32.65	40.82	55.1	57.14	59.18	59.18	63.27	67.35	85.71
layer 6	57.14	61.22	65.31	63.27	67.35	65.31	69.39	71.43	73.47	77.55
layer 5	93.88	97.96	97.96	100	100	100	100	100	100	95.92
layer 4	77.55	85.71	89.8	97.96	97.96	97.96	97.96	97.96	100	95.92
layer 3	71.43	65.31	81.63	87.76	87.76	95.92	97.96	97.96	97.96	100

Table 2: Varying the update depth. (Noise variance added = 0.1, width parameter =1.)