Surveillance video synopsis in the compressed domain for fast video browsing

Shi-zheng Wang\textsuperscript{a,b}, Zhong-yuan Wang\textsuperscript{a,*}, Rui-min Hu\textsuperscript{a}

\textsuperscript{a}Wuhan University, Luoqiaoshan Wuhan, Wuhan 430072, China
\textsuperscript{b}Institute of Automation Chinese Academy of Sciences, 95 Zhongguancun Danglu, Beijing 100190, China

Abstract

The traditional pixel-domain based video analysis methods have taken dominated places for long. However, due to the rapidly increasing volume and resolution of surveillance video, the desirable fast and scalable browsing encounters significant challenges in terms of efficiency and flexibility. Under this circumstance, operating surveillance video in compressed domain has aroused great concern in academy and industry. In order to perform the intelligent video analysis task on the premise of preserving accuracy and controlling complexity, this paper presents a compressed-domain approach for massive surveillance video synopsis generation, labeling and browsing. The main work and achievements include: (1) a compressed-domain scheme is established to condense the compressed surveillance video and record the synopsis results; (2) a background modeling method via the Motion Vector based Local Binary Pattern (MVLBP) is introduced to extract moving objects in an efficient way; (3) an object flags based synopsis labeling method is proposed to represent the object regions as well as their display modes in a flexible way. Experimental results show that the video analysis system based on this framework can provide not only efficient synopsis generation but also flexible scalable or playback browsing.

1. Introduction

With the increasing amount of video data, video surveillance is bringing considerable challenges to data storage, analysis, and fast browsing. Currently, there are many video compression techniques, which have been successfully used to store large volume surveillance video. However, the problem how to analyze the surveillance video effectively and efficiently and provide people access to key information has yet to be satisfactorily solved.

In recent years, surveillance video analysis in the pixel domain has become one of the most active issues in computer vision field. Many automatic analysis systems [1–4] of video surveillance have been devised and applied to military, commercial and industrial fields. However, the pixel domain methods would lead to a high computational complexity instead. Under this circumstance, how to process mass video with relatively low-complexity has received much attention. Some of the researchers choose to analyze videos in the compressed domain for reducing the time cost of video decoding and analysis [5–8]. In the compressed domain, Motion Vector (MV) and Discrete Cosine Transformation (DCT) coefficients are the main information available to extract objects in surveillance video. There are some improve-ments on efficiency with the video analysis methods in the compressed domain, but the high efficiency from incomplete decoding of video streams has not been fully exploited, or even been offset by high algorithm complexity involved in object extraction methods, such as energy minimization, region cluster, etc. In order to achieve the rapid extraction of moving objects for compressed surveillance video, background modeling methods in the compressed domain were presented and performed [7,8]. However, the compressed-domain based background modeling methods proposed so far either use DCT coefficients to model intra-frame backgrounds only, or rely on the introduction of additional information to model inter-frame, while lack effective algorithms to make use of the coding information of inter-frames directly in the compressed domain.

In the field of surveillance video browsing, video synopsis [9] was firstly presented and used by Rav-Acha. It has become the hot-spot in surveillance video field, with applications in video retrieval and video browsing. Generally, the synopsis video is generated from surveillance video in the pixel domain (an example is illustrated in Fig. 1), by extracting moving objects tubes, adjusting the synopsis modes and displaying the moving objects in different periods simultaneously. However, the main practical requirement in surveillance video analysis and browsing is to process compressed historical surveillance video, hence, the efficiency of video synopsis has to be considered. Otherwise, availability and
effectiveness for video synopsis in practical applications would be affected if there is long waiting time before browsing historical video. In [31], authors proposed a framework with online moving objects extraction and offline synopsis analysis. The waiting time for offline browsing is considerably reduced by preprocessing surveillance videos prior to storing. However, for the case involving online browsing, due to the fact that preprocessing cannot be achieved, it is impossible to save the time of generating synopsis video by this way.

In this paper, we propose a synopsis method for compressed surveillance video, which can extract the moving objects and perform the synopsis analysis in the compressed domain. Our main contributions include: (1) A compressed domain based framework of video synopsis for surveillance video browsing is presented, which can reduce the time cost of moving objects extraction while perform effective synopsis analysis; (2) An inter-frame background modeling with Motion Vector based Local Binary Pattern (MVLBP) is formulated to extract moving objects in the compressed domain. Meanwhile, residual information is employed to further refine the extracted objects; (3) An object-oriented synopsis labeling method is devised, which can reduce the synopsis storage cost and implement scalable synopsis browsing online or offline. Various experiments using real surveillance videos verify that the proposed framework can perform high efficiency in terms of synopsis generation, storage and browsing.

The remainder of this paper is organized as follows. Section 2 introduces the related work in surveillance video analysis and in surveillance video browsing. Section 3 particularly introduces the synopsis analysis in the compressed domain. In Section 4, the complete scheme of synopsis labeling and browsing based on object flags is addressed in details. Various experimental results are shown in Section 5. The conclusions and further research are given in Section 6.

2. Related work

We now discuss the related research in the literature and compare our work with the existing work. We classify existing researches into two categories: (1) surveillance video analysis, including video analysis methods both in the pixel and compressed domain; (2) surveillance video browsing, including various video summarization technologies which can obtain the key information of surveillance video in a quick way.

Surveillance video analysis in pixel domain. In recent years, has become one of the most active issues in computer vision field. Many automatic analysis systems of video surveillance have been devised and applied to military, commercial and industrial fields. Gaussian Mixture Model (GMM) [1] has been extensively used in moving object extraction for surveillance applications. After that, Boult et al. [2] presented a system, aiming at monitoring uncooperative and camouflaged targets, for military applications. Besides, Shah et al. [3] proposed an automatic surveillance system for commercial applications, named Knight. Akdemir et al. [33] proposed a systematic method to settle the problem of visual activity recognition, and applied it to bank and airport tarmac surveillance domains with efficiency and validity. Zhang et al. [4] displayed a probabilistic grammar approach to recognize complex events in videos. Recently, an immersive video surveillance system [20] is designed, which is applied to human behaviors learning in longitudinal. Huang et al. [21] has improved a set of biologically inspired models for the scene classification in the process of surveillance video analysis. Moreover, Li et al. [34] presented a method to recognize human action through trajectory analysis in the luminance field. Using background modeling schemes, some recent studies in [40–43] have introduced several very efficient and practical methods for efficient surveillance video coding and content analysis. For example, Paul et al. [40,41] presented the most common frame scene generated by dynamic background modeling for smartly coding intra-frame and inter-frame, respectively. However, the pixel domain methods would lead to a high computational complexity instead.

Surveillance video analysis in compressed domain, which has received much attention, could process mass video with relatively low-complexity. Motion Vector (MV) and Discrete Cosine Transformation (DCT) coefficients are the main materials available to extract objects in surveillance video. For example, Babu et al. [5] used the MV as the cue to segment objects in MPEG. Chen et al. [37] utilized a MV quantization method to preprocess MV and maximum a posteriori estimate to implement moving objects extraction in the compressed domain. Besides, [6] classified MV into four types and then used Markov Random Field (MRF) model to extract moving objects from background. [39] has proposed a behavioral analysis method with Motion Estimation in the compressed domain. Meanwhile, some beneficial explorations are also carried out to extract the objects by using DCT coefficients [22]. There are some improvements on efficiency, but the high efficiency from incomplete decoding of video stream has not been fully exploited, or even been covered up by high algorithm complexity involved in object extraction methods, such as energy minimization, region cluster, etc. To solve this problem, Wang et al. [7] came up with a high efficient algorithm to realize the intra-frame DCT coefficients based background modeling. In addition, Poppe et al. [8] introduce a method only based on the coding size of macroblock (MB) to segment moving objects in H.264. However, the information of inter frame has not been directly used to model background in the compressed domain yet. For this purpose, a background modeling method on the basis of MVLBP in the compressed domain is proposed in this paper. Instead, by coding the amplitude of MV with local binary pattern (LBP), and then modeling background with the LBP value, our method thus can exploit available compressed-domain information in inter frame to achieve rapid extraction of moving objects.

In the field of surveillance video browsing, video abstraction technique, expected to be the key technology emancipating people from intensive and tedious browsing work, has received more and more attention in recent years. By analyzing and processing
surveillance video, a short video abstraction would be generated with the most informative contents of original video. There are two kinds of techniques in video abstraction [23], namely, key frame extraction and video skimming. The former video summarizations also called as still-image abstract, selecting a series of representative frames from original video into a set, which helps to search and retrieve videos conveniently when combined with still-image retrieval [24,25]. Therefore, though it loses some original information in videos, time could be effectively saved to browse and retrieve surveillance video [26,29]. Moreover, in order to increase the processing speed, [38] has presented a method to perform video summarization in the compressed domain. The latter [27,28], also known as moving-image abstract, extracts video segments from the original video to obtain a shorter video, which is more coherent and expressive compared with those derived from the former technique. If effective selection cannot be implemented to the surveillance video, it is likely to spend large amounts of time on browsing and retrieving those video segments with mere information. Therefore, fused with the subjective visual attention, the browsing method for surveillance videos is gradually adopted [30]. Feng et al. [35] proposed a ROI cropping method in the compressed domain to browse high resolution videos efficiently.

As an improved technique for video abstraction, video synopsis [9] can make video abstractions shorter than original ones and preserve the basic contents of original video by displaying the moving objects from different periods simultaneously. Moreover, video synopsis by employing the dynamic programming [31] is proposed in order to adapt to the needs of real-time processing and fast browsing of surveillance video. To further decrease the waiting time for synopsis video generation, Pritch et al. put forward online process framework of video synopsis [11]. By real-time video analysis, they pre-extract moving object tubes and store them into database online, and then generate the favorable synopsis video for users offline. Whereas, for the processing to compressed surveillance videos or videos without online preprocessing, the time cost of generating synopsis video cannot be reduced through those methods above. Therefore, we focus on video synopsis method suitable for compressed surveillance video in this paper. Different from previous work in [9–11,31], we are the first to devise the framework on video synopsis analysis, labeling in the compressed domain.

### 3. Synopsis analysis in compressed domain

As shown in Fig. 2, a method to achieve synopsis video in the compressed domain is proposed. In this framework, a system is designed to support not only displaying synopsis video online, but also scalable browsing offline by combining the object tubes extraction module, synopsis analysis module and synopsis labeling module together. Firstly, in the object tubes extraction module, a novel background modeling approach based on MVLBP is employed. The initial foreground regions are extracted through background modeling, and then a modification based on $4 \times 4$ DCT coefficients and block partitions is implemented. Secondly, the display scheme for moving objects in synopsis video is achieved by synopsis analysis with energy function minimization after acquiring the moving object tubes. Finally, for the requirement of analyzing online and fast browsing offline, an object flags based method to label the object region and the display scheme is proposed, which facilitates the fast scalable browsing of synopsis video by writing object flags into bitstreams.

In this section, we focus on presenting the algorithms of three modules in video analysis: background modeling and foreground segmentation, object tracking and tubes extraction as well as synopsis analysis. The realization of synopsis labeling and video browsing modules will be described in the next section.

#### 3.1. Background modeling and foreground segmentation

In this paper, a background modeling method based on spatio-temporal MV modeling is proposed. At first, we preprocess the MV field for intra block and noise inter block; secondly, calculate the LBP value of MV based on the amplitude of MV with the accuracy of block partition; thirdly, introduce the LBP value and motion estimation into background modeling to extract moving objects. In H.264, each macroblock (MB) can be partitioned into various sizes of subblocks, and each subblock corresponds to a MV and a reference frame generally. However, following the coding-oriented RDO criterion [12], MB in the inter-frame may be predicted as intra mode for the homogeneity of adjacent blocks. When a background block is similar to a foreground block, a noisy MV would appear. To obtain a more reliable and dense MV field, initial MV field are

![Fig. 2. Framework of video synopsis and browsing in the compressed domain.](image)
preprocessed by a temporal accumulation within 3 inter frames and a $3 \times 3$ median filtering [37].

After the MV field is refined, LBP feature [14] is introduced to describe the spatial correlation among neighboring blocks. In our algorithm, as shown in Fig. 3, the LBP feature is extracted on the basis of each subblock in MV field, and the neighbour blocks are selected according to the block partitions. That is, neighboring blocks $D_3$–$D_8$ around current block $D_0$ are based on actual block partitions in the compressed surveillance video. The $D_i$ block will be set to 1, if its MV amplitude $MV_i$ is bigger than that of current block $D_0$, while The $D_i$ block will be set to 0, if its $MV_i$ is smaller than that of $D_0$. In this paper, we search for 4 blocks on the vertical angles directions of the current block at first, then search for 4-neighbour that connected with the 4 edges of the current block. The blocks in vertical angle direction is unique, while there might be two different situations for the blocks that connected with the 4 edges of current block: the first, the neighbour block is uniquely determined, including the situation that neighboring block is bigger than central block, like block $D_3$–$D_8$; the second, when neighbouring blocks are more than one block, like block $D_2$, we would calculate the average value of MV in neighboring blocks. Then we traverse neighboring blocks in clockwise order start with the top-left block.

Afterwards, we model background in the compressed domain with $4 \times 4$ block-partition accuracy. Because LBP is the mapping code from binary code to decimal code, not the ordinary numeric value in partial order (i.e. size relationship between numbers), the conventional methods of probability density function (pdf) estimation, neither the parametric GMM [1], nor the method of non-parametric kernel density estimation [14], can be directly used for estimating the probability density of LBP. Inspired by the existing methods of kernel density estimation [43], we define a distance function $d(p,q)$ as the number of different bits between two local patterns $p$ and $q$. Then we derive our local pattern kernel as $\Phi(p,q) = g(d(p,q)))$, where $g$ is a weighting function that can typically be a Gaussian. The probability density function of local pattern $q$ can be estimated smoothly by:

$$\hat{f}_{kl}(q) = (1-\lambda)\hat{f}_{kl-1}(q) + \lambda\Phi(p_t, q)$$  \hspace{1cm} (1)

Where $K$ is the number of models for the local pattern and $i$ is a learning rate. In the update procedure, the new pattern $p_t$ is matched to the weight sorted probability density functions in turn, and a match is found when $f_{kl-1}(p_t) > T_m$, where $T_m$ is a threshold parameter controlling the matching. Once a match is found, the matched probability density function would be updated [43], while other probability density functions remain the same. In addition, the $K$ weights are updated as:

$$w_{k} = (1-\lambda)w_{k-1} + \lambda\Phi(p_t, q)$$  \hspace{1cm} (2)

Where $w_{k}$ is an indicator variable being 1 for the matched model and 0 otherwise. If there is no matching in $K$ distributions to current pattern, a new probability distribution would be set according to the new pattern to replace the distribution with a low initialized weight. Then the probability of following input $4 \times 4$ block observations for each Gaussian density is presented by $P_{c}(p_t)$:

$$P_{c}(p_t) = \frac{\sum_{k=1}^{K}w_{k}\hat{f}_{k}(p_t)}{\sum_{k=1}^{K}w_{k}}$$  \hspace{1cm} (3)

Where $w_{k}$ is the weight of the $k$th Gaussian function at the $r$th frame and normalized to 1. Here, we have accomplished the temporal background modeling to current block $p_t$. Considering that if the current block is an inter block, as motion estimation is characterized by matching the similar blocks in inter-frame when coding surveillance video, we utilize MV to back trace current block $p_t$ to the corresponding $4 \times 4$ block in reference frame, and record the probability of reference block as $P_t$ calculated by Eq. (3). Then, the ultimate result $P_f$ is weighted with probabilities of the current block and its corresponding reference block:

$$P_f = \omega P_t + (1-\omega)P_c$$  \hspace{1cm} (4)

When reference block is inter block, $\omega = 0.5$, otherwise to be 0. Then we can classify the block $p_t$ into background or foreground by comparing its probability with a predefined threshold $Th_s$.

3.2. Object tubes extraction

After extracting moving objects by background modeling and foreground segmentation, we perform sticky tracking and tubes extraction on the extracted moving objects. As shown in Fig. 4, object tube is the 3D spatio-temporal representation of each object, and the goal of tracking moving object regions is to obtain object tubes. During the subsequent operation of video synopsis with energy functions minimization, each object tube would be regarded as a whole to calculate cost function and confirm the displaying way in synopsis video.

Considering that no more feature information can be extracted from video prediction information in compressed domain, e.g. texture and color features that are usually used in pixel-domain tracking [1], we select the sticky tracking [16], based on Nearest Neighbor Algorithm [13] as the moving objects tracking algorithm in our proposed method to track extracted moving object regions. The basic principle is to calculate the nearest distance between moving object regions from the current frame and frames in front or back of it, respectively. Assuming that there are $m$ objects at time $t$ and $n$ objects at time $t+1$, and $C_{mn}$ denotes the matrix of objects matching before sticky tracking, as described below:

$$C_{mn} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1m} \\ c_{21} & c_{22} & c_{23} & \cdots & c_{2m} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & \cdots & c_{nm} \end{bmatrix}$$  \hspace{1cm} (5)
Since objects may merge or split mutually, and background may be clutter, it may be difficult to directly match or track these m objects with the n objects; furthermore, flickering might occur easily. Hence, we merge the objects whose nearest distance d is less than 10 pixels to be one blob, and then perform tracking and object extraction to this blob (maybe moving object set). When there is a temporal overlapping or a spatial contact, the distance is set to be 0. If their distance between moving object regions in one frame is less than 10 pixels, we would assume moving objects belong to the same object tube, which would be regarded as a whole to operate energy function optimization hereafter.

In the process of sticky tracking, as the nearer objects are combined together, the number of combined moving objects is: \( m' < m \cdot n' < n \). The matching matrix \( C_{wr} \) after sticking tracking can be illustrated as:

\[
C_{wr} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1m'} \\
                  c_{21} & c_{22} & c_{23} & \cdots & c_{2m'} \\
                  \vdots & \vdots & \vdots & \ddots & \vdots \\
                c_{n1} & c_{n2} & c_{n3} & \cdots & c_{n'm'} \end{bmatrix}
\]

(6)

Experiment results show that sticky tracking is successful at reducing blinking effect, keeping the chronological order of intersectant objects, and amending over-segmentation, and thus efficiently optimizing the subjective performance of synopsis video browsing. In Fig. 4(a), it is the subjective effect of tube extraction with conventional nearest neighbour tracking algorithm [36] performed in pixel domain. Without sticky tracking, when current moving object is occluded by another moving object or they are intersected, the two moving objects are merged as one blob to be displayed together, but this blob will disappear when they break up. In the case of using the resulting tube directly in the subsequent video synopsis, there will appear a blinking effect in the obtained synopsis video. While with the sticky tracking, two intersectant moving objects are viewed as one object tube to calculate energy function. In this way, blinking effect of moving objects are suppressed while the chronological order of intersectant objects can be maintained, as shown in Fig. 4(b). In addition, sticky tracking can also settle the problem over-segmentation as shown in Fig. 4(c) and (d). Due to the collision of streetlight, the white car in the 394th frame of sequence “Daytime” splits into two moving object regions; however, under the sticky tracking, they are merged into the same tube as the distance between the two moving object regions is less than 10 pixels.

3.3. Synopsis analysis by energy minimization

Inspired by pixel-wise synopsis analysis [11], we design the synopsis analysis in the compressed domain to deal with the block-based objects after getting the set of object tubes. A new error concealment cost \( E_e(b) \) is introduced to the original energy function for calculating the cost of incomplete extraction of objects which are mapped into synopsis. In addition to \( E_e(b) \), the energy function \( E(M) \) contains other three conventional cost functions, including activity cost \( E_a(b) \), collision cost \( E_c(b, b') \) and temporal consistency cost \( E_t(b, b') \), and they are used to penalize the loss of object activities, object collision and the alteration of relative timing order of object tubes respectively. As mentioned above, the blocks within moving object region consist of seven types of divided block. Therefore, the synopsis analysis procedure is conducted with blocks as the unit in our algorithm. The energy function is shown as follow:

\[
E(M) = \sum_{b \in \mathbb{B}} (E_a(b) + \gamma E_e(b)) + \sum_{b, b' \in \mathbb{B}} (\alpha E_c(b, b') + \beta E_t(b, b'))
\]

(7)

Where \( \alpha, \beta, \gamma \) are weight factors. They can be adjusted according to the requirements of practical application. \( b \) and \( b' \) are different object tubes in the original video, while \( b \) and \( b' \) are those mapped into the synopsis video. \( B \) is the set of all tubes and \( M \) is the mapping relationship from original video to synopsis video, which indicates the display time of object tubes in set \( B \). The error concealment cost \( E_e(b) \) can be formulated by:

\[
E_e(b) = \sum_{t \in T} E_e(b_t)
\]

(8)

\[
E_e(b_t) = \begin{cases} D(b_t) > T_d & D(b_t) \geq T_d \\ 0 & \text{otherwise} \end{cases}
\]

(9)

Where \( T \) is the set of frame IDs mapped from object \( b \) to object \( b' \) in the original video, \( \|b\| \) is the number of blocks of object \( b \) in t-th frame, and \( D(b_t) \) is the area difference between the current object \( b_t \) and \( b_j \) is the mask generated by the temporal domain projection method [32], which is defined as the following:

![Image](https://example.com/image.png)
Instead of \( ^{1} \) in object region, and \( \mathcal{S} \) is the pixel-wise area of \( s_i \). When \( D(b_i) > D(b_i') \), we use the union of mask \( b' \) and original object region \( b \) instead of \( b \), to display the synopsis video for error concealment. The error concealment cost function can be used to suppress the occurrence of incomplete object regions in the synopsis video, while such segmentation defect may commonly exist in the primary synopsis video. Simultaneously, the incompletion of object tubes is also concerned in our algorithm. Therefore, we employ activity cost \( \mathcal{E}_a \) to penalize the loss of activities in object tubes, as follow:

\[
\mathcal{E}_a(b) = \sum_{i \in b} \chi(s_i) - \sum_{i \in b'} \chi(s_i)
\]

By using the \( \mathcal{E}_c(b) \) and \( \mathcal{E}_a(b) \), both the integrity of object regions and that of object tubes will be preserved, which improves the visual performance of synopsis video significantly. However, the collision among objects may arise in the synopsis video if it is short enough. Thus, we also employ the collision cost \( \mathcal{E}_c(b, b') \) to our algorithm and compute the collision cost at block level, instead of pixel level. The block level cost is:

\[
\mathcal{E}_c(b, b') = \sum_{k \in b \cap b'} \chi_b^c(s_k) \cdot \chi_{b'}^c(s_k)
\]

To reserve the temporal order of object tubes, the temporal consistency cost \( \mathcal{E}_t(b, b') \) is utilized. Assuming the occurrence of object tube \( b' \) is earlier than another object tube \( b' \) in the original video, it is formulated as:

\[
\mathcal{E}_t(b, b') = \exp(-{(t_b^e - t_b'^e)/\sigma_{time}})
\]

Where \( t_b^e \) is the ending time of object \( b \), and \( t_b'^e \) indicate the occurring time of object \( b' \) in the synopsis video. \( \sigma_{time} \) is a parameter defining the extent of temporal interaction between two objects. Consequently, having interpreted the four cost functions, the procedure of synopsis analysis can be represented as an optimal minimization problem as follow:

\[
M_{best} = \arg \min_M (\mathcal{E}(M))
\]

Fig. 5 explains our method intuitively. The minimization of energy function is performed by the iterative deterministic relaxation algorithm known as Simulated Annealing (SA) [9]. In particular, we employ the following customized SA algorithm to find satisfactory optimum for Eq. (13).

\begin{algorithm}
\caption{Minimize energy function by SA}
\begin{algorithmic}
\STATE \( E(M_0) = \max; t = 0; i = 1; j = 0 \)
\WHILE {\( t < 3 \) and \( i < 10 \)}
\STATE \( M_0 = \) random simulated annealing state of \( S(M) \)
\IF {\( E(M_0) < E(M_i) \)}
\STATE \( M_0 = M_i \)
\STATE \( j = 1; \)
\STATE \( t = t + 1 \)
\STATE \( i = i + 1 \)
\STATE \( M_j = \) output mapping pattern \( M_{best} \)
\ENDIF
\ENDWHILE
\RETURN \( M_j \)
\end{algorithmic}
\end{algorithm}

Where \( S(M) \) is the mapping set that SA algorithm may obtain, \( t \) is used to record the number of current mapping pattern being minimum mapping pattern, and \( i \) is used to record the number that the amount of energy is randomly compared. On the basis of above algorithm, if a mapping pattern \( M_i \) is the minimum cost solution in the third consecutive SA step (\( t > 3 \)) or the minimum energy solution within ten comparisons on the energy amount of random state (\( i > 10 \)), it will be output as the optimal mapping pattern.

4. Synopsis labeling and browsing based on object flags

4.1. Synopsis labeling

To generate and display the object-oriented synopsis video, it is crucial to record the regions of all moving objects and their mapping frame ID in the synopsis video. In this subsection, we suggest a coding and storage method for results of synopsis analysis based on object flags. Object flags consist of object region flags and object mapping flags. They are elaborated as follows:

4.1.1. Object region flags coding

Prediction coding can improve the coding efficiency of object flags, while ensure object region flags to be decoded using the corresponding prediction information in compressed domain. The moving object region with \( 4 \times 4 \) block-partition accuracy can be bounded with a minimum rectangle. Then, the moving object
blocks will be labeled as 1 and non-object blocks as 0 based on block partition in the rectangle.

In *intra-frame coding*, we propose a novel block traversing method to label the moving object region, which can reduce the labeling cost in the synopsis labeling compared to the traditional traversing method. First, the moving object within the rectangle is divided into several blocks, which can be denoted by a set \( \mathcal{B} = \{ \text{blk}_1, \text{blk}_2, \ldots, \text{blk}_N \} \). Given that the moving object is generally in the central region of rectangle, we introduce the horizontal and vertical axis into rectangle. Then the function is defined to calculate the distance from one block to the center of rectangle as following:

\[
d_n = \frac{d_n^2(\text{blk}_n)}{H_0^2} + \frac{d_n^2(\text{blk}_n)}{W_0^2}, \quad n = 1, 2, \ldots, N
\]

(16)

Where \( H_0, W_0 \) are normalized parameters, and they are proportional to the height and width of the rectangle. \( d_n(\cdot), d_n(\cdot) \) are used to calculate the pixel-wise minimum distance from block \( \text{blk}_n \) to the horizontal and vertical axis respectively. The distances of all blocks are sorted in ascending order, and then we traverse the blocks in the same order. As shown in Fig. 6, this method would lead to a more intensive appearance of 0 and 1, which would reduce the storage cost remarkably compared to the general order, which traverses from top to bottom and left to right. In this paper, the prefix and suffix flags are coded with RLC (run-length coding) [15], and the middle part is written into stream directly. As a result, the original bit-chain can be efficiently coded into a shorter one.

In *inter-frame coding*, in order to make better use of involved temporal correlation, block partition, MV and reference frame are adopted to code object region flags. Shown as Fig. 7, first, on the basis of the moving object region in reference frame, we divide the reference region into foreground region \( F \), background region \( B \) and boundary region \( C \), and set the width of \( C \) as one pixel. Second, map each block under object region flags coded in the rectangle back to reference frame with the corresponding MV: if the mapped block is in the region \( F \), the prediction is foreground, and a prediction flag is represented as 1; if in region \( B \), it is background, and represented as 0; if connected with region \( C \), regard it as the unconfirmed block. Then modification would be employed by the way of recording the coordinates of false predicted blocks. Finally, in inter-frame coding, lossless region labeling bitstream would include the flags of the unconfirmed blocks and the coordinates of false predicted blocks in the region \( F \) and \( B \).

**4.1.2. Object mapping flags coding**

Object mapping flags can be regarded as the flags information which records the mapping relationship of moving objects between original video and synopsis video. By using synopsis analysis described in Section 3, we can obtain the synopsis video and its corresponding mapping information automatically. We assume the mapping relationship function \( f(i,k) \), where \( i \) is the frame ID of original video and \( k \) is the object ID in frame \( i \). \( f(i,k) \) is the mapped frame ID of synopsis video when this object tube in current frame is mapped. Otherwise, \( f(i,k) \) is equal to \(-1\), when it is not mapped.

In detail, we assume that: there are \( M \) frames in original video and \( M \) frames in synopsis video. Assume the total number of objects extracted from original video is \( K \). \( T_k = [t^k_s, t^k_e] \) denotes the time interval of \( k \)th object in original video, while \( T_k = [t^k_s, t^k_e] \) denotes the time interval of object \( k \) in synopsis video, \( k \in \{1, 2, \ldots, K\} \).

Based on the assumptions above, object mapping flags can be expressed as below:

\[
f(i,k) = \begin{cases} 
    m & \text{if mapping} \\
    -1 & \text{otherwise} 
\end{cases} \quad i \in [t^k_s, t^k_e], \quad m \in \{1, 2, \ldots, M\}
\]

(17)

In our work, we adopt differential coding method [15] to encode object mapping flags in the scheme. The mapping relationship of objects in original video and synopsis video are shown in Fig. 8. As we can see, if we get the region flags of objects and their mapping relationships, the synopsis video can be represented intuitively and accurately.

As for storage of object flags, we combine the region flags obtained from synopsis labeling module and the mapping flags obtained from synopsis analysis module together at the encoder to get a group of object flags, and then write the sets of flags and synopsis information (frame number of synopsis video and object number of each frame) into the scalability extension parameter sets of H.264/AVC bitstream in original video using lossless coding method [12].

**4.2. Synopsis video generation and browsing**

Once receiving the binary bitstreams, the decoder can obtain the synopsis information and object flags immediately. In our experiment, the background would be generated and updated by the Principal Background Selection (PBS) method [16], when synopsis video display is required. Afterwards, the synopsis video can be reconstructed with the object flags, synopsis information, the decoded original video, MV, and block partition. A reconstructed schematic diagram of synopsis video is showed in Fig. 8.

Moreover, to expand two-layer mapping structure to multi-layer, we put forward a scalable reconstruction method. This method readjusts the mapping relationship of moving objects to reconstruct synopsis video with different video lengths and object condensations in a scalable way. Users can reconstruct a favorable synopsis video by setting the scalable level \( \rho \). If \( \rho = 0 \), the synopsis video will be reconstructed with initial mapping relationship \( f_o \).

We define \( f_s(i,k) \) as the mapping relationship of moving object \( k \) in frame \( i \), which is updated by setting the scalable level \( \rho \). The relationship can be expressed as below:

![Fig. 6. Traversing method of object region flags in intra-frame coding. (a) Original video. (b) Block partition of the 140th frame for QP = 30. (c) is the diagram of traversing order for subblocks. (d) is the diagram of labeled moving object region.](image-url)
Instead, we adjust the mappings as follows: If \( q_k > q_i \), we define \( f_q(i,k) \) as the frame ID of the moving object \( k \) in original video. Meanwhile, object segment, whose mapping ID is \( q_k = -1 \), will be set equal to the frame ID in original video. The following algorithm describes the procedure.

**Algorithm 2** Update the Mapping Relationship

**Input:** \( \rho \), scalable level for synopsis video reconstruction.  
\( f_0 \), initial mapping relationship with scalable level 0.  
\( l_{\text{max}} \), local variable.  
\( f_p \), new mapping relationship with scalable level \( p \).  

**Data:** \( f_0 \), initial mapping relationship with scalable level 0.  
\( f_p \), new mapping relationship with scalable level \( p \).  
\( l_{\text{max}} \), local variable.  

for \( k = 1 \) to \( K \) do  
for \( i = t_{k}^l \) to \( t_{k}^h \) do  
if \( f_q(i,k) = -1 \) then  
\( f_q(i,k) = f_0(i,k) + (k - 1) \cdot \rho \)  
end  
end  
for \( i = t_{k}^l \) to \( l_{\text{max}} \) do  
\( f_q(i,k) = i \)  
end

\[ f_q(i,k) = \begin{cases} f_0(i,k) + (k - 1) \cdot \rho & f_0(i,k) \neq -1 \\ -1 & \text{otherwise} \end{cases} \quad (18) \]

In our mapping procedure, we set a criterion: \( 0 \leq f_q(i,k) < i \). If \( \rho \) violates the criterion, mapping function (16) will not be implemented. Instead, we adjust the mappings as follows: If \( f_q(i,k) > i \), we define \( f_q(i,k) \) as the frame ID of the moving object \( k \) in original video. Meanwhile, object segment, whose mapping ID is \( q_k = -1 \), will be set equal to the frame ID in original video. The following algorithm describes the procedure.

5. Experiments

To employ three video sequences to evaluate the subjective and objective performance: the MPEG-4 sequence “Hall Monitor” [17], the AVS working group sequence “Daytime” [18] and the sequence named “F-building”, in which we captured the lossless BMP images with 25 frames per second using “Logitech C905” webcam, then transformed the images into YUV sequence. The main parameters are listed in Tables 1 and 2, and other parameters are set as: \( T_m = 0.01, T_h = 0.01, K = 3 \). All the codecs involved in this paper are on the basis of the JM12.4 platform [19]. The configurations are as follows: only the first frame is coded using intra-coded way, maximum reference frame number is 5, and the search range of MV is \([-32, 32]\). The widely used or even the sole parameter QP = 30 of video analysis in compressed domain [5, 7–9] is our experimental condition in this paper. All the experiments are conducted on a desktop PC with Intel Core i5, 2.67 GHz CPU, 2G RAM, and under the Microsoft Windows XP Professional operating system.

5.1. Precision and recall of object extraction

Performance evaluation is implemented on the basis of \( 4 \times 4 \) block precision. Wherein, we comprehensively compare the performance of proposed algorithm with moving object extraction algorithm, which modeling background in the pixel domain with \( 4 \times 4 \) block precision, and the moving objects extraction algorithm in the compressed domain [6, 37] respectively. The manual \( 4 \times 4 \) block-level ground truth is used to quantify the performance. The frame information used for performance evaluation is as followed: one frame from every ten frames for the sequence “F-building” (interval: 30–1000), one frame from every five frames for the sequence “Hall Monitor” (interval: 20–300), and one frame from every ten frames for “Daytime” (interval: frames with car).

Three criteria, precision (P), recall (R) and F-measure are used to quantify the performance. Table 3 shows the performance of moving object extraction in different test sequences with different algorithms. Since the coarse classification of MV is merely based on rigid threshold [6], this algorithm can be easily affected by MV noise, as shown in Fig. 9, its robustness is worse than the other anchor algorithm and proposed algorithm, and precision as well as recall are also lower than proposed algorithm. In the algorithm [37], when the modification is not implemented to the pixel domain, its recall is relatively higher than proposed algorithm, while its precision is far lower than proposed, which is not applicable to the requirement of accurate segmentation of surveillance videos. In contrast, the proposed method keeps the balance between higher precision and recall.

Due to the limitation of available information in the compressed domain, the performance of background modeling in the
pixel domain is better than compressed-domain methods. However, the complexity in pixel-domain video analysis is obviously higher than compressed-domain video analysis through the observation of processing time. About this point, we will discuss in Section 5.3 in details.

As indicated in Fig. 9, the subjective performance of moving objects extraction and synopsis videos based on the compressed domain is slightly worse than that on the pixel domain. However, the synopsis video itself is a visual data which helps users fast browse and retrieve surveillance video. Therefore, the decrease in objective quality does not damage the advantage of synopsis video that much.

### 5.2. Synopsis labeling and scalable browsing

Based on the scheme introduced in Section 3, we develop an synopsis labeling and synopsis browsing subsystem conducted on the JM12.4 platform [19]. We name the new codec as JM12.4-S below.

We make comparisons on the storage cost and objective quality of the synopsis video acquired in the two different domains. In our experiments, CABAC entropy coding mode is adopted. When original video is coded by JM12.4-S, object flags are not coded into the extension parameter sets, which lead to the same storage cost and objective quality with those by JM12.4. From the angle of objective criterion, the PSNR of synopsis video acquired by reconstructed original video and object flags in JM12.4-S is approximately the same with that from the reconstructed synopsis video in JM12.4. As shown in Table 4, compared with the 21.93% average increase in storage space, only an increase of 0.35% is presented in JM12.4-S based on object flags coding method. Furthermore, with the object flags, JM12.4-S could realize scalable display of synopsis video, while JM12.4 could not.

User interface of scalable browser and playback browser are shown in Figs. 10 and 11 respectively. Users can set the scalable level to adjust the mapping condensation of moving objects to realize scalable browsing, or click the corresponding target in synopsis video to achieve playback browsing of original video. Furthermore, in order to browse important contents of surveillance video, neither objects outside the region of interest nor small objects are mapped when high-condensation synopsis video is required. We set the range of scale from 0 to 50, where 50 is the original video and 0 is the synopsis video with the highest condensation.

Note that, the above experiments hardly involve the scene change scenarios or crowded places. Actually, if the camera only occasionally switches scenes in practical applications, we can first divide the original surveillance videos into different segmentations according to their respective backgrounds, and then perform video synopsis based on segmentations in a similar way. When applied to crowded places, we can treat it as fast browsing without blank frames if the crowd emerges sometimes. However, if the population is always very crowded, the video synopsis task cannot be effectively addressed by our method.

### 5.3. The comparison of time consumption

To evaluate the time consumption on fast browsing in different situations, synopsis analysis and labeling procedure can be divided into five modules: (a) video decoding; (b) background modeling and foreground segmentation; (c) sticky tracking and object tubes extraction; (d) video synopsis by energy minimization; (e) synopsis video labeling or coding, for two browsing modes: online and offline.
$T_d$ is the time of complete decoding for synopsis analysis in the pixel domain, and $T_{d2}$ is the time of partially decoding that includes block partitions, MVs, reference frames and residuals in the compressed domain. Besides, $T_b$ is the time of background modeling and foreground segmentation, $T_s$ is the time of sticky tracking and object tubes extraction, $T_c$ is the time of video synopsis by energy minimization, and $T_e$ is the time of synopsis video encoding. All the time is recorded in the two different domains: $T_{d1}, T_{b1}, T_{s1}, T_{c1}$, and $T_{e1}$ are in the pixel domain, while $T_{d2}, T_{b2}, T_{s2}, T_{c2}$ and $T_{e2}$ are in the compressed domain. The experimental results are shown in Tables 5 and 6. Compared with the complete decoding for pixel-domain synopsis analysis, there is a time saving ($T_{d1} - T_{d2}$).
generated from partially decoding in the compressed domain. Besides the time of decoding the fundamental information, the total time cost for synopsis analysis of surveillance video in the pixel domain and the compressed domain are \((T_{d1} + T_{d2} + T_{c1} + T_{c2})\) and \((T_{d2} + T_{d2} + T_{c2} + T_{c2})\), respectively. Their subtraction indicates the time saving of video analysis process in our framework. Compared to the synopsis video storage, the saving time by synopsis labeling is \((T_{c1} - T_{c2})\).

In the offline browsing mode, analysis and storage of synopsis video are required, while real time display is not, the saving time in the compressed domain is \((T_{d1} + T_{d2} + T_{c1} + T_{c1} - T_{c2} - a1 + T_{d2} + T_{d2} + T_{c2})\) and compared with the pixel domain. However, in the online browsing mode, real time display is required, complete decoding should be carried out, and thus the total time cost of decoding compressed surveillance video for synopsis browsing in our scheme is the same as that in the pixel domain. Therefore, the total time saving is \((T_{d1} + T_{d2} + T_{c1} - (T_{d2} + T_{d2} + T_{c2})\). As showed in the Tables 7 and 8, there is time saving in every browsing mode, generated from the method taking block as the compressed domain operation unit. As for the application, online browsing, the average percentage of the total processing time in the compressed domain accounting for the time in the pixel domain is 61.36%. Furthermore, as for the application, online analysis and storage for offline browsing, the average percentage is 5.85%. All the recorded time listed in Tables 5 and 6 are the average time of 10 times operating.

### 6. Conclusions

In this paper, a novel scheme for video synopsis based on compressed surveillance video is presented and discussed. A MLVPB based background modeling is devised for application in the compressed domain, with better utilization of inter-frame motion information. Meanwhile, an object-oriented synopsis labeling method is formulated to define the regions, map the objects based on the object flags, and finally fulfill scalable synopsis browsing at the decoder. This technology enjoys a promising surveillance oriented application especially in video browse and retrieval. To adapt to various application requirements, cloud computing for video synopsis may be useful and acceptable to accelerate our system in the future.

### Acknowledgments

This work was supported by the grants of the National Science Foundation of China (61231015, 61727137, 61070800, 61003184, 61170023), the major national science and technology special project (2010ZX03004-003-03), and the National Basic Research Program of China (973Program) (20-09CB320906).

### References


### Table 5

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Pixel domain</th>
<th>Compressed domain</th>
<th>Ratio (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (second)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F-building</td>
<td>11.495</td>
<td>34.418</td>
<td>27.855</td>
</tr>
<tr>
<td>Hall monitor</td>
<td>2.953</td>
<td>17.317</td>
<td>15.031</td>
</tr>
<tr>
<td>Daytime</td>
<td>9.734</td>
<td>27.347</td>
<td>19.173</td>
</tr>
</tbody>
</table>

### Table 6

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Time (second)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T_d1 T_d2 T_c1 T_c2 T_c2</td>
</tr>
<tr>
<td>F-building</td>
<td>1.334 12.446 19.721 16.879 8.190</td>
</tr>
<tr>
<td>Hall monitor</td>
<td>0.422 8.432 8.342 10.452 6.141</td>
</tr>
</tbody>
</table>

### Table 7

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Time (second)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T_e1 T_e1 T_e1 T_e1 T_e1</td>
</tr>
<tr>
<td>Pixel</td>
<td>939.79 57.87 6.25</td>
</tr>
<tr>
<td>Compressed</td>
<td>949.86 33.77 3.56</td>
</tr>
<tr>
<td>Ratio (%)</td>
<td>691.51 53.77 7.74</td>
</tr>
</tbody>
</table>

### Table 8

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Time (second)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T_e1 T_e1 T_e1 T_e1 T_e1</td>
</tr>
<tr>
<td>Pixel</td>
<td>102.89 60.54 58.84</td>
</tr>
<tr>
<td>Compressed</td>
<td>54.38 30.16 55.46</td>
</tr>
<tr>
<td>Ratio (%)</td>
<td>79.94 55.78 69.78</td>
</tr>
</tbody>
</table>


