Unsupervised Learning of Multiple Aspects of Moving Objects from Video

Michalis K. Titsias and Christopher K.I. Williams

School of Informatics, University of Edinburgh,
Edinburgh EH1 2QL, UK
M.Titsias@sms.ed.ac.uk, c.k.i.williams@ed.ac.uk

Abstract. A popular framework for the interpretation of image sequences is based on the layered model; see e.g. Wang and Adelson [8], Irani et al. [2]. Jojic and Frey [3] provide a generative probabilistic model framework for this task. However, this layered models do not explicitly account for variation due to changes in the pose and self occlusion. In this paper we show that if the motion of the object is large so that different aspects (or views) of the object are visible at different times in the sequence, we can learn appearance models of the different aspects using a mixture modelling approach.

1 Introduction

We are given as input a set of images containing views of multiple objects, and wish to learn appearance models of each of the objects. A popular framework for this problem is the layer-based approach which models an image as a composite of 2D layers each one modelling an object in terms of its appearance and region of support or mask, see e.g. [8] and [2].

A principled generative probabilistic framework for this task has been described in [3], where the background layer and the foreground layers are synthesized using a multiplicative or alpha matting rule which allows transparency of the objects. Learning using an exact Expectation-Maximization (EM) algorithm is intractable and the method in [3] uses a variational inference scheme considering translational motion of the objects. An alternative approach is that presented in [9] where the layers strictly combine by occlusion and learning of the objects is carried out sequentially by extracting one object at each stage.

Layered models do not explicitly represent variation in object appearance due to changes in the pose of the object and self occlusion. In this paper we describe how the generative model in [9] can be properly modified so that the pose of an object can vary significantly. We achieve this by introducing a set of mask and appearance pairs each one associated with a different viewpoint of the object. Such a model learns a set of different views (or aspects, [4]) of an object.

To learn different viewpoint object models we consider video training data and we first apply approximate tracking of the objects before knowing their full structure. This provides an estimate of the transformation of the object in each frame so that by reversing the effect of the transformation (frame stabilization) the viewpoint models for

P. Bozanis and E.N. Houstis (Eds.): PCI 2005, LNCS 3746, pp. 746–756, 2005.

that object can be learned using a mixture modelling approach. The tracking algorithm finds first the background while moving foreground objects are tracked at later stages. For the foreground objects our tracking algorithm is based on a dynamic appearance model of the object (appearance and mask) which is updated recursively as we process the frames.

The structure of the remainder of the paper is as follows: In section 2 we describe the layered generative model which can learn a single aspect for each foreground object. In section 3 we extent this model so that to learn multiple views of the same object. In section 4 we describe an algorithm for tracking multiple objects in image sequence. In section 5 we show some results in two video sequences and we conclude with a discussion in section 6.

2 Generative Layered Model

For simplicity we will present the generative model assuming that there are two layers, i.e. a foreground object and a static background. Later in this section we will discuss the case of arbitrary number of foreground layers and a moving background.

Let $\mathbf{b}$ denote the appearance image of the background arranged as a vector. Assuming that the background is static, $\mathbf{b}$ will have the same size as the data image size (although note that for moving backgrounds, $\mathbf{b}$ will need to be larger than the image size). Each entry b_i stores the ith pixel value which can either be a grayscale intensity value or a colour value. In our implementation we allow coloured images where $\mathbf{b}_i$ is a three-dimensional vector in the RGB space. However, for notation convenience next we assume that b_i is a scalar representing a grayscale value.

In contrast to the background, the foreground object occupies some region of the image and thus to describe this layer we need both an appearance $\mathbf{f}$ and mask $\boldsymbol{\pi}$. The foreground is allowed to move so there is an underlying transformation j that e.g. corresponds to translational or affine motion and a corresponding transformation matrix so that $T_j\mathbf{f}$ and $T_j\boldsymbol{\pi}$ is the transformed foreground and mask, respectively. We assume that the foreground and background strictly combine by occlusion, thus a pixel in an observed image is either foreground or the background. This is expressed by a vector of binary latent variables $\mathbf{s}$, one for each pixel drawn from the distribution [9]

$$P(\mathbf{s}|j) = \prod_{i=1}^{P} (T_j\boldsymbol{\pi})_i^{s_i} (\mathbf{1} - T_j\boldsymbol{\pi})_i^{1-s_i}. \tag{1}$$

Note that each variable s_i is drawn independently so that for pixel i, if $(T_j\boldsymbol{\pi})_i \simeq 0$, then the pixel will be ascribed to the background with high probability, and if $(T_j\boldsymbol{\pi})_i \simeq 1$, it will be ascribed to the foreground with high probability. Note that $\mathbf{s}$ is the binary mask of the foreground object in an example image, while $\boldsymbol{\pi}$ is the prior untransformed mask that captures roughly the shape of the object stored in $\mathbf{f}$.

Selecting a transformation j using an uniform prior P_j over J possible values and a binary mask $\mathbf{s}$, an image $\mathbf{x}$ is drawn by a Gaussian

$$p(\mathbf{x}|j, \mathbf{s}) = \prod_{i=1}^{P} N(x_i; (T_j\mathbf{f})_i, \sigma_f^2)^{s_i} N(x_i; b_i, \sigma_b^2)^{1-s_i}, \tag{2}$$

where each pixel is drawn independently from the above conditional density.

To express the likelihood of an observed image $p(\mathbf{x})$ we marginalise out the latent variables which are the transformation j and the binary mask $\mathbf{s}$. Particularly, we first sum out $\mathbf{s}$ using (1) and (2) and obtain

$$p(\mathbf{x}|j) = \prod_{i=1}^{P} (T_j \boldsymbol{\pi})_i N(x_i; (T_j \mathbf{f})_i, \sigma_f^2) + (\mathbf{1} - T_j \boldsymbol{\pi})_i N(x_i; b_i, \sigma_b^2). \quad (3)$$

Using now a uniform prior over the transformation P_j, the probability of an observed image $\mathbf{x}$ is $p(\mathbf{x}) = \sum_{j=1}^{J} P_j p(\mathbf{x}|j)$. Given a set of images $\{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ we can adapt the parameters $\theta = \{\mathbf{b}, \mathbf{f}, \boldsymbol{\pi}, \sigma_f^2, \sigma_b^2\}$ maximizing the log likelihood using the EM algorithm.

This model can be extended so that to have a moving background and L foreground objects. For example, for two foreground layers with parameters $(\mathbf{f}_1, \boldsymbol{\pi}_1, \sigma_1^2)$ and $(\mathbf{f}_2, \boldsymbol{\pi}_2, \sigma_2^2)$ and also a moving background the analogous of equation (3) is

$$p(\mathbf{x}|j_1, j_2, j_b) = \prod_{i=1}^{P} (T_{j_1} \boldsymbol{\pi}_1)_i N(x_i; (T_{j_1} \mathbf{f}_1)_i, \sigma_1^2) + (\mathbf{1} - T_{j_1} \boldsymbol{\pi}_1)_i \times \left[(T_{j_2} \boldsymbol{\pi}_2)_i N(x_i; (T_{j_2} \mathbf{f}_2)_i, \sigma_2^2) + (\mathbf{1} - T_{j_2} \boldsymbol{\pi}_2)_i N(x_i; (T_b \mathbf{b})_i, \sigma_b^2) \right], \quad (4)$$

where j_1, j_2 and j_b denote the transformation of the first foreground object, the second foreground object and the background, respectively.

Applying an exact EM algorithm to learn the parameters of the above model is in general intractable. For example, for the case of L foreground objects that can be transformed in J ways, there exist J^{L+1} configurations that can generate an observed image, which grows exponentially with the number of objects. For this reason approximate algorithms should be considered, e.g. in [3] an approximate variational method has been applied.

3 Incorporating Multiple Viewpoints

In this section we generalize the layer-based model for multiple moving objects so that the viewpoint of each foreground object can arbitrarily change. Section 3.1 describes the generative layered model for changeable viewpoints and section 3.2 discusses training the model.

3.1 Multiple Viewpoints

The layered model presented in section 2 assumes that each layer can change mainly due to a 2D planar motion. However, in many video sequences this assumption will be hardly true e.g. a foreground object can undergo 3D rotation so that at different times we may see the object from different viewpoints. For example, Figure 2a shows three frames of a sequence capturing a man walking; clearly the man's pose changes substantially during time. Next we generalize the layered model so that the appearance

of a foreground object can be chosen from a set of possible appearances associated with different viewpoints.

Assume again that there are two layers: one static background and one moving foreground object. We introduce a discrete latent variable v, that can obtain V possible values indexed by integers from 1 to V. For each value v we introduce a separate pair of appearance $\mathbf{f}^v$ and mask $\boldsymbol{\pi}^v$ defined as in section 2. Each pair $(\mathbf{f}^v, \boldsymbol{\pi}^v)$ models the appearance of the object under a certain viewpoint.

To generate an image $\mathbf{x}$ we first select a transformation j and a viewpoint v using uniform prior probabilities P_j and P_v respectively. Then we select a binary mask $\mathbf{s}$ from the distribution

$$P(\mathbf{s}|j,v) = \prod_{i=1}^{P} (T_j \boldsymbol{\pi}^v)_i^{s_i} (\mathbf{1} - T_j \boldsymbol{\pi}^v)_i^{1-s_i}, \tag{5}$$

and draw an image $\mathbf{x}$ from the Gausssian

$$p(\mathbf{x}|j,v,s) = \prod_{i=1}^{P} N(x_i; (T_j \mathbf{f}^v)_i, \sigma_f^2)^{s_i} N(x_i; b_i, \sigma_b^2)^{1-s_i}. \tag{6}$$

Note the similarity of the above expression with equation (2). The only difference is that now the appearance $\mathbf{f}$ and mask $\boldsymbol{\pi}$ are indexed by v to reflect the fact that we have also chosen a viewpoint for the foreground object.

To express the probability distribution according to which an image is generated we sum first out the binary mask and the viewpoint variable and obtain

$$p(\mathbf{x}|j) = \sum_{v=1}^{V} P_v p(\mathbf{x}|j,v), \tag{7}$$

where $p(\mathbf{x}|j,v)$ is given as in (3) with $\mathbf{f}$ and $\boldsymbol{\pi}$ indexed by v. Notice how the equation (7) relates to equation (3). Clearly now $p(\mathbf{x}|j)$ is a mixture model of the type of model given in (3). For example, if we choose to have a single viewpoint the latter expression reduces to the former one.

It is straightforward to extent the above model to the case of L foreground layers with varying viewpoints. In this case we need a separate viewpoint variable v_ℓ for each foreground object and a set of appearance and mask pairs: $\{\mathbf{f}_\ell^{v_\ell}, \boldsymbol{\pi}_\ell^{v_\ell}\}$, $v_\ell = 1, \ldots, V_\ell$. For example, when we have two foreground objects and a moving background the conditional $p(\mathbf{x}|j_1, j_2, j_b, v_1, v_2)$ is given exactly as in (4) by introducing suitable indexes to the foreground appearances and masks that indicate the choices made for the viewpoint variables.

3.2 Learning the Model

Training the above model using an exact EM algorithm is intractable. For L foreground objects and a moving background, each one undergoing J transformations and assuming V aspects for each foreground object, the time complexity is $O(J^{L+1}V^L)$. Approximate training methods such as the variational EM algorithm of [3] or the one-object-at-a-time method of [9] could be applied. However, it is clear that adding V views of each

object will complicate the training process, and there is a danger of confusion between views of one object and different objects.

A reliable method for training the model can be based on two stage learning framework. In the first stage we compute the 2D planar transformations of a foreground object in images, while in the second stage we learn the different viewpoint models by carrying out simple clustering. Particularly, we first approximate the transformation of an object in each frame, which is simply a motion according to which this frame is matched to a reference frame. Given these transformations it is easy to reverse their effect so as to transform each image into a reference frame where the viewpoint models for that object can be learned using a mixture model. Intuitively, what is happening here is that we are transforming the video so as to stabilize a given object; this greatly facilitates the learning of the viewpoint models for that object.

Assuming a set of training images $\{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ the steps of this algorithm are the following:

1. Track first the background $\mathbf{b}$ in order to approximate the transformation j_b^n of each image $\mathbf{x}^n$ and then learn the background by maximizing the log likelihood

$$L_b = \sum_{n=1}^{N} \log p_b(\mathbf{x}^n | j_b^n) = \sum_{n=1}^{N} \log \prod_{i=1}^{P} p_b(x_i^n; (T_{j_b^n} \mathbf{b})_i) \tag{8}$$

   and $p_b(x_i; (T_{j_b} \mathbf{b})_i)$ is given by equation (12).
2. Compute all the planar transformations $\{j_\ell^n\}$ of a foreground object ℓ using tracking (see section 4)
3. Learn the parameters of the object by maximizing the log likelihood

$$L_\ell = \sum_{n=1}^{N} \log \sum_{v_\ell=1}^{V_\ell} P_{v_\ell} p(\mathbf{x} | j_\ell^n, j_b^n, v_\ell). \tag{9}$$

In (9) the conditional density $p(\mathbf{x}|j_\ell^n, j_b^n, v_\ell)$ is given by

$$p(\mathbf{x}|j_\ell^n, j_b^n, v_\ell) = \prod_{i=1}^{P} (T_{j_\ell} \boldsymbol{\pi}_\ell^{v_\ell})_i p_{f_\ell}(x_i; (T_{j_\ell} \mathbf{f}_\ell^{v_\ell})_i) + (\mathbf{1} - T_{j_\ell} \boldsymbol{\pi}_\ell^{v_\ell})_i p_b(x_i; (T_{j_b^n} \mathbf{b})_i), \tag{10}$$

where we have replaced the Gaussian foreground and background pixel densities by the following robustified counterparts

$$p_f(x_i; f_i) = \alpha_f N(x_i; f_i, \sigma_f^2) + (1 - \alpha_f) U(x_i). \tag{11}$$

and

$$p_b(x_i; f_i) = \alpha_b N(x_i; b_i, \sigma_b^2) + (1 - \alpha_b) U(x_i). \tag{12}$$

Here $U(x_i)$ is an uniform distribution in the range of all possible pixel values and α_f and α_b express prior probabilities that a foreground (resp. background) pixel is not occluded. This robustification allow us to deal with occlusion caused by all the other foreground objects except the ℓth object. Clearly, these objects can occlude the background

and sometimes also the foreground object of interest. Thus, any time a foreground or background pixel will be occluded that will be explained by the uniform component $U(x_i)$ [5,9].

It is straightforward to maximize the log likelihood in (9) using the EM algorithm to deal with the missing information concerning the viewpoint variable, the binary mask $\mathbf{s}$ and the indicators of the outlier process. Once models for all objects have been learned in this fashion it is possible to refine the masks and appearances by optimizing them jointly, using an analogue of equation (4).

So far we have not discussed how we learn the background and approximate the transformations of the foreground objects. We doing this based on tracking that is described in the next section.

4 Tracking the Objects

In this section we present a tracking algorithm that applies to a sequence of frames $(\mathbf{x}^1, \ldots, \mathbf{x}^N)$ and approximates the corresponding set of transformations $(j^1, \ldots, j^N)$ that describe the motion of a single object.

We wish first to track the background and ignore all the other motions related to the foreground objects. To introduce the idea of our tracking algorithm assume that we know the appearance of the background $\mathbf{b}$ as well as the transformation j_b^1 that associates $\mathbf{b}$ with the first frame. Since motion between successive frames is expected to be relatively small we can determine the transformation j_b^2 for the second frame by searching over a small discrete set of neighbouring transformations centred at j_b^1 and inferring the most probable one, i.e. the one giving the highest likelihood $p_b(\mathbf{x}^2|j_b^2)$ (see equation (8)), assuming a uniform prior. This procedure can be applied recursively to determine the sequence of transformations in the entire video. However, the background $\mathbf{b}$ is not known in advance, but we can still apply roughly the same tracking algorithm by suitably initializing and updating the background $\mathbf{b}$ as we process the frames. This algorithm is described in detain in [7]. Once tracking of the background is completed we can learn its full structure by maximizing the log likelihood (8).

Assume now that the background has been learned. The pixels which are explained by the background in each image $\mathbf{x}^t$ are flagged by the background responsibilities $\mathbf{r}^t(j_b^t)$ computed by the equation

$$r_i(j_b) = \frac{\alpha_b N(x_i; (T_{j_b}\mathbf{b})_i, \sigma_b^2)}{\alpha_b N(x_i; (T_{j_b}\mathbf{b})_i, \sigma_b^2) + (1-\alpha_b)U(x_i)}. \tag{13}$$

Clearly the mask $\overline{\mathbf{r}}^t(j_b^t) = \mathbf{1} - \mathbf{r}^t(j_b^t)$ roughly indicates all the pixels of frame $\mathbf{x}^t$ that belong to the foreground objects. By focusing only on these pixels we wish to start tracking one of the foreground objects through the entire video sequence and ignore for the moment the rest foreground objects.

Our algorithm tracks the first object by simultaneously updating its mask $\boldsymbol{\pi}_1$ and appearance $\mathbf{f}_1$. The mask and the appearance are initialized so that $\boldsymbol{\pi}_1 = \mathbf{0.5} * \overline{\mathbf{r}}^t(j_b^t)$ and $\mathbf{f}_1 = \mathbf{x}^1$, where $\mathbf{0.5}$ denotes the vector with 0.5 values[1]. Due to this initialization

[1] The value of 0.5 is chosen to express our uncertainty about whether these pixels will ultimately be in the foreground mask or not.

we know that the first frame is untransformed, i.e. j_1^1 is the identity transformation. To determine the transformation of the second frame and in general the transformation j_1^{t+1}, with $t \geq 1$, of the frame $\mathbf{x}^{t+1}$ we find the most probable value of j_1^{t+1} according to the posterior

$$R(j_1^{t+1}) \propto \exp\Bigg\{ \sum_{i=1}^{P} (\mathbf{w}_1^{t+1})_i \log\Big((T_{j_1^{t+1}} \boldsymbol{\pi}_1^t)_i \times \\ p_f(x_i^{t+1}; (T_{j_1^{t+1}} \mathbf{f}_1^t)_i) + (\mathbf{1} - T_{j_1^{t+1}} \boldsymbol{\pi}_1^t)_i U(x_i^{t+1}) \Big) \Bigg\}, \quad (14)$$

where $\mathbf{w}_1^{t+1} = \overline{\mathbf{r}}^{t+1}(j_b^{t+1})$. $R(j_1^{t+1})$ measures the goodness of the match at those pixels of frame $\mathbf{x}^{t+1}$ which are not explained by the background. Note that as the objects will, in general, be of different sizes, the probability $R(j_1^{t+1})$ over the transformation variable will have greater mass on transformations relating to the largest object. Recall that $p_f(x_i^{t+1}; (T_{j_1^{t+1}} \mathbf{f}_1)_i)$ includes an outlier component so that some badly misfit pixels can be tolerated.

Once we determine j_1^{t+1} we update both the mask $\boldsymbol{\pi}_1$ and appearance $\mathbf{f}_1$. The mask is updated according to

$$\boldsymbol{\pi}_1^{t+1} = \boldsymbol{\pi}_1^t + \beta_{\boldsymbol{\pi}} \left(T^{-1}_{j_1^{t+1}} [\overline{\mathbf{s}}^{t+1}(j_1^{t+1})] - \boldsymbol{\pi}_1^t \right), \quad (15)$$

where T^{-1} denotes the inverse transformation and β_π takes values in the range $[0, 1]$. The vector $\overline{\mathbf{s}}^{t+1}(j_1^{t+1})$ expresses the segmentation of the object in the frame $\mathbf{x}^{t+1}$ so that each $\overline{s}_i^{t+1}(j_1^{t+1})$ stores the probability

$$\overline{s}_i^{t+1}(j_1^{t+1}) = \frac{(T_{j_1^{t+1}} \boldsymbol{\pi}_1^t)_i p_{f_1}(x_i^{t+1}; (T_{j_1^{t+1}} \mathbf{f}_1^t)_i)}{(T_{j_1^{t+1}} \boldsymbol{\pi}_1^t)_i p_{f_1}(x_i^{t+1}; (T_{j_1^{t+1}} \mathbf{f}_1^t)_i) + (\mathbf{1} - T_{j_1^{t+1}} \boldsymbol{\pi}_1^t)_i p_b(x_i^{t+1}; (T_{j_b^{t+1}} \mathbf{b})_i)}, \quad (16)$$

for the pixel i. The update (15) defines the new mask as a weighted average of the stabilized segmentation in the current frame (i.e. $T^{-1}_{j_1^{t+1}}[\overline{\mathbf{s}}^{t+1}(j_1^{t+1})]$) and the previous value of the mask. β_π is the weight of the stabilized segmentation in each current frame, e.g. if $\beta_\pi = 1$, then $\boldsymbol{\pi}_1^{t+1} = T^{-1}_{j_1^{t+1}} \overline{\mathbf{s}}^{t+1}(j_1^{t+1})$. The update for the foreground appearance $\mathbf{f}_1$ is given by

$$\mathbf{f}_1^{t+1} = \mathbf{f}_1^t + \beta_f \left(T^{-1}_{j_1^{t+1}} [\overline{\mathbf{s}}(j_1^{t+1}) * \mathbf{r}^{t+1}(j_1^{t+1}) * \mathbf{x}^{t+1}] - \mathbf{f}_1^t \right), \quad (17)$$

where $\mathbf{y} * \mathbf{z}$ denotes the element-wise product of the vectors $\mathbf{y}$ and $\mathbf{z}$. The vector $\mathbf{r}^{t+1}(j_1^{t+1})$ is defined similarly to equation (13) and stores the probabilities that the pixels of the object in the current frame have not changed dramatically (e.g. due to occlusion). Again the above update is very intuitive. For pixels which are ascribed to the ℓth foreground (i.e. $\overline{\mathbf{s}}^{t+1}(j_1^{t+1}) * \mathbf{r}^{t+1}(j_1^{t+1}) \simeq 1$), the values in $\mathbf{x}^n$ are transformed by $T^{-1}_{j_1^{t+1}}$ into the stabilized frame which allows the foreground pixels found in the current frame to be averaged with the old value $\mathbf{f}^t$ in order to produce $\mathbf{f}^{t+1}$. Notice that $\mathbf{f}_1$

adapts slowly to large changes of the object appearance (caused e.g. by occlusion) due to the semantics of the vector $\mathbf{r}^{t+1}(j_1^{t+1})$. Note also that as the frames are processed tracking becomes more stable since $\boldsymbol{\pi}_1$ approximates the mask of a single object and $\mathbf{f}_1$ will contain a sharp and clear view for only the one object being tracked while the rest of the objects will be blurred; see Figure 1b for an illustrative example.

Once the first object has been tracked we learn the different viewpoint models for that object as explained in section 3.2. When these models has been learned we can go through the images to find which pixels are explained by this object. Then we can remove these pixels from consideration by properly updating each $\mathbf{w}^t$ vector which allows tracking a different object on the next stage. Note also that the new mask $\boldsymbol{\pi}_\ell$ when we track the ℓth object is initialized to $\mathbf{0.5} * \mathbf{w}^t_{\ell+1}$, while the appearance $\mathbf{f}_\ell$ is always initialized to the first frame $\mathbf{x}^1$.

5 Experiments

We will consider two video sequences: the Frey-Jojic (FJ) sequence available from `http://www.psi.toronto.edu/layers.html` (see Figure 1) and the man-walking (see Figure 2). We will also assume that the number of different aspects that we wish to learn for each foreground object is known.

The FJ sequence consists of 44 118×248 images (excluding the black border); it was also used in experiments shown in [3,9]. Three frames of this sequence are displayed in Figure 1a. This sequence can be well modelled by assuming a single view for each of the foreground objects, thus we set $V = 1$ for both objects. The results in Figure 1c were obtained using a 15×15 window of translations in units of one pixel during the tracking stage. This learning stage requires EM which converged in about 30 iterations. Figure 1b shows the evolution of the initial appearance and mask ($t = 1$) through frames 10 and 20 as we track the first object (Frey). Notice that as we process the frames the mask focuses on only one of the two objects and the appearance remains sharp only for this object. The real running time of our MATLAB implementation for processing the whole sequence was 3 minutes.

The man-walking sequence consists of 85 144×360 coloured images. Figure 2a displays three frames of that sequence. We assume that the number of different aspects of the foreground object that we wish to learn is five, i.e. $V = 5$. Figure 2b shows the learned appearance and mask pairs of the different viewpoint models for the foreground object. When we applied the tracking algorithm we used a window of 15×15 translations in units of one pixel. Notice that each different pair of mask and appearance has modelled a different pose of the man. However, some of the masks are noisy. We hope to improve on that by adding spatial continuity constraints (e.g. using a MRF for the binary variable $\mathbf{s}$). Processing the whole video took about 20 mins, where the most of the time was spent in fitting the mixture model for learning the object views.

6 Discussion

Above we have extended the generative model for learning multiple moving objects so that to deal with large viewpoint variation. Particularly, we introduced multiple view-

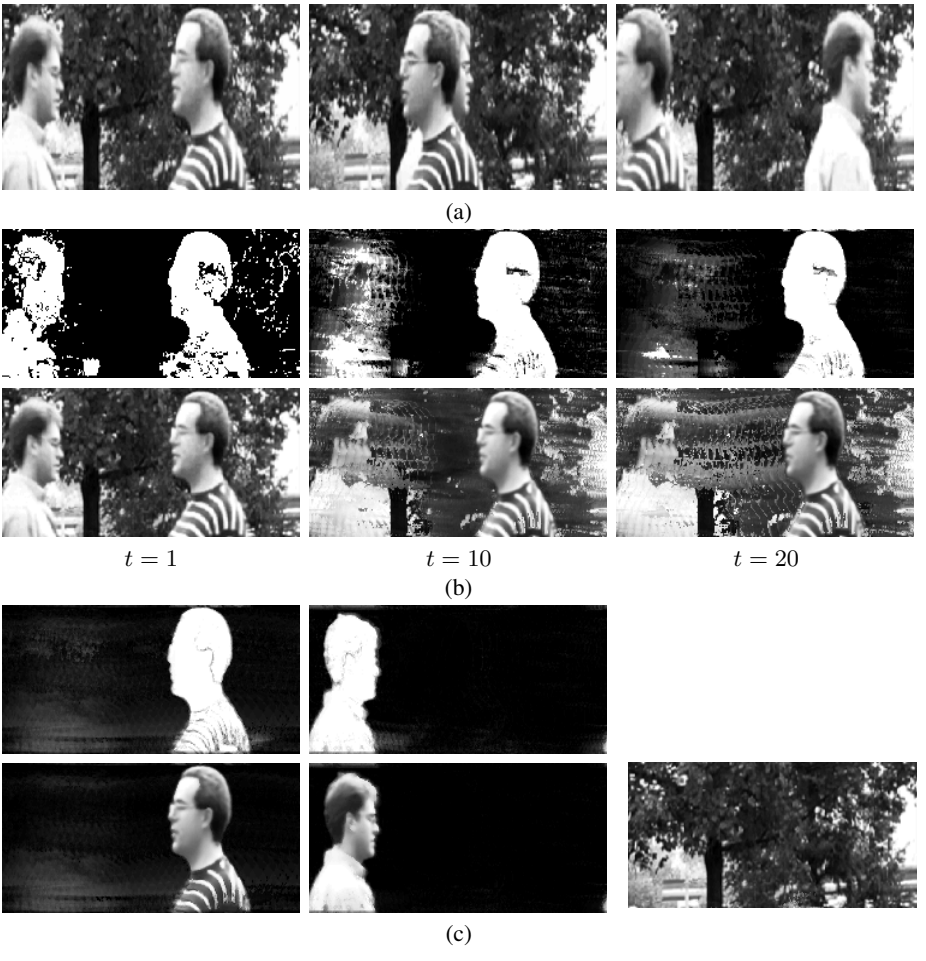

Fig. 1. Panel (a) shows three frames of the Frey-jojic sequence. Panel (b) shows the evolution of the mask $\boldsymbol{\pi}_1$ (top row) and the appearance $\mathbf{f}_1$ (bottom row) at times 1, 10 and 20 as we track the first object (Frey). Notice how the mask becomes focused on one of the objects (Frey) and how the appearance remains clear and sharp only for Frey. Panel (c) shows the mask and the element-wise product of the mask and appearance model ($\boldsymbol{\pi} * \mathbf{f}$) learned for Frey (first column from the left) and Jojic (second column) using the algorithm described in the text. The plot in the third column shows the learned background.

point models for each foreground object. These models are learned using a mixture modelling approach applied to the stabilized frames. To stabilize the frames we approximate the transformations of each object in the video using a tracking algorithm.

The mechanism for dealing with multiple viewpoints using mixture models has been considered before in [1]. However, in this method they consider a single object present in the images against a clutter background and only appearance images of different poses of the object are learned (not masks). In contrast, our method can be applied to

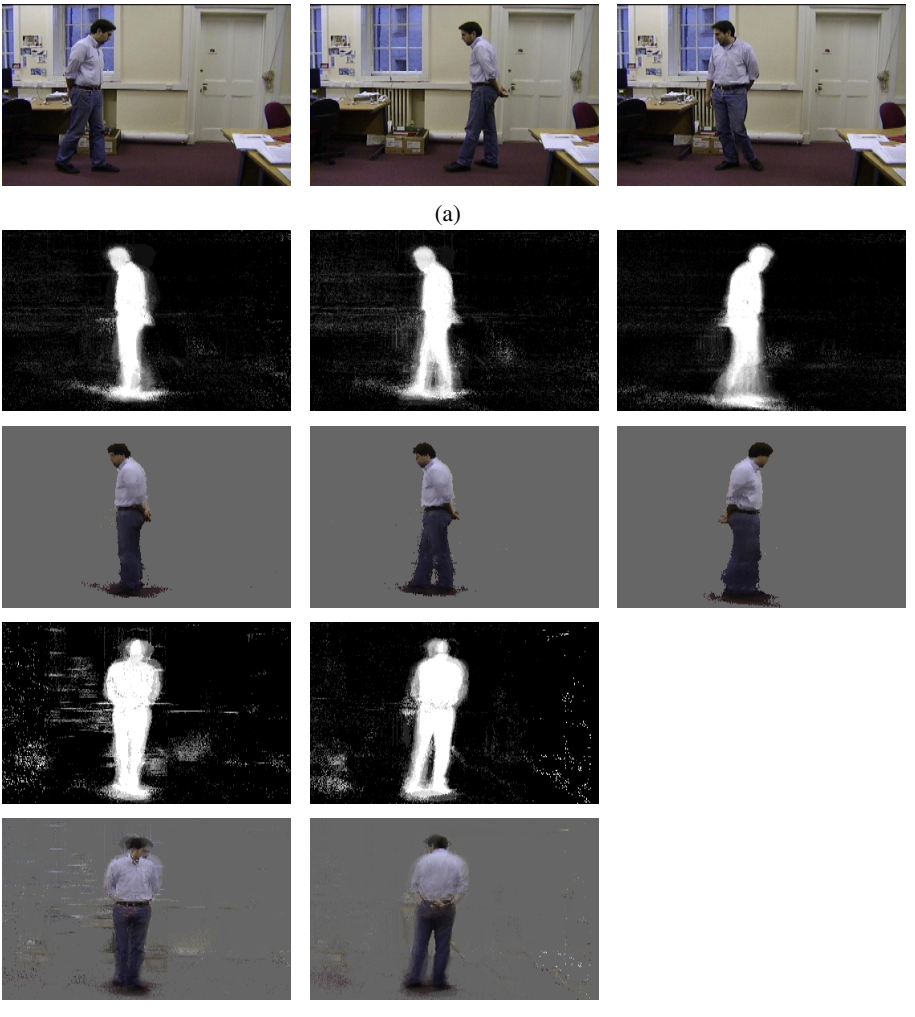

Fig. 2. Panel (a) shows three frames of the man-walking sequence. Panel (b) shows the the pairs of mask and the element-wise product of the mask and appearance model (showing against a grey background) for all different viewpoint models.

images with multiple objects and learn the background as well as the appearances and masks of the foreground objects.

Regarding tracking methods for learning moving layers, the method of [2] is much relevant to ours. They do motion estimation using optical flow by matching the current frame against an accumulative appearance image of the tracked object. Although they do not take into account issues of occlusion, so that if a tracked object becomes occluded for some frames, it may be lost. The work of [6] is also relevant in that it deals with a background model and object models defined in terms of masks and appear-

ances. However, note that the mask is assumed to be of elliptical shape (parameterised as a Gaussian) rather than a general mask. The mask and appearance models are dynamically updated during tracking, however the initialization of each model is handled by a "separate module", and is not obtained automatically.

Some issues for further work include dealing with objects that have internal variability, and modelling non-articulated moving objects. Another issue is to automatically identify how many views are needed to efficiently model the appearance of each object and also to determine the number of objects in the images.

References

1. B. J. Frey and N. Jojic. Transformation Invariant Clustering Using the EM Algorithm. *IEEE Trans Pattern Analysis and Machine Intelligence*, 25(1):1–17, 2003.
2. M. Irani, B. Rousso, and S. Peleg. Computing Occluding and Transparent Motions. *International Journal of Computer Vision*, 12(1):5–16, 1994.
3. N. Jojic and B. J. Frey. Learning Flexible Sprites in Video Layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2001*. IEEE Computer Society Press, 2001. Kauai, Hawaii.
4. J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.
5. S. Rowe and A. Blake. Statistical Background Modelling For Tracking With A Virtual Camera. In D. Pycock, editor, *Proceedings of the 6th British Machine Vision Conference*, volume volume 2, pages 423–432. BMVA Press, 1995.
6. H. Tao, H. S. Sawhney, and R. Kumar. Dynamic Layer Representation with Applications to Tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:134–141, 2000.
7. M. K. Titsias and C. K. I. Williams. Fast unsupervised greedy learning of multiple objects and parts from video. In *Proc. Generative-Model Based Vision Workshop*, 2004.
8. J. Y. A. Wang and E. H. Adelson. Representing Moving Images with Layers. *IEEE Transactions on Image Processing*, 3(5):625–638, 1994.
9. C. K. I. Williams and M. K. Titsias. Greedy Learning of Multiple Objects in Images using Robust Statistics and Factorial Learning. *Neural Computation*, 16(5):1039–1062, 2004.