Future Generation Computer Systems 86 (2018) 1371–1382

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Video big data in smart city: Background construction and optimization for surveillance video processing

Ling Tian, Hongyu Wang, Yimin Zhou *, Chengzong Peng

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China

HIGHLIGHTS

- A three-level video data fusion scheme is described for IoT BD fusion.
- A coding architecture with background picture is proposed for smart city video.
- A specific coding parameter optimization algorithm obtains notable performance.

ARTICLE INFO

Article history:
Received 27 September 2017
Received in revised form 23 November 2017
Accepted 29 December 2017
Available online 31 January 2018

Keywords:
Smart cities
Internet of Things
Video big data
Data fusion
Surveillance video processing

ABSTRACT

Transforming infrastructures, buildings and services with the sensed data from the Internet of Things (IoT) technique has drawn wide attention. Enormous video data from city surveillance cameras poses huge challenges of transmission, storage and analysis, which necessitates new video compression technologies. The fusion of video data generated from smart city could be used to support city management and urban policy. Based on the specific characteristics of surveillance video, which are successive pictures have very strong correlations and each picture can be divided into background and foreground, this work proposes a block-level background modeling (BBM) algorithm to support long-term reference structure for efficient surveillance video coding. A rate–distortion optimization for surveillance source (SRDO) algorithm is also developed to improve the coding performance. Experimental results show that the proposed BBM and SRDO can significantly improve the compression performance, which can effectively support diverse video applications in smart city.

1. Introduction

During the last few decades, the expansion of cities have resulted in city issues such as traffic, public security and crime tracking [1]. The concept smart cities will take advantages of the technologies about data sensing and big data analytics to gather the human activities information from all over the cities. It analyses the data and provides intelligent services for public application [2].

Particularly, transforming infrastructures, buildings and services with the Internet of Things (IoT) technique has attracted wide attention from both academia and industry [3]. With different sensors, IoT can generate numerous city data such as air composition, earth vibration and traffic [4]. Among such data, which shows that the pragmatic efficacy of clustering for collecting [5], the video source captured by the surveillance cameras has extensively increased. It is shown that the global video surveillance generated more than 560 Petabytes of data per day [6], which poses significant challenges for storage, transmission and analysis of such video data. Hence, there is a great need for robust administration system of large quantities of videos, which is capable of effective video compression, large, stable storage and high bandwidth of Ethernet or Internet transmission. Fig. 1 shows an example of the video big data cluster.

It can be seen from Fig. 1 that the video cluster has four layers. The basic capture layer collects the raw data from the surveillance cameras. It adopts the video compression technology to reduce the data size and stores them in storage devices. The management layer controls and classifies the videos and collected pictures. The analysis layer would extract features from pictures and analyze the data with intelligent algorithms and tools. After that, the analysis results are transferred to the application layer to improve city services such as surveillance control, observation, tracking and etc.

In the surveillance video big data cluster, compression is an important process. Compared with common cameras, surveillance cameras have a notable feature, that is, the scene, monitor scope and optical lens focus distance are relatively static. Hence, there is strong temporal relevance between successive pictures in the

* Corresponding author.
E-mail address: yiminzhou@uestc.edu.cn (Y. Zhou).

https://doi.org/10.1016/j.future.2017.12.065
0167-739X/

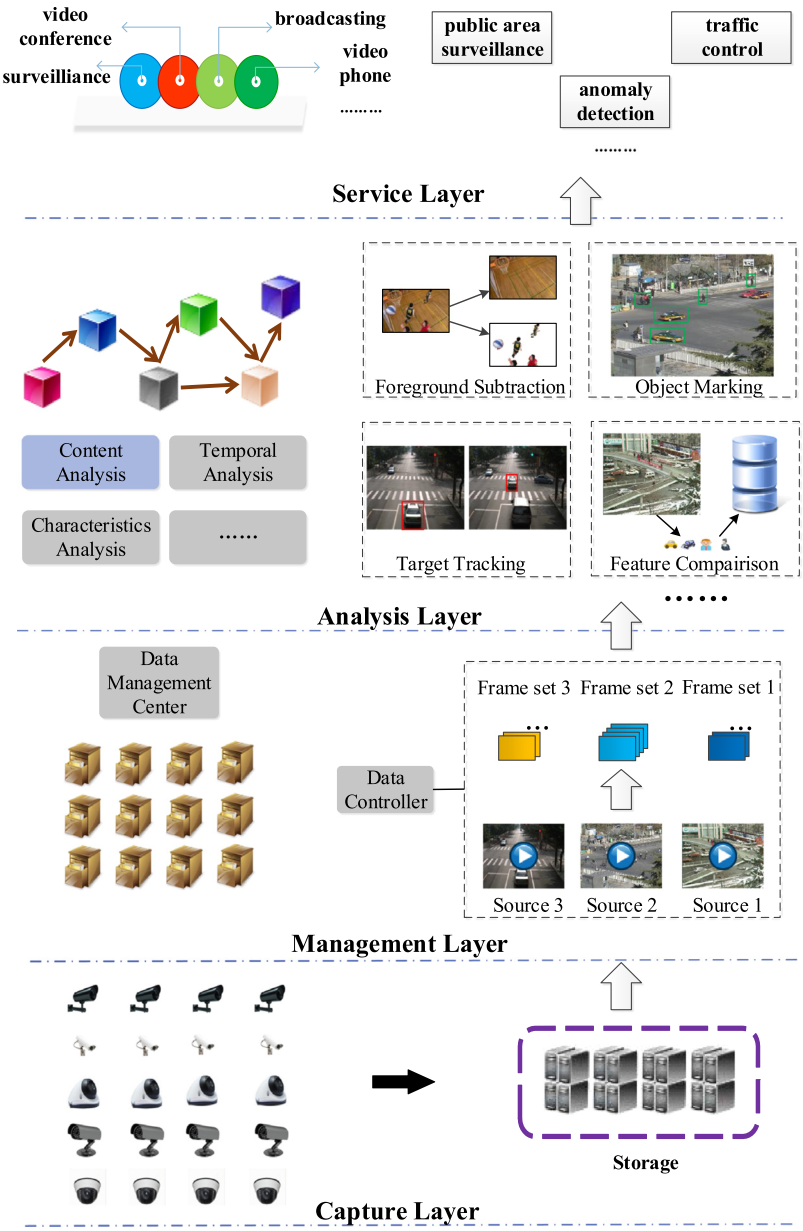


Fig. 1. Video big data cluster.

surveillance video and each picture can be divided into two parts, background and foreground. Relative to the foreground, the context of background is mostly constant, and the violent fluctuations of content occur in the foreground part. In this work, we consider this characteristic in video encoding to achieve high compression efficiency. We proposed a new block-level background modeling algorithm to build a global background picture for encoder reference. In addition, a surveillance video oriented rate–distortion optimization algorithm is developed to facilitate and optimize the compression of surveillance videos. The proposed algorithm can improve the video compression efficiency and effectively support diverse video applications in smart city.

The remainder of the paper is organized as follows. Section 2 discusses background knowledge of data fusion and video compression technologies. Section 3 proposes the background picture construction scheme. Section 4 presents the encoder architecture with background picture long-term reference and quantization parameter selection. Section 5 describes the surveillance rate–distortion optimization algorithm. Section 6 shows the experimental results and analysis. Finally, Section 7 summarizes this work with a conclusion.

2. Data fusion and video compression

2.1. IoT and video big data fusion

Since the IoT embedded in smart city produces quantities and various kinds of data, the collaborating and sharing of the generated data would be a considerable issue [7]. Data Fusion, which means the fusion of different forms and types of data, is a typical and useful solution that is widely adopted [8]. Combining data from multiple sensors and associating databases, data fusion techniques could be performed more preciseness and specific presumptions than the application of an individual sensor [9]. With an immediate big data fusion and analysis system, the ubiquitous environments such as IoT and smart city would acquire efficiency, reliability and accuracy in service quality and management [10].

In smart city applications, video data generated from the IoT poses some challenges. The distributed cameras in modern cities gather video sources in different scenarios, even in the same restricted area. Specifically, surveillance video content from different cameras for the same scene poses a challenge to the unified analysis and application. For instance, three cameras are distributed in three successive streets, where the environments are completely distinct. Some video analysis applications, like object tracking, will be evidently affected by the diversity of scenarios. The tracking or detecting precision is decreased, while the video data is full of redundant contents. An effective and reasonable fusion scheme would filter the video data, reduce the unnecessary information and improve the precision of content analysis applications in smart city.

Data fusion methods can be simply divided into three categories according to the mathematical attributes:

A. Probability-based methods including Bayesian analysis, statistics, and recursive operators [11].

B. Artificial Intelligence (AI) based techniques including classical machine learning, fuzzy Logic, Artificial neural networks (ANN) and genetic evaluation [12].

C. Theory of Evidence based Data Fusion methods [13].

These typical schemes are commonly applied and are already in use in traditional data fusion. However, the video data size from urban cities is tremendous. In a mid-size city, e.g. Chengdu in China, the camera amount setup by social organizations or public security system is about 480 thousand, which would generate Petabyte-level video data per day. This enormous unstructured data might lead to troubles in analysis and transmission. In addition, current big data systems have difficulty in handling such size of data. There exist two approaches to cope with such issues, the one is upgrading the hardware performance, and the other is to reduce and reorganize the data itself.

To effectively manage the huge amount of data, a three-level fusion system is shown in Fig. 2. After the video data is acquired, data fusion subsystem would receive data from several sensors in a local region. The video data combined with structured sensor data are handled with the simplified data fusion sub-process. The typical fusion schemes, such as Bayesian analysis, fuzzy logic and evidence based method would be adopted to reduce the data redundancy. Then the preliminary compressed and fused data would be transmitted to the data fusion cloud center [14–16] to be further precisely fused and analyzed. After that, the fused data or features generated from fusion cloud center would be classified and analyzed to produce fused decisions. Finally, the decisions are made to support all kinds of applications. For this system, a critical method to improve efficiency is to compress the cluster data [17–19] while keeping necessary content information.

The video data captured from the IoT in smart city is mainly contained of surveillance video. Since there is strong temporal relevance between successive pictures in the surveillance video, this feature distinctly differs the TV programs, live broadcasting and films. Thus, this work focuses on the background construction and optimization scheme, which is able to deduce the video data size and improve compression performance to effectively used in video data fusion.

2.2. Processing technologies for video big data

The advanced technology of High Efficiency Video Coding (HEVC/H.265) has become the state-of-the-art video compression standard since 2012. There are several optimized techniques in HEVC/H.265, including coding tree units (CTUs), the sampled representations of pictures, partitioning block and units by quadtree structures, slices, and tiles, intra picture and inter picture predictions [20], range extensions, and scalable extensions. Besides, HEVC/H.265 employs the hierarchical coding structure (HCS) [21]. HCS divides each picture into several layers according to its chronological order, using the length of a group of pictures (GOP) and its present order count (POC) to decide the index of each layer. The reference picture management is organized to improve coding efficiency and reduce distortion. In HEVC/H.265, HCS is used for bit-distribution, quality control, coding optimization, and the calculation of the quantization parameter(QP) [22]. With the help of these methods, HEVC/H.265 achieves the better efficiency than the previous coding standards [23].

In order to gain the trade-off between the coding performance and the quality for the surveillance source, researchers propose the background modeling and long-term reference technique [24]. Background modeling algorithms can separate the background and foreground of a picture, and produce a picture that contains the background content only. This background picture can be used as the long-term reference to encode the ensuing pictures. During the coding process, the encoder merely compresses the difference between the current picture and the background picture, then the bits are saved and the temporal redundancy of successive pictures is decreased.

Several techniques have been developed for the background modeling, such as normal distribution [25], Gaussian mixture model [26], image feature model [27], Poisson model [28] and non-parametric techniques [29]. Ding [30] proposed a background construction scheme based on motion compensation. Paul et al. [31] directly chose the latest picture to update the background picture, which is simple but imprecise. Schemes based on block substitution and coding unit classification [32]. F. Chen [33] proposed a probability based background construction scheme using reconstructed blocks during coding process to consider both the spatial and temporal relevance. Unfortunately, such strategy needs to modify the decoder structure, which is not available in the common decoder, resulting in limited application scenarios.

In 2014, X. Zhang et al. [34] proposed a different background modeling based on the HPS optimization (BHO) technique, which uses the statistics of temporal metrics and adaptive pixel averaging to construct the background picture. For the surveillance source with large temporal redundancy, BHO is proved to be effective. However, it has the following limitations.

1. BHO works in the pixel level and ignores the spatial relevance between blocks.
2. BHO processes the luminance and chrominance independently, taking no accounts of the potential relevance.
3. BHO uses the average of single pixel which may degrade the sensitivity to the content of the residual information.
4. BHO adopts the mean threshold which may retain the white noise in the picture.

In HEVC, under the initial QP parameter, the QP value is increased with the index of layer in each picture, and the QP difference between the successive layers is measured by the QP offset. However, there exists a strong temporal correlation among pictures in the surveillance video, and this feature can be exploited to enhance QP value allocation. For example, for surveillance video, if the inflection of a group of pictures (GOP) is smooth, the pictures in this GOP are highly correlated, and the first picture in this GOP will be referenced several times, so the QP value for the first picture can be smaller to ensure the quality and reduce the distortion of this GOP. Hence, the QP allocation strategy can be optimized by considering such features from the video source to achieve the global rate–distortion optimization (RDO) [35]. In our previous work, we proposed a Lagrangian multiplier adaptation algorithm

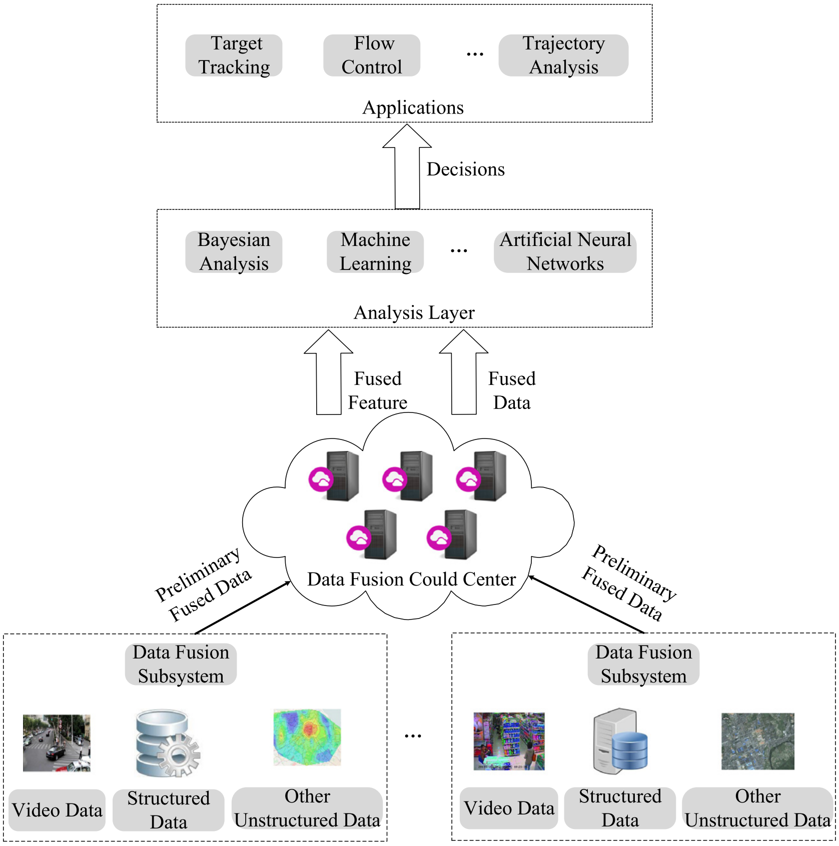


Fig. 2. Video data fusion system.

based on the layer-based fixed-QP (LF-QP) decision strategy for enhancing rate–distortion optimization performance [36]. However, LF-QP decision strategy used in HEVC ignores the content of picture and receives no feedback from the coding results, leading to poor encoding performance for videos with heterogeneous bit rates and resolutions.

In this work, we propose a block-level background modeling (BBM) algorithm based on the boundary match to overcome the aforementioned limitations and improve the coding efficiency. The BBM algorithm uses the residual gradient as the temporal information to distinguish the background blocks. While considering the boundary difference, the proposed BBM algorithm using the weighted average of pixel temporal value to smooth the background pixels. To optimize the objective and subjective quality of the encoded video, we also propose a surveillance rate–distortion optimization (SRDO) algorithm, which updates the largest coding unit (LCU) level Lagrange multiplier (λ) and picture level quantization parameter (QP) according to the motion intensity of pictures.

3. Background picture construction

In this section, the details of the proposed BBM algorithm are described, which consists of components such as residual gradient, edge comparison and pixel smoothness.

3.1. Residual gradient

The residuals of Y, U and V, which reflect the temporal relevance between the neighboring pictures, can be used to build a proper background picture. Generally, the residual of Y, U and V components for a certain block in the picture with POC t can be calculated as Eq. (1).

$$C^{\mathrm{D}}_{t,p,q}(i,j) = C_{t,p,q}(i,j) - C_{t-1,p,q}(i,j). \tag{1}$$

In Eq. (1), $C^{\mathrm{D}}_{t,p,q}(i,j)$ is the residual value. C represents one of the YUV components. $C_{t,p,q}(i,j)$ is the pixel value at picture t; i and j denote the horizontal and vertical index of pixel; p and q stand for the position of a block in a picture. With the residual formulation, the residual gradient for a block can be computed by

$$\nabla C_{t,p,q}(i,j) = \sqrt{\begin{array}{l}(C^{\mathrm{D}}_{t,p,q}(i,j) - C^{\mathrm{D}}_{t,p,q}(i,j+1))^2 + \\ (C^{\mathrm{D}}_{t,p,q}(i,j) - C^{\mathrm{D}}_{t,p,q}(i+1,j))^2\end{array}}, \tag{2}$$

where $\nabla C_{t,p,q}(i,j)$ denotes the L2-norm residual gradient.

3.2. Block classification

With the residual gradient derived by Eq. (2), one can decide the type of a block. First, the mean value of the residual gradient for a block with index (p,q) is

$$\mu(c) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\nabla C_{t,p,q}(i,j), \tag{3}$$

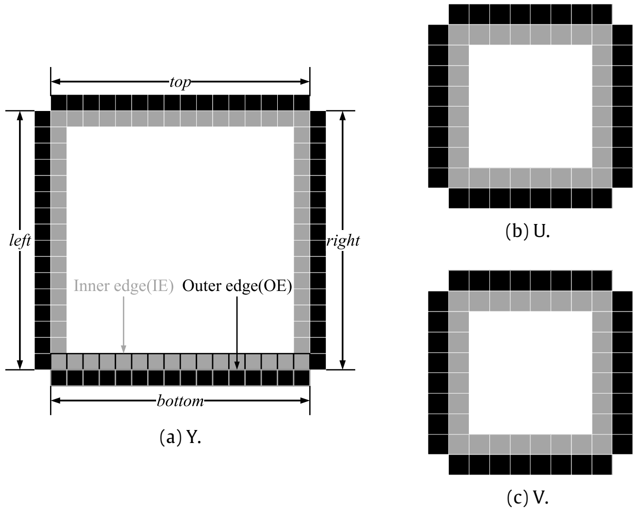
where n is the size of a block. Then, the variance is expressed by Eq. (4).

$$\sigma(c) = \sqrt{\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\nabla C_{t,p,q}(i,j) - \mu(c))^2}. \tag{4}$$

Then, a logarithm function for the residual gradient can be obtained as Eq. (5),

$$f(c) = \ln(1 + \frac{\sigma(c)}{\mu(c)+1}). \tag{5}$$

Eq. (5) is used as a description for the motion feature. Due to the 4:2:0 proportion of Y, U and V component, the sampling of U and V



(a) Y.

(b) U.

(c) V.

Fig. 3. Definition of inner and outer edge of a block.

is half in height and width of Y component. Thus, the weigh factor of U and V component is set to 1/4 when estimating the motion feature of a block, which is generated from Eq. (6).

$$\nabla = f(\mathrm{Y}) + \frac{1}{4}\cdot f(\mathrm{U}) + \frac{1}{4}\cdot f(\mathrm{V}). \tag{6}$$

The motion gradient ∇ reflects the temporal motion information for a block. If the content of a block varies slowly, the value of ∇ is likely to be 0. Correspondingly, when the block content varies rapidly, ∇ would evidently increase.

The block can then be classified to three different types according to the value of ∇, upper threshold ω and lower threshold ϖ:

- Ignorance type: if $\nabla > \omega$, then the block is ignored without any comparison and substitution.
- Replacement type: if $\nabla < \varpi$, then the block of background is strictly substituted.
- Detection type: if $\varpi \leq \nabla \leq \omega$, then the block type needs to be determined after the boundary comparison.

The experimental upper threshold ω and lower threshold ϖ are 1.15 and 0.50, which could achieve reasonable background construction performance.

3.3. Detection type with minimal edge error

The background picture is updated according to its type. As for the detection blocks, replace strategy can take advantage of the boundary difference. The boundaries in different directions are called as edges for a square block. An example of edges of a block is shown in Fig. 3. In Fig. 3(a), the block of Y component contains two types of edges, which are called the outer-edge (OE) and the inner-edge (IE). The IEs are made up of the *top*, *bottom*, *left* and *right* edge pixels on the block, while the OEs are the edge pixels on the four neighbor blocks. The same illustration can be applied to U and V components. For a block with detection type, we calculate the pixel difference between its four OEs and IEs to decide whether it should be replaced or not.

Here, we define $E_C(pos, p, q)$ as the absolute difference of IEs and OEs. The variable p and q are the horizontal and vertical index of a block, and *pos* denotes the direction label from the set $\mathrm{D}_{\mathrm{set}}$:

$$\mathrm{D}_{\mathrm{set}} = \{top, bottom, left, right\} \tag{7}$$

For a block with coordination index (p, q), the absolute difference of edges is calculated by Eq. (8).

$$E_C(p, q, d) = \frac{1}{n}\sum_{k=1}^{n} |C_{\mathrm{IE}}(p, q, d, k) - C_{\mathrm{OE}}(p, q, d, k)|, \tag{8}$$

where $C_{\mathrm{IE}}(p, q, d, k)$ denotes the IE pixel value of C component at direction *pos*, $C_{\mathrm{OE}}(p, q, d, k)$ represents the corresponding OE pixel value. Then, the total edge difference $E(p, q)$ of a block is generated by Eq. (9).

$$E(p, q) = \sum_{d\in \mathrm{D}_{\mathrm{set}}} \sum_{C=\mathrm{Y,U,V}} E_C(p, q, d). \tag{9}$$

With Eq. (9), the total edge difference of current block $E_t(p, q)$ at picture t and that of the corresponding block $E_{\mathrm{GB}}(p, q)$ at the global background picture can be determined. The smaller edge difference reflects the smaller disparity between the edge of a block and its neighbors, which indicates that such a block with smaller difference is able to match the background picture more properly. Therefore, we use $E_t(p, q)$ to compare $E_{\mathrm{GB}}(p, q)$, then the block with smaller edge difference is substituted at the identical position of the background picture.

3.4. Pixel smoothness

Unlike the video shotted by the high definition camera, the inferior device brings scene fluctuations and poor picture quality. Hence, in the surveillance video, the pixel intensity of luminance fluctuates frequently. This leads to the quality diversity of successive pictures, if the background picture is constructed based on such block substitution. To precisely establish the background picture, pixel temporal smoothness is an effective approach.

Linear smoothness, Gaussian smoothness and average smoothness are the classical smoothness functions. The linear and Gaussian smoothness functions require full data set, and the average smoothness function is suitable for diverse data set. In the background build processing, there is limited modeling data, and the data can be only acquired by the current and the latest coded pictures. Thus, this work adopts the simple but effective average pixel smoothness function.

The micro smoothness is the function that adopts the weight average of temporal value, and it is applied to pixels of the current background. Note that the temporal value of the background pixel varies respect to the foreground pixel in the surveillance video. The background and foreground pixels can be classified by the difference between two successive pixels. The smoothing result of micro pixels is affected by the choice of pixel classification threshold.

After the type of a block is determined, the average residual of the pixel at the corner is calculated as follow:

$$\begin{aligned}\rho = {} & \frac{1}{N}\sum_{t=1}^{N}\sum_{i=1}^{n} |C_t(i, 0) - C_0(i, 0)| \\ & + \frac{1}{N}\sum_{j=1}^{n} |C_t(0, j) - C_0(0, j)|,\end{aligned} \tag{10}$$

where $C_t(i, j)$ is the current pixel value at position (i, j), and N is the block substitution length. Then, the difference of background and current pixel is compared with the threshold ρ. Only those pixels with the difference smaller than ρ could be smoothed. The pixel smooth value with coordination (i, j) at background can be

$$P_t(i, j) = \begin{cases} C_t(i, j) & , t = t^* \\ \dfrac{P_{t-1}(i, j) \times (t - t^*) + C_t(i, j)}{t - t^* + 1} & , |P_t(i, j) - C_t(i, j)| < \rho \\ P_{t-1}(i, j) & , |P_t(i, j) - C_t(i, j)| \geqslant \rho \end{cases} \tag{11}$$

where $P_t(i, j)$ is the background pixel value at current POC t, t^* is the start POC of the pixel smoothness process. Such process only takes on Y-component, as the temporal pixel fluctuation of U and V component is ignorable.

3.5. Background construction algorithm

This section describes the algorithm of the proposed BBM, as shown in Algorithm 1. In Algorithm 1, line 7 to line 21 represents the background picture construction process, line 22 to line 33 illustrate the pixel smoothness process.

Algorithm 1 Block-Level Level Background Matching (BBM) Algorithm

```
Input: The YUV video sequence;
 1: Model picture count $\phi = 120$;
 2: Smooth picture count $\theta = 30$;
 3: Picture width $w$, picture height $h$;
 4: Vertical block count $H$, Horizontal block count $W$;
 5: Classification threshold $\varpi = 1$ and $\omega = \sqrt{2}$;
Output: Background picture;
 6: Begin
 7:   for $i = 1 : \phi - \theta$ do
 8:     for $j = 1 : H$ do
 9:       for $k = 1 : W$ do Calculate:
10:         A. The residual gradients with Eq.(1) and (2);
11:         B. The mean value and variance of residual gradient by Eq.(3) and (4);
12:         C. The motion gradient $\nabla$ by Eq.(5) and (6);
13:         D. The edge difference $E_t$ and $E_{GB}$ with Eq.(8) and (9);
14:         if $(\nabla < \varpi)$ or $(\nabla < \omega$ and $E_t < E_{GB})$ then
15:           Substitute block at coordination $(j, k)$;
16:         else
17:           continue;
18:         end if
19:       end for
20:     end for
21:   end for
22:   for $i = \phi - \theta + 1 : \phi$ do
23:     for $j = 1 : H$ do
24:       for $k = 1 : W$ do
25:         Calculate the Smooth threshold $\rho$ by Eq.(10);
26:         if $(|P_i(j, k) - C_i(j, k)| < \rho)$ then
27:           Calculate background pixel value $P_i(j, k)$ with Eq.(11);
28:         else
29:           continue;
30:         end if
31:       end for
32:     end for
33:   end for
34: End
```

4. Encoder architecture with background picture long-term reference

The long-term reference model is essential to encode the source video with the background picture. This section discusses the architecture of the encoder based on the background picture and the long-term reference structure.

4.1. Overview of background based encoder

The proposed hybrid encoder architecture with background picture and long-term reference is shown in Fig. 4. The novel architecture adopts an extra constructed background picture as the input of long-term reference picture, which is ignored by the typical HEVC coding structure. In the surveillance video, the background in video stream does not change, so the background picture needs to be encoded only once, and this reduces the temporal redundancy deeply and saves the coding bits.

4.2. Long-term reference structure

Fig. 5 shows different applications of the long-term reference structure. The rectangles in the figures denote the pictures and the arrows represent the reference relationship. L denotes the reference length, which is decided by the intra picture period. Fig. 5(a) adopts the first I picture as the long-term reference picture, while Fig. 5(b) uses the constructed background. The structure is similar to the typical Low-Delay (LD) reference profile. For the picture with picture order count (POC) L, its reference would be the picture with POC $L - 12$ and $L - 2$. In other words, one picture could refer to its former two neighbors. The difference between the classical LD and long-term structure is that the long-term picture will be referred by all the successive pictures during the coding process.

4.3. Hierarchical coding structure

Fig. 6 shows an example of the organization of the hierarchical structure with Low-delay (LD) and Random-Access (RA) coding. The curves between pictures denote the reference relationship. In Fig. 6(a), LD adopts the forward reference structure. There are 8 pictures in the GOP and the display order is the same as the coding order. However, as shown in Fig. 6(b), RA typically selects the binary pictures with the neighbor temporal sequence as the coding reference. Hence, the display order and the coding order are different in RA.

4.4. QP offset in hierarchical structure

HEVC introduces the hierarchical coding structure (HCS) as its novel feature, in which pictures are divided into several layers virtually based on their chronological order. The layer index is decided by the picture order and the length of current GOP. The value of QP offset is set by the layer index in advance.

In general, pictures with low layer index will be referenced by the high layer index pictures. Thus, the quality of those pictures with low layer index are more essential and their QP values should be small to reduce the distortion. Similarly, along with the increasing of the layer index, the QP value for that picture could be gradually decreased. H.264/AVC, H.264/SVC and HEVC/H.265 choose to simply decrease QP value by 1 for successive layers. In fact, the basic QP determination can be expressed by

$$QP_t = QP_{\text{ini}} + \Lambda + QP_{\text{offset}} \tag{12}$$

where QP_t denotes the QP value for Picture t, QP_{ini} indicates the initial QP value set by the user, the picture type modifier Λ is 0 for intra pictures and 1 otherwise in Eq. (12), the QP_{offset} represents the offset value of QP for Picture t, which can be calculated as in Eq. (13).

$$QP_{\text{offset}} = \text{layer}(L, t) \tag{13}$$

In Eq. (13), the function layer(L, t) denotes the layer index based on the input of the GOP length L and the POC t, which can be represented in the following formula,

$$\begin{aligned} &\text{layer}(L, t) = \min\{\log_2 L - k\} \\ &\text{s.t.}\;\; L - t \bmod L = (L - t \bmod L) \gg k \ll k, \\ &\qquad k \in [0, \log_2 L] \end{aligned} \tag{14}$$

where min{} is used to select the minimum value from a set, k is the layer index, symbol $\ll$ and $\gg$ denote binary zero-fill left shift and right shift, respectively.

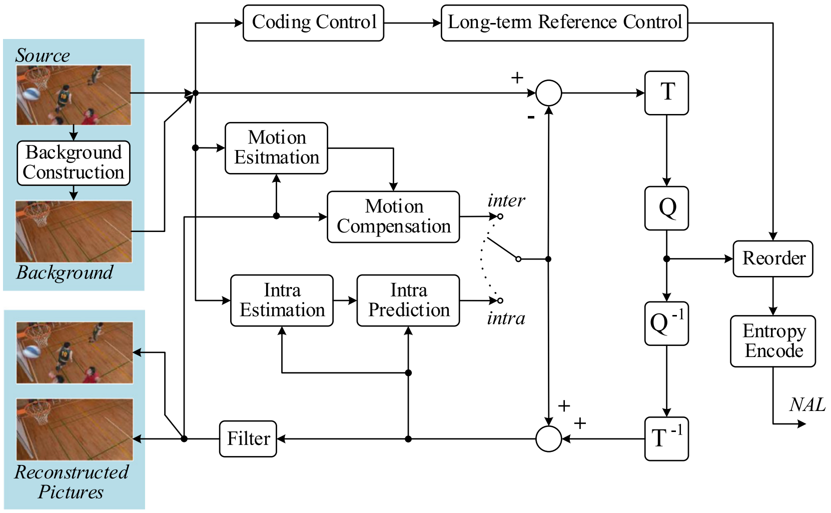
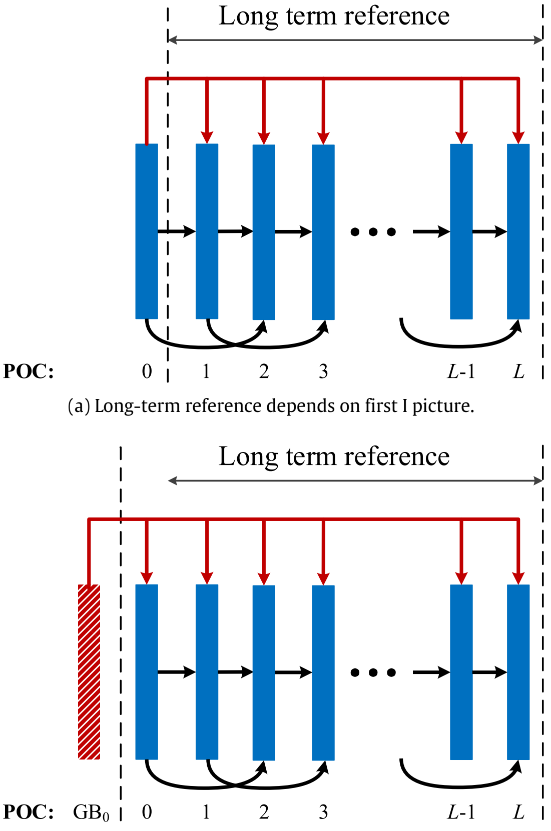


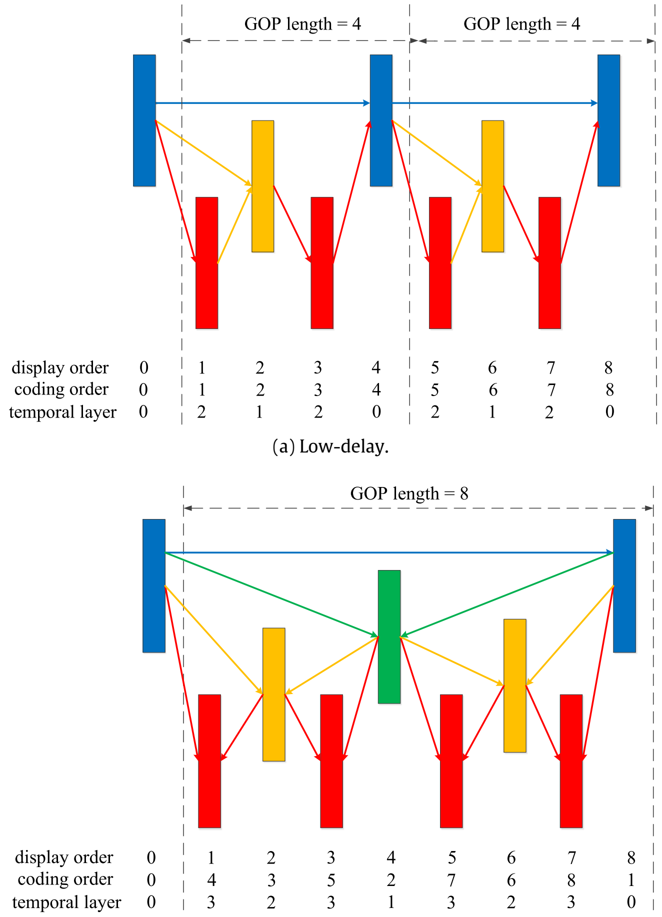
Fig. 4. Hybrid encoder architecture with background picture reference.



(a) Long-term reference depends on first I picture.

(b) Long-term reference depends on the constructed background.

Fig. 5. Different types of long-term reference structure.



display order	0	1	2	3	4	5	6	7	8
coding order	0	1	2	3	4	5	6	7	8
temporal layer	0	2	1	2	0	2	1	2	0

(a) Low-delay.

display order	0	1	2	3	4	5	6	7	8
coding order	0	4	3	5	2	7	6	8	1
temporal layer	0	3	2	3	1	3	2	3	0

(b) Random-access.

Fig. 6. Hierarchical coding structure with four temporal layers.

The final QP value can be computed by the combination of Eqs. (12)–(14). Since this QP decision strategy is not adaptable, X. Li et al. [37] introduced a QP adaptation approach with an offset value using the error propagation and distortion from the inter-frame dependency. In [38,39], the authors propose an adaptive QP calculation method based on the temporal pumping artifact to take the temporal effect between pictures into consideration. However, the above works ignore the motion characteristics of the source and lack effective adaptability in scenarios with different bit-rates, resolutions, or frame rates during the coding process. T. Zhao et al. [40] proposed a complex QP adaptation scheme which is deducted from traditional rate distortion model. In this work, we take the advantages of the motion characteristics and the temporal redundancy information to help allocate the QP values dynamically.

In the hierarchical structure coding, the higher layer has a larger amount of pictures, which deserve better QP values to ensure better encoding quality. Hence, we propose the following equation to calculate QP values,

$$QP_t = QP_{\text{ini}} + \max\{\log_2 L\} + \Lambda - QP_{\text{offset}} \tag{15}$$

where max{} is used to select the maximum value from a set, L is a variable, which denotes the GOP length, and QP_{offset} is a non-negative number used for feeding back the rangeability of QP, which can be collected from the layer index and picture content variety.

Unlike the top-down QP allocation model in Eq. (12), Eq. (15) adopts a bottom-up QP allocation strategy, which can take better advantages of the rich picture contents from the higher layer. Note that QP_{offset} can be presented as,

$$QP_{\text{offset}} = \varphi(t) \cdot \text{layer}(L, t) \tag{16}$$

where the function $\varphi(t)$ is the feedback of the temporal redundancy for Picture t. More about this feedback function $\varphi(t)$ will be discussed in the subsection 5.2.

According to the HCS rule, a half of all the pictures belong to the highest layer and are allocated with the same QP value in HEVC. This QP is the biggest QP in the coding process. In other words, the highest layer has the biggest QP. Furthermore, the bigger allocated QP value is, the worse quality distortion has. So all the picture in the highest layer could almost achieve the same distortion under the biggest QP value by Eqs. (15) and (16).

5. Surveillance video coding optimization

Since the constructed background picture is different from the pictures in common reference structure, the QP and λ values could further optimize the coding performance.

5.1. Evaluation of distortion

The moving contents of a picture can be partly reflected by the distortion between pictures in the surveillance video. The picture with more distinct moving parts respects that its distortion would be amplified and result in the loss of objective quality. Hence, the picture with more fluctuating should be allocated with bigger QP value than the less fluctuating pictures.

To evaluate whether a picture is fluctuated or static, we choose the mean squared error (MSE) to evaluate the motion compensated distortion (MCD) between pictures.

For a LCU with coordinate (r, s) in picture t, the background distortion is calculated by Eq. (17).

$$M_t(r, s) = \|B_t(r, s) - \widehat{G}(r, s)\|, \tag{17}$$

where $B_t(r, s)$ is the block motion estimation value of the LCU with coordination (r, s) in the picture t, $\widehat{G}(r, s)$ is the value from the background picture at the corresponding coordinate. Then, the MCD can be deducted from Eq. (18).

$$C_t^{\text{M}}(r, s) = \min\{M_t(r, s), \|B_t(r, s) - B_{t-1}(r, s)\|\}. \tag{18}$$

5.2. Picture level QP determination

Without loss of generality, we can use Eq. (19) to deduce the mean value of MCD for each LCU to represent the picture level MCD.

$$\overline{C}^{\text{M}}(t) = \frac{\sum_{r=1}^{M}\sum_{s=1}^{N} C_t^{\text{M}}(r, s)}{M \cdot N}, \tag{19}$$

In Eq. (19), $\overline{C}^{\text{M}}(t)$ is the MCD for picture t; n denotes the number of LCUs in the picture; and C_b^{M} represents MCD value for LCU b.

For picture i, let m_i be the feedback of the picture level temporal redundancy, which can be calculated in Eq. (20).

$$m_t = \ln \overline{C}^{\text{M}}(t) \tag{20}$$

To estimate the temporal redundancy of Picture t, the mean value of m can be obtained via pictures within a window as follows,

$$\mu_t = \frac{1}{W} \cdot \sum_{w=1}^{W} m_{t-w} \tag{21}$$

where μ_t is the mean value of m, and W is the window size.

Similarly, the variance of m_t can be calculated as Eq. (22).

$$\sigma_t = \sqrt{\frac{1}{\omega} \cdot \sum_{w=1}^{\omega}(m_{t-w} - \mu_t)^2} \tag{22}$$

Accordingly, the coefficient for the QP value of picture t is decided by Eqs. (23) and (24).

$$\varphi(t) = \kappa \cdot \log_2 \frac{\mu_t}{\sigma_t} \tag{23}$$

$$\begin{aligned} QP_t &= QP_{\text{ini}} + \max\{\log_2 L\} + \Lambda \\ &\quad - \kappa \cdot \log_2 \frac{\mu_t}{\sigma_t} \cdot \text{layer}(L, t) \end{aligned} \tag{24}$$

With the reallocated QP from Eq. (24), the QP for each picture t is accurately determined, which would benefit the coding process to achieve bit-rate saving.

5.3. LCU level λ adjustment

The picture with intensive motion change costs more bits. To enhance the coding result and visual quality, it is necessary to allocate the bits to LCU reasonably. Thus a proper QP and λ for each LCU in a picture should be allocated. The mean value of motion estimation $\overline{M_t}$ for the current picture is calculated by Eq. (25).

$$\overline{M_t} = \frac{\sum_{r=1}^{M}\sum_{s=1}^{N} M_t(r, s)}{M \cdot N}, \tag{25}$$

where M is the horizontal LCU count and N is the vertical LCU count. Then, the motion factor for $M_t(r, s)$ is determined by Eq. (26).

$$f_t(r, s) = \frac{\overline{M_t} + \varepsilon}{M_t(r, s) + \varepsilon}, \tag{26}$$

where $f_m(r, s)$ is the motion factor and ε is the minimum value that prevents division by zero. The motion factor partly reflects the variety of moving content. With Eq. (26) the adjustment of λ can be as the following:

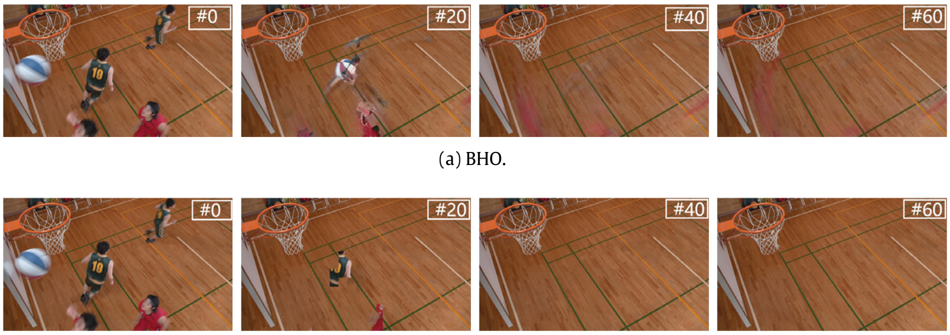
$$\lambda_t^{*}(r, s) = \lambda_{\text{sys}}(r, s) \cdot f_t(r, s), \tag{27}$$

where λ_{sys} is the global value from configuration.

Eq. (27) is taking the motion characteristics into consideration for λ allocation at each LCU, which can improve the coding quality for moving content in the surveillance source.

5.4. Surveillance video optimization algorithm

In this section, **Algorithm 2** gives the workflow of the entire coding process of the proposed SRDO algorithm, which can efficiently adjust the QP and λ. The calculation of parameters is not compute-intensive when compared with the coding process, hence the temporal complexity remains the same level.



(a) BHO.

(b) BBM.

Fig. 7. Background construction stages of *BasketballDrill*.

Algorithm 2 Rate–distortion optimization algorithm

```
Input: : The YUV video sequence;
  Model picture count $\phi = 120$;
  The background YUV picture;
  Total coding picture amount $n$;
  Horizontal LCU count $M$, vertical LCU count $N$;
  Background picture generated by Algorithm 1;
Output: : Bit stream;
  Begin
    for $i = 1 : \phi$ do
      Encode picture $i$ and construct the background picture;
    end for
    Configure long-term reference mode;
    for $i = \phi + 1 : n$ do
      Encode picture $i$ using background picture as reference;
      Determine the block MSE from Eqs. (17) and (18)
      Calculate the ratio value from Eqs. (19), (20), (21), (22) and (23);
      Compute QP value for next picture by Eq. (24);
      for $j = 1 : M$ do
        for $k = 1 : N$ do
          Calculate the motion factor by Eqs. (25) and (26);
          Compute $\lambda$ using Eq. (27);
          Encode LCU $(j, k)$ with adjusted $\lambda$;
        end for
      end for
    end for
  End
```

Table 1
Test sequences.

Sequence name	Resolution	Frames per second
BasketballDrill	832 × 480	50
BasketballDrillText	832 × 480	50
Crossroad	720 × 576	30
Office	720 × 576	30
Overbridge	720 × 576	30
FourPeople	1280 × 720	60
KristenAndSara	1280 × 720	60
vidyo1	1280 × 720	60
vidyo3	1280 × 720	60
Intersection	16000 × 1200	30
Mainroad	1600 × 1200	30
Traffic_crop	2560 × 1600	30

5.5. Complexity analysis

The complexity of the SRDO and BBM algorithm are both low. The complexity for background construction part and λ adjustment is $O(n^2)$ for a picture, but for the encoder, it requires thousands of MSE, discrete cosine transform (DCT) and other $O(n^2)$ or $O(n^3)$ algorithms. Therefore, the extra cost for SRDO and BBM could be ignored compared with the encoding process.

6. Experiment and analysis

To evaluate the performance of the proposed BBM algorithm, we integrated BBM into HM16.15 [41]. Extensive experiments are carried out under the Low-Delay (LD) structures. The test sequences are separately selected from the common test condition (CTC) [42] and audio video coding standard(AVS) surveillance sequence, which are listed in Table 1. We compare the performance of BBM and SRDO algorithms with that of BHO algorithm.

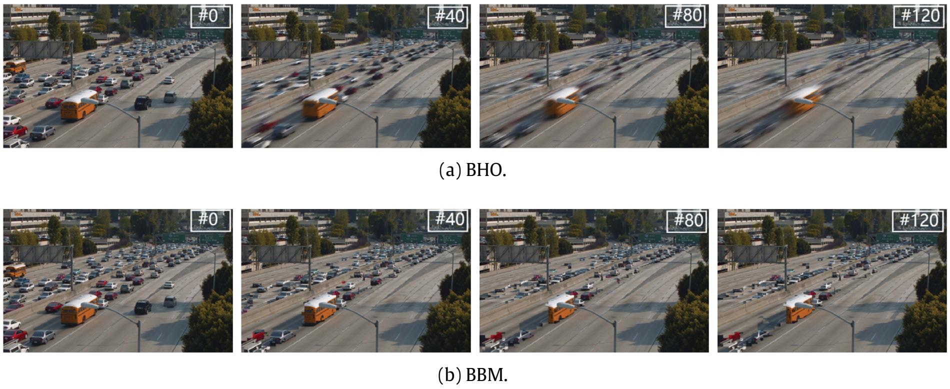
6.1. Background modeling

The background modeling processes are shown in Figs. 7 and 8 from sequence *BasketballDrill* and *Traffic*, respectively. Each figure contains two lines of pictures, the first line shows the test results of BHO, and the second line is the result of our proposed BBM algorithm. In each picture, the number in the top-right corner represents the iteration count. For *BasketballDrill*, the comparison is made every 20 times of iteration and for *Traffic*, the same comparison is made every 40 times. Fig. 7 shows that BBM algorithm adopts completely different strategy compared from the BHO algorithm. After 40 times of iteration, BBM algorithm reaches the smooth background, but there are still some dazzle lights after 60 times of iteration in the result of the BHO algorithm. In addition, the subjective quality of encoded picture by BBM is better than BHO algorithm.

One of the blocks replace processing is shown in Fig. 9. Fig. 9(a) denotes a part of the background picture after 12 iterations and Fig. 9(e) is from the 13th iterations. As Fig. 9 shows, the outlined block is replaced according to the difference of Y, U and V components at IEs and OEs. The differences are significant in Fig. 9(b) and less evident in Fig. 9(f), which denotes that the substituted block after 13 iterations in the training are more likely to be a background block. The replacement also maintains reasonable subjective quality, as the replaced block is a floor block, which looks like a static background compared to the foot block before replacement.

6.2. Low-delay coding performance

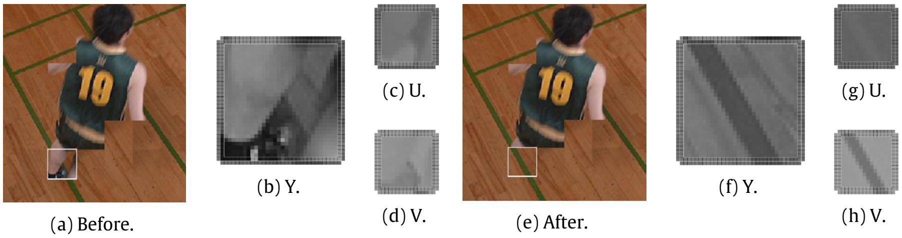
Tables 2 and 3 are the BD-Rate result of BBM and BBM with SRDO under LD configuration, respectively. The BD-Rate can represent the bit saving versus anchor under the same objective quality. From Table 2, it can be seen that both BBM and BHO achieves BD-Rate gain compared with anchor for all the sequence under LD configuration. The average BD-Rate gain for BHO is 6.03%, while that of the proposed BBM is 7.01%. In Table 3, the average BD-rate gain rises up to 16.25%, which is also better than BHO. Note that



(a) BHO.

(b) BBM.

Fig. 8. Background construction stages of *Traffic*.



(a) Before.

(b) Y.

(c) U.

(d) V.

(e) After.

(f) Y.

(g) U.

(h) V.

Fig. 9. Block substitution of *BasketballDrill* at the 13th iteration.

Table 2
BD-rate comparison on BBM.

	Long-term vs Anchor			BHO vs Anchor			BBM vs Anchor		
	Y	U	V	Y	U	V	Y	U	V
BasketballDrill	−11.6%	−15.2%	−10.3%	−9.40%	−10.74%	−6.15%	−9.57%	−13.44%	−5.53%
720 × 576	−1.93%	−7.99%	−8.62%	−6.03%	−8.62%	−9.71%	−7.06%	−11.63%	−12.36%
1280 × 720	−2.58%	−2.47%	−2.98%	−3.04%	−1.33%	−2.29%	−3.71%	−2.39%	−1.95%
1600 × 1200	−5.44%	−1.36%	−2.02%	−12.22%	−11.77%	−14.97%	−13.54%	−7.27%	−9.02%
2560 × 1600	−3.66%	−3.41%	−3.36%	−1.11%	2.00%	2.82%	−1.94%	−1.77%	−0.99%
Overall	−4.21%	−5.56%	−5.28%	−6.03%	−6.08%	−6.48%	−7.01%	−7.31%	−6.25%

Table 3
BD-rate comparison on BBM with SRDO.

	Long-term vs Anchor			BHO vs Anchor			BBM vs Anchor		
	Y	U	V	Y	U	V	Y	U	V
832 × 480	−18.89%	−28.18%	−23.20%	−19.41%	−25.96%	−21.35%	−19.18%	−27.56%	−19.51%
720 × 576	−5.54%	−29.29%	−28.96%	−11.86%	−30.89%	−30.40%	−13.14%	−33.48%	−34.12%
1280 × 720	−8.33%	−19.58%	−20.61%	−9.66%	−18.43%	−20.68%	−10.27%	−18.59%	−21.39%
1600 × 1200	−19.29%	−19.57%	−20.38%	−32.23%	−38.97%	−41.10%	−33.07%	−28.67%	−33.36%
2560 × 1600	−10.99%	−22.15%	−23.55%	−5.94%	−16.50%	−15.59%	−10.02%	−20.12%	−20.34%
Overall	−11.44%	−23.65%	−23.33%	−15.29%	−26.06%	−26.20%	−16.25%	−25.62%	−26.17%

the long-term reference can achieve better BD-rate gain for the surveillance sequences. The average gain on U and V component are also domesticate the advantages of the proposed algorithm.

As one type of the most important unstructured data, original video source is difficult to be directly adopted in information fusion to support all kinds of applications. The prominent BD-rate saving indicates the proposed algorithm can provide impactful compression efficiency of video data. This can benefit to the video fusion pretreatment and improve the performance of video transmission, storage and analysis.

7. Conclusion

In this work, we have proposed a new video compression scheme, namely, block-level boundary matching (BBM) algorithm to compress the surveillance video effectively. It is also collaborated with the surveillance rate–distortion optimization algorithm (SRDO) to further improve the compression performance. Experiment results have shown that BBM and SRDO can significantly enhance the compression rate of surveillance video. For video big data cluster, fusion and Internet of Things (IoT) in smart cities, the proposed scheme would require less storage requirement, strong coding efficiency and high transmission performance. It could be widely adopted by extensive video applications in smart city. For the future work, we will concentrate on improvement of modeling strategy, content-based optimization to achieve better video compression efficiency, explore approaches to meliorate the structure of video big data cluster and develop the application of video data fusion in smart city.

Acknowledgement

This work is supported by the foundation of science and technology department of Sichuan province (No. 2017JZ0031 and 2017HH0075).

References

[1] T. Anagnostopoulos, A. Zaslavsky, K. Kolomvatsos, A. Medvedev, P. Amirian, J. Morley, S. Hadjieftymiades, Challenges and opportunities of waste management in IoT-enabled smart cities: A survey, IEEE Trans. Sustain. Comput. 2 (3) (2017) 275–289. http://dx.doi.org/10.1109/TSUSC.2017.2691049.

[2] Z. Shao, J. Cai, Z. Wang, Smart monitoring cameras driven intelligent processing to big surveillance video data, IEEE Trans. Big Data PP (99) (2017). http://dx.doi.org/10.1109/TBDATA.2017.2715815. 1–1.

[3] B. Kang, D. Kim, H. Choo, Internet of everything: A large-scale autonomic IoT gateway, IEEE Trans. Multi-Scale Comput. Syst. 3 (3) (2017) 206–214. http://dx.doi.org/10.1109/TMSCS.2017.2705683.

[4] B. Cheng, G. Solmaz, F. Cirillo, E. Kovacs, K. Terasawa, A. Kitazawa, FogFlow: Easy programming of iot services over cloud and edges for smart cities, IEEE Internet Things J. PP (99) (2017). http://dx.doi.org/10.1109/JIOT.2017.2747214. 1–1.

[5] Y. Yang, Z. Ma, Y. Yang, F. Nie, H.T. Shen, Multitask spectral clustering by exploring intertask correlation, IEEE Trans. Cybernet. 45 (5) (2015) 1083–1094. http://dx.doi.org/10.1109/TCYB.2014.2344015.

[6] Data generation from new video surveillance cameras was 566 petabytes per day in 2015, https://storageservers.wordpress.com/2016/01/22/data-generation-from-new-video-surveillance-cameras-was-566-petabytes-per-day-in-2015.html,[EB/OL].

[7] R. Mehmood, F. Alam, N.N. Albogami, I. Katib, A. Albeshri, S.M. Altowaijri, UTiLearn: A personalised ubiquitous teaching and learning system for smart societies, IEEE Access 5 (2017) 2615–2635. http://dx.doi.org/10.1109/ACCESS.2017.2668840.

[8] J. Dong, D. Zhuang, Y. Huang, J. Fu, Advances in multi-sensor data fusion: Algorithms and applications, Sensors 9 (10) (2009) 7771.

[9] H.B. Mitchell, Multi-Sensor Data Fusion: An Introduction, Springer Publishing Company, Incorporated, 2007, p. 97108.

[10] F. Alam, R. Mehmood, I. Katib, N.N. Albogami, A. Albeshri, Data fusion and IoT for smart ubiquitous environments: A survey, IEEE Access 5 (2017) 9533–9554. http://dx.doi.org/10.1109/ACCESS.2017.2697839.

[11] Hall, L. David, Handbook of Multisensor Data Fusion, CRC Press, 2001, pp. 180–184.

[12] H.B. Azad, S. Mekhilef, V.G. Ganapathy, Long-term wind speed forecasting and general pattern recognition using neural networks, IEEE Trans. Sustain. Energy 5 (2) (2014) 546–553. http://dx.doi.org/10.1109/TSTE.2014.2300150.

[13] H. Janssen, W. Niehsen, Vehicle surround sensing based on information fusion of monocular video and digital map, in: IEEE Intelligent Vehicles Symposium, 2004, 2004, pp. 244–249. http://dx.doi.org/10.1109/IVS.2004.1336389.

[14] G. Sun, V. Chang, G. Yang, D. Liao, The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion, Inform. Sci. (2017).

[15] G. Sun, D. Liao, D. Zhao, Z. Sun, V. Chang, The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion, Future Gener. Comput. Syst. (2017).

[16] G. Sun, D. Liao, D. Zhao, Z. Xu, H. Yu, Live migration for multiple correlated virtual machines in cloud-based data centers, IEEE Trans. Serv. Comput. PP (99) (2017). http://dx.doi.org/10.1109/TSC.2015.2477825. 1–1.

[17] G. Sun, D. Liao, S. Bu, H. Yu, Z. Sun, V. Chang, The efficient framework and algorithm for provisioning evolving VDC in federated data centers, Future Gener. Comput. Syst. 73 (2016).

[18] G. Sun, D. Liao, V. Anand, D. Zhao, H. Yu, A new technique for efficient live migration of multiple virtual machines, Future Gener. Comput. Syst. 55 (C) (2016) 74–86.

[19] G. Sun, V. Anand, D. Liao, C. Lu, X. Zhang, N.H. Bao, Power-efficient provisioning for online virtual network requests in cloud-based data centers, IEEE Syst. J. 9 (2) (2015) 427–441. http://dx.doi.org/10.1109/JSYST.2013.2289584.

[20] G.J. Sullivan, J.R. Ohm, W.-J. Han, T. Wiegand, Overview of the high efficiency video coding (HEVC) Standard, IEEE Trans. Circuits Syst. Video Technol. 22 (12) (2012) 1649–1668. http://dx.doi.org/10.1109/TCSVT.2012.2221191.

[21] H. Schwarz, D. Marpe, T. Wiegand, Overview of the scalable video coding extension of the H.264/AVC standard, IEEE Trans. Circuits Syst. Video Technol. 17 (9) (2007) 1103–1120. http://dx.doi.org/10.1109/TCSVT.2007.905532.

[22] H.L. Cycon, T.C. Schmidt, M. Wahlisch, D. Marpe, M. Winken, A temporally scalable video codec and its applications to a video conferencing system with dynamic network adaption for mobiles, IEEE Trans. Consum. Electron. 57 (3) (2011) 1408–1415. http://dx.doi.org/10.1109/TCE.2011.6018901.

[23] D. Grois, D. Marpe, A. Mulayoff, B. Itzhaky, O. Hadar, Performance comparison of H.265/MPEG-HEVC, VP9, and H.264/MPEG-AVC encoders, in: Picture Coding Symposium (PCS), 2013, 2013, pp. 394–397. http://dx.doi.org/10.1109/PCS.2013.6737766.

[24] M. Paul, W. Lin, C.T. Lau, B.S. Lee, A long-term reference frame for hierarchical B-picture-based video coding, IEEE Trans. Circuits Syst. Video Technol. 24 (10) (2014) 1729–1742. http://dx.doi.org/10.1109/TCSVT.2014.2302555.

[25] N. Friedman, S. Russell, Image segmentation in video sequences: A probabilistic approach, Comput. Sci. (2013) 175–181.

[26] C. Stauffer, W.E.L. Grimson, Adaptive background mixture models for real-time tracking, in: Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Vol. 2, 1999, p. 252. http://dx.doi.org/10.1109/CVPR.1999.784637.

[27] B. Klare, S. Sarkar, Background subtraction in varying illuminations using an ensemble based on an enlarged feature set, in: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2009, pp. 66–73. http://dx.doi.org/10.1109/CVPRW.2009.5204078.

[28] E.C. Hall, R.M. Willett, Foreground and background reconstruction in poisson video, in: 2013 IEEE International Conference on Image Processing, 2013, pp. 2484–2488. http://dx.doi.org/10.1109/ICIP.2013.6738512.

[29] R. Zhang, W. Gong, A. Yaworski, M. Greenspan, Nonparametric on-line background generation for surveillance video, in: Proceedings of the 21st International Conference on Pattern Recognition, ICPR2012, 2012, pp. 1177–1180.

[30] R. Ding, Q. Dai, W. Xu, D. Zhu, H. Yin, Background-frame based motion compensation for video compression, in: Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on Vol. 2, 2004, pp. 1487–1490. http://dx.doi.org/10.1109/ICME.2004.1394518.

[31] M. Paul, W. Lin, C.T. Lau, B.s. Lee, Video coding using the most common frame in scene, in: 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, 2010, pp. 734–737. http://dx.doi.org/10.1109/ICASSP.2010.5495033.

[32] L. Ma, H. Qi, S. Zhu, S. Ma, A fast background model based surveillance video coding in hevc, in: 2014 IEEE Visual Communications and Image Processing Conference, 2014, pp. 237–240. http://dx.doi.org/10.1109/VCIP.2014.7051548.

[33] F. Chen, H. Li, L. Li, D. Liu, F. Wu, Block-composed background reference for high efficiency video coding, IEEE Trans. Circuits Syst. Video Technol. PP (99) (2016). http://dx.doi.org/10.1109/TCSVT.2016.2593599. 1–1.

[34] X. Zhang, Y. Tian, T. Huang, S. Dong, W. Gao, Optimizing the hierarchical prediction and coding in HEVC for surveillance and conference videos with background modeling, IEEE Trans. Image Process. 23 (10) (2014) 4511–4526. http://dx.doi.org/10.1109/TIP.2014.2352036.

[35] M. Winken, D. Marpe, T. Wiegand, Global and local rate-distortion optimization for lapped biorthogonal transform coding, in: Image Processing (ICIP), 2010 17th IEEE International Conference on, 2010, pp. 173–176. http://dx.doi.org/10.1109/ICIP.2010.5650962.

[36] S. Li, C. Zhu, Y. Gao, Y. Zhou, F. Dufaux, M.T. Sun, Lagrangian multiplier adaptation for rate-distortion optimization with inter-frame dependency, IEEE Trans. Circuits Syst. Video Technol. 26 (1) (2016) 117–129. http://dx.doi.org/10.1109/TCSVT.2015.2450131.

[37] X. Li, P. Amon, A. Hutter, A. Kaup, Model based analysis for quantization parameter cascading in hierarchical video coding, in: 16th IEEE International Conference on Image Processing, ICIP, 2009, pp. 3765–3768. http://dx.doi.org/10.1109/ICIP.2009.5414354.

[38] S. Wan, Y. Gong, F. Yang, Perception of temporal pumping artifact in video coding with the hierarchical prediction structure, in: IEEE International Conference on Multimedia and Expo, ICME, 2012, pp. 503–508. http://dx.doi.org/10.1109/ICME.2012.149.

[39] Y. Gong, S. Wan, F. Yang, H.R. Wu, B. Li, A frame level metric for just noticeable temporal pumping artifact in videos encoded with the hierarchical prediction structure, in: IEEE International Conference on Image Processing, ICIP, 2015, pp. 3034–3038. http://dx.doi.org/10.1109/ICIP.2015.7351360.

[40] T. Zhao, Z. Wang, C.W. Chen, Adaptive quantization parameter cascading in hevc hierarchical coding, IEEE Trans. Image Process. 25 (7) (2016) 2997–3009. http://dx.doi.org/10.1109/TIP.2016.2556941.

[41] HEVC reference software (HM16.7) https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.7/.

[42] F. Bossen, Common test conditions and software reference configurations, Document JCTVC-L1100, 2013.

Ling Tian received the B.S., M.S., and Ph.D. degrees from the school of computer science and engineering, University of Electronic Science and Technology of China (UESTC) in 2003, 2006 and 2010, respectively. She is currently an Associate Professor in UESTC. She had been a visiting scholar in Georgia State University (GSU) during 2013 in the United States. She has edited 2 books and holds over 10 China patents. She has contributed over 10 technology proposals to the standardizations such as China Audio and Video Standard (AVS) and China Cloud Computing standard. Her research interests include image/video coding, streaming and processing, visual perception and applications.

Hongyu Wang received the B.S. degree from University of Electronic Science and Technology of China (UESTC) in 2016. He is currently working toward B.S. degree in School of Computer Science, UESTC. His research interests include video coding, streaming and processing.

Yimin Zhou got his B.S., M.S., and Ph.D. degrees in computer science from the school computer science and engineering, University of Electronic Science and Technology of China (UESTC) in 2003, 2006, and 2009, respectively. He was a joint Ph.D. student with University of Central Arkansas (UCA), USA, from 2007 to 2009. He was a visiting scholar at Georgia State University (GSU) and University of California, Santa Barbara (UCSB), USA, in 2013 and 2017 separately. He has been a post Ph.D. major in signal processing since 2013. He is currently an Associate Professor in UESTC. His research interests include video coding, streaming and processing. He owns 5 granted parents and concerned the video encoding standards like HEVC, IVC and AVS. His 2 proposals were adopted to the MPEG and over 20 proposals adopted to the AVS.

Chengzong Peng received the B.S. degree from University of California, Irvine (UCI) in 2017. He is currently working as a research assistant in School of Computer Science, UESTC. His research interests include database and big data technologies.