Mining Appearance Models Directly from Compressed Video

Datong Chen
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213
datong@cs.cmu.edu

Qiang Liu
Department of Neurological Surgery
University of Pittsburgh
Pittsburgh PA 15213
qliu@neuronet.pitt.edu

Mingui Sun
Department of Neurological Surgery
University of Pittsburgh
Pittsburgh PA 15213
mrsun@neuronet.pitt.edu

ABSTRACT

In this paper, we propose an approach to learning appearance models of moving objects directly from compressed video. The appearance of a moving object changes dynamically in video due to varying body poses, lighting conditions, and partial occlusions. Efficiently mining the appearance models of objects is a crucial and challenging technology to support content-based video coding, clustering, indexing, and retrieval at the object level. The proposed approach learns the appearance models of moving objects in the spatial-temporal dimension of video data by taking advantage of the MPEG video compression format. It detects a moving object and recovers the trajectory of each macro-block covered by the object using the motion vector present in the compressed stream. The appearances are then reconstructed in the DCT domain along the object's trajectory, and modeled as a mixture of Gaussians (MoG) using DCT coefficients. We prove that, under certain assumptions, the MoG model learned from the DCT domain can achieve pixel-level accuracy when transformed back to the spatial domain, and has a better band-selectivity compared to the MoG model learned in the spatial domain. We finally cluster the MoG models to merge the appearance models of the same object together for object-level content analysis.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Content Analysis nd Indexing; I.5.1 [**Pattern Recognition**]: Models

General Terms

object modeling

Keywords

video mining, appearance modeling, mixture of Gaussian, DCT transform, compressed video

MDM/KDD'06, August 20, 2006, Philadelphia, Pennsylvania, USA.

1. INTRODUCTION

As video sources grow explosively, efficient methods for indexing and retrieving unstructured video data at the object-level are in high demand for many different applications. Content analysis at the object-level allows us to create links among objects in video; for example, jumping to the next shot where the same object appears. Appearances of a moving object changes dynamically in video due to varying body poses, lighting conditions, and partial occlusions. Modeling object appearances of an object in video, therefore, requires a large amount of training data, which is expensive to label manually. Therefore, unsupervised object modeling in video is desired, which depends on accurate object detection and tracking. However, these two technical components usually involve high computational cost. And, as the video data are mostly presented in compressed digital formats, there is an extra cost for video decompression if detecting and tracking are conducted in the spatial domain. In this paper, we are interested in the unsupervised mining of objects' appearance models in video at a low computational cost.

Extracting and modeling video objects have been investigated by many researchers in both the computer vision and multimedia communities. Much research has been directed to extracting a video object from its background to segment the boundary (arbitrary shape) of the object in each frame. Although color [9], edge [8] and contour [16] are useful cues, motion has been a dominant feature used for extracting objects from video [18, 9, 8, 16]. These methods are able to extract a video object precisely at the pixel-level, which is then coded as a collection of video object planes. However, high accurate pixel-level extraction is very time consuming and is not practical to application to large amounts of compressed videos for learning appearance models.

An alternative approach is to learn a model to represent the entire collection of video object appearances. To represent the appearances of a video object, the model uses a compact format to remember the appearances of all the video object planes while at the same time keeping most of the appearance information to support future video indexing and visual analysis tasks. The video object planes of a video object can contain rich and dynamic appearance information, such as shape, color, texture and motion, that benefits content-based video indexing and retrieval, and visual analysis such as object classification. To build an appearance model, a

video object has to be tracked and extracted from its background first. By using a probabilistic model, background pixels included in object appearances due to the inaccurate extraction can be implicitly distinguished to achieve a sort of pixel-level accuracy. Therefore, the extraction does not require a pixel-level accuracy, but a block or macro-block level. A large amount of work has been proposed by computer vision researchers to model objects in a scene, such as a mixture of Gaussians (MoG) method [5], adaptive filter methods [10], minimal and maximal intensity value methods [14], PDE level set [6], Hidden Markov models (HMMs) [12], and kernel density estimation techniques [13]. However, these algorithms are designed for accurate extraction and paid little attention to efficiency in the application to video in a compressed format, which we have to uncompress before applying these methods.

In order to improve efficiency, some researchers in the multimedia community have proposed to perform video indexing and retrieval in compressed video directly. For example, motion vectors in the MPEG codec have been used for video object extraction. There are also some methods proposed to build background models in the compressed domain [15, 17]. These methods have been developed for extracting moving objects or for encoding purposes. Most algorithms use only discrete cosine transform (DCT) energy (DC) coefficient and work on the block-level. For example, the algorithm in [17] extracts objects at a size of 8×8 blocks and cannot obtain accurate object contour. [1] presented a method that could extract video object precisely at the pixel-level. However, the method requires a decoding step to obtain appearances of each video object. Another effort in the compression domain is to use DCT coefficients to support image and video indexing and retrieval [11, 19]. In this class of work, DCT coefficients were used as texture features to train classifiers or clusters to retrieve images or identify a specific object in high-level visual analysis (e.g., a known human face). In summary, the previous work on object appearance extraction and modeling has been studied in the compressed domain at the block-level.

In this paper, we propose an unsupervised algorithm for mining object appearances in compressed video streams. We construct appearance models using MPEG motion vectors and DCT coefficients embedded in the MPEG video. The proposed algorithm detects a moving object and recovers the trajectory of each macro-block covered using the motion vector. The appearances are then reconstructed in the DCT domain following the object's trajectory, and modeled as a mixture of Gaussian (MoG) using DCT coefficients. We prove that, under certain assumptions, the MoG model learned from the DCT domain can achieve pixel-level accuracy after being transformed back to the spatial domain and has better band-selectivity compared to the MoG model learned in the spatial domain. Finally, the algorithm clusters the extracted MoG models to merge the appearance models of the same object together. Using the resulting DCT domain appearance models, object-level video indexing and retrieval can be performed in compressed MPEG video directly without decompression.

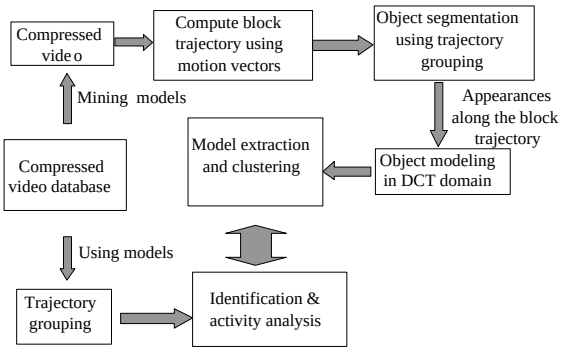


Figure 1: The diagram of the proposed algorithm.

2. A SCHEME FOR MINING OBJECT APPEARANCE MODELS IN COMPRESSED VIDEO

Figure 1 illustrates the proposed scheme for mining video objects from compressed video without decompression. The decompression step can be computationally expensive when we have to access a huge amount of video data in a limited time, for example, building an index for a specific person in 10 years of news video. The proposed scheme saves the decompression step in both the appearance modeling and object retrieval to improve the efficiency.

In this scheme, we first recover the trajectory of each block using motion vectors coded in the compression domain. The challenge is that the block partitions are all in regular locations in each frame, but a trajectory of a macro-block through the video computed from motion vectors may not always follow these regular locations. In fact, the trajectory of a macro-block of a moving object aligns most likely between two or four regular blocks in reference frames. Section 3 discusses the trajectory recovery for regular and irregular aligned blocks. Second, blocks within a macro-block are assumed to have the same trajectory as the macro-block. We then group together to segment individual objects (see section 4). Third, appearance of the object is reconstructed following its trajectory. The MoG model of the appearances of each object is learned along its trajectory in the DCT domain. Finally, we cluster the extracted MoG models to merge the appearance models of the same object together.

3. RECOVERING BLOCK TRAJECTORIES IN COMPRESSED VIDEO

Most current video data is stored in popular compression formats, such as MPEG-1, 2, 4, and H.26X. As a common feature of these formats, video frames are organized as a sequence of three types of frames: I, P, B. Each I frame consists of a list of 64-dimensional DCT coefficient vectors in which each of the coefficient vectors is transformed from an 8×8 size partition block of pixels in the original spatial domain. A P frame also consists of a list of DCT coefficient vectors. However, these coefficient vectors are not transformed directly from the original spatial domain. Four adjacent blocks are treated as a macro-block, which associates with a motion vector pointing to the most similar macro-block in a reference frame. The reference frame is usually

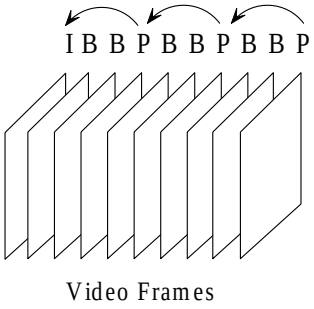


Figure 2: The GOP structure of compressed video.

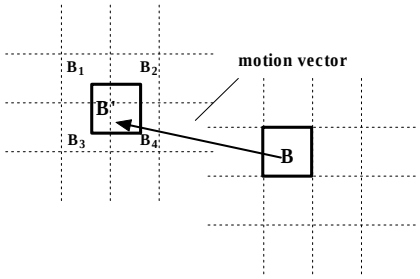


Figure 3: An illustration of an irregular block referenced by a motion vector.

the previous I or P frame as shown in Figure 2. The DCT coefficient vectors in a P frame are transformed from the residues of the macro-block and its reference. A B-frame usually contains much fewer residues, which is not used in this research.

Since the motion vector of a macro-block points to the location of this macro-block in the previous reference frame, it is possible to back-track the locations of the macro-block in the previous sequence. This sequence of locations is the trajectory of the macro-block, which is recovered first in our approach. For the blocks in each macro-block, we assume that they share the same motion vector. Recovering the trajectory of a block that is always at regular locations in video is a trivial task. However, blocks are usually located at irregular locations in reference frames, as shown in Figure 3.

The reference block B' of the current block B in its reference frame is located between four blocks B_1, B_2, B_3, and B_4, since the block B' is not located at a regular place. There is no motion vector corresponding to the block B' to the next reference frame. To obtain the motion vector of the block B', we perform interpolation using the motion vectors of the four covered blocks $B_{1,2,3,4}$. The motion vectors of the blocks $B_{1,2,3,4}$ are denoted by $mv(B_i)$, $i = 1, 2, 3, 4$. The common sub-blocks shared by blocks B_i and B' are defined as $B' \bigcap B_i$. The area of a block or sub-block is defined as $|B|$. We therefore interpolate the motion vector $mv(B')$ of the block B' to the next reference frame as:

$$mv(B') = \sum_{i=1}^{4} \frac{|B' \bigcap B_i|}{|B_i|} mv(B_i). \quad (1)$$

4. GROUPING AN OBJECT USING BLOCK TRAJECTORIES

Blocks of a video object are believed to have similar motion behaviors in a short period of time as assumed by pixel-or feature- based motion extraction approaches. Let us consider the trajectory of each block as a feature and then group the blocks that have similar trajectories into one video object. Formally, a trajectory $T_B = (mv_0(B), \ldots, mv_t(B))$ of a block B is a sequence of velocities in the x and y direction from frame 0 to frame t. To simplify the grouping process, we extract trajectory vectors with a sliding window of n frames overlapped by $\frac{n}{2}$ frames. The trajectory vectors of all blocks in a frame are first clustered using K-means in the euclidian space. Since the goal of this step is to extract foreground motion and background motion, the number of clusters is set as 2. We then group the outlier motion blocks into objects according to their spatial correlations. Isolated blocks are merged into their neighbors to obtain fewer video objects in each sliding window. A video object may cross more than n frame in a video. To simplify this research, we did not track video objects across the sliding window though it is possible to do so.

5. OBJECT APPEARANCE MODELING

Object appearance modeling is essential to identify an object or track an object in video. The goal is to remember what an object looks like in previous frames, and then recognize and localize the object in a new frame. The appearance of a video object in a frame is represented by the appearances of the blocks of pixels belonging to the video object. In the spatial domain, the appearances of a block object can be collected by cropping the reference blocks in its trajectory. However, the extraction of appearances of a block in the DCT domain is not easy because spatial cropping is not straight-forward in the DCT domain. In this section, we first present the method of extracting appearances of blocks in the DCT domain.

5.1 Extracting appearance of an irregular block in DCT domain

The appearance of a regular block in the DCT domain of an I frame is a 64-dimensional DCT coefficient vector packed in the compressed video stream. The appearance of an irregular block in an I frame, which may be referenced by a block in the nearest P frame, is unfortunately not directly coded in the compression video. For example, Figure 4 shows that an irregular block in a reference frame. It contains the contents from four regular blocks, which are coded in DCT coefficients.

To compute the DCT coefficient vector $DCT(B')$ of the block B', an intuitive method is to perform inverse DCT transform to the block $DCT(B_{1,2,3,4})$, then crop the sub-blocks $B_{13,31,24,42}$ to reconstruct the block B' in the spatial domain, and finally compute the $DCT(B')$. However, this method consists of DCT transform and inverse DCT transform steps, which are computationally expensive.

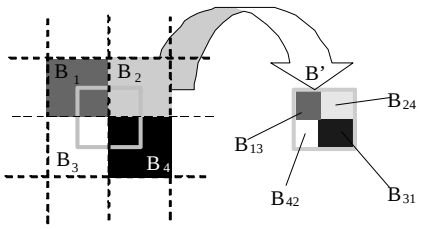


Figure 4: The appearance of an irregular block in an *I* frame consists of the information from all the regular blocks it covers.

To avoid performing DCT and inverse DCT transforms, Chang and Messerschmitt [3] proposed a method to compute $DCT(B')$ through transcoding. In their method, a sub-block (for example B_{13}) can be cropped from the block B_1 and shifted to the top-left corner in the spatial domain by two sparse 8×8 matrices:

$$B_{13} = V_1 B_1 H_1, \tag{2}$$

where,

$$V_1 = \begin{bmatrix} 0 & I_h \\ 0 & 0 \end{bmatrix}, \tag{3}$$

$$H_1 = \begin{bmatrix} 0 & 0 \\ I_w & 0 \end{bmatrix}. \tag{4}$$

h and w are overlapped height and width, respectively. The block B' can then be reconstructed as:

$$B' = \sum_{i=1}^{4} V_i B_i H_i. \tag{5}$$

According the orthogonality of the DCT, the $DCT(B')$ is therefore computed as:

$$DCT(B') = \sum_{i=1}^{4} DCT(V_i) DCT(B_i) DCT(H_i). \tag{6}$$

The DCT transformations of all the possible V_i and H_i can be pre-computed in this algorithm. To make this method even faster, Chaparro et. al. [4] proposed a fast algorithm exploiting the similarities among these matrices and saved 44.3% computation cost on in average in comparison with the intuitive inverse DCT plus DCT approach.

The obtained $DCT(B')$ is just the residual from the block and its reference. Because the DCT is a linear transform, the DCT appearance of a block in a P frame can be extracted using motion compensation once we obtain the appearance of its reference block. Let us denote the residue values in block B in a P frame as $\overline{B}$: $B = \overline{B} + B'$. The DCT appearance of the block B can be calculated as:

$$DCT(B) = DCT(\overline{B}) + DCT(B'). \tag{7}$$

Thus, all the DCT appearances of a block on its trajectory can be extracted without transforming the compressed video back to the spatial domain.

5.2 A mixture of Gaussians in spatial domain

Mixture of Gaussian was proposed to model the background and foreground of a video by Stauffer et al. [2]. The idea is that for each location x in image, we model the pixel values $f_{1\ldots t}(x)$ in t frames by a mixture of k Gaussian $N(\mu(x), \sigma(x), w(x))$, where $\mu(x)$, $\sigma(x)$ and $w(x)$ are three k-dimensional vectors, $\mu(x) = (\mu_i^x)_{i=1,\ldots,k}$, $\sigma(x) = (\sigma_i^x)_{i=1,\ldots,k}$ and $w(x) = (w_i^x)_{i=1,\ldots,k}$. The model is optimized theoretically by maximizing the log-likelihood of pixel values stream $f_t(x)$:

$$(\mu^*(x), \sigma^*(x)) = \arg \max_{\mu(x),\sigma(x)} \sum_{j=1}^{t} log \left(\sum_{i=1}^{k} w_i(x) p_{ij}(x) \right), \tag{8}$$

where

$$\begin{aligned} p_{ij}(x) &= p\left(f_j(x) | \mu_i^x, \sigma_i^x\right) \\ &= \frac{1}{\sqrt{2\pi}\sigma_i^x} e^{\left(-\frac{\left(f_j(x) - \mu_i^x\right)^2}{2{\sigma_i^x}^2} \right)}, \end{aligned} \tag{9}$$

and

$$\begin{aligned} w_i(x) &= p(\mu_i^x, \sigma_i^x) \\ &= \frac{\sum_{j=1}^{t} p\left(f_j(x) | \mu_i^x, \sigma_i^x\right)}{\sum_{h=1}^{k} \sum_{j=1}^{t} p\left(f_j(x) | \mu_h^x, \sigma_h^x\right)}. \end{aligned} \tag{10}$$

The more number of Gaussian components k used in the model, the model fits better the appearances in the training data. To train a model with a standard EM algorithm is very time consuming and requires predefined k in advance. In practice, the number of Gaussians k is determined using a Direchlet-like process, which can learn the parameter k in the model training. By following the trajectory of each pixel, where x is a variable along time t, this algorithm is can model a moving object as well as stationary backgrounds.

The pixel-level models treat pixels independently of each other, and can be extended to the block-level. Considering the total 64 locations in an 8×8 block B, we define the vector of pixel values in the block B at time t as $F_t = (f_t(x))_{x \in B}$. Here, we use a vector form of the block B instead of the matrix form to simplify the representation in the future discussion. We should emphasize that x are relative locations in block B, and the location of the block B varies along its trajectory.

Correspondingly, we define the mean vector of the ith gaussian component as $\mu_i = (\mu_i(x))_{x \in B}$, and define the variance vector as $\sigma_i = (\sigma_i(x))_{x \in B}$. The Gaussian mixtures are still represented as three vectors $\mu = (\mu_i)_{i=1,\ldots,k'}$, $\sigma = (\sigma_i)_{i=1,\ldots,k'}$ and $w = (w_i)_{i=1,\ldots,k'}$. Eq. 8 can then be extended to the block level as:

$$(\mu^*, \sigma^*) = \arg \max_{\mu,\sigma} \sum_{j=1}^{t} log \left(\sum_{i=1}^{k'} w_i P_{ij} \right), \tag{11}$$

where

$$P_{ij} = p(F_j|\mu_i,\sigma_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{\left(-\frac{1}{2}(F_j-\mu_i)^T(\sigma I\sigma^T)^{-1}(F_j-\mu_i)\right)}, \quad (12)$$

and

$$w_i = p(\mu_i,\sigma_i) = \frac{\sum_{j=1}^{t} p\left(F_j|\mu_i,\sigma_i\right)}{\sum_{h=1}^{k'}\sum_{j=1}^{t} p\left(F_j|\mu_h,\sigma_h\right)}. \quad (13)$$

I is a 64×64 identity matrix. Since there is a strong independent assumption among pixels in a block, the covariance matrix $\sigma I\sigma^T$ of the Gaussian mixtures is constrained to be diagonal.

If the independence assumption is satisfied and the trajectory is long enough, the block level model is a combination of 64 pixel-level models associated with the block, and the number of block-level Gaussians k' theoretically can be k^{64} for all the possible combinations. Given the 64 pixel-level models, this block-level model can be uniquely constructed by assigning each possible combinations a 64-dimensional Gaussian, and conversely, pixel-level models can be recovered from this constructed block-level model. In practice, most of the pixels are dependent and the trajectory is not long enough to estimate all Gaussians. Therefore, a block-level model directly estimated from a video is a good approximation, but not completely equivalent to the original pixel-level models defined in Eq. 8. We can use the k' Gaussians $(\mu_i(x),\sigma_i(x))$ and weights w_i provided by the block-level model for each pixel x to approximate a pixel-level model by using sampling-based estimation. When there are many similar Gaussians, we can easily reduce the number of Gaussians by merging similar Gaussians with associated weights together.

5.3 A mixture of Gaussians in DCT domain

The same modeling can be performed in the DCT domain. Let us transform a block B into the DCT domain to obtain $DCT(B)$. By keeping the same ordering method as we did in the spatial domain, the $DCT(B)$ is a 64-dimensional vector denoted as D. The DCT transform process can be represented as an orthogonal and unique DCT matrix K multiplied by the pixel-value vector F of the block B, which is formally defined as the following:

$$D = KF, \quad (14)$$

The inverse DCT transform is then defined as:

$$F = K^T D, \quad (15)$$

where the orthogonal matrix K has the property that:

$$K^{-1} = K^T. \quad (16)$$

The MoG model is then built on top of coefficient vector D through t frames using the same optimization approach as in the spatial domain. The 64-dimensional vector of the mean of the ith Gaussian component is denoted as Λ_i, while the variance vector is denoted as Ξ_i. The k'-dimensional model vectors are defined as $\Lambda = (\Lambda_i)_{i=1,\ldots,k'}$, $\Xi = (\Xi_i)_{i=1,\ldots,k'}$ and $w^d = (w_i^d)_{i=1,\ldots,k'}$. Then following Eq. 11, the MoG model in the DCT domain can be estimated as:

$$(\Lambda^*,\Xi^*) = \arg\max_{\Lambda,\Xi} \sum_{j=1}^{t} log\left(\sum_{i=1}^{k'} w_i^d P_{ij}^d\right) \quad (17)$$

where

$$P_{ij}^d = p(D_j|\Lambda_i^*,\Xi_i^*) = \frac{1}{(2\pi)^{32}|\Xi_i^*|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}\left(D_j-\Lambda_i^*\right)^T\Xi_i^{*-1}\left(D_j-\Lambda_i^*\right)\right)}, \quad (18)$$

and

$$w_i^d = p(\Lambda_i^*,\Xi_i^*) = \frac{\sum_{j=1}^{t} p\left(D_j|\Lambda_i^*,\Xi_i^*\right)}{\sum_{h=1}^{k'}\sum_{j=1}^{t} p\left(D_j|\Lambda_h^*,\Xi_h^*\right)}. \quad (19)$$

This block-level model has a close relationship with the block-level in spatial domain, which is discussed in the next subsection. We can also build coefficient-level models by considering each coefficient independently. Fortunately, since DCT is a orthogonal transform, the 64 dimensions of the coefficient are linearly independent. Therefore, the inverse covariance matrix Ξ_i^{-1} of each Gaussian component is diagonal and the coefficient-level MoG model can be estimated for each of the 64 dimensions individually without imposing an additional independence assumption.

5.4 Relationship between MoG models in DCT and spatial domains

The MoG model estimated in the DCT domain using the method proposed in subsection 5.3 has a close relationship with the MoG model of the same block estimated in the spatial domain using the method described in subsection 5.2. To exploit this relationship, let us first transform the mean and variance vectors of the MoG model in DCT domain into the spatial domain using Eq. 15:

$$\overline{\mu} = K^T\Lambda, \quad (20)$$

and

$$\Sigma = K^T\Xi K. \quad (21)$$

Substituting the above two equations and Eq. 14 into Eq. 17, we convert a the optimization function from the DCT domain back to the spatial domain, in which the target function is:

$$(\overline{\mu}^*,\Sigma^*) = \arg\max_{\overline{\mu},\Sigma} \sum_{j=1}^{t} log\left(\sum_{i=1}^{k'} \overline{w}_i\overline{P}_{ij}\right), \quad (22)$$

where

$$\overline{P}_{ij} = p(D_j|\Lambda_i^*,\Xi_i^*) = \frac{1}{(2\pi)^{32}|K\Sigma_i^*K^T|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}\left(KF_j-K\overline{\mu}_i^*\right)^TK\Sigma_i^{*-1}K^T\left(KF_j-K\overline{\mu}_i^*\right)\right)} = p\left(F_j|\overline{\mu}_i^*,\Sigma_i^*\right). \quad (23)$$

and

$$\overline{w}_i = p(\Lambda_i^*,\Xi_i^*) = p(\overline{\mu}_i^*,\Sigma_i^*). \quad (24)$$

Table 1: An algorithm to estimate Gaussian mixtures for one DCT coefficient

D^c	a sequence of the cth DCT coefficients in t frames;
M^c	an empty set for Gaussian mixtures.
1.	From the first frame $j = 0$ to t, perform step 2 to 5.
2.	For each Gaussian $(\lambda_i^c, \xi_i^c, w_j^c)$ in M^c, check if the current D_j^c matches the Gaussian by examining $\lvert D_j^c - \lambda_i^c \rvert < 2.5\xi_i^c$:
3.	If true, update (λ_i^c, ξ_i^c) as: $\lambda_i^c(new) = (1-\rho)\lambda_i^c + \rho D_j^c$ $\xi_i^{c2}(new) = (1-\rho)xi_i^{c2} + \rho(D_j^c - \lambda_i^c(new))^2$ $\rho = \frac{\beta}{\sqrt{2\pi}} e^{-\frac{(D_j^c - \lambda_i^c(new))^2}{2xi_i^{c2}}}$
4.	If false, update the weight as: $w_j^c(new) = (1-\beta)w_j^c$
5.	If none of the Gaussian in M matches the coefficient, add a new Gaussian $(D_j^c, \alpha^c, 1.0)$ into M^c. α^c is a constant.

The vectors ($\overline{\mu}^*$ and Σ^*) are the optimal values of the MoG model in the spatial domain of the block B. In comparing with the MoG model estimated using Eq. 8, the only difference is that the covariance matrix $\sigma_i^* I \sigma_i^{*T}$ in Eq. 8 is diagonal, which is a special case of the covariance matrix Σ_i^*. This is because the model in the spatial domain is constrained by imposing the independence assumption among pixels. In other words, if the independence assumption made by the MoG method in the spatial domain is satisfied, every covariance matrix Σ_i^* is then a diagonal matrix and Eq. 22 is equivalent to Eq. 8. This also means that the MoG model estimated in the DCT domain is equivalent to the MoG model estimated in the spatial domain if the optimal vector of the model is unique.

In practice, the independence assumption is rarely satisfied. The MoG method in the DCT domain provides a potentially better model because it models the correlations between pixels in a block.

6. THE ALGORITHM OF APPEARANCE MODEL EXTRACTION

Instead of modeling at the block-level directly, we can model each coefficient individually because the DCT coefficients are independent of each other. The formal presentation of this modeling is very similar to the pixel-level modeling, and therefore is not repeated here. Since the conventional optimization includes an EM process, which is expensive to apply on a large data set (e.g., video data), we use the Direchlet-like process proposed by Stauffer et. al. [2]. Let (λ_i^c, ξ_i^c) denotes the ith Gaussian of the cth DCT coefficient and w_i^c denotes its weight. The algorithm is briefly described in Table 1.

This algorithm has no iteration, and estimates a MoG model by only scanning the video data once. The obtained DCT coefficient models M^c have many advantages compared to the corresponding pixel-level models estimated in the spatial domain to perform video object clustering, indexing, and retrieval. First, the estimation of DCT models is faster than the estimation of pixel-level models because many DCT coefficients are zero after MPEG quantization. Second, DCT models are able to provide band-selective filtering during the estimation, which may enhance the video object appearances for special purposes. For example, low-pass filter is preferred to obtain smoothed appearances while high-pass filter leads to edge-enhanced appearances. To apply a band-selective filter, we can estimate the selected frequency or orientations with more Gaussians by adjusting the constants α^c of the corresponding coefficients. Finally, model matching in the DCT domain can also be band-selective. Different weights can be used in the matching process to emphasize selected frequencies or orientations in a video-object clustering or retrieval task.

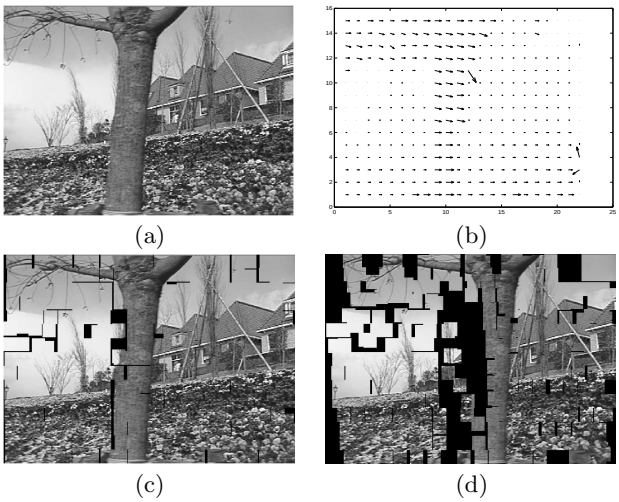
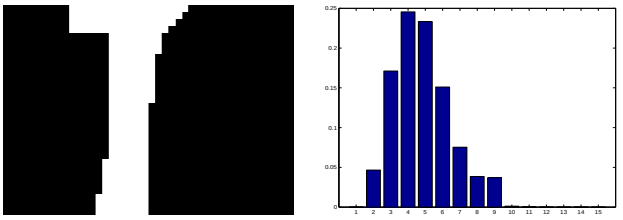


Figure 5: Recover trajectories of macro-blocks using MPEG motion vectors. (a) original frame; (b) MPEG motion vectors; (c, d) locations of macro-blocks on the trajectories.

Statistical object modeling approaches, such as the MoG method, provide object appearance models and the background models at the same time. Learned object models implicitly discriminate the background from the object because the pdfs of object pixels concentrate on several Gaussians due to the tracking, but the pdfs of background pixels are more flat. In other words, background pixels usually have low data likelihood values conditioned on the model of the video object. Therefore, the background pixels have less effect on the matching decision in video object matching.

DCT domain models have the same function, omitting the effects of background in video object matching. To illustrate this function, we first build a DCT domain model for a video object using the proposed approach. For example the video object can be a "tree" in a garden sequence compressed in MPEG-1 format. Figure 5 illustrates the recovering of macro-block trajectories using MPEG motion vectors. Large percentage macro-blocks are well-tracked by the motion vectors. Some are missing due to the noise of the motion vectors. Therefore, some recovered trajectories may also be noisy and have various lengths, which may introduce errors at the boundary of a video object.



(a) (b)

Figure 6: Grouped macro-blocks for the video object "tree", and the histogram of number of Gaussians in the learned DCT model.

By grouping the trajectories, the macro-blocks of each video object can be clustered together. Figure 6 (a) shows the macro-blocks of the "tree". All small and isolated macro-blocks are merged into their neighbors. A DCT model is then learned at the coefficient-level along the trajectories. Different coefficients may have various number of Gaussians. Figure 6 (b) shows the histogram of the number of Gaussians in the DCT model of the "tree". Most of the coefficients have about 3 to 6 Gaussians.

Figure 7: A reconstruction image from the learned DCT model using the means of the best match Gaussians.

The drawback of DCT models is that they are difficult to visualize. To visualize DCT coefficient models, the models need to be reconstructed into a block-level model and then transformed back to the spatial domain using Eq. 20 and 21. The construction of a block-level model from individual DCT coefficient models is a similar process to constructing a block-level model using pixel-level models in spatial domain, which is discussed in Section 5.2. Theoretically, a block-level model should consist of all possible combinations of Gaussians of individual coefficients. This may lead to a huge number of Gaussians in a block-level model, which is not practically computable. To address this problem, Monte Carlo (MC) methods [7] can be used to approximate a block-level model by sampling the best matches in the DCT domain. MC methods ensure the convergence of approximation of the block-level model given enough samples. In the worst case, the MC method may sample all frames in the trajectory and the obtained model is exactly same as estimated directly in block unit using Eq. 22.

To visualize the learned DCT model, we convert it back to the spatial domain using Monte Carlo methods. Figure 7 shows the reconstructed image from the model using the means of the best matched Gaussians to the original image displayed in the Figure 5. The function of discriminating the background can be shown by computing the data likelihood of the same image given the model. In Figure 8, we plot the log values of the data likelihoods of the video object "tree". The values are reverse-displayed to emphasize the background pixels, which means that the low values corresponding to the background are assigned high gray values in the figure. Most of the background pixels are correctly located in the boundary blocks of the video object "tree". The spatial model has a pixel-level accuracy, probabilistically. Since the spatial model is directly transformed from the DCT model we learned in compressed video, the DCT model also performs the function of reducing the effects of the background pixels in video object matching, and provides a sort of pixel-level or sub-block-level accuracy of modeling.

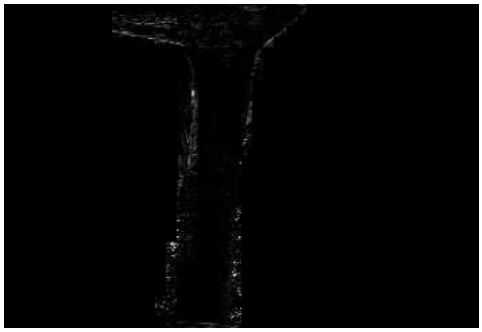


Figure 8: The data likelihood values (log values) of the "tree" (reversed grayscale for display).

7. CLUSTERING APPEARANCE MODELS

Using the above algorithm, we can extract appearance models of an object in a short period of time. Tracking failure of a moving object after a couple of seconds may break the appearances of the same object in a video into separated models. Objects entering or existence video scene also produce separated models. To obtain models corresponding to each object, we cluster the extracted MoG models using the spectral clustering algorithm.

In a spectral clustering algorithm, MoG models are treated as nodes in a graph $G = (S, E)$, where E is the affinity matrix encoding the similarity between any two nodes in the node set S. An affinity E_{ij} is defined as:

$$E_{ij} = \exp -\frac{d^2(s_i, s_j)}{2\sigma^2}, \tag{25}$$

where $d(s_i, s_j)$ is denoted as a distance measure between the node s_i and the node s_j. We use the Bhattacharyya coefficient to measure this distance. The parameter σ is usually a predefined constant.

We construct a matrix $L(E) = \left(D^{-\frac{1}{2}}(E)\ E\ D^{-\frac{1}{2}}(E)\right)$, where $D(E)$ is the degree matrix of E, and compute the eigenvectors of the matrix L. A matrix $Y = [y_1 y_2 ... y_h]$ is then constructed by stacking the h largest eigenvectors of the matrix L in columns. We then normalize each row of the matrix

Y to have unit length and cluster the normalized rows into K clusters using the K-means algorithm. Each row of the matrix Y corresponds to one node (a MoG model). Therefore if the K corresponds to the number of objects in the video, we can merge the models in the same cluster into one to model the appearances of the object in the video.

8. EXPERIMENTS

To evaluate the proposed algorithm, we perform the model extraction algorithm on a 24-hour surveillance video captured at a door entrance. The algorithm extracts 409 appearance models for 24 people and a food cart. Appearances consist of object coming in and out at the entrance. Both the persons and the food cart have various of appearances because of the motions in the video. Figure 9 shows some video snap-shots of extracted models.

Figure 9: Some snap-shots extracted from the video stream used in our experiments. Each row corresponds to a learned appearance model.

We evaluate the extracted model with manually labeled ground truth. The algorithm has a 100% recall, which means all the entering and exiting activities are extracted. 68 models are extracted because of tracking failures in the motion vectors. The individual models are clustered using the spectral clustering algorithm. The affinity matrix is depicted in Figure 10, in which models are ordered using the ground truth. The diagonal values are set to be always zero. We selected only 10 middle frequency coefficients in the DCT models to compute the distance in the infinity matrix. We can see that most models from the same person are similar to each other, which form the squares along the diagonal of the matrix.

Performance of spectral clustering is plotted in Figure 11. We applied the spectral clustering algorithm onto the infinity matrix with varying the number of eigenvectors h. The algorithm achieves a 78% percent accuracy according to the ground truth. The performance increases quickly when the value h is increased from one and reaches the peek at $h = 6$.

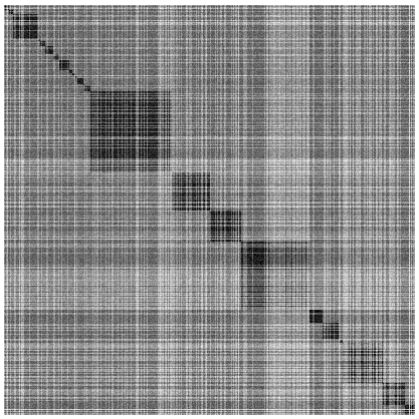


Figure 10: The affinity matrix of the extracted models.

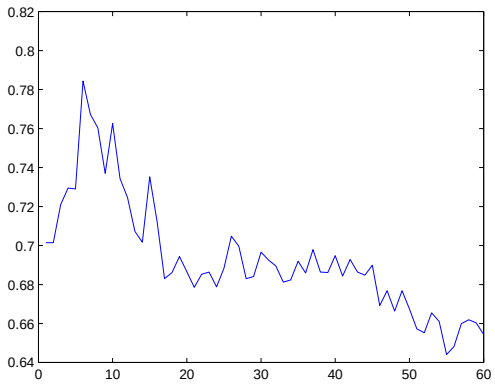


Figure 11: The performance of appearance clustering using the Spectral algorithm.

9. CONCLUSIONS

We have proposed an approach to extract and learn video object models directly from compressed video data. The proposed method effectively utilizes the motion and appearance information contained in motion vectors and DCT coefficients in the compressed video to extract video objects and model their appearances. The approach recovers the trajectories of macro-blocks using MPEG motion vectors and extracts video objects by clustering these macro-blocks. It further reconstructs the appearances of each macro-block along its trajectory in the DCT domain and learns an MoG model to represent the video object. Extracted models are then clustered using the spectral method to merge the appearances of the same object together. Experiments have shown that object appearance models can be automatically learned and extracted in a large quantity of compressed

video streams at a reasonable accuracy.

10. ACKNOWLEDGMENTS

This research is supported by the National Science Foundation (NSF) under Grant No. IIS-0205219.

11. ADDITIONAL AUTHORS

Additional author: Jie Yang (School of Computer Science, Carnegie Mellon University, email:jie.yang@cs.cmu.edu

12. REFERENCES

[1] R. V. Babu, K. R. Ramakrishnan, , and S. H. Srinivasan. Video object segmentation: A compressed domain approach. In *IEEE Trans. on circuits and systems for video technology, 14(4), APRIL 2004*, pages 462–474, 2004.

[2] W. G. C. Stauffer. Adaptive background mixture models for real-time tracking. In *Proc. of CVPR, Vol 2*, pages 246–252, 1999.

[3] S.-F. Chang and D. G. Messerschmitt. A new approach to decoding and compositing motion-compensated dct-based images. In *Proc. of ICASSP 1993*, 1993.

[4] L. F. Chaparro and C. C. Li. New algorithm for dct-based transcoding and compositing in multi-point video conferencing. In *Tech. report of Iniv. of Pittsburgh 2003*, 2003.

[5] M. Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. In *Proc. of the 6th ECCV*, pages 751–767, 2000.

[6] B. Han, D. Comaniciu, and L. Davis. Sequential kernel density approximation through mode propagation: applications to background modeling. In *Proc. ACCV 2004*, 2004.

[7] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *4th European Conf. Computer Vision*, volume 1, pages 343–356, 1996.

[8] C. Kim and J.-N. Hwang. An integrated scheme for object-based video abstraction. In *MULTIMEDIA '00: Proceedings of the eighth ACM international conference on Multimedia*, pages 303–311, New York, NY, USA, 2000. ACM Press.

[9] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Foreground object detection from videos containing complex background. In *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10, New York, NY, USA, 2003. ACM Press.

[10] B. Lo and S. Velastin. Automatic congestion detection system for underground platforms. In *Proc. of 2001 Int. Symp. on Intell. Multimedia, Video and Speech Processing*, pages 158–161, 2001.

[11] C.-W. Ngo, T.-C. Pong, and R. T. Chin. Exploiting image indexing techniques in dct domain. In *Pattern Recognition 34(9), 2001*, pages 1841–1851, 2001.

[12] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. In *IEEE Trans. on PAMI, 22(8)*, pages 831–843, 2000.

[13] J. Phillips, S. Sarkar, I. Robledo, P. Grother, and K. Bowyer. Baseline results for the challenge problem of human id using gait analysis. In *Proc. of Face and Gesture Recognition*, 2002.

[14] A. Z. R.S. Aygun. Stationary background generation in mpeg compressed video sequences. In *IEEE ICME*, pages 701–70, 2001.

[15] D. Z. W. Zeng, W. Gao. Automatic moving object extraction in mpeg video. In *Proc. of ISCAS'03*, pages 524–527, 2003.

[16] Z. Wu and C. Chen. A new foreground extraction scheme for video streams. In *MULTIMEDIA '01: Proceedings of the ninth ACM international conference on Multimedia*, pages 552–554, New York, NY, USA, 2001. ACM Press.

[17] Q. T. X. Yu, L. Duan. Robust moving video object segmentation in the mpeg compressed domain. In *Proc. ICIP vol.2*, pages 933–936, 2003.

[18] T. Yang, S. Z. Li, Q. Pan, and J. Li. Real-time and accurate segmentation of moving objects in dynamic scene. In *VSSN '04: Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks*, pages 136–143, New York, NY, USA, 2004. ACM Press.

[19] H. Zhang, S. Y. Tan, S. W. Smo-liar, and G. Yihong. Automatic parsing of newsvideo. In *Multimedia Systems vol. 2, 1995*, pages 256–266, 1995.