An Object-based Video Coding Framework for Video Sequences Obtained From Static Cameras

Asaad Hakeem, Khurram Shafique, and Mubarak Shah
School of Computer Science, University of Central Florida
Orlando, FL, 32816
{ahakeem,khurram,shah}@cs.ucf.edu

ABSTRACT

This paper presents a novel object-based video coding framework for videos obtained from a static camera. As opposed to most existing methods, the proposed method does not require explicit 2D or 3D models of objects and hence is general enough to cater for varying types of objects in the scene. The proposed system detects and tracks objects in the scene and learns the appearance model of each object online using incremental principal component analysis (IPCA). Each object is then coded using the coefficients of the most significant principal components of its learned appearance space. Due to smooth transitions between limited number of poses of an object, usually a limited number of significant principal components contribute to most of the variance in the object's appearance space and therefore only a small number of coefficients are required to code the object. The rigid component of the object's motion is coded in terms of its affine parameters. The framework is applied to compressing videos in surveillance and video phone domains. The proposed method is evaluated on videos containing a variety of scenarios such as multiple objects undergoing occlusion, splitting, merging, entering and exiting, as well as a changing background. Results on standard MPEG-7 videos are also presented. For all the videos, the proposed method displays higher Peak Signal to Noise Ratio (PSNR) compared to MPEG-2 and MPEG-4 methods, and provides comparable or better compression.

Categories and Subject Descriptors

E.4 [**Data**]: Coding and Information Theory—*Data Compaction and Compression*; H.4.3 [**Information Systems**]: Applications: Communications-Teleconferencing; I.4.2 [**Image Processing and Computer Vision**]: Compression (Coding)

General Terms

Algorithms, Performance, Experimentation

Keywords

Object-based Video Coding, Background Subtraction, Tracking, Incremental PCA, Contour-based Tracking, Affine and Projective transformation.

1. INTRODUCTION

Video compression is an area of research that has received considerable attention over the last few decades. Various compression techniques have been used to store, transmit, and manipulate video data efficiently. Video compression is commonly achieved by removing redundancies in the frequency, spatial and temporal domains. Standard coding techniques, such as predictive coding, transform coding, and vector quantization, treat the image/video as random signals and exploit their stochastic properties to achieve compression [2].

While most of the existing coding methods belong to the above defined category, there is a major research effort underway to produce video compression systems that use 2D or 3D models of objects and/or scenes to succinctly describe them (in the form of model parameters) for compression. Due to the inability to segment objects from a general scene and nonexistence of models for most real world objects and scenes, the current methods are not suitable for general videos and reduce the complexity of the problem by making some assumption about the problem domain. Once a problem domain is selected, one can use prior knowledge about the domain to construct models and to detect and segment objects from the scene. The domains widely considered in this regard include video telephony [2, 5], video surveillance [12, 19], and medical imaging [13, 17]. One of the features shared by the videos in these domains is that they are usually acquired by a static camera (which is the assumption made in this paper).

In this paper, we present a video coding frame work for videos that are acquired by a static camera. The proposed system extracts arbitrary shaped objects in the video frames using computer vision techniques and does not require explicit models of these objects. Instead, the system learns the models online for each object using incremental principal component analysis. The basic assumption here is that the appearance space (that includes non-rigid motion, different poses, and views of the object) of an object can be represented by a small number of principal components. Such an assumption about appearance space is commonly used in object recognition literature [18], and is also justified by the experimental results presented in Section 4. Also note that the transitions between different poses and views of an

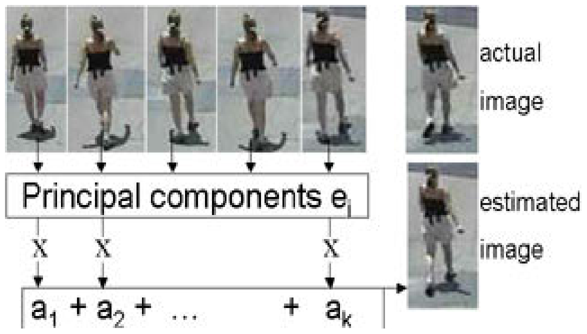


Figure 1: Sequence of a walking woman with different poses are shown. Each pose has different local transformations for each body part, therefore a single global motion compensation between frames is not sufficient to obtain a good object estimate $\hat{O}_i^t$. Thus the key poses are retained in the form of eigen vectors and a linear combination $(a_1, a_2, ..., a_k)$ of these eigen vectors $(e_1, ..., e_k)$ are used to estimate the new object pose $\hat{P}_i^t$. Further application of a global motion compensation to $\hat{P}_i^t$ will result in a good object estimate $\hat{O}_i^t$.

object in a video are usually smooth, and thus, the appearance of an object in the kth frame of a video sequence can be generated by using the subspace of the object's appearances that were observed till frame $k-1$. The object's appearance in the kth frame is coded in terms of the coefficients of the principal components of this subspace. The rigid component of the object's motion is estimated by affine motion parameters.

The framework is applied to both surveillance and video phone domains. The videos in these domains have different characteristics and thus require different methods for detecting and segmenting the objects from the background. The videos in surveillance domains usually consists of objects that are smaller in size (as compared to the background) and are fast moving. There are usually multiple objects in the scene and they have frequent occlusions with each other. Background subtraction techniques [15, 7] are commonly used to detect objects in these videos. Due to the presence of concurrent multiple objects in the scene, it is important to have consistent labelling of these objects through out the video sequence. That is, given two frames and a set of objects in each frame, in order to learn the appearance model of each object, we must know which object in the first set corresponds to which object in the second set. The problem becomes even more complex when these objects occlude each other in the scene. Thus, in the surveillance domain, a system must detect and track objects as well as handle occlusion, entries and exits in the scene.

In contrast to surveillance domain, sequences in video phone domain have large foreground to background ratio, where foreground largely consists of the face and torso of a person, and less occlusion. In addition, the motion in these videos is usually the local motion of the human face and global motion of the head. The objects typically do not enter or leave the scene. The standard methods of detecting objects in this domain include manual initialization [5], deformable contours [20], shape constraints [4], and facial feature extraction [24]. Both 2D generic models [14, 1, 11] and 3D facial models [6] have been used in this domain for video coding.

The significance of our method, compared to the above mentioned methods, is to extract arbitrary shaped objects from the video frame. The object extraction is based on background modelling or contour-based tracking methods, where object regions are compactly represented by motion and appearance models. Our method estimates a single set of motion compensation parameters for each object, and thus has a significantly lower number of motion parameters compared to other methods that encode arbitrary shaped objects. Another major advantage of object-based video coding is that it provides a semantic and structural description of the scene and may be use for the generation and analysis of videos with similar semantics. In addition, it allows the assignment of different priorities to different objects and the background in terms of encoding bit-rate based on their significance in the domain of concern.

The organization of the paper is as follows: In the next section, we present the object-based compression framework, where we discuss the encoding and decoding processes. The object detection and tracking algorithms for the surveillance and video phone domains are discussed in Sections 3.1 and 3.2 respectively. Finally, the results and discussion of our experiments are demonstrated in Section 4.

2. OBJECT-BASED VIDEO CODING FRAMEWORK

In this section, we present the proposed framework for object-based video coding for videos acquired from a static camera. We first assume that the objects in each frame F_i are segmented from the background and the temporal correspondences of these objects are known over the video sequence. Consider a video segment of n frames and let $O_1, O_2, \ldots, O_n$ be the sequence of observations of some object O in $F_1, F_2, \ldots, F_n$ respectively. In this paper, we treat each observation O_i as a vector in some (fixed) high dimensional space. We want a parameterized and succinct representation of each observation O_i, $2 \leq i \leq n$ of the object O using its previous observations $O_1, \ldots, O_{i-1}$. One solution is to apply motion compensation to the observation O_{i-1} to estimate O_i. This method works well for rigid objects such as cars, trucks, and boxes that undergo a single in-plane motion in consecutive frames. However, for objects having out of plane motion, such as a car taking a U-turn or non-rigid objects, and for non-rigid objects, such a humans, applying a single rigid motion compensation on O_{i-1} does not result in a good approximation of its appearance O_i in the frame F_i. Another possible solution is to have a prior 2D or 3D model of the object under question and use the model parameters to represent the object. However, obtaining prior models of all possible objects in a general scenario is a difficult task. Even in the presence of prior models, parameter estimation for these models (or model fitting) is by no means a trivial problem.

To avoid using prior models for the objects in question, we learn the appearance of each object online as shown in Figure 1. The learned model is weak when the object is first observed but becomes stronger and stronger as more observations of the object are available. We assume that the transitions between different poses and views of an object are smooth, and hence, the appearance O_{i+1} of the object O in frame F_{i+1} can be represented as a linear combination of its previous i appearances, i.e., $O_{i+1} = \sum_{j=1}^{i} a_j O_j$. Thus, the appearance O_{i+1} of an object can be represented by the

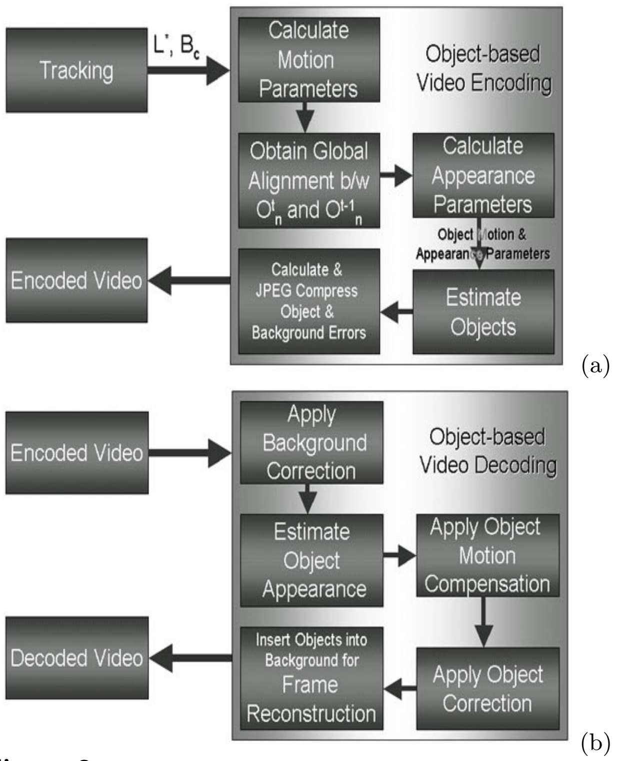

Tracking
L*, B_c
Calculate Motion Parameters
Object-based Video Encoding
Obtain Global Alignment b/w O^t_n and O^{t-1}_n
Calculate Appearance Parameters
Object Motion & Appearance Parameters
Encoded Video
Calculate & JPEG Compress Object & Background Errors
Estimate Objects
(a)
Encoded Video
Apply Background Correction
Object-based Video Decoding
Estimate Object Appearance
Apply Object Motion Compensation
Decoded Video
Insert Objects into Background for Frame Reconstruction
Apply Object Correction
(b)

Figure 2: Flow chart of the object-based video compression framework. (a) Encoding process is shown where each frame is encoded in a similar fashion to obtain the encoded video. The output of the tracking component is the labels and silhouettes of objects $L*$ and the background model B_C. The silhouettes of the same object in two consecutive frames are aligned using the affine motion parameters and the appearance is encoded in the form of the coefficients of principal components. The encoded video for each frame consists of the affine motion parameters and coefficients of principal components of each object along with the JPEG compressed object and background errors. (b) Decoding process is shown where each frame is reconstructed in a similar manner to obtain the decoded video. The initialization (not shown) creates a background region and initializes the object models and then inserts the objects in the background to reconstruct the first video frame. The rest is simply the inverse of the encoding process.

coefficient vector $a_1, \dots, a_i$. However, the size of this vector increases over time and because of the linear dependence among the appearance vectors $O_1, \dots, O_i$, is not the best succinct representation of O_{i+1}. Therefore, we compute the basis of the appearance subspace of object O from the observations $O_1, \dots, O_i$ using Principal Component Analysis (PCA) [10, 16], and the appearance vector O_{i+1} (of d dimensions) is then represented by the coefficient vector z_{i+1} (of k dimensions, $k \ll d$) of the k most significant principal components of this space as follows:

$$z_{i+1} = WO_{i+1}$$

The appearance model is represented by the projection matrix W, of dimensions $k \times d$ consisting of the k most significant eigen vectors, $v_1(i), v_2(i), \dots, v_k(i)$;

The principal components $v_1(i), v_2(i), \dots, v_d(i)$ and the corresponding eigen values $\lambda_1(i), \lambda_2(i), \dots, \lambda_d(i)$ are the solution of the following linear system:

$$(S - \lambda_j(i)I)v_j(i) = 0, \quad j = 1, ..., d \tag{1}$$

where S is the sample covariance matrix, and is given by:

$$S = \frac{1}{i}\sum_{j=1}^{i}(O_j - \mu_i)(O_j - \mu_i)^T, \tag{2}$$

$\mu_i = \frac{1}{i}\sum_{j=1}^{i} O_j$ is the sample mean.

To compute the principal components efficiently for each object at each frame, we update the existing principal components by using an incremental method (IPCA) based on [21]. At each time frame F_{i+1}, the IPCA method iteratively computes the new principal components $v_j(i+1)$ (for $j = 1, 2, ...d$), as follows:

```
1. $u_1(i+1) = O_{i+1}$.
2. For $j = 1, 2, ..., min(d, i+1)$ do,
   (a) If $j = i+1$,
   initialize the $j$th eigenvector as $v_j(i+1) = u_j(i+1)$;
   (b) Otherwise,
```

$$v_j(i+1) = \frac{i-l}{t}v_j(i) + \frac{1+l}{i+1}u_j(i+1)u_j^T(i+1)\frac{v_j(i)}{||v_j(i)||} \tag{3}$$

$$u_{j+1}(i+1) = u_j(i+1) - u_j^T(i+1)\frac{v_j(i+1)}{||v_j(i+1)||}\frac{v_j(i+1)}{||v_j(i+1)||} \tag{4}$$

where l is the amnesiac parameter giving larger weights to newer samples, and $||v||$ is the eigenvalue of v. Intuitively, eigenvectors $v_j(i)$ are pulled towards the data $u_j(i+1)$, for the current eigenvector estimate $v_j(i+1)$ in (3). Since the eigenvectors have to be orthogonal, therefore (4) shifts the data $u_{j+1}(i+1)$ normal to the estimated eigenvector $v_j(i+1)$. This data $u_{j+1}(i+1)$ is used for the estimating the $(j+1)$th eigenvector $v_{j+1}(i+1)$. Note that the IPCA method converges to the true eigenvectors in fewer computations than PCA (see proof in [22]).

Once the object appearance is coded, we use the affine motion model to compute its position along with other motion transformations such as rotation, scaling and shearing from its previous position. Under the affine transformation, the position of each point in the new frame is defined by the six parameters as follows:

$$x_i = a_{11}x_{i-1} + a_{12}y_{i-1} + T_x \tag{5}$$

$$y_i = a_{21}x_{i-1} + a_{22}y_{i-1} + T_y \tag{6}$$

where $a_{11}, ..., a_{22}$ are the rotation, shearing and scaling parameters, and T_x, T_y are the translation parameters. These parameters are estimated using least square method in a hierarchical coarse to fine procedure as presented in [3].

For each frame, the transmitter encodes the estimated objects $\hat{O}_1^t, ..., \hat{O}_n^t$ (in the form of coefficients of principal components and affine parameters) and the reconstruction error of background and objects as the base difference image B_d^* and object difference image O_d^* respectively. The background error is not encoded in case the sum of squared differences ($SSD = \frac{1}{width*height*3}\sum_{i=1}^{width}\sum_{j=1}^{height}\sum_{k=1}^{3}(B_{ijk} - Image_{ijk})^2$) is below a certain threshold T_b. The encoded video stream, at each time instance, contains the compressed base difference image B_d^*, object difference image O_d^*, labelled object silhouettes L^*, and the motion and appearance parameters of each object. Note that L^* contains only the new object silhouettes, as motion compensation can be applied to the object silhouettes in previous frame, to obtain object regions in the current frame. Also, the appearance parameters only contain the eigen coefficients of each object,

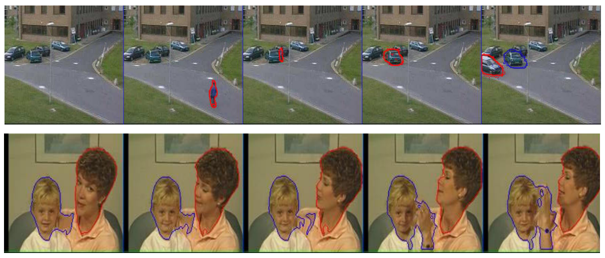

(a)
(b)

Figure 3: **Sequence of frames from videos in the surveillance and video phone domain where the red and blue contours are the tracked objects. (a) Surveillance Video: Objects enter and exit the field of view of the camera, therefore a background model can be learnt and utilized for tracking. (b) Video Phone Scene: Objects persist inside the field of view of the camera and thus parts of the background remain occluded during the complete sequence.**

rather than the eigen vectors, since the principal components can be estimated (and maintained) using the reconstructed objects during decoding at the receiver end.

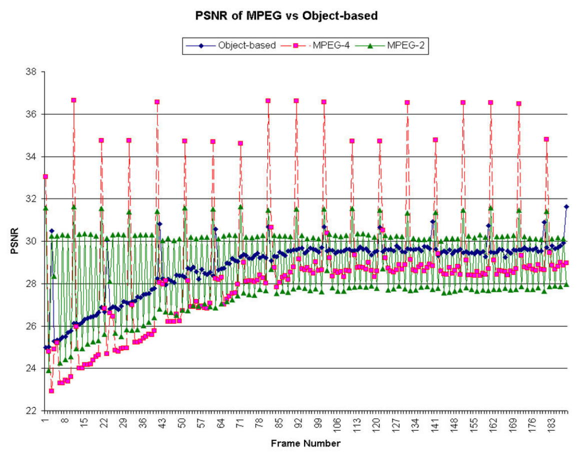
Decoding of the video is achieved by decompressing the first frame and creating a background B_c. Also, each new object's appearance model is initialized using PCA. For the subsequent frames, we obtain the estimated objects $\hat{O}_1^t, ..., \hat{O}_n^t$ by their appearance and motion parameters, and remove the object estimation error by adding the object difference image O_d^* to obtain $\dot{O}_1^t, ..., \dot{O}_n^t$. Furthermore, the error in the background region B_c is removed by adding the base difference image B_d^*. The video frame is reconstructed by inserting the objects $\dot{O}_1^t, ..., \dot{O}_n^t$ into the corrected background. Finally, each object's appearance model and silhouette are updated by the IPCA and motion compensation (using motion parameters) respectively, and the above process is repeated for all the frames. The object-based compression framework is summarized in Figure 2.

3. DETECTION AND TRACKING OF OBJECTS

Thus far in this paper, we have assumed that the segmentation of objects in each frame and their temporal correspondence throughout the video sequence is readily available. In the next two sub-sections, we will present solutions to these problems for both video surveillance and video phone domains. Video surveillance systems track objects in indoor and outdoor environments, where objects enter and exit the field of view of the camera (see Figure 3a). Since the objects do not persist in the field of view, one can use background modelling techniques to detect and track the objects. In contrast, video phone scenes have objects persisting in the field of view of the camera and the complete background region is not visible (see Figure 3b) during the entire sequence. Therefore the background subtraction techniques are not suitable for tracking in these videos. Instead, we utilize a contour-based tracking method that models a contour using an energy functional, and tracks the object from frame to frame by minimizing the contour energy. We now describe both of these methods in detail.

3.1 Tracking in Surveillance Videos

In this section, we address some issues related to finding the correspondence between objects at each time instant for surveillance videos. There are several problems during detection and tracking, inherent in the surveillance domain, that include rapidly changing lighting conditions (due to cloud cover), shadows, different types of occlusions (such



(a)

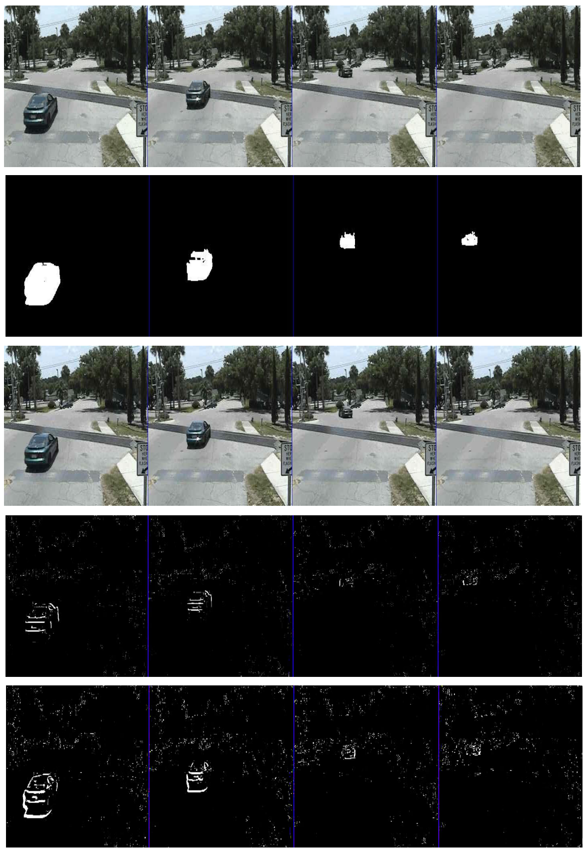

(b)
(c)
(d)
(e)
(f)

Figure 4: **Car moving on a road with a perspective view and taking a turn (Video1) (a) PSNR comparison of Object-based compression with MPEG-2 and MPEG-4 (b) Sequence of frames from original video (c) Object silhouettes using background subtraction (d) Reconstructed images using the object-based compression method (e) Difference images between original and object-based compressed (f) Difference images between original and MPEG4 compressed. Note that the object-based method has less error at the object boundaries compared to the MPEG-4 method. This is because MPEG-4 approximates the objects using blocks, while our method uses actual object silhouettes.**

as object to object, and object to background region), and entry/exit of objects.

We use background subtraction for object detection, that is based on an extension of Stauffer et al. [15] by Javed et al. [7]. In their extension, they propose multiple levels of processing during background subtraction. The first level is the pixel level processing that separately uses color-based and gradient-based distributions, to find pixels belonging to the foreground or the background. The second level is the region level processing that integrates the gradient and color information. A connected component algorithm is applied to group all foreground pixels into regions. Any foreground region that corresponds to an object will have high values of gradient-based background subtraction at its boundaries. This will not be true for falsely detected regions and they are added to the background regions. This method handles the common problems in most background subtraction algorithms such as quick illumination changes due to adverse weather conditions, relocation of the background objects (e.g. repositioning of a chair), and initializing the background model with moving objects.

Once the object silhouettes are obtained, we use a tracking method proposed by Javed et al. [8]. In this method, each object is modelled by the color and spatial probably density functions *pdfs*. The color *pdf* is approximated by a normalized histogram, while the spatial *pdf* is represented by a Gaussian distribution, with the variance equal to the sample variance of the object silhouette. For a new video frame, the color and spatial probabilities (belonging to each labelled object) are calculated for each pixel in the foreground region. Each of those pixels will vote for that object label, for which the product of the spatial and color probabilities is the highest. Thus each foreground region is assigned that object label, which received the highest number of votes. Object occlusion is also handled during tracking, the details of which can be found in [8], and [9].

3.2 Tracking in Video Phone Scenes

Video phone scenes usually have objects present at the start of the video and they persist in field of view of the camera throughout the entire sequence. Since we cannot model the background in such videos, we cannot use background subtraction techniques for tracking. Thus we use the contour-based object tracking method as proposed by Yilmaz et al. [23]. Object tracking is treated as a two-class discriminant analysis of pixels into regions belonging to object R_{obj} and background R_{bck}, which depends upon object features, energy functional, and the energy minimization technique.

The object features under consideration are appearance and shape, and the appearance features are composed of color and texture. Pixels are clustered as object or background by the *independent opinion polling* strategy. The shape of the object is learnt over time (t), based on the object contour Γ and is given by $P_{shape} = P(\varphi(R_i^t)|\Gamma^t)$, where R_i are the object regions and φ is the partitioning operator that divides the image into object and background regions.

Tracking the energy functional is formulated as a maximum a posteriori estimate (MAP) of the object contour in the t^{th} frame, and can be calculated by maximizing the probability P_Γ over the subsets $\Gamma \subset \Omega$, where Ω is the space of all object contours. The MAP estimate can be written in

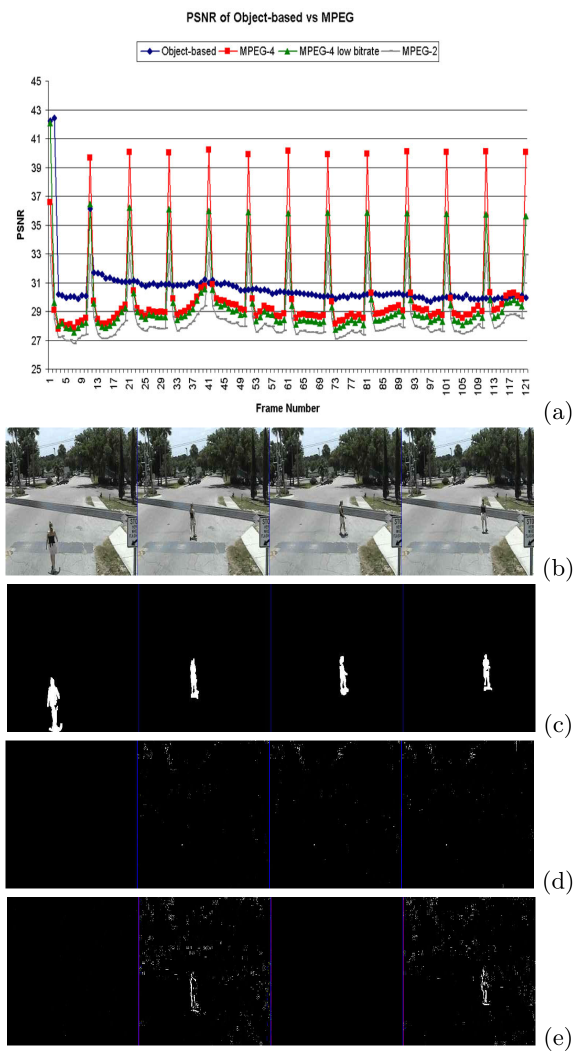


Figure 5: Person walking on a road with a perspective view and taking a turn (Video7) (a) PSNR comparison of Object-based compression with MPEG-2 and MPEG-4 (b) Sequence of frames from original video (c) Object silhouettes using background subtraction (d) Difference images between original and object-based compressed (e) Difference images between original and MPEG4 compressed. Note that this video sequence was used to compare the compression factors of MPEG-4 vs. the object-based method.

terms of sub-regions (obtained from the Bayes' rule) as:

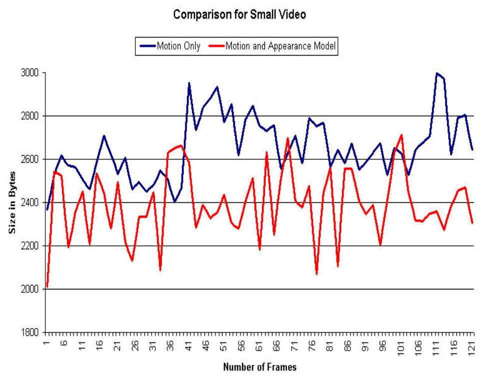
$$\Gamma^t = arg\max_{\Gamma\subset\Omega}\prod_{x_1}[\prod_{x_2} P_{R^{\Gamma}_{obj}}(I^t(x_2))\prod_{x_3} P_{R^{\Gamma}_{bck}}(I^t(x_3))P^t_{shape}]$$

where $x_1 \subset \Gamma$, $x_2 \in R_{obj}$ and $x_3 \in R_{bck}$. Converting this to energy functional by considering the negative log-likelihood of the probabilities, we obtain:

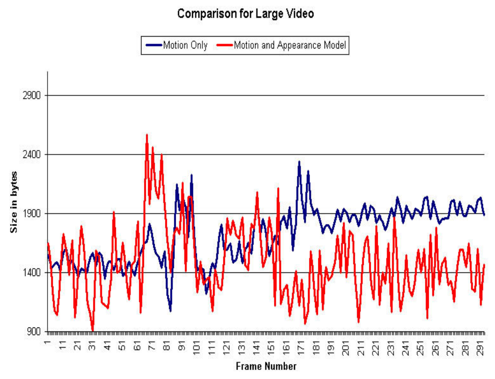
$$E = \int_{x_1}[\int\int_{x_2}\Psi_{obj}(x_2)dx_2 + \int\int_{x_3}\Psi_{bck}(x_3)dx_3 - logP^t_{shape}]dx_1 \qquad (7)$$

where $x_1 \subset \Gamma$, $x_2 \in R_{obj}(x_1)$, $x_3 \in R_{bck}(x_1)$, and $\Psi_\alpha(x) = -logP_{R_\alpha}(I^t(x)) : \alpha \in \{obj, bck\}$.

Object tracking is achieved by evolving the contour in each frame such that the energy functional given in (7) is minimized. The minimization is achieved by finding the



(a)



(b)

Figure 6: The difference image comparison for motion compensation only vs object motion and appearance model encoded videos. (a) For a smaller video (video7) the difference image sizes are almost equal. (b) For a larger video (caviar3) the difference image sizes decreases for the object appearance model encoded video as the new pose can be reconstructed with existing model poses.

derivative of the energy functional, which is associated by the Euler Lagrange equations of the functional given in (7) by:

$$\frac{\delta E}{\delta x} = -[\int\int_{-m}^{m}(-\Psi_{obj}(x) - \Psi_{bck}(x))dx - S]\frac{\delta y}{\delta s} \quad (8)$$

$$\frac{\delta E}{\delta y} = [\int\int_{-m}^{m}(-\Psi_{obj}(x) - \Psi_{bck}(x))dx - S]\frac{\delta x}{\delta s} \quad (9)$$

where s is the contour parameter and S is the contour shape. Once all the objects are tracked using contour evolution, the background is obtained using $B_i^t = I_i^t - (O_1^t + O_2^t + ... + O_n^t)$ in the current frame I_i^t of n objects.

4. RESULTS AND DISCUSSION

We perform an online encoding of videos from a static camera using the proposed object-based compression framework. The encoding scheme is demonstrated on nine videos, three are from traffic surveillance dataset, one is from standard Performance Evaluation of Tracking and Surveillance (PETS) dataset [26], 3 are from the standard CAVIAR dataset [25], while 2 videos are from the standard MPEG-7 dataset. Vid-eo1 is of a car that crosses the railway tracks and then turns left as seen in Figure 4. Video5 is of one car emerging from behind the trees, while the other is occluded by the tree. Video7 is of a woman walking on the road then she turns and walk away from the camera as shown in Figure 5. PETS video contains both people and cars that are moving

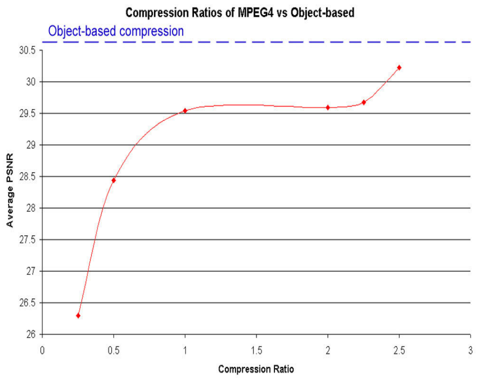


Figure 7: The average PSNR for the MPEG-4 method with different compression ratios compared to object-based method for video7. The horizontal axis portrays the compression factor of MPEG-4 vs object-based method e.g. 0.25 depicts MPEG-4 is 4 times smaller than the object-based video size.

with a changing background as detailed in Figure 9. Caviar 1, 2 and 3 are from the standard surveillance videos from the shopping mall as shown in Figure 8. Akiyo is a video of the Japanese newscaster as in Figure 10, while the mother daughter sequence is shown in Figure 11. The objects in the first seven videos are obtained using background modelling techniques, while we use a contour-based object tracking for Akiyo and mother daughter videos. This is because the objects occlude part of the background throughout the video and background modelling techniques cannot be used for tracking.

The decompressed frames are tested for the Peak Signal to Noise Ratio (PSNR), and is computed using the Mean Squared Error (MSE) that is calculated by:

$$MSE = \frac{1}{width * height}\sum_{i=1}^{width}\sum_{j=1}^{height}(A_{ij} - B_{ij})^2$$

where A is the original image, B is the reconstructed image, and *width* and *height* are image dimensions. Note that this difference is performed over three color channels (RGB) and averaged for the channels to obtain a single value. The PSNR is defined in terms of MSE as follows:

$$PSNR = 10 \times log_{10}\frac{(2^n - 1)^2}{MSE} \quad (10)$$

where the numerator term in the fraction is the peak signal value and MSE is the noise (error) in the signal. Also, the mean PSNR for all the frames in the video is given by $\mu PSNR = \frac{1}{frames}\sum_{i=1}^{frames} PSNR_i$.

There are twofold discussions on the results in the next sections. First, the advantages of using an object appearance model for object-based video compression is shown. Compression ratios for compressing with and without an object appearance model are compared. Second, the compression of our method is compared to the leading commercially available MPEG encoders. The next section details the necessity of object appearance model for attaining better compression.

4.1 Requirement of an Object Appearance Model

As discussed in Section 2, an object appearance model is necessary to model the local motion in non-rigid moving

Table 1: Difference image size comparison with and without object appearance modelling

	Frames	With Appearance Model		Without Appearance Model	
		Size(KB)	μPSNR	Size(KB)	μPSNR
video7	121	289.4	30.6	290.4	30.2
caviar1	212	287	31.5	294	31.1
caviar2	331	613.2	32.1	672.1	31.4
caviar3	293	443	31.9	508	30.5

Table 2: PSNR comparison of Object-based compression with MPEG-2 and MPEG-4

	Frames	Original	MPEG-2		MPEG-4		Object-based	
		Size(MB)	Size(MB)	μPSNR	Size(MB)	μPSNR	Size(MB)	μPSNR
video1	188	46.4	7.83	28.2	1.85	28.8	0.64	28.4
video5	251	62.4	7.90	29.7	2.48	30.2	0.61	31.7
video7	121	29.9	5.21	28.4	0.97	30.2	0.45	30.6
pets	555	308.0	33.9	33.5	6.48	34.5	3.36	36.8
caviar1	212	30.5	6.92	29.9	0.92	31.3	0.43	31.5
caviar2	331	81.3	10.2	30.2	1.91	31.9	0.78	32.1
caviar3	293	68.0	8.54	30.1	1.43	30.1	0.64	31.9
akiyo	300	21.7	2.75	32.9	0.17	32.5	0.19	32.4
mother	301	21.8	3.06	32.5	0.23	32.4	0.21	31.4

objects. We conducted experiments for videos containing non-rigid objects and encoded it without modelling its appearance (using motion compensation only). We then compared the compression results with the object appearance model being encoded. This is shown in Table 1 where compression is equally good for smaller videos while it is better for longer videos encoded with object appearance model. The reason behind it is that, for smaller videos, the object appearance model initially does not have large sample poses to reconstruct the new object pose from the existing poses. Hence, the error (difference image) is initially larger compared to motion compensated encoding, which decreases as more poses are available. This is also shown in Figure 6, where the size of the difference image from the two encoding methods is compared.

4.2 Comparison with the MPEG Encoders

We also show the results of the video frames compressed using the MPEG-2 and MPEG-4 standards. The compression tool used for encoding the MPEG videos is Adobe Premiere, where MPEG-2 encoding uses 'Microsoft Windows Media Video 9' codec at 'One-Pass Quality VBR and Main Profile' setting and MPEG-4 encoding utilizes 'XVid MPEG-4' codec at '1 Pass - Quality and Main Profile' setting. The summary of results for the different compression methods is shown in Table 2. The proposed object-based method's compressed video contains the difference (error) image, k largest eigen coefficients, labelled object silhouettes, and the affine motion parameters of each object.

The average PSNR value is kept the same for object-based and MPEG-4 compression, however it is not possible to control the PSNR quality for MPEG-2 using the current codec. The compression ratios of the three methods are compared, and our method outperforms the MPEG-4 compression by a factor ranging from 1.6 to 5 (see Table 2), and MPEG-2 by a factor ranging from 9 to 17. The PSNR values are better for object-based method, except at the I-frames of MPEG-2 and MPEG-4 methods, where the actual images are used rather than approximations, hence our method has lower compression for those frames.

We have also compared the compression factors and μPSNR of MPEG-4 vs the fixed object-based compression for video7 as shown in Figure 7. The μPSNR for object-based method is 30.6db and is kept fixed, while the MPEG-4's μPSNR topped at 30.2db at maximum quality. The video was compressed with different bit-rates for the MPEG-4 method to portray the quality at different compression factors compared to the object-based compression method. The horizontal axis portrays the compression ratio of MPEG-4 vs object-based method (e.g. 0.25 depicts MPEG4 is 4 times smaller than the object-based video size). As shown in the figure, our method has better quality even at the maximum bit-rate of MPEG-4 video, at 2.5 times the size, compared to the object-based compressed video.

The performance of our object-based video compression method depends upon the size of the object and change in the background. The difference image at the boundary of the object will be small in size for our method compared to the MPEG-4 method that approximates the object boundary, thus the compression depends upon the size of the object. If the object size is small (about 30x50 pixels) then the compression will be almost the same as MPEG-4, since the size of the object compared to the size of the frame is relatively small. On the other hand, if the object size is large (greater than 50x80 pixels) then our method will have far better compression compared to MPEG-4 as shown in Table 3, where Compression Factor = (Size of MPEG-4 compressed Video)/(Size of Object-based compressed Video). We also compare the compression of the background difference image for our object-based vs MPEG-4 encoding. The background image sequence is generated by the background model for video7 and caviar1, and the results are illustrated in Table 4. Our method outperforms the MPEG-4 method since our method only updates the background when it has a significant change, where as MPEG-4 updates the background at each frame. Though our method has

Table 3: Comparison of Object-based with MPEG-4 compression at different object sizes

	Frames	Object Size	Object-based Size(KB)	MPEG-4 Size(KB)	Compression Factor
pets	555	Small (30x50)	3360	6480	1.9
video7	121	Small (30x50)	450	970	2.1
caviar3	293	Medium (45x70)	645	1434	2.2
caviar2	331	Large (50x80)	779	1910	2.4
video1	188	Large (75x65)	640	1850	2.9

Table 4: Comparison of Object-based with MPEG-4 compression for background difference images

	Frames	Object-based		MPEG-4			
		Size(KB)	μPSNR	Size(KB)	μPSNR	Size(KB)	μPSNR
video7	121	15.4	30.2	74.4	30.7	19.6	18.3
caviar1	293	23.9	30.8	178.0	31.1	48.0	20.4

worse μPSNR than MPEG-4 (at higher bit-rate) as can be seen in Table 4, visually the changes in the background are insignificant (e.g. small tree with moving leaves), and hence can be ignored for surveillance purposes.

5. CONCLUSION

This paper presented a novel object-based video coding framework for videos obtained from a static camera in surveillance and video telephone domains. As opposed to most existing methods, the proposed method does not require explicit 2D or 3D models of objects and hence is general enough to cater for varying types of objects in the scene. The proposed system detects and tracks objects in the scene and learns the appearance model of each object online using incremental principal component analysis (IPCA). Each object is then coded using the coefficients of the most significant principal components of its learned appearance space. The rigid component of the object's motion is coded in terms of its affine parameters. The proposed method is evaluated on videos containing a variety of scenarios such as multiple objects undergoing occlusion, splitting, merging, entering and exiting, as well as a changing background. Compression results are demonstrated using standard MPEG-7 as well as other videos and the proposed method displays higher Peak Signal to Noise Ratio (PSNR) compared to MPEG-2 and MPEG-4 methods and also provides comparable or better compression.

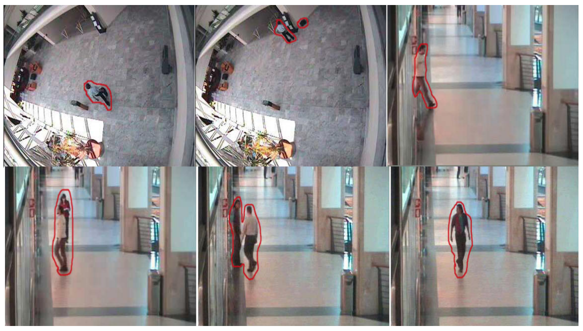


Figure 8: Different video frames from the CAVIAR dataset. The first two frames are from caviar1 where person enters the scene, puts down his bag and browses around. The next two frames are from caviar2 where a person enters into a corridor through a door and turns around goes back in the door. The final two frames are from caviar3 where two people cross each other in a corridor.

6. REFERENCES

[1] Y. Altunbasak and A. M. Tekalp, "Occlusion-adaptive content-based 2-D mesh design and tracking for object-based coding," In *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp.1270-1280, 1997.

[2] K. Aizawa and T. Huang, "Model Based Image Coding: Advanced Video Coding Techniques for Very Low Bit-Rate Applications," In *Proceedings of the IEEE*, vol. 83, no. 2, pp.259-271, 1995.

[3] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. "Hierarchical model-based motion estimation". In *European Conference on Computer Vision*, pp.237-252, 1992.

[4] Buck, "Segmentation of moving head-and-shoulder shapes," In *Picture Coding Symposium,* Boston, 1990.

[5] C. S. Choi and T. Tekebe, "Analysis and synthesis of facial image sequences in model-based image coding," In *IEEE Transactions on Video Technology*, pp.257-275, 1994.

[6] D. DeCarlo, D. Metaxas, and M.Stone, "An Anthropometric Face Model using Variational Techniques," In *Proc. SIGGRAPH*, pp.67-74, 1998.

[7] O. Javed, K. Shafique, and M. Shah. "A hierarchical approach to robust background subtraction using color and gradient Information". In *Workshop on Motion and Video Computing*, pp.22-27, 2002.

[8] O. Javed, Z. Rasheed, O. Alatas, and M. Shah, "KNIGHTM:A Real Time Surveillance System for Multiple Overlapping and Non-Overlapping Cameras". In *The Fourth International Conference on Multimedia and Expo*, Baltimore, Maryland, 2003.

[9] O. Javed, and M. Shah. "Tracking and Object Classification for Automated Surveillance". In *European Conference on Computer Vision*, pp.343-357, 2002.

[10] I. T. Jolliffe. "Principal component analysis". New York: Springer-Verlag, 1986.

[11] R. Koch, "Dynamic 3D scene analysis through synthesis feedback control", In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15(6), June, 1993.

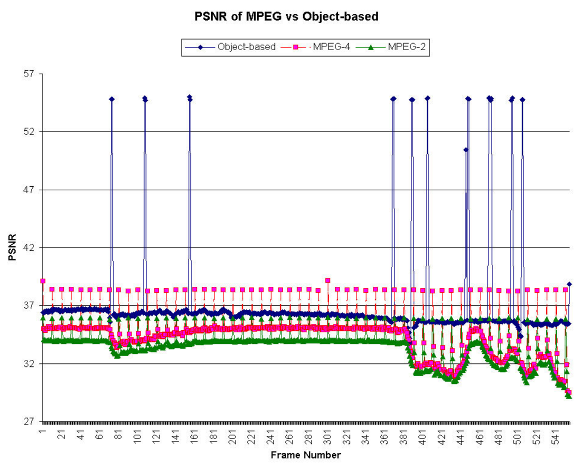
[12] I. Martins and L. Corte-Real, "A Video-coder using 3D model-based background for video surveillance applications,". In *IEEE Internation Conference on Image Processing* , pp.919-923, 1998.

[13] G. Menegaz and J. Thiran, "Lossy to lossless object-based coding of 3D MRI data". In *IEEE Transactions on Image Processing* , Vol.11(9), pp.1053-1061, 2002.

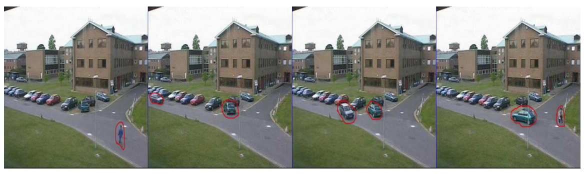
[14] Y. Nakaya, K. Aizawa, and H. Harashima, "Texture updating methods in model-based coding of facial images,". In *Picture Coding Symposium* , Boston, 1990.

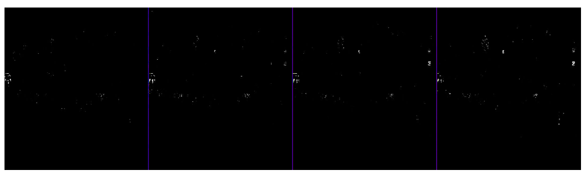
[15] C. Staffer, and W. E. L. Grimson. "Learning patterns of actvity using real-time tracking." In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.22(8), pp.747-757, 2000.

[16] G. Strang. "Linear algebra and its applications". New York: Academic, 1980.



(a)


(b)


(c)

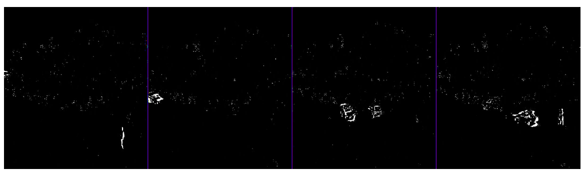
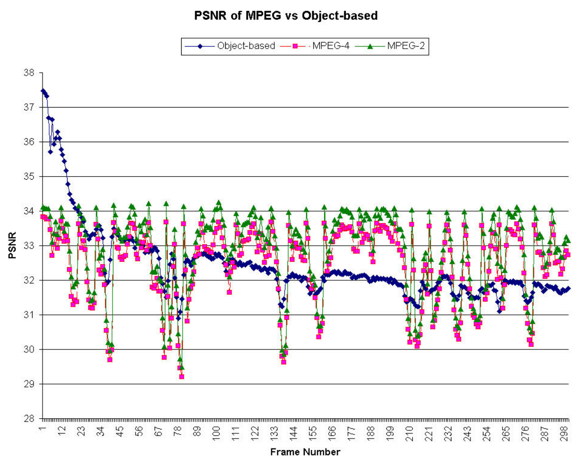
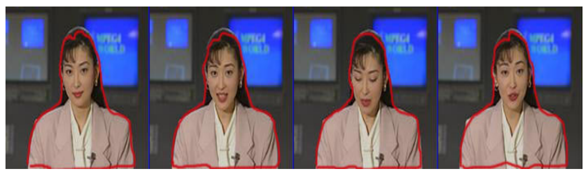

(d)

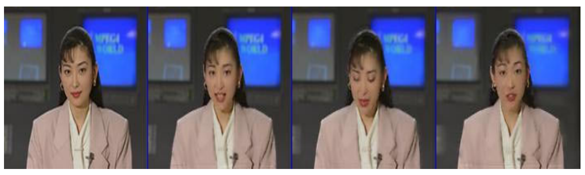
Figure 9: Standard Performance Evaluation for Tracking and Surveillance (PETS) video containing people walking and cars moving with changing background (Video8) (a) PSNR comparison of Object-based compression with MPEG-2 and MPEG-4 (b) Sequence of frames from original video (c) Difference image sequence between original and object-based compressed (d) Difference image sequence between original and MPEG4 compressed. Note that this sequence is quite complex as it has a changing background (addition and removal of cars from the background model). Also, people walk to the car, sit in it and drive away. Thus our method is able to track objects, and update and learn the background, and is able to compress the video with better quality compared to the MPEG-4 method.

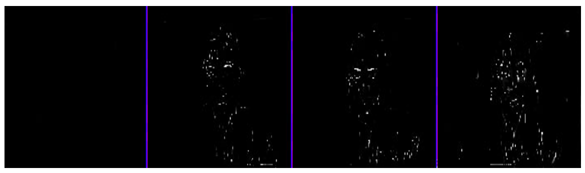


(a)

(b)


(c)


(d)


(e)

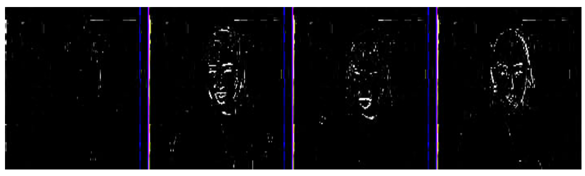
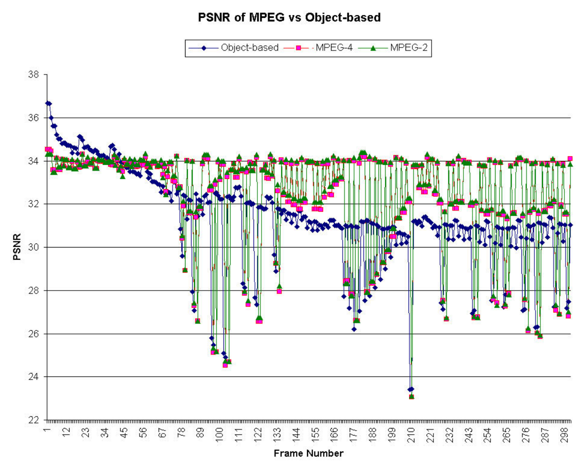
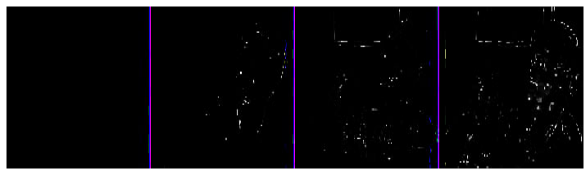

(f)

Figure 10: Standard Japanese newscaster Akiyo video (a) PSNR comparison of object-based compression with MPEG-2 and MPEG-4 (b) Sequence of frames from original video (c) Object layers obtained using contour-based object tracking (d) Reconstructed images using the contour-based object tracking and object-based video compression (e) Difference image sequence between original and object-based compressed (f) Difference image sequence between original and MPEG4 compressed. Note that the objects in this video are tracked by the contour-based method, as the background was occluded by the newscaster.



(a)

(b)


(c)

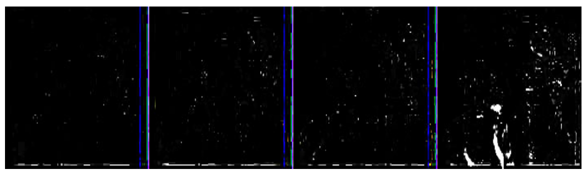

(d)

Figure 11: Standard mother daughter video (mother) (a) PSNR comparison of Object-based compression with MPEG-2 and MPEG-4 (b) Sequence of frames from original video (c) Difference image sequence between original and object-based compressed (d) Difference image sequence between original and MPEG4 compressed.

[17] M. G. Strinzis, "Object-based coding of stereospic and 3D image sequences,". In *IEEE Signal Processing Magazine*, pp.14-28, 1999.

[18] M. J. Turk, and A. Pentland. "Eigenfaces for recognition". In *Workshop on Human Computer Interaction*, Vol.3, pp.71-86, 1991.

[19] A. Vetro, T Haga, K. Sumi, and H. Sun, "Object based coding for long term archive of surveillance video". In *Technical Report, TR-2003-98, MERL*, July 2003.

[20] W. J. Welsh "Model-based coding of videophone images," In *Electronic and Communication Engineering Journal* , Vol.3(1), pp.29-36, 1991.

[21] J. Weng, Y. Zhang, and W. Hwang. "Candid covariance-free incremental principal component analysis". In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.25(8), pp.1034-1040, 2003.

[22] Y. Zhang, and J. Weng. "Convergence analysis of complementary candid incremental principal component analysis". Technical Report MSU-CSE-01-23, Michigan State University, East Lansing, 2001.

[23] A. Yilmaz, X. Li, and M. Shah. "Contour-based object tracking with occlusion handling in video acquired using mobile cameras". In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.26(11), 2004.

[24] A. L. Yuille, P. W. Hallina, and D. S. Cohen, "Feature extraction from faces using deformable templates," In *International Journal of Computer Vision*, Vol.8(2), pp.99-111, 1992.

[25] "CAVIAR: Context Aware Vision using Image-based Active Recognition", Downloadable from *http://homepages.inf.ed.ac.uk/rbf/CAVIAR/*.

[26] "PETS: Performance Evaluation of Tracking and Surveillance", Downloadable from *http://pets2002.visualsurveillance.org/*.