Signal Processing: Image Communication 82 (2020) 115775

Contents lists available at ScienceDirect

Signal Processing: Image Communication

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

Background foreground boundary aware efficient motion search for surveillance videos☆

Tushar Shankar Shinde [a,b], Anil Kumar Tiwari [a], Weiyao Lin [b,*], Liquan Shen [c]

[a] Department of Electrical Engineering, Indian Institute of Technology Jodhpur, Jodhpur, 342037, India
[b] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China
[c] School of Communication and Information Engineering, Shanghai University, Shanghai, 200444, China

ARTICLE INFO

MSC:
41A05
41A10
65D05
65D17

Keywords:
Block matching
Surveillance video
Block classification
Search complexity
Directional motion

ABSTRACT

The huge amount of data in surveillance video coding demands high compression rates with lower computational requirements for efficient storage and archival. The motion estimation is a very time-consuming process in the traditional video coding framework, and hence reducing computational complexity is a pressing task, especially for surveillance videos. The presence of significant background proportion in surveillance videos makes its special case for coding. The existing surveillance video coding methods propose separate search mechanisms for background and foreground regions. However, they still suffer from misclassification and inefficient search strategies since it does not consider the inherent motion characteristics of the foreground regions. In this paper, a background-foreground-boundary aware block matching algorithm is proposed to exploit special characteristics of the surveillance videos. A novel three-step framework is proposed for boundary aware block matching process. For this, firstly, the blocks are categorized into three classes, namely, background, foreground, and boundary blocks. Secondly, the motion search is performed by employing different search strategies for each class. The zero-motion vector-based search is employed for background blocks. Whereas, to exploit fast and directional motion characteristics of the boundary and foreground blocks, the eight rotating uni-wing diamond search patterns are proposed. Thirdly, the speed-up is achieved through the novel region-based sub-sampled structure. The experimental results demonstrate that two to four times speed-up over existing methods can be achieved through this scheme while maintaining better matching accuracy.

1. Introduction and related work

Today, surveillance cameras play an important role in home-care, public safety and security, traffic management, business enhancement, and pedestrian detection [1]. The increase in a significant market for surveillance videos has brought a great challenge for the storage and maintenance of thousands of Terabytes data that would be produced per minute. Thus for long-time archival and real-time monitoring of surveillance videos, there is a great need for fast and efficient coding methods. The surveillance-based visual tracking also needs computationally efficient algorithms [2,3]. Considering these challenges and requirements, this work focuses on the computationally efficient surveillance video coding methods.

In the traditional video coding framework, the motion estimation component plays a crucial role in reducing the temporal redundancy. Hence, for better motion estimation, computationally expensive methods are employed. For example, in the Full Search (FS) motion estimation method, the best matching block corresponding to the block in the current frame is matched with the all candidate blocks present in the entire search window in the reference frame [4]. Although this process provided the best matching performance, it has severe computational disadvantages. Thus to increase the speed of the block matching process, several fast and efficient search algorithms are proposed in the literature. The algorithms can be categorized into two approaches: (1) reducing the total number of search points, and (2) reducing the number of computations at each search point.

For example, the H.264 standard [5] uses different variants of Diamond Search (DS) [6] and Hexagon Search [7,8] patterns, whereas the HEVC [9] employed Test Zone Search (TZS) [10] pattern for better motion estimation accuracy. Various attempts were made to reduce the computational complexity of TZS in [11–16]. The modified

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.image.2019.115775.

* Corresponding author.
E-mail address: wylin@sjtu.edu.cn (W. Lin).

https://doi.org/10.1016/j.image.2019.115775
Received 22 April 2019; Received in revised form 14 October 2019; Accepted 30 December 2019
Available online 7 January 2020

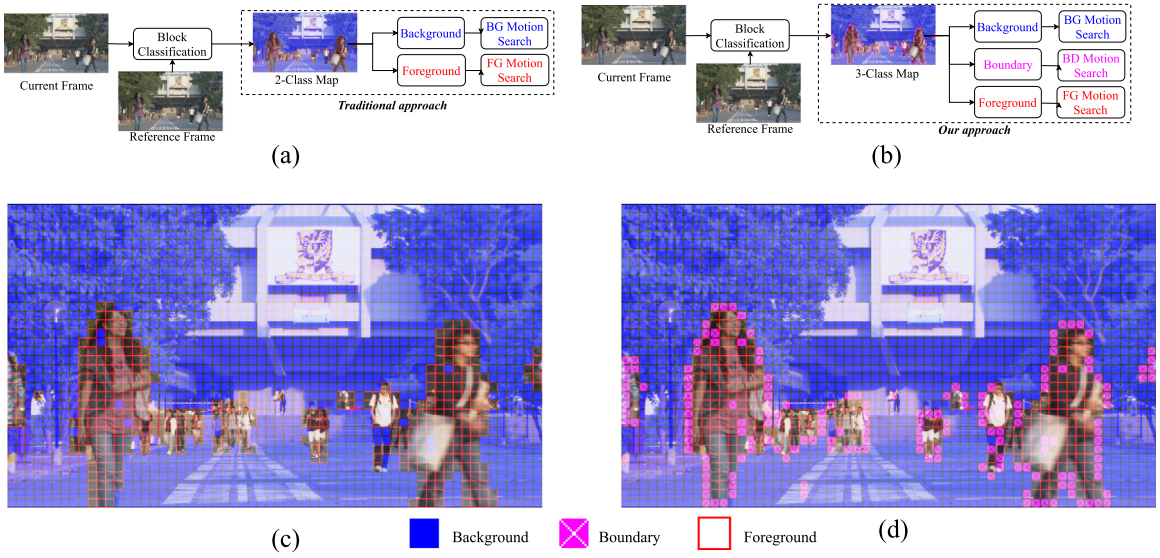

Fig. 1. (a) Illustration of traditional block classification approach with two classes: background and foreground, (b) proposed approach with three classes: background, foreground, and boundary, (c) two classes example (square sequence), (d) three classes example (square sequence). (Best viewed in color).

hexagon grid search (MHGS) [12] uses different mechanisms of complexity reduction in addition to hexagon patterns. The motion search complexity reduction has been extended for fast activity recognition applications [17,18]. These algorithms use different search patterns for reducing the total number of search points. In literature, the number of search points is also reduced by different early search termination methods [19]. The zero-motion termination method in [20] compared the distortion measure: the sum of absolute difference (SAD) value with a threshold at the zero motion vector (MV) point. The search process is terminated if the SAD value is less than the threshold. In another approach, the partial distortion measures (partial SAD) are employed to reduce the total number of computations involved at each search point [21]. The work on efficient direction oriented motion search in [22] uses both a reduction in search points and partial distortion measure.

Although these fast motion search algorithms demonstrated the significant computational gain, they do not consider specific properties of surveillance videos. In general, the surveillance videos are captured from fixed-angle or fixed-view cameras. For better compression, a pre-built dictionary-based coding scheme for traffic surveillance videos is presented in [23]. These videos always have considerable static background regions, unless the camera is moving. The background-foreground-division-based search (BFDS) [24] method exploited these surveillance-specific characteristics for fast motion estimation. The BFDS outperformed TZS in search complexity reduction. The computational complexity is reduced in BFDS while maintaining a similar block matching accuracy. Although BFDS searched only zero-MV point for background areas and used variants of TZS in the foreground areas for fast and efficient motion estimation, it failed to exploit inherent directional motion characteristics. Besides, BFDS needs additional computations for generating and updating the background frame. Moreover, only the bi-level classification of each block is done: either background or foreground. However, multilevel classification is desired for better matching accuracy. Efficient block classification for screen content coding in HEVC is proposed in [25]. For example, the illustration of the traditional two-class block classification approach and our idea of three class block classification for surveillance video coding is shown in Fig. 1.

The proposed work targets to address the above-mentioned problems. The detailed discussion on problem analysis is presented in the next Section. To overcome these constraints, this paper presents a novel background foreground boundary aware block matching algorithm. Our three major contributions are:

- A novel framework for three-level block classification,
- New search strategies for different classes, and
- Speed-up mechanism for different classes

In our work, firstly, the blocks are categorized into three classes, namely, background (BG), foreground (FG), and boundary (BD) blocks. Secondly, the motion search is performed by employing different search strategies for each class. The zero-motion vector-based search is employed for background blocks. Whereas, to exploit fast and directional motion characteristics of the boundary and foreground blocks, the eight rotating uni-wing diamond search patterns are proposed. Thirdly, the speed-up is achieved through the novel region-based sub-sampled structure.

The rest of the paper is organized as follows: Section 2 discusses problems in existing block matching works. The proposed algorithm is described in Section 3. Experimental settings, results, and performance comparisons are presented in Section 4. Section 5 concludes the paper.

2. Problem analysis

2.1. Characteristics of surveillance videos

The key characteristic of surveillance video lies in the high proportion of static blocks corresponding to the BG region. A large number of static blocks in the BG region provides a two-fold advantage: (1) better matching accuracy, and (2) fast motion estimation convergence. The static nature of BG regions has two direct implications: (1) the motion vectors (MV) for BG are concentrated near zero, and (2) the distortion values are smaller compared to FG regions.

The experiments were carried on twelve different surveillance test sequences taken from standard datasets [26–29]. The representative frames of each sequence are shown in Fig. 2. The details are mentioned in Table 1. The MV distribution for BG and FG regions shown in Fig. 3(a) clearly depicts that over 95% of MVs for BG are within unit area distance from the center due to its static nature. On the other hand, the MVs for the FG region are dynamically distributed since FG always contain moving objects. The MV distribution for fast motion FG region indicates that about 30% of MVs lie within a unit area from zero-MV, but the rest of the MVs are present in all the directions. Whereas, for the directional motion FG region, the MVs are mostly situated in a particular direction. About 70% of MVs are observed to lie along a particular distance path.

Table 1
Surveillance test sequences.

Sr. No.	Sequence name	Spatial resolution	Total frames	Frame rate	Frame starts	Dataset	Description	Motion type	Motion area	Sequence type
1	Campus	720 × 576	3001	30	2701	PKU [26]	Students and car moving in the campus.			
2	Classover	720 × 576	3001	30	351	PKU [26]	Students walking and cycling after class is over.	Slow	Small	A
3	Hall monitor	352 × 288	300	30	21	NCTU [27]	Person moving and picking briefcase.			
4	Ice	352 × 288	240	30	21	NCTU [27]	Crowd scene: people doing ice skating.			
5	Office	720 × 576	3001	30	1351	PKU [26]	Office working scene: a person entering.	Slow	Large	B
6	Square	1920 × 1088	250	25	11	IVPL [28]	Students moving in front of building.			
7	Bank	720 × 576	3001	30	201	PKU [26]	Vehicles moving in perpendicular direction.	Fast	Small	C
8	Crossroad	720 × 576	3001	30	1501	PKU [26]	Buses moving at crossroad.			
9	Intersection	1600 × 1200	1001	30	101	PKU [26]	Vehicles moving in speed at intersection.			
10	Mainroad	1600 × 1200	1001	30	501	PKU [26]	Fast moving cars seen from closer top-view.	Fast	Large	D
11	Overbridge	720 × 576	3001	30	1001	PKU [26]	Moving vehicles and people walking over bridge.			
12	Pedestrian area	768 × 432	250	25	81	LIVE [29]	Passerby encountering on open square.			

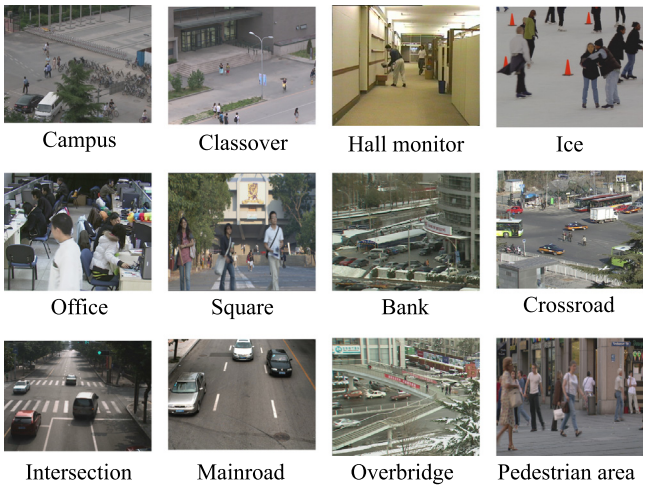

Fig. 2. Representative frames of surveillance sequences used in the analysis.

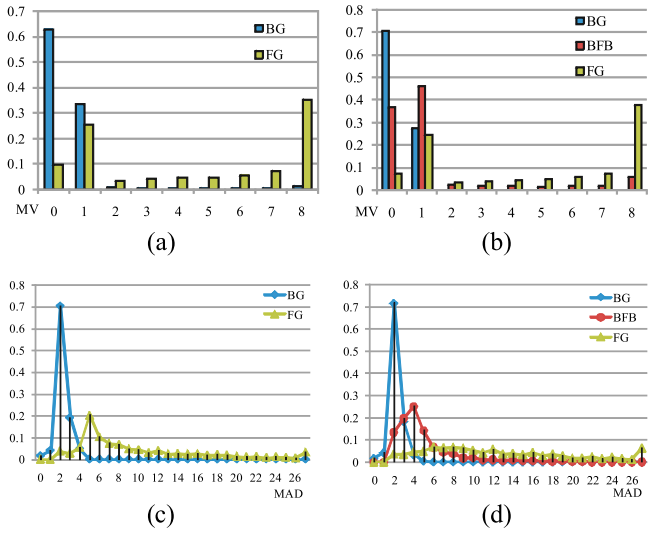

Fig. 3. Statistics of MV distribution: (a) traditional BG-FG approach, (b) Our BG-FG-boundary approach, statistics of average MAD distribution: (c) traditional BG-FG approach, (d) Our BG-FG-boundary approach.

The better matching accuracy for static BG regions resulted in lower distortion values as compared to FG regions. The MAD distribution for BG and FG regions is shown in Fig. 3(c). The bi-modal MAD distribution for BG and FG regions can be further used to improve classification accuracy.

It is evident from MV distribution that zero-MV biased search could be employed for BG regions to reduce the number of search points. On the other hand, the existing search methods could be employed for FG regions resulting in optimum motion estimation.

2.2. Problems in the existing works

We investigated three major problems in the existing works. First, the traditional approach mentioned in [24], the blocks may be wrongly classified as BG or FG due to bi-level classification. The block classification could be correct for smaller block sizes like 4 × 4, whereas question lies for the block sizes larger than 8 × 8. There exists a considerate proportion of blocks on object boundary, which contains both BG and FG regions partially (called as a boundary (BD) blocks hereafter). The block matching process for these blocks is not straightforward. Performing the BG-based search for BD blocks could result in poor matching accuracy due to the presence of the FG region. Whereas performing FG-based search could still result in poor matching accuracy due to the presence of the BG region. Hence, there is a need to investigate newer search strategies for these boundary blocks.

It should also be noted that the video frames are divided into non-overlapping blocks in the existing works. In the non-overlapping scheme, some of the internal regions of the moving objects could go undetected if the object has a size larger than the current block size, and if the change in pixel values is relatively small. Hence, low distortion FG region maybe wrongly classified as static BG region. Hence, there is a need to employ overlapping block classification mechanism for better classification accuracy.

Second, the existing fast block matching algorithms suffer from a high probability of becoming trapped in the local distortion minimum [30]. The primary reasons for this are (1) lower number of search points in the pattern, and (2) the non-sparsity in the location of search points in selected patterns. These problems make the DS [6] algorithm ineffective. Although TZS [10] and BFDS [24] overcome the problem of being trapped in a local minimum, they still require a considerable number of search points. Conventional MV estimation methods use either zero $(0,0)$ location [5] or median location [31] as initial search center for block matching process. The median value is computed using MVs of adjacent blocks located at the left, top, and top-right position. It is observed that the median start location is more useful for FG blocks containing higher motion content. Although this approach speeds-up the matching process, the matching accuracy is severely compromised for blocks with object boundaries. Since these blocks may follow a different motion path than its neighboring blocks. The DS and TZS have widely adopted search algorithms in practice, and BFDS is a variation of the TZS algorithm designed for surveillance sequences. However, none of them could exploit the special motion characteristics of FG blocks in the surveillance sequences. Hence, there is a need to innovate newer

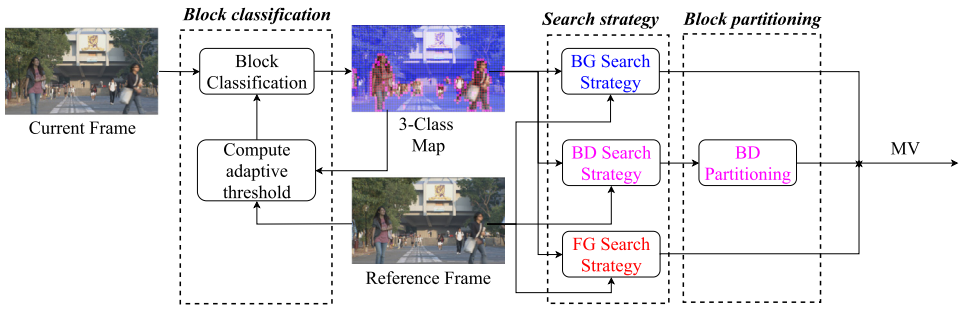

Fig. 4. Framework of the proposed approach.

direction oriented search patterns for the faster and accurate matching process.

Third, the computational complexity involved in the search process is still high. The sub-sampling in the spatial domain substantially speeds up the computation of distortion measure. Since the number of pixels used for distortion computation is reduced [32]. In literature, $1:2$ and $1:4$ sub-sampled distortion measure is widely adopted. However, to achieve better classification accuracy and improve block partitioning for boundary blocks present in the surveillance videos, a novel region-based sub-sampling is desired.

3. The proposed method

A background foreground boundary aware motion search method is proposed to address the problems present in the existing works.

3.1. Overview of the proposed method

The framework of the proposed method is shown in Fig. 4. The proposed framework contains three parts: (1) three class block classification, (2) class-based search strategy, and (3) block partitioning mechanism. First, the current frame is divided into non-overlapping blocks of size $M \times N$. Then, each block is classified into one of the three classes: (1) background (BG), (2) foreground (FG), and (3) boundary (BD) based on the self-adaptive threshold. The foreground is the region with moving objects. On the other hand, the background is the static region in nature without any moving objects. In our work, a simple approach for the block classification, which is solely based block distortion measure is presented. The current frame, reference frame, and the generated class map is used for motion estimation. Second, different search strategies are employed for each class to exploit the inherent motion characteristics in the surveillance videos. Third, the block partitioning mechanism is incorporated for BD blocks for efficient matching.

3.2. Block classification

The block classification framework for our work is shown in Fig. 5. The current frame is first divided into non-overlapping blocks of size $M \times N$ and partial SAD (pSAD) is computed for each block. The partial SAD is a popularly used efficient measure of similarity between two frame-blocks: candidate block (C) and reference block (R) and it is computed as:

$$pSAD(x, y, m_x, m_y) = \sum_{i \in h_x} \sum_{j \in h_y} \left| \begin{matrix} C_{(x+i,y+j)} - \\ R_{(m_x+x+i,m_y+y+j)} \end{matrix} \right| \quad (1)$$

In (1), (x, y) represents position of the candidate block and (m_x, m_y) denotes relative displacement from the candidate block location. Whereas, (h_x, h_y) denotes position of highlighted pixels (in $M \times N$ sized block) used in partial SAD computation. The traditional SAD uses all the pixels present in the candidate block for distortion measure computation. Whereas, on the other hand, partial SAD uses a subset of

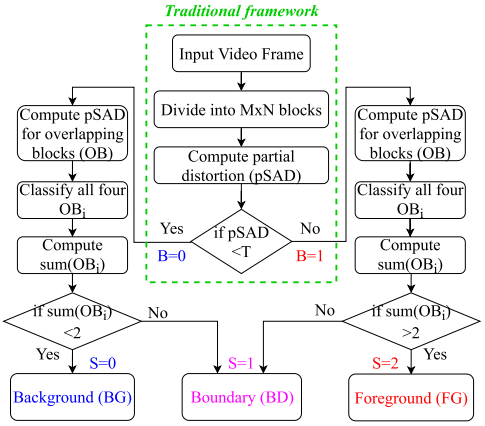

Fig. 5. Block classification framework.

pixels, which are highlighted explicitly for computational advantage. Then based on the partial SAD value, the block is classified into two intermediate classes: (1) intermediate-BG, and (2) intermediate-FG, which can be expressed as:

$$B = \begin{cases} 0, & pSAD < T \\ 1, & otherwise. \end{cases} \quad (2)$$

where T is the SAD threshold used for classification. It is understood that the fixed thresholds will mostly provide judgment error in block classification. Hence, the block classification threshold needs to be adaptive in nature. To this end, the work in [33] considers average distortion value for BG blocks belonging to the reference frame as an adaptive threshold. However, this mechanism fails to adjust the threshold in slight initial misclassification. To avoid this scenario, we propose to use a better adaptive threshold than used in [33].

$$T = \overline{SAD}_{pfbg} = \beta \times \frac{\sum_{(x,y) \in pfbg} SAD_{(x,y)}}{n_{pfbg}} \quad (3)$$

where β is multiplying factor, $pfbg$ is the background region in the previous frame and n_{pfbg} is the total number of BG blocks in the previous frame. The detailed discussion on selection of β is carried out in the experimental settings section.

The non-overlapping scheme used in traditional block classification framework has the disadvantage that some of the internal regions of the moving objects could go undetected if the object has a size larger than the current block size, and if the change in pixel values is relatively small. To address this, we propose to use four overlapping blocks (OB) centered at the center of the current block. The illustration of the current block and corresponding overlapping blocks is shown in Fig. 9(b). The overlapping blocks are also classified based on the threshold T.

$$OB_i = \begin{cases} 0, & pSAD < T \\ 1, & otherwise. \end{cases} \quad (4)$$

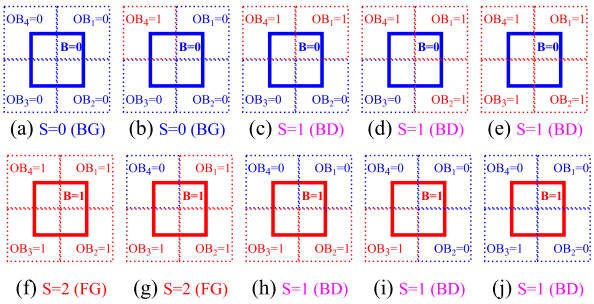

Fig. 6. Different block overlap scenarios for block classification.

The different overlapped-block classification scenarios are shown in Fig. 6. The block is classified into three different classes namely: (1) BG block, (2) FG block, and (3) BD block based on the number of overlapped blocks belong to the respective intermediate class.

$$S = \begin{cases} 0, & \sum_{i=1}^{4} OB_i < 2 \ and \ B = 0 \\ 2, & \sum_{i=1}^{4} OB_i > 2 \ and \ B = 1 \\ 1, & otherwise. \end{cases} \quad (5)$$

This classification, with the inclusion of overlapping blocks, has not only resulted in better three-level saliency detection but also reduced the chances of block misclassification. The inclusion of boundary class along with traditional background and foreground classes provide the following advantages: (1) improved motion vector prediction since boundary blocks cannot be trusted for predicting possible motion direction, (2) improved matching accuracy due to multiple classes, and (3) fast motion search convergence for each class.

3.3. Search strategy

Different search strategies are employed for each class in the surveillance video coding framework based on their MV distribution characteristics. The MV distribution corresponding to the FS algorithm is used for better understanding and analysis.

3.3.1. Zero-biased search for BG blocks

The MV distribution shown in Fig. 3(b) clearly indicates that the BG blocks are static in nature and seldom move. Hence, no block matching is performed for these blocks, and the MV is set to zero. It should be noted that the use of better search patterns would only increase computational requirements without any significant improvement in matching accuracy for these blocks. The higher chances of MV process to be trapped in local optimum makes fast search patterns ineffective for BG blocks. The fixed zero-MV strategy for BG blocks has achieved better trade-off compared to other fast block matching methods. The BFDS [24] and TZS [10] needs 1 and at least 24 points, respectively. It should be noted that although our method is searching 1 zero-MV point, our computational complexity is 75% better than the traditional zero-MV approach due to the utilization of $1:4$ sub-sampling during the block classification process. The proposed sub-sampling structure is shown in Fig. 9(a). Detailed discussion on the sub-sampling structure is carried in the section related to the speed-up mechanism.

3.3.2. Direction-oriented search for FG blocks

The MV distribution shown in Fig. 3(b) for fast motion FG region indicates that about 30% of MVs lie within a unit area from zero-MV, but the rest of the MVs are present in all the directions. Whereas, for directional motion content FG region, the MVs are mostly located along a particular direction. About 70% of MVs for these content lie in a particular direction. More appropriate points are included in traditional search patterns to avoid the problem of being trapped in a local minimum. Further, to exploit the directional motion characteristics of FG blocks, this work introduces novel directional search patterns shown in Fig. 7. For efficient directional motion tracking, the novel search patterns are created by adding uni-wing to the small diamond search pattern (SDSP) [6] in respective rotating directions. These patterns are termed as rotating uni-wing diamond search pattern (RUWDS). Eight possible combinations of uni-wing patterns: $0°, \pm45°, \pm90°, \pm135°, 180°$ are shown in Fig. 7 (f–m). These patterns can track horizontal, vertical as well as diagonal movements very effectively. These patterns can provide faster convergence due to the inclusion of extra points in the direction of motion.

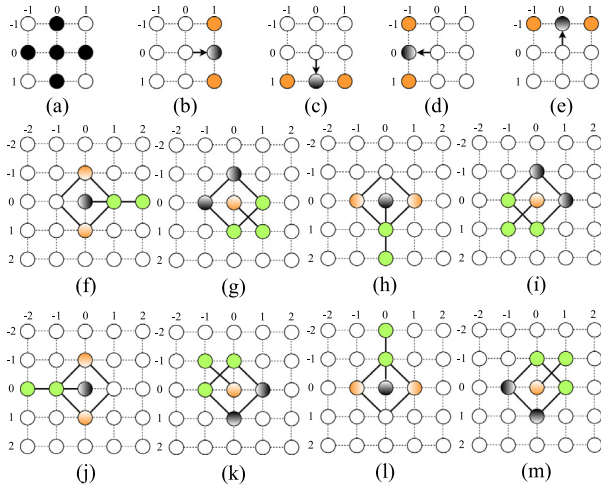

Fig. 7. Search patterns: (a) 1st step pattern: SDSP, (b) 2nd step east directional pattern, (c) 2nd step south directional pattern, (d) 2nd step west directional pattern, (e) 2nd step north directional pattern, (f–m) 3rd step onward search patterns: (f) 0° directional pattern: RUWDS, (g) −45°, (h) −90°, (i) −135°, (j) 180°, (k) 135°, (l) 90°, and (m) 45°.

The MV obtained during intermediate steps of search process is referred as intermediate-MV (IMV), whereas the final MV obtained after completion of the search process is referred as final-MV (FMV). The search process is described in following steps:

Step 1: The search process starts from search center: either default zero-MV or median-MV and uses the small diamond search pattern (SDSP) shown in Fig. 7(a). The block distortion values for the five points of SDSP (highlighted in black color) are obtained and minimum distortion location is treated as IMV of the first step. The search process is terminated if the minimum distortion location lies at the search center.

Step 2: One of the three-point search patterns shown in Fig. 7 (b-e) is chosen such that IMV obtained during the first step, is the new search center. The minimum distortion location is treated as IMV of the second step.

Step 3: The new search pattern selection for the ith step is based on the coordinate difference of the IMVs obtained in previous two search steps: $(i-1)$th IMV and $(i-2)$th IMV. Let coordinates of IMV point at the ith search step be (x_i, y_i). Let coordinate difference of IMVs of the $(i-1)$th and $(i-2)$th search steps be denoted as $(\Delta x_i, \Delta y_i)$, such that:

$$(\Delta x_i, \Delta y_i) = (x_{i-1} - x_{i-2}, y_{i-1} - y_{i-2}) \quad (6)$$

The search pattern for the subsequent steps are chosen based on the current direction of motion identified by $(\Delta x_i, \Delta y_i)$. The illustration of coordinate $(\Delta x_i, \Delta y_i)$ mapping is shown in Fig. 8(a). This mapping helps in understanding the pattern to be used during next search step. The search process is terminated if $(\Delta x_i, \Delta y_i) = (0,0)$. Otherwise, the

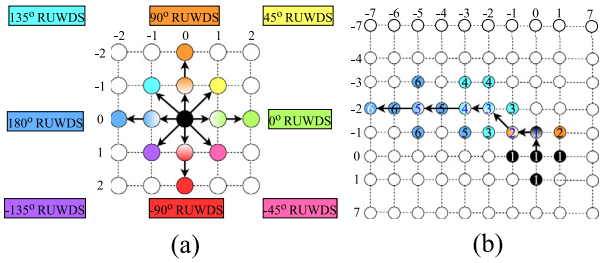

Fig. 8. (a) The illustration of coordinate location $(\Delta x_i, \Delta y_i)$. One of the eight direction-oriented search patterns is chosen based on $(\Delta x_i, \Delta y_i)$, (b) Typical search path example. The corresponding step number is marked in all the traversed search points.

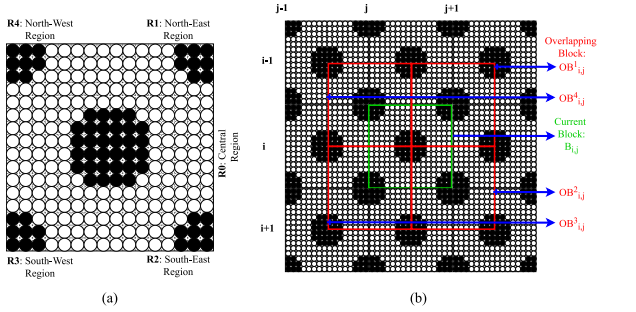

Fig. 9. (a) 16 × 16 block (pixel grid) with region-based 1 : 4 sub-sampling. Highlighted pixels of five different regions are used for partial distortion computation, (b) Illustration of 16 × 16 current block (highlighted in green) and corresponding four overlap blocks (highlighted in red).

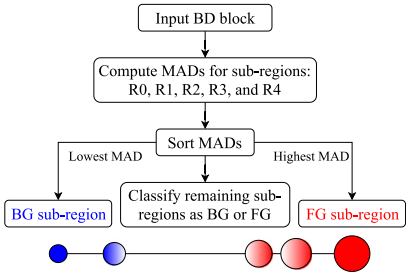

Fig. 10. BD block partitioning process. The size of circle is proportional to the MAD values. BG and FG sub-regions are highlighted in blue and red color respectively.

process is continued with Step 3 till the pattern touches search region boundary.

The path traversed by each search step for the block matching is referred to as the search path. The typical search path example is shown in Fig. 8(b). The search process uses SDSP in 1st step, two vertex point north-directional search pattern in 2nd step, 135° RUWDS pattern in 3rd and 4th steps, followed by 180° RUWDS pattern in the last two steps, before search is terminated since the pattern touched the search region boundary. It should be noted that our patterns can jump two-pixel locations at once in horizontal and vertical directions. Hence, these patterns converge faster without increasing the number of points searched.

3.3.3. BD Block partitioning and center-biased direction-oriented search for BD blocks

The MV distribution shown in Fig. 3(b) for boundary blocks indicates that more than 70% of MVs lie within a unit area from zero-MV, but the rest of the MVs are present in all the directions. Although BD blocks have static nature similar to BG blocks, they also tend to have MVs located at all the directions. This is mainly because of the fact that these blocks contain both BG and FG regions simultaneously. This means that the partial region belonging to BG would have zero-MV, and the remaining region would have object motion based larger MV. This has resulted in matching ambiguity in the motion search process. The zero-biased search strategy could result in poorer matching accuracy, whereas a direction-oriented search strategy could also provide sub-optimal performance due to the presence of the BG region.

To address this problem, we partitioned the BD blocks into five non-overlapping sub-regions, as shown in Fig. 9(a). The sub-regions are intentionally spaced apart for a better understanding of BG and FG portions. The BD partitioning process is illustrated in Fig. 10. The sub-regions: R0, R1, R2, R3, and R4 are classified into two categories based on the distortion values. The MAD distribution shown in Fig. 3(d) indicates that the MADs for BG are quite lower and concentrated near zero, whereas MADs for FG is uniformly distributed and large in general. Based on this understanding, the MADs for the five sub-regions are computed, and the lowest distortion sub-region is assumed to belong to the BG area, whereas the highest distortion sub-region is assumed to belong to the FG area. The remaining three sub-regions are classified based on the linear threshold. Different BD partitioned scenarios are illustrated in Fig. 11. The five sub-regions could result in a total of 32 different partitioned scenarios. However, our assumption of at least one sub-region belongs to FG and BG each, results in the following two cases to be non-existent: (1) all sub-regions are BG, and (2) all sub-regions are FG.

Once the sub-regions are classified, we propose to use only FG sub-regions for the block matching process. This mechanism not only improves the matching accuracy but also increases the search speed due to exclusion of BG regions from the current BD block. Although FG sub-regions in the BD block are considered for block matching process, it should be noted that remaining BG sub-regions in the BD block are considered to be static in nature, and more specifically, they are treated as zero-MV sub-regions. Hence, region segmentation for the BD block provided not only better matching accuracy for FG sub-regions but also for BG sub-regions. In a nutshell, if the BG region is the main part of the BD block, the matching accuracy will not be compromised. The matching accuracy could only be further improved as compared to the traditional block matching approaches.

It should also be noted that traditionally considered median-MV as an initial search center approach would be counter-productive for these blocks since the motion characteristics at object boundaries are quite different and difficult to predict. Hence we propose to use a zero-biased direction oriented search for BD blocks. The search patterns proposed for FG blocks are utilized on partitioned BD sub-regions for better matching accuracy.

3.4. Block partitioning mechanism for speed-up

The sub-sampling in the spatial domain for distortion measure computation provides additional speed-up. Since the number of pixels used for distortion computation is reduced [32]. In literature, 1 : 2 and 1 : 4 sub-sampled distortion measure is widely adopted. It resulted in 50% and 75% savings in the number of pixels used for each distortion measure computation, respectively. However, to achieve better classification accuracy and improve block partitioning for boundary blocks present in the surveillance videos, region-based sub-sampling is desired. To address this, we present region-based 1 : 4 sub-sampling. The illustration of 16 × 16 block (pixel grid) with 1 : 4 sub-sampling is shown in Fig. 9(a). The sub-sampled grid contains a total of five sub-regions: one central sub-region referred to as R0 and four corner sub-regions, namely north-east region, south-east region, south-west region, and north-west region referred as R1, R2, R3, and R4, respectively. The advantages of our region-based partial distortion measure as three-fold: (1) it provides desired computational saving without compromise in matching accuracy, (2) it would be used to further improve block classification accuracy, and (3) the region-based sub-sampling is symmetric and hence it is easily scalable for different block sizes.

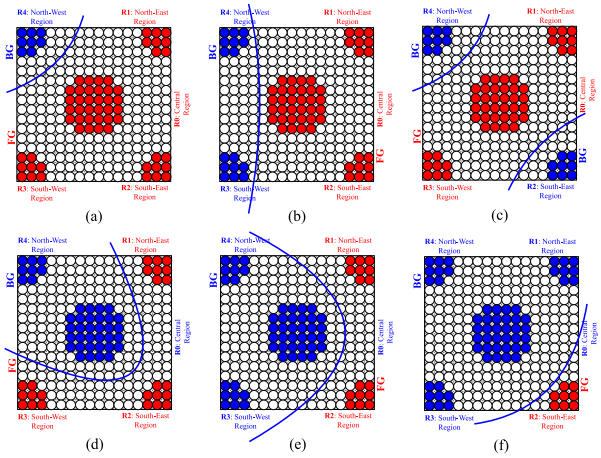

Fig. 11. Different BD block partitioning scenarios: (a) single region R4, (b) two adjacent regions R4 and R3, (c) opposite regions R4 and R2, (d) R4 and central region R0, (e) R4, R3 and R0, (f) R4, R1, R3 and R0. BG and FG regions are highlighted in blue and red color respectively.

4. Experiments

4.1. Datasets

Experiments were carried on different surveillance test sequences listed in Table 1. Twelve test sequences with varying foreground motion content, different background proportion, and of different spatial frame sizes are chosen from benchmark databases [26–29]. Among them, eight sequences are taken from PKU-SVD-A dataset [26], two sequences are taken from standard CIF sequences [27], and a sequence is taken from each LIVE dataset [29], and IVPL dataset [28]. The background is mostly static throughout the length of the test sequence. The sequences are categorized into four different types: {A, B, C, and D}, based on motion content and proportion of motion area. All test sequences are in YUV 4: 2: 0 (uncompressed) color format, and only the luminance (Y) component of each test sequence with the 8-bit resolution is used for block matching experiments. Total 100 frames of each sequence starting from the frame start mentioned in Table 1 are considered for experiments.

4.2. Evaluation metrics

The peak signal-to-noise ratio (PSNR) is widely adopted objective quality metric to evaluate the performance of block matching methods. Although bit-rate provides a better understanding of compression efficiency, the standard block matching algorithms only focus on matching accuracy. It is evident that better matching accuracy could lead to better compression performance. Moreover, our analysis is focused only on the motion estimation component of the surveillance coding engine. Hence, in this paper, bit-rate, PSNR, and search complexity are considered during the experimental evaluation.

4.2.1. Prediction quality

The traditional PSNR inherently provides equal weight to each pixel during mean-squared-error (MSE) computation. It clearly lacks to provide different importance to different areas in the sequence based on their relevance. However, for surveillance sequences, it is highly desired that the foreground area is given more weight based on its higher relevance over background areas. Evaluation metric Weighted PSNR (WPSNR) based on Weighted MSE (WMSE) is used in [34]. WPSNR provided a better understanding of the surveillance sequence coding performance. Hence, WPSNR is considered for objective performance evaluation.

$$WPSNR = 10\log_{10}\left(\frac{V_{\max}^2}{WMSE}\right) \tag{7}$$

$$WMSE = \sum_{s=0}^{S-1} \frac{\omega_s}{|C_s|} \sum_{(m,n)\in C_s} MSE(m,n) \tag{8}$$

where V_{max} is the maximum pixel intensity for the given bit resolution, S is the number of classes, ω_s is the weight of class s, C_s is the set of blocks belonging to the class s, $|C_s|$ is the number blocks belonging to the class s, and $MSE(m,n)$ is the mean-squared-error of (m,n)th block.

4.2.2. Search complexity

The search complexity in the motion search algorithm not only depends on the total number of search points traversed in the search path but also on the number of computations performed at each search point. Hence the parameter is proposed to include both: (1) average number of search points and (2) proportion of pixels used for partial distortion measure. To this end, the average number of checked pixels per candidate block (ANCPB) is computed as:

$$ANCPB_{\alpha} = \alpha \times \text{Average number of search points} \tag{9}$$

where α represents the proportion of pixels used in the sub-sampled SAD computation.

4.2.3. Bits per motion vector

Motion vector coding not only depends on MV distribution but also on the proportion of BG blocks. The MV is traditionally coded using a variable-length coder (VLC) table. Hence, we use a similar VLC coding mechanism, which allocates a lower number of bits to the MVs close to (0,0), whereas, a higher number of bits are assigned to the MVs away from (0,0).

4.3. Experimental settings

4.3.1. Search parameters

Our framework is general and scalable. Hence, it can be easily extended to different block sizes. However, we consider widely adopted search parameters in our experiments. The block size as 16×16 and search range as $p = \pm 8$ is considered.

The effect of variation in thresholds for block classification is shown in Fig. 12. As expected, fixed-threshold value resulted in judgment error. An increase in the fixed-threshold value from 256 to 2048 resulted in an increase in the proportion of blocks classified as BG. The adaptive-threshold selected with $\beta = 1$ resulted in sudden decay in the proportion of BG blocks. To overcome this, the $\beta > 1$ is considered in our experiments. We have empirically found that multiplying factor $\beta = 2$ in (3) has two advantages: (1) the adaptive-threshold value is independent of initial-threshold value, it is self-adaptive in nature and varies smoothly, and (2) as desired, the proportion of BG blocks does not change drastically for consecutive frames.

For BD block partitioning, different sub-sampling structures were explored. The criteria for selection of particular structure lies in three main reasons: (1) the sub-regions must have considerably enough pixels in it, (2) the pixels must be close-enough or intact in particular sub-region, and (3) different sub-regions must be spaced apart. It is found that sub-region with a circle radius $r = 3$ provided the best performance. The structure shown in 9 (a) closely resembles the above-mentioned criteria and hence chosen in our method.

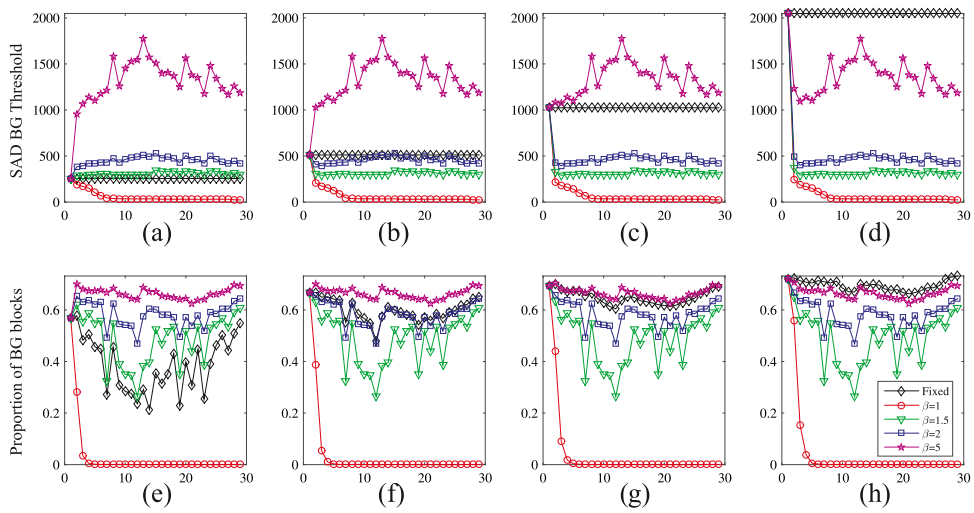

Fig. 12. The effect of variation in fixed and adaptive thresholds with different multiplying factor β for block classification. Column 1–4 considers the initial-threshold value as 256, 512, 1024, 2048, respectively.

4.3.2. Test setup for rate–distortion comparison

The test sequences listed in Table 1, are also used for rate–distortion (RD) comparisons between different block matching algorithms. In this pursuit, all the block matching algorithms, including the proposed algorithm, are integrated into simple video encoder and simulated under QP values of 22, 27, 32, and 37. For comparing the average difference between RD curves, the Bjontegaard's method [35] is used to calculate the average Bit-rate and PSNR.

All the block matching algorithms are implemented to best of our knowledge and understanding in MATLAB 2015a running on 64-bit Windows 7 platform with Intel Xeon(R) CPU E5-2650 v2 @ 2.60 GHz with 32.0 GB RAM.

4.4. Results

Several experiments were carried out to validate the efficacy of the proposed method. We explore the effect of different components of the proposed method and demonstrate the performance improvement of the proposed algorithm over other existing methods.

4.4.1. Comparison of different components of the proposed method

In this section, we compare the effect of various components of the proposed approach. The proposed framework contains three components: (1) 3-class block classification (BC), (2) class-based search strategy (SS), and (3) block partitioning (BP) mechanism for speed-up. To study the effect of an individual component, we keep the remaining two components in the proposed framework unchanged. The experimental results for different combinations of three components are shown in Table 2.

First, we evaluate the effect of our 3-class block classification framework against the traditional 2-class block classification approach. To this end, we only change component BC and keep components SS and BP unchanged. Our 3-class approach in BC+SS+BP yielded, on average, better PSNR results than the traditionally used 2-class approach in SS+BP. The inclusion of additional boundary class to the existing BG and FG classes provided a better classification for surveillance videos. The improved motion search strategy for boundary class provided better matching accuracy. It is evident from Table 2, that improvement up to 0.25 dB in PSNR is achieved mainly because of the better classification accuracy. However, the motion search complexity for the 3-class approach is about 15% higher than the 2-class approach.

The subjective evaluation for the traditional 2-class approach and our 3-class approach are shown in Fig. 13. It is evident from the results that the proposed approach can extract important characteristics in the surveillance sequences. The blocks at the object boundaries are either classified as BG or FG in the traditional approach. But these blocks contain both BG and FG partially, and hence our approach with additional boundary class addresses this problem. In turn, the block misclassification is significantly reduced.

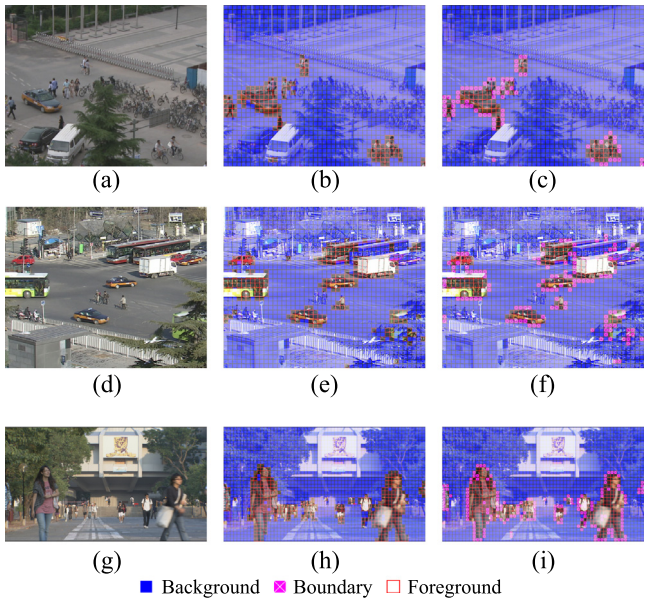

Fig. 13. Results of the proposed block classification method. (a) Original frame, Campus (46th frame), (d) Original frame, Crossroad (6th frame), (g) Original frame, Square (50th frame), (b), (e), and (h) 2-Class classification, (c), (f), and (i) Our 3-Class classification.

Second, we evaluate the effect of our direction-oriented search patterns against traditional TZS patterns. To this end, we only change component SS and keep components BC and BP unchanged in our framework. For fair comparisons, the motion estimation process for background blocks in both BC+SS+BP and BC+BP is unchanged. The direction oriented search patterns with adaptive search center is utilized for FG and boundary blocks in BC+SS+BP, whereas TZS patterns are utilized in BC+BP. Our block classification based adaptive search strategy clearly outperformed the traditional search strategy. The improvement up to 3.19 dB in PSNR for Mainroad sequence and improvement of about 0.28 dB in PSNR on an average is achieved

Table 2
PSNR and motion search complexity (ANCPB) comparisons for different components of proposed method. The full version of proposed method contains all three components: 3-class block classification (BC), class-based search strategy (SS), and block partitioning (BP).

PSNR (in dB)													
Algorithm	Campus	Class	Hall	Ice	Office	Square	Bank	Crossroad	Intersect	Mainroad	Overbridge	Pedestrian	Average
Proposed	35.0525	34.2832	33.5433	28.9309	30.0303	31.6335	32.8658	27.067	32.3646	26.4812	30.972	30.2226	31.1206
SS + BP	−0.0263	0.0021	−0.0413	0.0070	−0.0971	−0.0431	−0.0155	−0.0270	0.0011	−0.2592	−0.0445	−0.0518	−0.0496
BC + BP	0.0627	0.1213	0.0643	0.0758	0.1090	−0.1014	−0.0050	0.2306	−0.3002	−3.1928	−0.0327	−0.4706	−0.2866
BC + SS	0.0135	0.0136	−0.0360	0.0080	−0.0093	−0.0215	−0.0111	0.0107	0.0030	0.0039	−0.0370	−0.1426	−0.0171
Motion search complexity (ANCPB)													
Algorithm	Campus	Class	Hall	Ice	Office	Square	Bank	Crossroad	Intersect	Mainroad	Overbridge	Pedestrian	Average
Proposed	1.1334	1.7343	1.1414	4.1082	2.1701	2.6796	0.7464	2.5171	6.3104	3.3798	1.2049	7.1805	2.8588
SS + BP	1.0254	1.2629	1.0227	3.8589	2.0002	2.5281	0.6870	2.3242	3.1179	3.3471	1.1252	6.6849	2.4154
BC + BP	5.0241	8.5913	4.9901	19.7940	8.4303	12.1350	2.7297	11.6462	28.9561	9.6702	4.2971	29.0001	12.1053
BC + SS	1.8089	1.9550	1.8386	4.6634	2.8293	3.2716	1.4665	3.0866	3.7022	4.0191	1.8945	6.9065	3.1202

Table 3
PSNR comparisons for different traditional and surveillance-based block matching algorithms.

Algorithm	Campus	Class	Hall	Ice	Office	Square	Bank	Crossroad	Intersect	Mainroad	Overbridge	Pedestrian	Average
FS	35.2069	34.4575	33.6495	29.0614	30.3416	31.5965	32.8852	27.6075	32.2507	23.3813	31.2733	29.8607	30.9643
DS [6]	−0.1145	−0.1198	−0.0568	−0.2114	−0.6990	−0.4534	−0.0165	−1.0377	−0.8369	−0.7596	−0.7872	−0.7626	−0.4879
TZS [10]	−0.0185	−0.0243	−0.0160	−0.0623	−0.1396	−0.0555	−0.0066	−0.0732	−0.2235	−0.2088	−0.1548	−0.1775	−0.0967
MHGS [12]	−0.0455	−0.0590	−0.0543	−0.0657	−0.1841	−0.0734	−0.0121	−0.1304	−0.2448	−0.2443	−0.2602	−0.2167	−0.1325
FAME [36]	−0.0645	−0.0825	−0.0799	−0.2830	−0.6673	−0.5380	−0.0185	−0.2179	−0.7251	−0.7440	−0.5217	−1.0666	−0.4174
BFDS [24]	−0.0946	−0.0500	−0.0427	−0.0633	−0.2453	−0.0815	−0.0179	−0.1336	−0.2265	−0.2129	−0.1621	−0.2206	−0.1293
[24] + [12]	−0.1213	−0.0850	−0.0811	−0.0664	−0.2962	−0.0996	−0.0231	−0.1835	−0.2479	−0.2456	−0.2674	−0.2630	−0.1650
[24] + [36]	−0.1336	−0.1063	−0.1121	−0.2824	−0.8711	−0.5588	−0.0297	−0.2819	−0.7292	−0.7515	−0.5266	−1.1168	−0.4583
Proposed	−0.1543	−0.1743	−0.1063	−0.1306	−0.3113	0.0370	−0.0194	−0.5405	0.1139	3.0999	−0.3013	0.3619	0.1562
(WPSNR)	−0.1822	−0.1918	−0.1206	−0.1414	−0.3434	0.0750	−0.0257	−0.6281	0.1339	3.5498	−0.3855	0.4163	0.1797

mainly because of our direction oriented search patterns and adaptive search center selection. Not only this, the motion search complexity for our search pattern is about five times better than the traditional search pattern. Hence, we have been able to achieve both improvements in matching accuracy and a significant reduction in search complexity at the same time.

Third, we evaluate the effect of our speed-up approach against the proposed approach without BP. To this end, we only change component BP and keep components BC and SS unchanged in our framework. Our partial distortion mechanism in BC+SS+BP yielded about 0.02 dB improvement in PSNR on an average over BC+SS. Moreover, our partial distortion approach also provided about 10% improvement in search complexity over BC+SS on an average, mainly because of the utilization of lower of a number of pixels for each distortion computation.

Overall, the proposed approach with all three components BC + SS + BP provided the best PSNR results on an average mainly due to the inclusion of direction oriented search patterns. On the other hand, SS+BP provided the least search complexity due to a 2-class classification approach and partial distortion measure. The trade-off for matching accuracy and motion search complexity is obtained by BC+SS+BP combination. Next, we compare the effect of proposed BC+SS+BP against other existing methods.

4.4.2. Comparison of the proposed and the existing methods

The performance of the proposed method is compared against five traditional block matching algorithms: FS, DS [6], TZS [10], modified hexagon grid search (MHGS) [12], and fast adaptive motion estimation (FAME) [36]. The performance is also compared against a surveillance-based block matching method BFDS [24], which uses the TZS pattern for FG blocks. There are very few block matching methods designed specifically for surveillance sequences. For better performance evaluation, we create two new methods by incorporating the latest MHGS and FAME methods in the place of TZS in the BFDS framework. The block classification in these two methods is kept the same as BFDS. Hence, in total, we compare the proposed method against five traditional and three surveillance-based block matching algorithms.

The PSNR and motion search complexity results for different block matching algorithms are shown in Tables 3 and 4. As expected, the block matching accuracy is the best for FS algorithm among existing algorithms due to its exhaustive search nature. The PSNR results indicate that the DS performs worst among compared algorithms. This is mainly because DS has a typical structure which gets trapped in a local optimum. However, TZS and BFDS performed better than the DS due to the inherent capability of the TZS pattern and its variants. MHGS achieve more than 18% computational saving than TZS but come with slightly poorer matching accuracy. The FAME algorithm outperforms DS in both matching accuracies and search complexity due to advanced search mechanism. The PSNR and WPSNR values for the proposed algorithm clearly outperformed existing methods on an average. It should be noted that our algorithm not only outperformed all fast block matching methods but also the FS algorithm. For the Mainroad sequence, the proposed algorithm achieved more than 3.5 dB improvement in WPSNR over FS.

The block-wise PSNR comparisons shown in Fig. 14 indicates that the proposed algorithm outperforms existing methods for BD and FG blocks. Our method provided better results for BD blocks due to the effective utilization block partitioning mechanism. Hence, the proposed method is observed to be superior for these foreground motion contents. This is achieved mainly because of the four reasons: (1) the search starts from either zero-MV or median-MV, (2) the new search window with new search-center is chosen, (3) only FG portion in BD blocks in chosen for block matching, and (4) due to the capability of our proposed direction oriented search patterns to capture directional motion search characteristics of the moving objects.

FS is the computationally demanding method and requires $(2p+1)^2$ search points for the single MV search operation. Whereas, DS algorithm requires $9+5\times(nSteps)+4$ number of search points. The ANCPB results shown in Table 4 indicates that the computational complexity is in the order FS > TZS > MHGS > DS > BFDS > BFDS+MHGS > FAME > Proposed > BFDS+FAME. Although TZS and MHGS provided better PSNR results, still it comes at a high computational expense. The BFDS algorithm used only a single search point for BG blocks as expected and cleverly outperformed TZS, which uses at least 24 search points without any notable degradation in matching performance. On the other hand, our method used a sub-sampling pattern for BG blocks

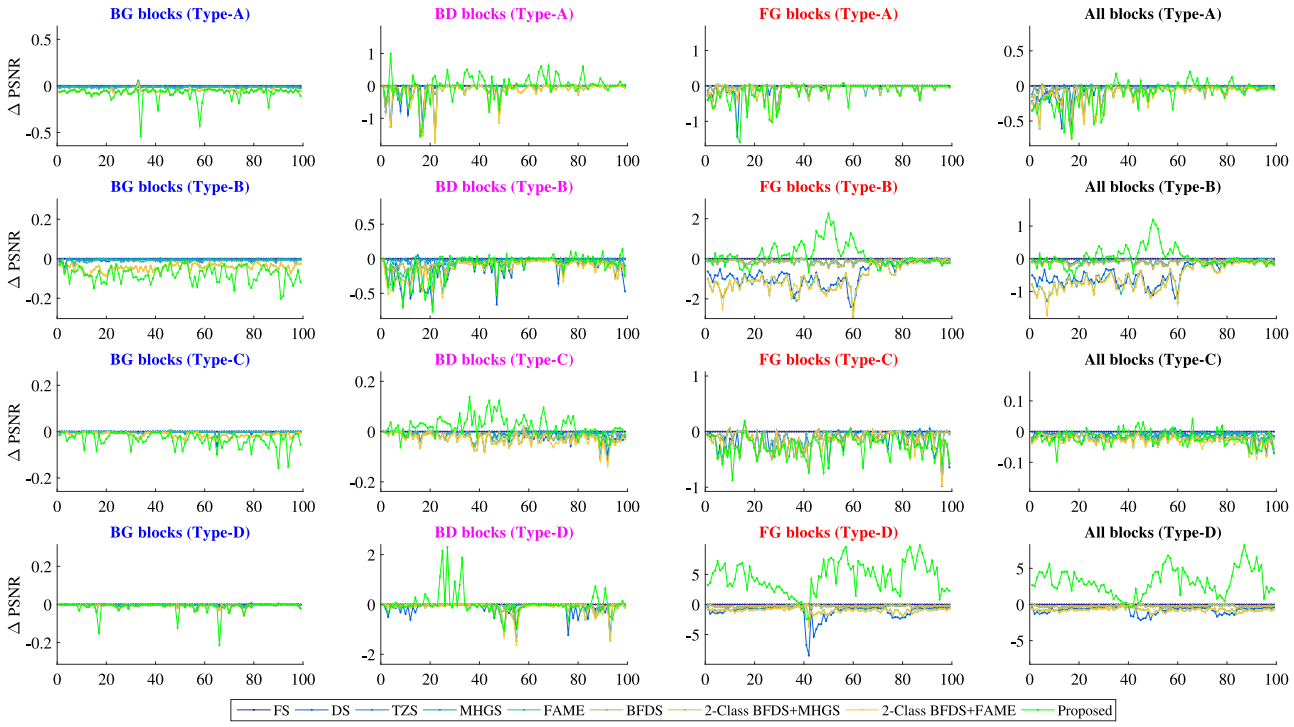

Fig. 14. Block-class-wise PSNR (in dB) comparison for different algorithms against FS. Row 1–4 corresponds to different sequences: {Hall monitor (type-A), Square (type-B), Bank (type-C), and Mainroad (type-D)}, respectively. (Best viewed in color).

Table 4
Motion search complexity (ANCPB) comparisons for different traditional and surveillance-based block matching algorithms.

Algorithm	Campus	Class	Hall	Ice	Office	Square	Bank	Crossroad	Intersect	Mainroad	Overbridge	Pedestrian	Average
FS	289.000	289.000	289.000	289.000	289.000	289.000	289.000	289.000	289.000	289.000	289.000	289.000	289.000
DS [6]	13.898	13.869	13.726	14.942	15.172	14.971	13.192	14.969	14.723	15.051	13.401	17.730	14.637
TZS [10]	31.341	33.296	32.169	30.745	30.461	29.772	29.664	29.373	28.701	27.419	28.977	27.298	29.935
MHGS [12]	25.366	27.059	26.168	25.406	24.962	24.196	23.740	24.339	23.391	22.457	23.313	23.447	24.487
FAME [36]	5.539	5.765	5.587	5.758	5.670	5.419	5.155	5.635	5.401	5.192	5.119	5.652	5.491
BFDS [24]	3.193	5.203	4.003	13.215	4.441	6.898	2.341	6.079	9.085	4.341	2.953	14.132	6.324
[24] + [12]	2.934	4.437	3.551	11.621	4.107	6.104	2.208	5.760	7.976	4.309	2.813	12.928	5.729
[24] + [36]	1.462	1.660	1.503	3.249	1.820	2.158	1.249	2.249	2.594	1.887	1.395	3.735	2.080
Proposed	1.133	1.734	1.141	4.108	2.170	2.680	0.746	2.517	6.310	3.380	1.205	7.181	2.859

and achieved further improvement of 75% in computations for those blocks. Not only this but the block-wise ANCPB comparisons shown in Fig. 15 indicates that our algorithm significantly outperformed existing methods (except BFDS+FAME) for BD and FG blocks as well. Our method is four times computationally better than the next-best method for BG blocks. Whereas, the proposed method is at least two times computationally better for BD and FG blocks. This is achieved mainly because of the three reasons: (1) the search could directly start from the highly probable vicinity of true MV, (2) the new direction-oriented search patterns are chosen based on the motion characteristics, and (3) region-based partial distortion measure.

The effect on BD-PSNR, BD-Bitrate, and ANCPB are presented in Table 5. Among existing block matching algorithms, the degradation in BD-PSNR and BD-Bitrate is the smallest for TZS with the worst ANCPB performance. On the other hand, BFDS+FAME provides the best ANCPB performance with significant compromise in BD-PSNR and BD-Bitrate. In contrast, the proposed algorithm not only outperformed in BD-PSNR but also in BD-Bitrate with more than 99% savings in ANCPB. The RD curves for video sequences Square and Pedestrian area under different QP values of 22, 27, 32, and 37 are shown in Fig. 16. It is evident from Fig. 16 that, increase in PSNR is proportional to an increase in bitrate. Moreover, the proposed algorithm outperforms existing block matching algorithms in terms of PSNR, Bit-rate, and ANCPB.

Table 5
Average BD-PSNR, BD-Bitrate, and ANCPB results for different block matching algorithms in comparison to FS.

Algorithm	BD-PSNR	BD-Bitrate	ΔANCPB (in %)
DS [6]	−0.529	15.165	−94.94
TZS [10]	−0.096	2.002	−89.64
MHGS [12]	−0.173	2.782	−91.53
FAME [36]	−0.483	10.125	−98.10
BFDS [24]	−0.209	5.048	−97.81
[24] + [12]	−0.283	5.654	−98.02
[24] + [36]	−0.583	13.583	−99.28
Proposed	0.168	−6.557	−99.01

The simplest method to encode MVs could be to use fixed-length coding. For $(2p+1)\times(2p+1)$ search region, the fixed-length coding requires at least $\lceil log_2(2p+1)\rceil$ bits to represent motion in each horizontal and vertical direction. However, it is observed that a large proportion of MVs lies in the vicinity of the search center (0,0), and a variable-length coder (VLC) could provide improved MV coding efficiency. The bits per MV (BMV) results shown in Table 6 indicates that VLC provided better coding efficiency on an average. However, BMVs are slightly larger in a couple of sequences containing mainly large and non-homogeneous motion vectors. The BMV is in the order FS > TZS > MHGS > FAME

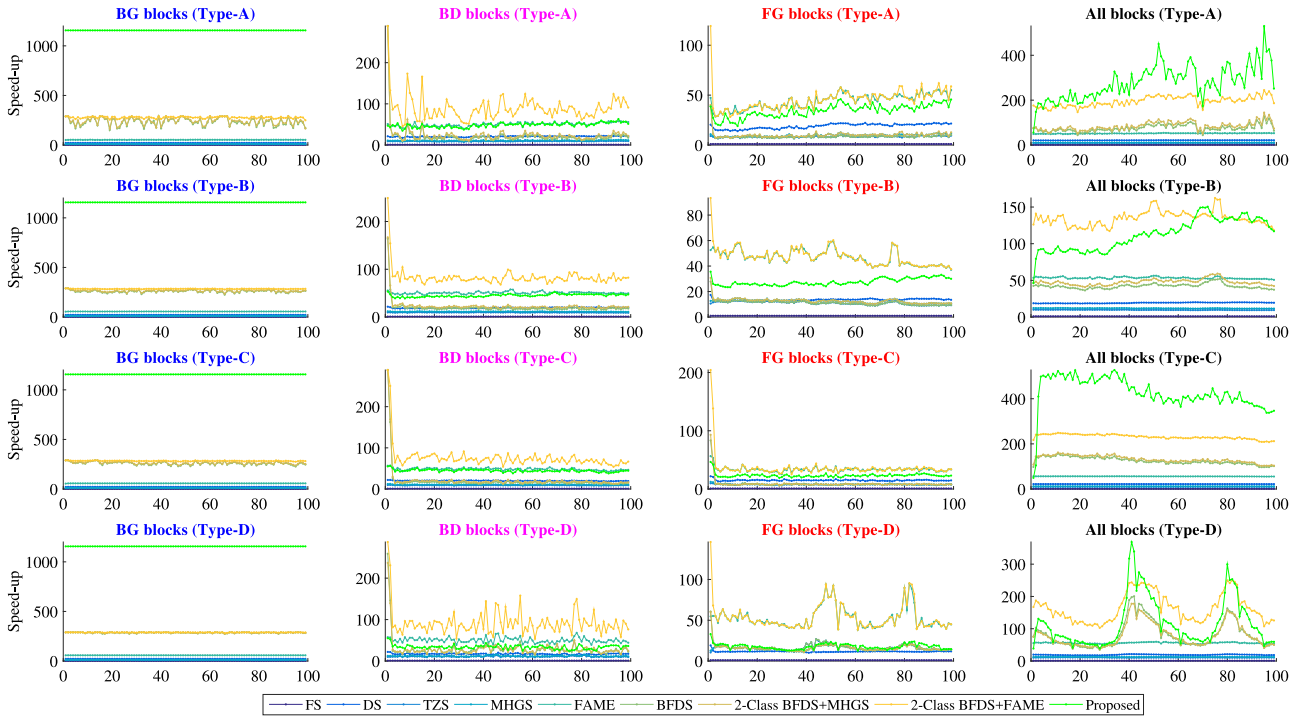

Fig. 15. Block-class-wise Speed-up (in times) comparison for different algorithms against FS. Row 1–4 corresponds to different sequences: {Hall monitor (type-A), Square (type-B), Bank (type-C), and Mainroad (type-D)}, respectively. (Best viewed in color).

Table 6
Motion vector coding (in BMV) comparisons for different traditional and surveillance-based block matching algorithms.

Algorithm	Campus	Class	Hall	Ice	Office	Square	Bank	Crossroad	Intersect	Mainroad	Overbridge	Pedestrian	Average
FS	2.884	3.170	2.889	3.455	4.221	3.762	2.184	3.661	3.489	4.376	2.492	6.265	3.571
DS [6]	2.750	2.876	2.603	3.301	3.570	3.413	2.155	3.263	3.159	3.500	2.307	5.550	3.204
TZS [10]	2.794	2.924	2.720	3.415	4.046	3.638	2.163	3.516	3.467	4.356	2.428	6.192	3.471
MHGS [12]	2.754	2.860	2.624	3.407	4.013	3.616	2.152	3.460	3.465	4.367	2.399	6.148	3.439
FAME [36]	2.706	2.765	2.536	3.325	3.683	3.416	2.148	3.423	3.336	3.815	2.375	5.799	3.277
BFDS [24]	2.355	2.177	2.155	3.378	2.932	3.313	2.100	3.132	3.450	4.282	2.378	5.730	3.115
[24] + [12]	2.350	2.174	2.147	3.372	2.928	3.306	2.096	3.106	3.448	4.293	2.352	5.689	3.105
[24] + [36]	2.347	2.172	2.143	3.291	2.836	3.225	2.096	3.098	3.320	3.764	2.333	5.359	2.999
Proposed	1.438	1.244	1.217	2.572	2.143	2.480	1.128	2.259	2.476	3.324	1.355	5.146	2.232

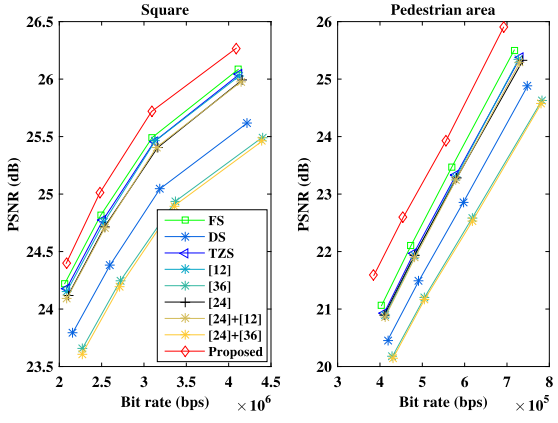

Fig. 16. Rate–distortion results for video sequences Square and Pedestrian area under different QP values of 22, 27, 32, and 37. (Best viewed in color).

> DS > BFDS > BFDS+MHGS > BFDS+FAME > Proposed. For the FS algorithm, the BMV is highest, because FS MV distribution is comparatively sparse. On the other hand, an early search termination mechanism in fast search algorithms like TZS, MHGS, FAME, and DS restrict MV distribution to be as sparse as FS. Hence, the BMV for fast search algorithms is comparatively lower. In fact, BFDS+FAME provided about 16% BMV savings as compared to FS. On the other hand, the proposed method provided a significant improvement of about 37% in BMV. This improvement is largely achieved through our efficient search mechanism.

5. Conclusion

In this paper, a fast background-foreground-boundary aware block matching algorithm is proposed for surveillance videos. Our method firstly used a three-level saliency detection mechanism to classify blocks into three classes: BG, BD, and FG blocks. This classification captured different characteristics of surveillance videos. Secondly, the proposed method used different search strategies for different classes. Our search patterns are found to be useful for directional motion content blocks in the surveillance sequences. Thirdly, we have also proposed BD block partitioning criteria for efficient block matching. The proposed approach provided better matching quality as compared to other existing algorithms. At the same time, it has also been able to provide four times speed-up for BG blocks and more than two times speed-up for BD and FG blocks as compared to the BFDS method. An improvement in the block matching accuracy as compared to FS and speed-up over existing methods is the major contribution of our work. Experimental results

confirm the efficacy of the proposed method in different scenarios. In further work, we aim to include the proposed scheme in the HEVC framework.

Acknowledgments

The paper is supported in part by the following grants: China Major Project for New Generation of AI Grant (No. 2018AAA0100400), National Natural Science Foundation of China (No. 61971277), the Visvesvaraya Ph.D. Scheme for Electronics and IT Research Fellowship (MeitY, India).

References

[1] Y. Fang, G. Ding, Y. Yuan, W. Lin, H. Liu, Robustness analysis of pedestrian detectors for surveillance, IEEE Access 6 (2018) 28890–28902.
[2] Y. Fang, Y. Yuan, L. Li, J. Wu, W. Lin, Z. Li, Performance evaluation of visual tracking algorithms on video sequences with quality degradation, IEEE Access 5 (2017) 2430–2441.
[3] Y. Fang, Y. Yuan, L. Xu, W. Lin, A benchmark for robustness analysis of visual tracking algorithms, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 1120–1124, http://dx.doi.org/10.1109/ICASSP.2016.7471850.
[4] Y.-C. Lin, S.-C. Tai, Fast full-search block-matching algorithm for motion-compensated video compression, IEEE Trans. Commun. 45 (5) (1997) 527–531.
[5] D. Marpe, T. Wiegand, G.J. Sullivan, The H.264/MPEG4 advanced video coding standard and its applications, IEEE Commun. Mag. 44 (8) (2006) 134–143.
[6] S. Zhu, K.-K. Ma, A new diamond search algorithm for fast block-matching motion estimation, IEEE Trans. Image Process. 9 (2) (2000) 287–290.
[7] C. Zhu, X. Lin, L.-P. Chau, Hexagon-based search pattern for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol. 12 (5) (2002) 349–355.
[8] C. Zhu, X. Lin, L. Chau, L.-M. Po, Enhanced hexagonal search for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol. 14 (10) (2004) 1210–1214.
[9] G.J. Sullivan, J. 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.
[10] JCT-VC, HMReference Software, 2013, https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/.
[11] P. Nalluri, L.N. Alves, A. Navarro, Complexity reduction methods for fast motion estimation in HEVC, Signal Process., Image Commun. 39 (2015) 280–292.
[12] K. Singh, S.R. Ahamed, Low power motion estimation algorithm and architecture of HEVC/H. 265 for consumer applications, IEEE Trans. Consum. Electron. 64 (3) (2018) 267–275.
[13] A.F. Guarda, J.M. Santos, L.A. da Silva Cruz, P.A. Assunção, N.M. Rodrigues, S.M. de Faria, A method to improve HEVC lossless coding of volumetric medical images, Signal Process., Image Commun. 59 (2017) 96–104.
[14] L. Jia, C.-Y. Tsui, O.C. Au, K. Jia, A new rate-complexity-distortion model for fast motion estimation algorithm in HEVC, IEEE Trans. Multimed. 21 (4) (2019) 835–850.
[15] B.T. Oh, Enhanced zonal search algorithm for motion estimation in depth-map coding, Signal Image Video Process. 12 (3) (2018) 523–530.
[16] Y. Zhang, C. Zhu, Y. Lin, J. Zheng, Y. Wang, An efficient partition scheme for depth-based block partitioning in 3D-HEVC, in: Pacific Rim Conference on Multimedia, Springer, 2015, pp. 428–436.
[17] S. Poularakis, K. Avgerinakis, A. Briassouli, I. Kompatsiaris, Efficient motion estimation methods for fast recognition of activities of daily living, Signal Process., Image Commun. 53 (2017) 1–12.
[18] M. Manafifard, H. Ebadi, H. Moghaddam, Appearance-based multiple hypothesis tracking: Application to soccer broadcast videos analysis, Signal Process., Image Commun. 55 (2017) 157–170.
[19] C. Zhu, W.-S. Qi, W. Ser, Predictive fine granularity successive elimination for fast optimal block-matching motion estimation, IEEE Trans. Image Process. 14 (2) (2005) 213–221.
[20] J.-F. Yang, S.-C. Chang, C.-Y. Chen, Computation reduction for motion search in low rate video coders, IEEE Trans. Circuits Syst. Video Technol. 12 (10) (2002) 948–951.
[21] C.-H. Cheung, L.-M. Po, Adjustable partial distortion search algorithm for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol. 13 (1) (2003) 100–110.
[22] T.S. Shinde, A.K. Tiwari, Efficient direction-oriented search algorithm for block motion estimation, IET Image Process. 12 (9) (2018) 1567–1576.
[23] C. Ma, D. Liu, X. Peng, L. Li, F. Wu, Traffic surveillance video coding with libraries of vehicles and background, J. Vis. Commun. Image Represent. (2019).
[24] L. Zhao, Y. Tian, T. Huang, Background-foreground division based search for motion estimation in surveillance video coding, in: 2014 IEEE International Conference on Multimedia and Expo (ICME), IEEE, 2014, pp. 1–6.
[25] Y. Lu, H. Liu, Y. Lin, L. Shen, H. Yin, Efficient coding mode and partition decision for screen content intra coding, Signal Process., Image Commun. 68 (2018) 249–257.
[26] W. Gao, Y. Tian, T. Huang, S. Ma, X. Zhang, The IEEE 1857 standard: Empowering smart video surveillance systems, IEEE Intell. Syst. 29 (5) (2014) 30–39.
[27] N.-C V. Lab, YUV test sequences, 2019, http://vip.cs.nctu.edu.tw/resource_seq.html.
[28] I. Lab, Subjective quality video database, 2011, http://ivp.ee.cuhk.edu.hk/research/database/subjective/index.html.
[29] K. Seshadrinathan, R. Soundararajan, A.C. Bovik, L.K. Cormack, Study of subjective and objective quality assessment of video, IEEE Trans. Image Process. 19 (6) (2010) 1427–1441.
[30] C. Lin, Y. Lin, H. Hsieh, Multi-direction search algorithm for block motion estimation in H. 264/AVC, IET Image Process. 3 (2) (2009) 88–99.
[31] L. Lin, I.-C. Wey, J.-H. Ding, Fast predictive motion estimation algorithm with adaptive search mode based on motion type classification, Signal Image Video Process. 10 (1) (2016) 171–180.
[32] X. Yi, N. Ling, Improved normalized partial distortion search with dual-halfway-stop for rapid block motion estimation, IEEE Trans. Multimedia 9 (5) (2007) 995–1003.
[33] P. Liu, K. Jia, Low-complexity saliency detection algorithm for fast perceptual video coding, Sci. World J. 2013 (2013).
[34] S. Zhang, K. Wei, H. Jia, X. Xie, W. Gao, An efficient foreground-based surveillance video coding scheme in low bit-rate compression, in: 2012 Visual Communications and Image Processing, IEEE, 2012, pp. 1–6.
[35] G. Bjontegaard, Calculation of average PSNR differences between RD curves, presented at the 13th VCEG-M33 Meeting, Austin, TX, Apr. 2001, 2001.
[36] R. Mukherjee, P. Saha, I. Chakrabarti, P.K. Dutta, A.K. Ray, Fast adaptive motion estimation algorithm and its efficient VLSI system for high definition videos, Expert Syst. Appl. 101 (2018) 159–175.