The trade-off between accuracy and the complexity of real-time background subtraction

Md Alamgir Hossain | VanDung Nguyen | Eui-Nam Huh

Department of Computer Science and Engineering, Kyung Hee University, Yongin-si, Gyeonggi-do, Republic of Korea

Correspondence
Eui-Nam Huh, Department of Computer Science and Engineering, Kyung Hee University, 1732, Deogyeong-daero, Gijeong-gu, Yongin-si, Gyeonggi-do, 17104, Republic of Korea.
Email: johnhuh@khu.ac.kr

Abstract
Background subtraction used in object detection, tracking and action recognition is a typical method that separates foreground objects from the background. These applications require accuracy and a complexity reduction technique. Some approaches have been proposed to either increase accuracy or decrease complexity. However, the trade-off between increasing accuracy and reducing the complexity of background subtraction is a big challenge. To address this issue, a background subtraction-based real-time moving object-detection approach is proposed. The key contribution in authors’ proposal is to use a colour image and a novel colour-gradient blending fused image to achieve accurate background/foreground segmentation. The fused image is a combination of a gradient image and a colour image to correct illumination variations and preserve the edge information. Also, thresholds are adaptively selected based on the dynamic background behaviour to attain a more robust classification system. The proposed model based on real-time and complex videos from the CD-2012 and CD-2014 change detection data sets, and the CMD data set is evaluated. Experimental results indicate that authors’ method processes around 43 frames per second and requires six bytes of memory per pixel, which is noticeably more efficient and less complex than other background subtraction methods.

1 | INTRODUCTION
For many years, background subtraction (BGS) has attracted a lot of attention in computer vision and image processing for its broad array of applications. BGS extracts moving objects from a video. The approach needs to model the background and must capture different types of background to detect moving foreground (FG) objects. A frame-difference BGS approach detects moving objects from the absolute intensity difference between the observed frame and the reference frame. On the other hand, some sample consensus approaches used local binary pattern (LBP) [1], local binary similarity pattern (LBSP) [2, 3], local singular value decomposition binary pattern (SVDBP) [4], or gradient magnitude (GM) [5] in addition to colour to increase accuracy. Although the additional features with colour-based BGS increase accuracy to some extent, the method brings extraneous complexity. Therefore, to solve the complexity-versus-accuracy problem, we utilize a colour-feature and novel colour-gradient blending fused feature for background/foreground segmentation. The new fused feature is created from the colour feature and the gradient feature.

There have been many proposed approaches to object detection, all with different goals. We classify object detection methods into two categories: typical object detection methods and moving-object detection methods. The typical object detection approach is the process of segmenting objects, such as humans, cars, trees, and buildings in images or videos. On the other hand, the latter approach classifies objects in motion, such as humans, vehicles, and animals. BGS is mainly used in background reconstruction, moving-object classification, and moving-object detection. The moving object-related applications are in intelligent visual surveillance systems, like road traffic surveillance, video surveillance, and maritime and airport monitoring [6, 7]. Other applications include human–computer interaction, traffic monitoring, real-time gesture recognition, content-based video coding, optical motion capture, object tracking, and activity recognition [6, 8, 9]. The demand for real-time moving-object detection is growing nowadays. Real-time
moving-object detection and tracking are promising applications in the smart home, the smart environment, smart cities, robotics, and drones. The Internet of Things, cloud computing, fog computing, and edge computing also use detection services for surveillance [10]. Many sensors have limited memory and processing capabilities. Hence, we need applications that utilize fewer resources and yet still provide accuracy. According to our recent review of moving-object detection, we conclude that although some methods are quite effective in terms of accuracy, their complexity is not acceptable for extensive use. Compared to a shallow learning-based method, an accurate deep learning-based method incurs huge computational costs, and requires a large volume of ground truth data to train the model. Bouwmans and Garcia-Garcia [6] identified the present trends in the BGS approaches in terms of effectiveness, memory usage, and time complexity. Therefore, we propose a BGS-based real-time moving-object detection approach. The proposed method can be applied in limited-computing and low memory-space devices, and is effective and efficient. Moreover, it can work well despite dynamic backgrounds, thermal effects, unstable video, bad weather, and turbulence.

There are many challenges in BGS or moving foreground detection, such as illumination variations, light switching, dynamic backgrounds, shadows, motionless moving objects, camouflage, sleeping foreground objects, camera jitter [6], noisy images, inserted background objects [11], and so on. To solve such challenges, many methods have been proposed. A frame-difference method [12, 13] was proposed for background/foreground segmentation that was effective to some extent for non-challenging video cases. Although the method can quickly classify background and FG, it is very sensitive to illumination variations and camera jitter. But, this kind of approach fails when there are either mild or large variations in the background of a video. Later on, the statistical method [14–20] and the codebook method [21] tried to make an illumination-invariant pixel model. Although the accuracy of this method increased compared to the frame-difference method, the method underperforms when dealing with background variations, just like the frame-difference method. Also, the sample consensus-based approaches [2, 22, 23] achieved some success with intensity changes, but the methods sharply decreased in accuracy when there are large, unexpected changes in a video. To work in dynamic backgrounds (either mild or large changes in background), sample consensus-based approaches [2–5, 24, 25] added some pixel descriptors to create an illumination- and shadow-invariant model, and introduced a dynamic background control mechanism. Later on, deep learning-based BGS (DeepBS) [26] exploited two methods [3, 19] to create backgrounds to train and test the model. Although the methods [3, 26] increased performance to some extent, they incurred unnecessary costs. Therefore, it is difficult to exploit these approaches in real-time applications.

Some BGS approaches [1, 5, 27, 28] dispense local descriptors, like LBP, LBSP, and GM to get the texture and neighbour information, correct the intensity, and remove shadow. The LBP- or LBSP-based BGS incurs a higher cost than gradient-based BGS. In LBP [1, 27], a local binary pattern emerges from matching each neighbouring pixel with the centre pixel based on a constant threshold. The threshold is some fraction of the centre pixel. Unlike LBP, LBSP [28] encodes the pattern based on the similarity between the centre pixel and the neighbouring pixels. Although LBP and LBSP descriptors are said to be intensity-invariant, they fall in a flat region, as shown in Figure 1, which results in false detections. The window size of the descriptors affects intensity correction. For example, considering the same window size of 5 \times 5 (Figure 1), the LBSP pattern shows more variations in frame #160 than the corresponding patterns in frame #1 and frame #83. Therefore, large block-size descriptors are very sensitive to mild variations in the background. In the same way, LBP and GM also exhibited different patterns and magnitudes for those three frames. On the other hand, in our design, a colour-gradient blending fused image (FI) with a 3 \times 3 kernel shows the same values in cases with only the background, the shadow, and the front side of the foreground. The FI can reduce illumination variance. In conclusion, the novel FI can detect all the edges and correct the intensity to some extent at the same time.

Deep learning-based typical object-detection approaches, such as the fast region-based convolutional neural network (Fast R-CNN) [29], Faster R-CNN [30], you only look once (YOLO) [31], and Mask R-CNN [32] were proposed for accurate object detection. However, those methods did not mention how to detect moving objects. Alternatively, the traditional image processing-based methods such as Visual Background extractor (ViBe) [23], ViBe+ [24], pixel-based adaptive segmenter (PBAS) [5], LOBSTER [2], Self-balanced Sensitivity Segmenter (SuBSENSE) [3], and Sub-super pixel-based Background Subtraction (SBS) [20], and machine learning-based approaches such as the weightless neural network (WNN) [33], the CNN [34], and DeepBS [26] were proposed for moving object segmentation. PBAS and SuBSENSE used pixel features and dynamic background control mechanisms for accurate segmentation. Our proposal enhances PBAS and SuBSENSE using the novel FI. Our method outperforms the existing approaches in terms of accuracy and computational complexity.

In this research, we propose a novel FI-based background subtraction (FBGS) approach. First, we create the FI, which is a combination of both the gradient image and the colour image and which leads to reduced noise while preserving edge information. Afterwards, we fuse the FI and the colour image, model the dynamic background characteristics, update the background samples, and select an adaptive threshold for background/foreground classification. Finally, we do post-processing to discard noise in raw background/foreground segmentation. We list our key contributions as follows: (1) We design a novel colour-gradient FI, which is a weighted combination of colour and the GM for effective background/foreground classification. (2) Under different pixel intensity variations, thresholds are effectively selected at runtime to achieve accurate segmentation. (3) Our system is based on colour, the FI, and an adaptive threshold, which can improve the segmentation accuracy in challenging cases. (4) We test our system with 54 videos from the CD-2012 and CD-2014 change detection data sets, and the CMD data set. Our experimental
results demonstrate the performance effectiveness of our proposed system.

In the rest of the paper, Section 2 describes other relevant work. The hyperparameters setup, background model, background/foreground classification, background change measurement, dynamic or non-dynamic region detection, blinking pixel level, threshold factor update, background factor update, background model update decision, background diffusion, unstable video detection, and post-processing are presented in Section 3. We describe the parameters settings, the image denosing effects, the importance of FI magnitude, a comparison of methods, and complexity measurements in Section 4, and we discuss the important aspects of our method in Section 5. Conclusions and future work are elaborated on in Section 6.

2 RELATED WORKS

From the aspect of high-accuracy performance, many BGS approaches have been proposed. BGS approaches can be classified into basic/simple methods, statistical- and cluster-based methods, sample consensus-based methods, and learning-based methods. Figure 2 depicts the taxonomy of the BGS algorithm.

(1) Simple method. A background model is based on two- or three-frame differences, persistent frame differences, averages, medians, and histograms of pixels in a video. In the two-frame difference method [13] and the block-frame difference method [35], an observed video frame is subtracted from a reference background model to detect moving objects. While the absolute difference between two frames extracts some moving objects, it results in false classifications if there are dynamic backgrounds, intensity changes, colour variations, shadows, unstable video, start-up objects, and so on. The method can only detect the boundary edge of evenly coloured intensity or grey intensity objects. In a median-based BGS, the median of each pixel over the temporal direction separates the background from the FG. The temporal median-based BGS fails when each pixel is observed in less than 50% of the total observations. Spectral-360 [36] exploited an image formation model and colour spectrum reflectance, and classified the foreground
from the video based on full-colour spectrum reflectance feature resemblance, instead of colour/grey intensity similarity in a pixel.

(2) Statistical- and cluster-based BGS methods. In the mixture of Gaussians (MOG) model [37], a pixel belongs to more than one Gaussian distribution, whereas Zivkovic and Van Der Heijden [17] used the K-nearest neighbours (KNN) algorithm for BGS, where a pixel belongs to a single cluster. Although MOG and KNN operate well for non-dynamic backgrounds, they fail to segment a moving object from a multi-modal background. In the Gaussian mixture model (GMM) [14], Stauffer and Grimson observed the pixels’ intensity behaviour (road and water surface pixels, and the monitor flickering pixel) over time, and found a lot of intensity variations. The variation remained approximately constant for a static environment under a fixed lighting condition. The GMM cannot accurately track an object inserted into a scene, since it does not consider recent pixel behaviour. In addition, the GMM results in a false classification when an object stays static for a long time, and then starts to change position. A Gaussian distribution can be exploited for BGS when there are small, grey-intensity or colour-intensity variations in a pixel, rather than dynamic backgrounds such as rippling water, a plant’s leaves swinging, and the like. Wang et al. [19] presented a method named flux tensor with a split Gaussian (FTSG) model, a combination of the spatio-temporal motion detector called a flux tensor and the mixture of a Gaussian distribution for BGS. Unlike MOG, which shares the same fixed number of Gaussian distributions for both foreground and background, FTSG brings one Gaussian distribution for the foreground and one Gaussian mixture for the background. It is known that MOG is not illumination- and shadow-invariant. Also, the flux tensor cannot detect soft motion. Instead of the per-pixel similarity measure, Chen et al. [20] measured a superpixel-by-superpixel analogy and called it SBS. The superpixel was constructed using a simple linear iterative clustering (SLIC) algorithm, and then, the superpixel was subsequently clustered into a sub-superpixel with the help of a K-means algorithm. Moreover, the process of intensity correction could not be more effective, because intensity varies randomly in reality.

(3) Sample consensus-based method. At first, Wang and Suter [22] proposed the novel concept of a sample consensus (SACON) for BGS. Later on, Barnich and Van Droogenbroeck presented ViBe [23] to classify the background or FG. Like SACON, ViBe is also a sample consensus-based BGS approach. Unlike SACON, which updates blob samples and pixel samples simultaneously, ViBe updates only pixel samples randomly over time, based on a conservative update policy. In the conservative update policy, the foreground will never be considered as background in the whole BGS process. Therefore, if moving objects stop, the objects will not be background objects anymore. Afterwards, Van Droogenbroeck et al. improved ViBe, naming it ViBe$^+ [24]$, introducing an updating factor of $1/5$ (because of large global motion due to unstable video) instead of one random selection out of 20 pixels (background sample), and added a new post-processing operation. Unlike ViBe, St. Charles et al. utilized the LBP descriptor and colour feature in the LOBSTER [2] and SubSENSE [3] methods. Both ViBe and LOBSTER create the same background modelling policy, diffusion mechanism, and median filter-based post-processing. In the same way, Hofmann et al. [5] also came across a sample consensus-based BGS called the pixel-based adaptive segmenter. For the first time, PBAS created a dynamic background modelling mechanism after realizing the essence of updating stored background samples and thresholds for the dynamic background. Therefore, PBAS introduced two parameters: the decision threshold, which was updated dynamically at a fixed scale, and a learning rate calculating the degree of background dynamics in order to update background at runtime. An extension of PBAS introduced by Tiefenbacher et al. [25] used a proportional-integral-derivative (PID) regulator to manage the decision threshold and to change the update rate variables more effectively. In 2015, St. Charles et al. improved their previous LOBSTER method by introducing intra-LBSB, inter-LBSB, blind update (the background could be updated by both foreground pixel and background pixel), a dynamic background modelling mechanism like PBAS, and adaptive post-processing. The goal of this research work is to detect moving objects on a dynamic environment like PBAS and SubSENSE. To clearly explain the difference between PBAS, SubSENSE, and our FBGS architecture with classic dynamic-update-based methods, we present the key performance indicators used in each approach and the corresponding operations in Table 1. In 2016, to make a shadow- and illumination-invariant BGS system, Guo et al. [4] converted SVD of an image block (normalized SVD), and finally encoded the LBP pattern of the normalized SVD. Although Guo et al. [4] claimed that SVDBP was more illumination- and shadow-invariant than LBP, the performance of this approach was not as good as the LSBP-based SubSENSE [3].

(4) Learning-based method. A self-organizing method called spatially coherent self-organizing background subtraction (SC-SOBS) [38] automatically creates a neural network model for background/foreground segmentation. SC-SOBS was proposed to work in a challenging environment, such as a dynamic background, with slow illumination changes, and with camouflage problems. In 2014, the WNN [33] was proposed to adapt to random behaviour in the background and to control the effectiveness of the BGS system using colour intensity history. Braham et al. [34] proposed a CNN-based BGS that matched patchwise background images created by the median-based method [39]. Babaei et al. [26] developed DeepBS, and DeepBS matched the image patch of the background frame generated by the combination of SubSENSE [3] and FTSG [19], rather than only the frame-difference BGS in the previous approach [34]. The DeepBS method is not completely error-free, because a portion of the image patch could be foreground and background at the same time. Also, a background image created by the
TABLE 1

| Key performance indicator | Method | FE1 | FE2 | background model | TH | Operation |
|---------------------------|--------|-----|-----|------------------|----|-----------|
|                           | PBAS   | Colour | GM | Dynamic          | Dynamic | The segmentation decision is only based on a weighted fusion of colour and gradient magnitude. In addition, in PBAS, the background and the threshold, respectively, are updated according to a conservative policy or the minimal distance between an observed pixel and its stored background pixel. |
|                           | SuBS   | Colour | LBSP | Dynamic          | Dynamic | The segmentation decision is based on colour and LBSP. Moreover, SuBS updates the background and the threshold according to a blind policy and a threshold factor, respectively. The threshold factor is determined not only based on the minimal distance between an observed pixel and its stored background pixel, like PBAS, but also from the noise of the raw segmentation of the previous frame. |
|                           | FBGS   | Colour | FL | Dynamic          | Dynamic | FBGS effectively uses colour, the FL, and a fusion of colour and the FI for the segmentation decision. Also, like SuBS, FBGS uses a blind policy and a threshold factor to update the background and threshold, respectively. But under different pixel intensity variations, the thresholds are effectively selected at runtime to achieve accurate segmentation. |

Abbreviations: BG, background; FBGS, fused image-based background subtraction; FE1, Feature 1; FE2, Feature 2; FI, fused image; GM, gradient magnitude; LBSP, local binary similarity pattern; PBAS, pixel-based adaptive segmenter; SuBS, self-balanced sensitivity segmenter; TH, threshold factor.

3 | METHOD PRELIMINARIES

The sample consensus-based method is a pixel-by-pixel background/foreground classification method. Hereinafter, we define background samples to represent background pixels. In sample consensus-based BGS, neighbours of each pixel are separately stored as background samples from the first video frame. The background samples are updated based on the background condition over time. Then, an observed pixel from the next video sequences is segmented as an foreground if a certain number of background pixels do not match the observed pixel. The proposed FBGS method uses our newly created colour-gradient blending FL and a colour image to extract the foreground from the BG. We define notations in Table 2. According to LOBSTER [2], SuBS [3], PBAS [5], ViBe [23], and ViBe [24], these hyperparameters are found empirically. The best values of these hyperparameters are used to achieve the best performance results. Figure 3 depicts the FBGS method. The FBGS model has 13 blocks, as follows.

1. **Fl creation.** The FI is a combination of a colour image and a gradient image. The gradient image is defined as an image that takes a GM of a colour intensity image for each pixel. The GM shows a directional change in the colour intensity of a colour-intensity image. The specific percentage of colour/grey intensity and GM are combined to create the new FI (see Section 3.1).

2. **Background model.** The background model is created from the background samples. The background sample is the neighbour pixels of a pixel for which the background model is initialized. Two background samples are created for each pixel from the first colour/grey image and the FL frame separately (see Section 3.2).

3. **Background/foreground classification.** The classification starts from the second frame of the video sequence based on similarity/dissimilarity of the pixel’s features. The colour/grey intensity and fused magnitude are the features of each pixel (see Section 3.3).

4. **Background change measurement.** The minimum distance between a pixel and its background samples is estimated in order to measure the background change, which is used to update the threshold factor and background factor (see Section 3.4).

5. **Dynamic or non-dynamic region detection.** Tree leaves rustling, a fountain, light fluctuations, and the like constitute a dynamic background; otherwise, the background is static (see Section 3.5).

6. **Blinking pixel detection.** A blinking pixel is defined as a pixel that switches between foreground and BG. The number of blinking pixels is estimated from raw segmentation. This level is used to decide the degree of change in the threshold factor and background factor (see Section 3.6).

7. **Threshold factor update.** The threshold factor is updated over time based on a minimal distance between a pixel and its background samples. This factor is utilized to compute the decision thresholds (see Section 3.7).

8. **Background factor update.** The background factor is used to calculate the probability of a background sample update and the probability of diffusion (see Section 3.8).

9. **Decision threshold.** The decision threshold determines background or foreground pixels in the background/foreground classification (see Section 3.9).

10. **Background model update decision.** The background samples need to be updated over time. The detected background pixel replaces the background sample based on the probability of pixel replacement (see Section 3.10).
### TABLE 2  Description of variables and notations

| Variable/notation | Value | Description/Definition |
|-------------------|-------|------------------------|
| $x$               | −     | The row number of a video frame |
| $y$               | −     | The column number of a video frame |
| $I$               | −     | The colour/intensity an image frame |
| $F$               | −     | The fused image frame |
| $K_x$             | −     | The horizontal convolution kernel |
| $K_y$             | −     | The vertical convolution kernel |
| $#_{match}$      | −     | The number of matching similarities |
| $\lor$            | −     | The logical OR operator |
| $L_1(A, B)$       | −     | The Manhattan distance between $A$ and $B$; let the intensity of a pixel $= A$, and the pixel's background sample intensity or the pixel's intensity of the previous frame $= B$ |
| $L_{1\text{ min}}(A, B)$ | − | The minimum Manhattan distance between $A$ and $B$ |
| $T_I(x, y)$       | −     | The decision threshold of a colour/grey image pixel |
| $T^F_I(x, y)$     | −     | The decision threshold of a fused image pixel |
| $H(x, y)$         | −     | The threshold factor; initially, $H(x, y) = 1$ |
| $S(x, y)$         | −     | The stable or unstable background region indicator; initially, $S(x, y) = 0$ |
| $\neg$            | −     | The complement of $S(x, y)$ |
| $G(x, y)$         | −     | The ghost artefacts |
| $B(x, y)$         | −     | The blinking pixel |
| $G_{\text{max}}$ | 0.995 | The minimum ghost artefacts |
| $G_{\text{min}}$ | 0.010 | The maximum ghost artefacts |
| $\alpha$          | 25    | The short-term moving average weight factor |
| $\beta$           | 100   | The long-term moving average weight factor |
| $R_{\text{seg}}$  | 3     | The minimum segmentation fault |
| $R_{\text{bg}}$   | 0.1   | The minimum unstable ratio of the background region |
| $W_{icre}$        | 1     | The degree of increment in blinking pixel approximation |
| $W_{dcre}$        | 0.1   | The degree of decrement in blinking pixel approximation |
| $Z$               | 0.01  | The threshold factor step size |
| $U_{icre}$        | 0.50  | The degree of the background factor increment |
| $U_{dcre}$        | 0.25  | The degree of the background factor decrement |
| $\gamma$          | 0.60  | The weight factor to fuse colour and normalized gradient pixels |
| $N$               | 49    | The number of background samples |
| $t$               | 1, 3  | The fusion weight factor of colour/grey image pixel and fused image pixel; for a colour pixel, $t = 1$; for a grey pixel, $t = 3$ |
| $#_{\text{min}}$ | 2     | The minimum number of required matches |
| $T_m^I$           | 30    | The minimum threshold of a colour/grey image pixel |
| $T_m^O$           | $T_m^I / 5$ | The threshold offset of a colour/grey image pixel |
| $T_m^F$           | 26    | The minimum threshold of a fused image pixel |
| $T_O^F$           | $T_m^F / 5$ | The threshold offset of a fused image pixel |
| $U(x, y)$         | [4, 256] | The background factor; initially, $U(x, y) = 4$; the lower limit of $U(x, y) = 4$; the upper limit of $U(x, y) = 256$ |
| $P(x, y)$         | $1 / U(x, y)$ | The probability of a background model update |
| $\text{maxind}$  | 255   | The maximum intensity distance |
(11) **Background diffusion.** In addition to background sample replacement, the neighbouring pixels’ background samples are replaced with the detected background pixels based on the probability of a background update, the ghost artefacts, and the background condition (dynamic or static) (see Section 3.11).

(12) **Unstable video detection.** An unstable video can be caused by a non-static camera. The unstable video replaces the background sample with background pixels more frequently, and increases the threshold more sharply (see Section 3.12).

(13) **Post-processing.** Post-processing discards noise (isolated foreground pixels) and fills the unfilled region of objects detected during background/foreground classification (see Section 3.13).

### 3.1 FI creation

In this subsection, we detail our FI-generation model, which takes red–green–blue (RGB) or greyscale images as input. Figure 4 illustrates the FI creation process. The model includes three main phases: the gradient image, the normalized image, and the FI.

(1) **Gradient image.** Input to the model can be video frames or image sequences. The video frame and the gradient image are, respectively, denoted as \( I^I(x,y) \) and \( I^G(x,y) \). Here, \( x \) and \( y \), respectively, represent the row and column of the frame. \( I^I(x,y) \) is taken without any smoothing to create \( I^G(x,y) \). We do not smooth \( I^I(x,y) \) during convolution because smoothing erodes edge information in addition to reducing noise. \( I^I(x,y) \) is convoluted with horizontal convolution kernel \( K_x \) and vertical convolution kernel \( K_y \) to, respectively, get the gradient in horizontal direction \( \nabla X \) and vertical direction \( \nabla Y \). Both \( K_x \) and \( K_y \) of \( 3 \times 3 \) kernels are called the Sobel kernel. The asterisk * indicates a two-dimensional convolution operator. Therefore, the gradient vector \( \nabla M \) is represented as \( [\nabla X, \nabla Y]^T \), as in (5). The gradient frame, \( I^G(x,y) \), is calculated using (6).

\[
K_x = \begin{bmatrix}
-1 & 0 & +1 \\
-2 & 0 & +2 \\
-1 & 0 & +1 
\end{bmatrix}, \quad (1)
\]

\[
K_y = \begin{bmatrix}
-1 & -2 & -1 \\
0 & 0 & 0 \\
+1 & +2 & +1 
\end{bmatrix}, \quad (2)
\]

\[
\nabla X = K_x \ast I^I(x,y), \quad (3)
\]

\[
\nabla Y = K_y \ast I^I(x,y), \quad (4)
\]

\[
\nabla M = [\nabla X \nabla Y]^T, \quad (5)
\]

\[
I^G(x,y) = \sqrt{\nabla X^2 + \nabla Y^2}. \quad (6)
\]

(2) **Normalized image.** The gradient image, \( I^G(x,y) \), is normalized to ensure it is in the range \([0, 255]\) as input image \( I^I(x,y) \). Normalized gradient image, \( I^{NG}(x,y) \), is calculated as

\[
I^{NG}(x,y) = \frac{255 \times I^G(x,y) \max(I^G(x,y))}{\max(I^G(x,y))}. \quad (7)
\]

(3) **FI.** Before fusion, a Gaussian function is used to smooth \( I^I(x,y) \), which results in \( I^I(x,y) \). Consequently, we combine
The colour-gradient blending rule-based fused image construction model

$\nabla X = K_x \cdot I^I(x, y)$

$\nabla Y = K_y \cdot I^I(x, y)$

$I^F(x, y) = \sqrt{(\nabla X)^2 + (\nabla Y)^2}$

$\gamma$ is set to 0.60, after experiment explained in Section 4.1.1. $I^F(x, y)$ preserves sharp edges and reduces noise at the same time. The GM of an image is not illumination-invariant. When the intensity of the neighbours of observed pixel $I^I(x, y)$ increases, the GM of the observed pixel also increases; and while the intensity of the neighbours of observed pixel $I^I(x, y)$ decreases, the GM also decreases. That is why we smooth $I^I(x, y)$ when combining it with $I^NG(x, y)$ in (8), because we are interested in taking an average colour/grey intensity change of a pixel, since colour/grey intensity change can be noise or real motion. Although $I^F(x, y)$ is not completely illumination-invariant, $I^F(x, y)$ reduces noise while keeping actual motion and edge information.

3.2 | Background model

RGB or greyscale image $I^I(x, y)$ and FI $I^F(x, y)$ are the first frames, colour/grey image background model $B^I(x, y)$ from $I^I(x, y)$ and FI background model $B^F(x, y)$ from $I^F(x, y)$ are initialized for the first time before background/foreground segmentation, as follows:

$B^I(x, y) = \{B^I_0(x, y), B^I_1(x, y) ... B^I_{N-1}(x, y)\}$

$B^F(x, y) = \{B^F_0(x, y), B^F_1(x, y) ... B^F_{N-1}(x, y)\}$

where each $B^I_i(x, y)$ or $B^F_i(x, y)$, $i \in [0, N - 1]$, is, respectively, called background sample of background model $B^I(x, y)$ or $B^F(x, y)$. Two background samples, $B^I_i(x, y)$ and $B^F_i(x, y)$, are randomly selected from the neighbours of a pixel at coordinates $(x, y)$. The randomly selected background samples are temporarily stored from sample position 0 to sample position $(N - 1)$, consecutively, for $N$ samples. $N$ was fixed at 49 after the many time experiments explained in Section 4.1.2. Noted that the background samples are updated dynamically based on the nature of the region (static or dynamic).

3.3 | Background/foreground classification

After background sample initialization from the first frame, $I^I_1(x, y)$, background/foreground classification starts from the second frame, $I^I_2(x, y)$. According to the sample consensus-based BGS, our FBGS compares each observed pixel, $I^I_2(x, y)$ and $I^I_0(x, y)$ at coordinates $(x, y)$, with background samples
\( B_N^f(x,y) \) and \( B_N^b(x,y) \), which were previously initialized and continuously updated. If the similarity matching number, \#match, of each observed pixel at coordinates \((x,y)\) is less than the minimum number of required matches, \#min, observed pixel \((x,y)\) will be classified as foreground (i.e. 1); otherwise, the pixel will be segmented as background (i.e. 0). The raw segmentation, \( F_R(x,y) \), is represented as expressed in (11). In this equation, \( T_f(x,y) \) is the decision threshold of \( I_f(x,y) \), and \( T_l(x,y) \) is the decision threshold of \( I_l(x,y) \). \( L_f(I_f(x,y), B_N^f(x,y)) \) is the Manhattan distance between intensity pixel \( I_f(x,y) \) and its background sample, \( B_N^f(x,y) \), and \( L_l(I_l(x,y), B_N^l(x,y)) \) is the Manhattan distance between fused pixel \( I_f(x,y) \) and its background sample \( B_N^l(x,y) \). \( L_f(I_f(x,y), B_N^f(x,y)) + 1/t (L_f(I_f(x,y), B_N^f(x,y))) \) is the combination of the Manhattan distances for pixel \((x,y)\). Here, \( t \) is the fusion weight factor.

### 3.4 | Background change measurement

Our background change measurement is similar to those in [3] and [5]. We mention that the dynamic region or non-dynamic region of a video control, \( H(x,y) \) and \( U(x,y) \), can increase or decrease detection accuracy. The difference between the observed background pixel and its background samples, noise level (approximated by blinking pixels), and type of video (stable or unstable) leads to successively determining the nature (dynamic or non-dynamic) of the background. Before calculating short-term moving average \( M_f^s(x,y) \) and long-term moving average \( M_f^l(x,y) \) between a pixel at coordinates \((x,y)\) and its background samples, \( B_N^f(x,y) \) and \( B_N^l(x,y) \), the minimum average distance, \( D_{\text{min}}(x,y) \) between a pixel at coordinates \((x,y)\) and its background samples is estimated as in seen in (12). In that equation, \( L_{\text{min}}(I_f(x,y), B_N^f(x,y)) \) is the minimum Manhattan distance between observed pixel \( I_f(x,y) \) and its background samples, \( B_N^f(x,y) \); \( L_{\text{min}}(I_l(x,y), B_N^l(x,y)) \) is the minimum Manhattan distance between observed pixel \( I_l(x,y) \) and its background samples, \( B_N^l(x,y) \); and \( \text{maxIDst} \) is the maximum intensity distance used to normalize the distance. The range of \( D_{\text{min}}(x,y) \) is \([0, 1]\).

Afterwards, we determine short-term moving average, \( M_f^s(x,y) \), and long-term moving average, \( M_f^l(x,y) \), as follows:

\[
M_f^s(x,y) = (1 - \alpha) M_f^s(x,y) + \alpha D_{\text{min}}(x,y),
\]

\[
M_f^l(x,y) = (1 - \beta) M_f^l(x,y) + \beta D_{\text{min}}(x,y),
\]

where \( \beta \) is the long-term learning rate, and \( \alpha \) is the short-term learning rate. Note that these two parameters, \( \alpha \) and \( \beta \), have the same meaning throughout our proposed method. \( M_f^s(x,y) \) and \( M_f^l(x,y) \) fall within the range \([0, 1]\). In a completely static background area, \( M_f^s(x,y) \approx 0 \) and \( M_f^l(x,y) \approx 0 \). And for a fully dynamic region, \( M_f^s(x,y) \approx 1 \) and \( M_f^l(x,y) \approx 1 \).

\[
F_R(x,y) = \begin{cases} 
1, & \text{if } \#\text{match} \left\{ L_f(I_f(x,y), B_N^f(x,y)) < T_f(x,y) \lor (L_f(I_f(x,y), B_N^f(x,y)) \lor L_l(I_l(x,y), B_N^l(x,y)) < T_l(x,y) \right\} < \#\text{min}, \\
0, & \text{otherwise,}
\end{cases}
\]

\[
D_{\text{min}}(x,y) = \min(1, (L_{\text{min}}(I_f(x,y), B_N^f(x,y))/\text{maxIDst} + L_{\text{min}}(I_l(x,y), B_N^l(x,y))/\text{maxIDst}))/2),
\]

\[
S(x,y) = \begin{cases} 
1 & \text{if } H(x,y) > R_d \lor R_I(x,y) > R_n \lor R_c(x,y) > R_n, \\
0 & \text{otherwise},
\end{cases}
\]

\[
W'(x,y) = \begin{cases} 
W'(x,y) + W_{\text{ser}}, & \text{if } \max(M_f^s(x,y), M_f^l(x,y)) > R_n \land B(x,y) = T, \\
W'(x,y) - \frac{4}{2} W_{\text{ser}}, & \text{else if } F_{O-1}^R(x,y) = T, \\
W'(x,y) - \frac{2}{2} W_{\text{ser}}, & \text{else if } S(x,y) = T, \\
W'(x,y) - W_{\text{der}}, & \text{otherwise},
\end{cases}
\]

\[
H(x,y) = \begin{cases} 
H(x,y) + Z(W'(x,y) - W_{\text{ser}}), & \text{if } H(x,y) \leq (1 + \min(M_f^s(x,y), M_f^l(x,y))2)^2, \\
H(x,y) - Z W'(x,y), & \text{otherwise},
\end{cases}
\]

\[
U(x,y) = \begin{cases} 
U(x,y) + \frac{U_{\text{ser}}}{W'(x,y) \max(M_f^s(x,y), M_f^l(x,y))}, & \text{if } F_{O-1}^R(x,y) = T \land (\min(M_f^s(x,y), M_f^l(x,y)) < R_n \land F_{O-1}^R(x,y) = T), \\
U(x,y) - \frac{U_{\text{ser}}}{\max(M_f^s(x,y), M_f^l(x,y))}, & \text{otherwise.}
\end{cases}
\]
3.5 Dynamic or non-dynamic region detection

The state (dynamic or non-dynamic) of a region has a very important effect on segmentation. Rather than global measurement, we calculate the local behaviour of the background. A dynamic background does not appear at the same magnitude, and also does not happen in all the regions of the frame, except in unstable videos. In the case of an unstable video, we know that all the pixels transform the same amount between consecutive frames, on average, because of the camera movement. In a dynamic BG, for example, in some regions, tree leaves move, fountains spray water, and in the rest of the frame, nothing happens. Also, pixel-to-pixel motion within a dynamic background region varies. Therefore, we approximate the segmentation fault to separately identify dynamic or non-dynamic background \( S(x, y) \) per region. We calculate per-pixel short-term segmentation fault \( R^T(x, y) \) and long-term segmentation fault \( R^L(x, y) \) in the following way:

\[
R^T(x, y) = (1 - \alpha)R^T(x, y) + \frac{\alpha}{\text{newD} \times \text{oldD}}F^R_O(x, y),
\]

\[
R^L(x, y) = (1 - \beta)R^L(x, y) + \frac{\beta}{\text{newD} \times \text{oldD}}F^R_O(x, y).
\]

We apply this normalization to keep the range of \( R^T(x, y) \) and \( R^L(x, y) \) within [0, 1]. Wrongly classified foreground produces a false positive. Therefore, dynamic or non-dynamic region \( S(x, y) \) is calculated as expressed in (17). In that equation, \( H(x, y) \) is the threshold factor in (19), \( R_{dm} \) is the minimum threshold, and \( R_m \) is the minimum unstable ratio of the background region. Logical variable \( S(x, y) \) is either 1 or 0. \( S(x, y) = 1 \) means a dynamic pixel; otherwise \( S(x, y) = 0 \) indicates a non-dynamic pixel. The flag \( S(x, y) \), increases or decreases decision thresholds \( T^T(x, y) \) and \( T^L(x, y) \) in (21) and (22).

3.6 Blinking pixel level

FBGS calculates pixel-level classification errors by identifying blinking pixels and average background noise. Note that the blinking pixels are identified like the methods [3, 5, 24]. In FBGS, blinking pixels reside either in observed classified frame \( F^R_O(x, y) \) or the previously segmented frame, \( F^R_{O-1}(x, y) \), but the pixels do not remain in both frames. Thus, the easiest way to find blinking pixel \( B(x, y) \) is to use the XOR operation between \( F^R_O(x, y) \) and \( F^R_{O-1}(x, y) \). Some errors occur if a direct XOR is carried out between \( F^R_O(x, y) \) and post-processed \( F^R_{O-1}(x, y) \) since the brink of moving foreground in \( F^R_O(x, y) \) produces extra blinking pixels. Therefore, XOR is taken between the dilated \( F^R_D(x, y) \) and post-processed \( F^R_{O-1}(x, y) \). Therefore, pixel-level classification error \( W'(x, y) \) is adjusted dynamically, as expressed in (18). In that equation, \( R_m \) is the minimum unstable ratio, \( W_{\text{inc}} \) is the variational increment, and \( W_{\text{dec}} \) is the variational decrement. \( W'(x, y) \) is always \( \geq 0 \). \( T \) indicates true. With a small amount of noise, \( W'(x, y) \approx 0 \). This happens when a region has a very low-intensity variation, a completely static background, or very near-to-static background. Alternatively, \( W'(x, y) \approx 1 \) with a completely dynamic region, such as fountains, tree leaves rustling, or very large-intensity variations due to camera internal errors, light fluctuations, and so on.

3.7 Threshold factor update

Now, local distance threshold factor \( H(x, y) \) is updated as expressed in (19) based on \( M^L(x, y) \) and \( M^T(x, y) \), and \( W'(x, y) \). In (19), \( Z \) is the threshold factor step size. \( H(x, y) \) is always \( \geq 1 \).

3.8 Background factor update

In addition to the threshold factor update, background factor \( U(x, y) \) is also continuously updated, as seen in (20). In the equation, \( U_{\text{inc}} \) and \( U_{\text{dec}} \), respectively, are the degree of background factor increment and decrement, while \( R_B \) is the minimum unstable ratio of the background region, and \( U(x, y) \) is the background factor.

3.9 Decision threshold

Two decision thresholds, \( T^T(x, y) \) and \( T^L(x, y) \) in (11), are determined as follows:

\[
T^T(x, y) = (T^T_m H(x, y) - (\neg S(x, y)) T^T_0) / 2,
\]

\[
T^L(x, y) = (T^L_m H(x, y) - (\neg S(x, y)) T^L_0) / 2,
\]

where \( T^T_0 \) is the colour threshold offset, \( T^L_0 \) is the fused threshold offset, \( T^T_m \) is the minimum colour distance threshold, and \( T^L_m \) is the minimum fused distance threshold.

A static region leads to smaller values of \( T^T(x, y) \) and \( T^L(x, y) \). Any changes in the videos are detected as foreground objects owing to these smaller values. On the other hand, a high dynamic region (rustling tree leaves, rippling water, a shaking video, quick light changes etc.) leads to large values for \( T^T(x, y) \) and \( T^L(x, y) \). But then, we cannot detect soft-motion objects, which will increase the false negatives. Therefore, a robust strategy is needed to avoid false positives and false negatives. That is why we introduce pixel-level threshold factor, \( H(x, y) \), just like in [3, 5]. In a dynamic region, \( H(x, y) \) needs to step up to increase \( T^T(x, y) \) and \( T^L(x, y) \) in order to reduce false classifications. On the other hand, for a non-dynamic (static) area, \( H(x, y) \) needs to diminish to decrease \( T^T(x, y) \) and \( T^L(x, y) \) and manage camouflage problems. Also, we bring in pixel-level background factor \( U(x, y) \) like the methods in [3, 5], to measure the degree of dynamic background or non-dynamic background. Note that the probability of a background sample update, \( P(x, y) \), is equal to \( 1 / U(x, y) \). The proposed FBGS randomly updates background samples in \( B_N' (x, y) \) and \( B_N (x, y) \).
based on $P(x, y)$. The dynamic region steps down $U(x, y)$ which, in turn, increases $P(x, y)$. And non-dynamic increases for $U(x, y)$ eventually decrease $P(x, y)$. Moreover, $H(x, y)$ and $U(x, y)$ are two important variables that can increase or decrease the accuracy of segmentation.

### 3.10 Background model update decision

Background model update plays a very important role in a sample consensus-based background/foreground classification. The decision variables $T^f(x, y)$ and $T^F(x, y)$, the minimum number of required samples, $\#_{min}$, the neighbour window size ($(3 \times 3)$ or $(5 \times 5)$), and sample size $N$ all directly affect detection accuracy. There are two background update strategies: a conservative update and a blind update. The conservative update never takes into account foreground-classified pixels in the background samples. So, the problem is that a badly recognized foreground, the object’s moving place, and the foreground classified pixels in the background, whether it is background or foreground. In our proposed method, we consider this policy of update despite one of its drawbacks: slowly moving objects can be assimilated into the background if any condition does not bind it. For this reason, our method increases $U(x, y)$ in the static (non-dynamic) region, and will decrease it in a dynamic region. As a result, the probability of a background sample update becomes low in a non-dynamic region and becomes high in a dynamic region. When an observed pixel, $I^I(x, y)$ or $I^E(x, y)$, is classified as either foreground or background, then the foreground or background pixel has $1/U(x, y)$ chances to randomly replace the background samples in $B^I_{k}(x, y)$ and $B^E_{k}(x, y)$. A dynamic or a non-dynamic background updates $H(x, y)$ and $U(x, y)$ at runtime. Incorrectly increasing or decreasing $H(x, y)$ and $U(x, y)$ results in false detections. Therefore, accurate identification of the nature of a region is needed to correctly adapt these two variables. Based on some conditions, neighbours’ background samples are also replaced in order to preserve spatial consistency, called diffusion. And more background samples are randomly replaced because of unstable video.

### 3.11 Background diffusion

We diffuse the background based on the background’s nature (dynamic or non-dynamic) and ghost artefacts, $G(x, y)$. $G(x, y)$ arises when an object starts moving for the first time. Therefore, $G(x, y)$ is calculated as

$$G(x, y) = \begin{cases} 1 & \text{if } M^I_A(x, y) > G_{min} || M^F_A(x, y) < G_{max} \\ 0 & \text{else} \end{cases}$$

$$D^N(x, y) = \frac{L(I^I(x, y), I^I_{o-1}(x, y))} \text{mxdDist} + \frac{L(I^F(x, y), I^F_{o-1}(x, y))} \text{mxdDist}/2$$

where $G_{min} = 0.995$ is the ghost minimum, $G_{max} = 0.010$ is the ghost maximum, and $M^I_A(x, y)$ is the short-term moving average between consecutive pixels of consecutive frames, not like the short-term moving average between a pixel and its background samples, $M^F_A(x, y)$. $G(x, y) = 1$ if it has a ghost effect. $G(x, y) = 0$ if it does not have a ghost artefact. If $G(x, y) = 1$, the dynamic or non-dynamic region, $S(x, y)$, will lead to, respectively, diffuse $5 \times 5$ and $3 \times 3$ neighbours randomly, based on probability $P(x, y)$.

Before measuring short-term moving average between consecutive pixels of consecutive frames, $M^A_N(x, y)$, the mean of the normalized combined colour and fused distance, $D^N(x, y)$, is approximated, as seen in (24). In this equation, $L(I^I(x, y), I^I_{o-1}(x, y))$ is the Manhattan distance between observed pixel $I^I(x, y)$ and its previous pixel, $I^I_{o-1}(x, y)$, and $L(I^F(x, y), I^F_{o-1}(x, y))$ is the Manhattan distance between observed pixel $I^F(x, y)$ and its previous pixel, $I^F_{o-1}(x, y)$.

Thereafter, per-pixel normalized short-term moving-average distance between consecutive frames, $M^A_N(x, y)$, is estimated as

$$M^A_N(x, y) = (1 - \alpha)M^A_N(x, y) + \alpha D^N(x, y),$$

where the short-term learning rate is $\alpha$. The range of $M^A_N(x, y)$ is $[0, 1]$. For a sudden high dynamic-background region and the foreground, $M^A_N(x, y) \approx 1$. For a static (non-dynamic) background region, $M^A_N(x, y) \approx 0$.

### 3.12 Unstable video detection

Unstable video segmentation is a very challenging problem because both the foreground and the background move. We measured short-term moving average $V^I_m(x, y)$ and long-term moving average $V^E_m(x, y)$ of input video frames $I^I(x, y)$. To find the moving averages, we downsampled $I^I(x, y)$ to create down-sampled version $I^D(x, y)$ to reduce the complexity. Therefore, we calculate the moving averages as

$$V^I_m(x, y) = (1 - \alpha)V^I_m(x, y) + \alpha I^D(x, y),$$

$$V^E_m(x, y) = (1 - \alpha)V^E_m(x, y) + \alpha I^D(x, y).$$

Afterwards, frame-level motion distance $dist(x, y)$ is estimated as

$$dist(x, y) = \frac{|V^I_m(x, y) - V^E_m(x, y)|}{2},$$

when $dist(x, y) > T^I_m/2$, 10% of $B^I(x, y)$ and 10% of $B^E(x, y)$ are randomly replaced from the last video frame, $I^I_{o-1}(x, y)$ and $I^E_{o-1}(x, y)$. In addition, the lower limit and the upper limit of $U(x, y)$ are respectively updated to 2 and 256 when $dist(x, y) > T^I_m/4$. 
### TABLE 3  Six different video categories from CD-2012

| Category                  | Denoted | Description                                                                 |
|---------------------------|---------|-----------------------------------------------------------------------------|
| Baseline                  | BL      | This simple category has two indoor videos and two outdoor videos. Some of the videos contain soft background motion, and others have pedestrians and abandoned objects. |
| Thermal                   | TH      | The videos captured by an infrared camera include the thermal artefact and the camouflage problem. |
| Shadows                   | SH      | This category contains four outdoor and two indoor videos of soft shadows and cast shadows. |
| Dynamic background        | DB      | All six outdoor videos have a lot of background motion.                      |
| Camera jitter             | CJ      | This category includes one indoor and four outdoor unstable videos with different degrees of unwanted jitter. |
| Intermittent object motion| IM      | This video group also contains six videos that have ghost artefacts and intermittent object motion. |

### TABLE 4  Five different video categories from CD-2014

| Category              | Denoted | Description                                                                                                                                 |
|-----------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Low frame rate        | LF      | This category keeps four videos with frames-per-second (fps) changes from 0.17 to 1 fps.                                                   |
| Night video           | NV      | This category incorporates six videos of moving vehicles with low visibility.                                                              |
| Pan-tilt-zoom         | PTZ     | This category includes four videos with different degrees of camera pan, as well as zoom-in and zoom-out.                                   |
| Turbulence            | TB      | This category holds four videos, which were captured at 5–15 km away from the scene by an infrared camera. The video involves heat artefacts, distortion, and different sizes of moving objects. |
| Bad weather           | BW      | The four videos in this category contain a strong winter storm with low visibility.                                                        |

### 3.13  Post-processing

According to [40], a post-processing operation can play a key role in background/foreground segmentation. The post-processing operation is carried out as follows. On the raw segmentation frame, $F_R(x, y)$, a morphological closing is applied followed by a flood fill and ending with the complement operation. A morphological close operation is performed on the raw segmentation frame, $F_R(x, y)$, which is followed by a flood fill operation. Then, the complement operation is accomplished. After that, the bitwise OR operation is performed between $F_R(x, y)$ and this complement output. Finally, we use a median filter with this OR output to get an error-free moving object silhouette, $F_F(x, y)$. The size of the median filter is chosen dynamically, based on the input frame size.

### 3.14  Performance analysis process

We used three data sets (change detection data sets CD-2012 [41] and CD-2014 [42], and CMD [43]) to appraise the performance of our proposed FBGS. CD-2012 contains 31 videos divided into six categories: baseline (BL), thermal (TH), shadow (SH), dynamic background (DB), camera jitter (CJ), and intermittent object motion (IM). Table 3 describes the six different video categories of this data set. And CD-2014 presents 53 actual, complex video sequences of five more categories in addition to the six categories from CD-2012. The five new categories are low frame rate (LF), night video (NV), PTZ, turbulence (TB), and bad weather (BW). We describe these five categories in Table 4.

CMD is a small data set that has one video of 500 frames. The video contains soft camera jitter. We used these data sets because they have challenging sequences in different categories. Also, we can easily compare our FBGS with most of the recent BGS approaches, because those methods measured their performance using these data sets as well. The key performance metrics (KPM) from [41] are utilized to assess the effectiveness of our FBGS approach. Table 5 lists the metrics used for the accuracy calculation.

All approaches are developed in OpenCV C++ 3.3.0 inside the Microsoft Visual Studio 2015 in a computer with an Intel Core i5-8400 2.80 GHz CPU, running the Windows 10 (64 bit) OS with 16 GB of RAM. Memory usage and processing time are estimated in terms of bytes per pixel (bpp) and frames per second (fps), respectively. We take RGB videos with the dimensions of $320 \times 240$ for the complexity comparison.

### 4  EXPERIMENTS AND RESULTS EVALUATIONS

This section includes (1) the determination of parameters, (2) an image denoising effect experiment, (3) the effect of GM and FI magnitude, (4) accuracy under different metrics, and (5) a complexity measurement.

#### 4.1  Determination of parameters

As shown in Figure 3, our system is robust because it consists of many functions that are related together. It is difficult to achieve
TABLE 5  Key performance metrics

| Metric                  | Denoted | Description                                                                 |
|-------------------------|---------|-----------------------------------------------------------------------------|
| True Positive           | TP      | Pixel segmented as foreground pixels.                                       |
| False Positive          | FP      | Background pixel inaccurately segmented as a foreground pixel.             |
| True Negative           | TN      | Pixel that is a background pixel detected as a background pixel.            |
| False Negative          | FN      | Foreground pixel wrongly classified as a background pixel.                  |
| Recall                  | Re      | \( \frac{TP}{TP + FN} \) How many samples a system segments correctly within all positive classes; a large value of Re is better. |
| Specificity             | Sp      | \( \frac{TN}{TP + TN} \) Correctly segmented background samples within the total number of background samples. Sp is also named the true negative rate (TNR); a large value of Sp is better. |
| False Positive Rate     | FPR     | \( \frac{FP}{FP + TN} \) The number of incorrect foreground pixels from all the background pixels; a small value for FPR is better. |
| False Negative Rate     | FNR     | \( \frac{FN}{(FP + FN) \cdot (FP + TN)} \) The number of foreground pixels wrongly segmented as background pixels from the total background pixels squared; a small value for FNR is better. |
| Percentage of Wrong Classification | PWC | \( \frac{1000(FN + FP)}{(FP + FN + FP + TN)} \) The percentage of foreground pixels incorrectly classified as background, and background pixels classified as foreground, out of the total true foreground and background pixels; a small value for PWC/PBC is better. |
| Precision               | Pr      | \( \frac{TP}{TP + FP} \) How correctly a system classifies all positive detections; a large value for Pr is better. |
| F-measure               | F1-score| \( \frac{(2 \cdot Pr \cdot Re)}{Pr + Re} \) Detection accuracy, measured by taking a harmonic mean between recall and precision. F-measure is also called F-score or F1-score; a large value for FM is better. |

FIGURE 5  Input image, raw gradient image, normalized gradient image, and the fused image

Experimental results with different combinations of parameters to determine optimal hyperparameters. Moreover, in each function, we calculate the optimal parameters to get the best performance for the next function. Therefore, we inherited some parameters from existing approaches such as LOBSTER [2], SuBSENSE [3], PBAS [5], ViBe [23], and ViBe + [24] and rest of the parameters are determined the way the existing approaches achieved. We take the following hyperparameters from the existing approaches:

1. \( \#_{\text{min}} = 2 \): The required number of matching samples for classification is chosen similar to ViBe, ViBe +, and PBAS.
2. \( U(x, y) = [4 \ldots 256] \): The lower limit and upper limit of the background factor are similar to SuBSENSE.
3. \( H(x, y) = 1 \): The initial value of the threshold factor is chosen like LOBSTER. Then, this parameter is dynamically changed.
4. \( T_m^I = 30 \): The minimum threshold of a colour/grey image pixel is taken from SuBSENSE.
5. \( W_{\text{inc}}^I = 1, W_{\text{dec}}^I = 0.1 \): The degree of increment/decrement in blinking pixel approximation is similar to SuBSENSE.

However, the optimal values of the rest of hyperparameters were determined experimentally on CD-2012 [41] and CD-2014 [42] data sets, as discussed in the next subsection.

4.1.1  |  FI experiment

Figure 5 shows the input colour image, the raw gradient image, the normalized gradient image, and the novel FI. We created the raw gradient image, normalized gradient image, and novel FI from the input colour image. The raw gradient image looks noisier than the normalized gradient image and the FI. The FI looks sharper, and with less error, than the raw gradient image and the normalized gradient image. The normalized gradient image contains only edge information, whereas the FI exhibits smoothed edge information as well as non-edge information.
The bold face values mean the best performances and the average values.

Table 6

| Colour (%) | GM (%) | CD-2012 | CD-2014 |
|------------|--------|---------|---------|
| 40         | 60     | 0.8088  | 0.7230  |
| 60         | 40     | 0.7995  | 0.7163  |
| 20         | 80     | 0.8085  | 0.7217  |
| 80         | 20     | 0.7926  | 0.7116  |
| 30         | 70     | 0.8067  | 0.7214  |
| 70         | 30     | 0.7966  | 0.7143  |
| 50         | 50     | 0.8035  | 0.7191  |

Table 7

| F1-score |
|----------|
| 0.8088   |

Abbreviations: GM, gradient magnitude

**Figure 6**

F1-score for different numbers of background samples

We experimented to choose the value of $\gamma$ described in Section 3.1. Table 6 shows the colour and GM ratio to create an FI. We selected the ratio based on the accuracy in terms of overall average F1-score for the CD-2012 and CD-2014 data sets. From Table 6, we conclude that accuracy in terms of F1-score increases with the increase in GM. Otherwise, the accuracy decreases with an increase in colour. Moreover, the sum of colour and GM is constant: 100%. Consequently, we find the best accuracy with 40% colour and 60% GM. Therefore, we chose $\gamma = 0.60$ to create the FI.

**4.1.2 Effect of the number of background samples**

The complexity of FBGS does not depend much on the number of background samples, $N$. We selected $N$ based on the effect of a different $N$ on accuracy regarding the overall F1-score. Figure 6 shows the accuracy from a different $N$ with CD-2012, CD-2014 data sets. We see that for CD-2014, the accuracy increased with the increased $N$, but for CD-2012, the accuracy escalated until $N = 49$, and then, the accuracy fell. Note that CD-2014 is a more challenging data set than CD-2012. Therefore, a large number for $N$ performs slightly better for challenging video cases. And we find the best scores for $N = 49$ with all data sets. Therefore, we fixed $N$ at 49.

**4.1.3 Other parameters settings**

According to the experiment shown in Table 7, we get the optimal values of the minimum ghost artefacts, $G_{min} = 0.995$, and the maximum ghost artefacts, $G_{max} = 0.010$. Also, we take the optimal values of the minimum segmentation fault, $R_{min} = 3.000$, and the minimum unstable ratio of the background region, $R_{m} = 0.100$ according to Table 8. Moreover, the optimal value of the threshold factor step size, $Z = 0.01$, is determined based on the experiments shown in Table 9. Consequently, in the same way we determine the rest of the hyperparameters according to experimental results.

**4.2 Image denosing effect**

We analysed the denoising impact on input image $I^I(x, y)$ for background/foreground segmentation by considering the two inputs: $I^F(x, y) + \text{Gaussian smoothing of } I^I(x, y)$ and $I^F(x, y) + \text{non-smoothing of } I^I(x, y)$. Table 10 shows that F1-scores of $I^F(x, y) + \text{non-smoothing of } I^I(x, y)$ are always better than $I^F(x, y) + \text{Gaussian smoothing of } I^I(x, y)$. This happens because the
TABLE 9  
F1-scores for the different values of Z on CD-2012 [41] and CD-2014 [42] data sets

| Z    | F1-score CD-2012 | F1-score CD-2014 |
|------|------------------|------------------|
| 0.005| 0.8060           | 0.7203           |
| 0.010| 0.8088           | 0.7230           |
| 0.015| 0.8041           | 0.7205           |
| 0.020| 0.8048           | 0.7215           |
| 0.025| 0.8039           | 0.7215           |
| 0.030| 0.8043           | 0.7219           |
| 0.035| 0.8050           | 0.7221           |

The bold face values mean the best performances and the average values.

TABLE 10  
Effects on accuracy of smoothing versus non-smoothing in terms of F1-score

| Dataset    | Smoothing | Non-smoothing |
|------------|-----------|---------------|
| CD-2012 [41] | 0.7996    | 0.8088        |
| CD-2014 [42] | 0.7165    | 0.7230        |
| CMD [43]    | 0.9338    | 0.9481        |

The bold face values mean the best performances and the average values.

By analysis, the accuracy difference between color + GM and color + FI, shown in Table 11, we observe that the proposed color + FI always performs better than color + GM. Therefore, we use the color image and the FI for background/foreground segmentation.

TABLE 11  
A comparison of accuracy between Color + GM and Color + FI

| Dataset    | F1-score Color + GM | F1-score Color + FI |
|------------|---------------------|---------------------|
| CD-2012 [41] | 0.7940              | 0.8088              |
| CD-2014 [42] | 0.7165              | 0.7230              |
| CMD [43]    | 0.9338              | 0.9481              |

The bold face values mean the best performances and the average values.

Abbreviations: FI, fused image; GM, gradient magnitude

TABLE 12  
Results of our proposed method for each category of CD-2012 [41], CD-2014 [42], and CMD [43] data sets in term of KPM

| Cate. | Re     | Sp     | FPR    | FNR    | PWC   | Pr     | F1     |
|-------|--------|--------|--------|--------|-------|--------|--------|
| BL    | 0.9401 | 0.9960 | 0.0032 | 0.0599 | 0.5347 | 0.8864 | 0.9112 |
| TH    | 0.7370 | 0.9906 | 0.0094 | 0.2630 | 1.928  | 0.9073 | 0.7924 |
| SH    | 0.8988 | 0.9906 | 0.0094 | 0.1012 | 1.461  | 0.8364 | 0.8623 |
| DB    | 0.9181 | 0.9992 | 0.0008 | 0.0819 | 0.1682 | 0.8625 | 0.8832 |
| IM    | 0.7658 | 0.9365 | 0.0635 | 0.2342 | 6.928  | 0.6572 | 0.6450 |
| CJ    | 0.8729 | 0.9736 | 0.0265 | 0.1271 | 2.953  | 0.7138 | 0.7588 |
| CD-2012 [41] | 0.8554 | 0.9821 | 0.0179 | 0.1446 | 2.329  | 0.8106 | 0.8088 |
| BW    | 0.8078 | 0.9988 | 0.0012 | 0.1922 | 0.4658 | 0.9088 | 0.8517 |
| LF    | 0.7619 | 0.9947 | 0.0053 | 0.2381 | 1.131  | 0.6414 | 0.5982 |
| NV    | 0.7427 | 0.9724 | 0.0276 | 0.2573 | 3.359  | 0.3721 | 0.4615 |
| PTZ   | 0.9077 | 0.9582 | 0.0418 | 0.0923 | 4.235  | 0.2817 | 0.3564 |
| TB    | 0.8089 | 0.9996 | 0.0003 | 0.1911 | 0.1986 | 0.8819 | 0.8317 |
| CD-2014 [42] | 0.8329 | 0.9833 | 0.0167 | 0.1671 | 2.124  | 0.7202 | 0.7230 |
| CMD [43] | 0.9851 | 0.9948 | 0.0052 | 0.0149 | 0.5675 | 0.9138 | 0.9481 |

The bold face values mean the best performances and the average values.

4.3  Effect of GM and FI magnitude

By analysing the accuracy difference between colour + GM and colour + FI, shown in Table 11, we observe that the proposed colour + FI always performs better than colour + GM. Therefore, we use the colour image and the FI for background/foreground segmentation.

4.4  Accuracy under different metrics

We measured the effectiveness of our proposed FBGS with CD-2012, CD-2014, and CMD based on the metrics described in Table 5.

4.4.1  Performance of our method on the three data sets

Table 12 shows the performance of our method for each category (Tables 3 and 4) of CD-2012 and the CD-2014, and CMD data sets in terms of recall (Re), specificity (Sp), FPR, FNR, PWC, precision (Pr), and F1-score. Furthermore, we show the overall average scores of these metrics for these data sets. The overall-average F1-scores are 0.8088, 0.7223, and 0.9481 for the CD-2012, the CD-2014, and the CMD data sets, respectively. We see that F1-scores do not deviate much for the different categories of the CD-2012 dataset. But for LF, NV, and PTZ from the CD-2014 data set, the accuracy in terms of F1-score is lower. The precision falls in challenging cases more than non-challenging cases, but recall does not fluctuate much for the different categories. Most importantly, our proposed FBGS mainly shows better accuracy in challenging cases, like those with shadows, dynamic backgrounds, camera jitter, bad weather, PTZ, and turbulence, compared to the rest of the categories.

Therefore, the performance shows the effectiveness of our proposed approach. Since not all changes in the videos are noise, we created an FI that reduces noise rather than completely removing it. The remaining noise does not affect segmentation as we adaptively determine the dynamic threshold, update the background samples, and do the post-processing.

4.4.2  Comparison of our method with existing methods for the CMD data set

We show the comparison of our proposed FBGS approach with the existing methods in Table 13 in terms of Re, Sp, FPR, FNR, PWC, Pr, and F1-score. Moreover, for the comparison, F1-score and Re mainly focus on representing detection accuracy for
TABLE 13 Comparisons of our method with the existing methods in terms of KPM for the CMD [43] data set

| Method     | Sp  | FPR | FNR | PWC | Re  | Pr  | F1  |
|------------|-----|-----|-----|-----|-----|-----|-----|
| FBGS       | 0.948 | 0.0052 | 0.0149 | 0.5675 | 0.9851 | 0.9138 | 0.9481 |
| SuBS [3]   | 0.9962 | 0.0038 | 0.0405 | 0.5738 | 0.9594 | 0.9333 | 0.9462 |
| PBAS [5]   | 0.9939 | 0.0061 | 0.0433 | 0.8017 | 0.9567 | 0.8955 | 0.9251 |
| SBS [20]   | 0.9975 | 0.0024 | 0.0014 | 0.3862 | 0.9170 | 0.8672 | 0.8914 |
| LOB [2]    | 0.9980 | 0.0020 | 0.1893 | 1.191 | 0.8107 | 0.9565 | 0.8776 |
| ViBE [23]  | 0.9939 | 0.0060 | 0.0018 | 0.780 | 0.8950 | 0.7209 | 0.7986 |
| KDE [44]   | 0.9929 | 0.0070 | 0.0022 | 0.9128 | 0.8754 | 0.6945 | 0.7745 |
| GMM [14]   | 0.9891 | 0.0108 | 0.0026 | 1.319 | 0.8684 | 0.6156 | 0.7205 |

The bold face values mean the best performances and the average values.

background/foreground segmentation in our experiment. The bold, italic, and underlining indicate first, second, and third positions. For the CMD data set in terms of the Re performance metric, our FBGS outperforms at 98.51%. The reason is that our non-smoothing grey colour image, $I(x,y)$, does not lose intensity changes, and our novel FI, $I^F(x,y)$, preserves subtle edge information, which increases true positives. On the other hand, while considering accuracy in terms of F1-score, our proposed method was the best, with 94.81% accuracy, as shown in Table 13. This happened because of our novel FI, which reduces noise due to pixel intensity change, and our threshold selection, which is based on background behaviour.

4.4.3 Comparison of the methods with CD-2012 and CD-2014 data sets

Unlike the accuracy assessment for the CMD data set, we compared our proposed approach with the state-of-the-art approaches based on average F1-score of each video category in the CD-2012 and CD-2014 data sets. The comparison results are shown in Table 14. Like Table 13, bold, italic, and underlining denote first, second, and third positions. Our proposal outperforms for DB and PTZ videos, with 88.32% and 35.64% average F1-scores, respectively. Recall that the backgrounds of DB videos are dynamic, and the PTZ video has the pan-tilt-zoom effect. Our method outperforms these in two cases for two reasons. (1) Our proposed novel FI not only reduces noise but also preserves the edge information, which leads to an increase in Re and Pr. (2) We determine decision thresholds per pixel, based on the nature of the local background region, and we update the background model dynamically, which reduces false positives and improves true positive detection.

4.5 Complexity measurement

According to Bouwmans and Garcia-Garcia [6], lowering the complexity and maintaining the accuracy of BGS is a present research trend. Therefore, complexity reduction in addition to increasing accuracy is an important concern. Table 15 shows the complexity comparison of PBAS, LOBSTER (LOB), SuBS, and ViBE with our FBGS method. The bold, italic, and underlining represent the same positions as in previous tables. In terms of frames per second, PBAS performed the best accuracy, and ViBE took last among the five methods. On the other hand, the proposed FBGS and LOB finished second and third, respectively. PBAS processes 3.73 more frames per second than our FBGS. SuBS displayed 25.02 fps, whereas our method exhibited 42.78 fps, which is 17.58 fps more than SuBS. Although accuracy with SuBS in terms of average F1-score is slightly better, the complexity is greater in terms of frames per second and bytes per pixel. Because DeepBS uses SuBS and FTS for background construction, the complexity of DeepBS is higher than SuBS and our proposed method. In terms of memory requirements, ViBE requires the least space among the five methods. Ours uses 6 bps, which is 9 bps fewer than PBAS, LOB, and SuBS. PBAS, LOB, and SuBS allocate 15 bps. Our proposed FBGS placed second in terms of time complexity and memory usage. On average, our method achieved the best position.

5 DISCUSSION

FBGS proved its effectiveness when we assessed the performance with the three data sets (Tables 12, 13, and 14). Our proposed method exhibited a 0.7230 F1-score, a 0.8088 F1-score, and a 0.9481 F1-score for the CD-2012, CD-2014, and CMD data sets, respectively (Tables 12, 13, and 14).

Tables 13, 14, and 15 show the advantages of our system as follows. First, according to Table 13, our FBGS achieved the best recall (Re) and F1-score on CMD data set [43]. Also, in Table 14, our method performed the best accuracy on dynamic background (DB) and pan-tilt-zoom (PTZ) videos. Second, according to Table 15, the proposed method noticeably reduced algorithmic complexity so that it can be used in limited computing devices. Considering real-time BGS, DB and PTZ are major video categories that are used to compare each method in our proposal. Although it seems the performance of the proposed method is not as promising as the deep learning-based approaches such as DeepBS, our approach shows better results than DeepBS on DB and PTZ videos (Table 14). Even SuBSSENSE does not include a CNN framework that shows better performance in many scenarios. However, F1-score is known as a very important KPI used to measure segmentation accuracy. Our proposal achieved better F1-score than SuBSSENSE on CMD data set as well as on DB and PTZ videos of CD-2014 data set [42] (Tables 13 and 14).

We observe that, for unstabilized video, the colour feature of a pixel needs to be considered for better classification than the colour + FI features. The colour + FI features result in more false positives for the unstabilized video (PTZ) than the other cases. For PTZ, we obtained a 0.5425 F1-score and a 0.3564 F1-score for colour and colour + FI, respectively. But, for the rest of the categories, colour + FI always performed better than only colour. We also observed that adaptive threshold determination, background sample update, and post-processing play a key role in accurate segmentation.
### TABLE 14  
Average F1-score comparison between our proposed approach and existing approaches with CD-2012 [41] and CD-2014 [42] data sets

| Cate. | DeepBS [26] | SuBS [3] | FBGS | LOB [2] | PBAS [5] | KNN [17] | ViBe [23] | MOG [45] | ViBe+ | GMM [14] | S-360 [36] | SOBS [38] | SBS [20] |
|-------|-------------|----------|------|---------|---------|---------|---------|--------|------|--------|----------|----------|---------|
| BL    | 0.9580      | 0.9480   | 0.9112 | 0.9321  | 0.7363  | 0.8315  | 0.7961  | 0.6423 | 0.7950 | 0.6612 | –        | –        | –       |
| TH    | 0.7583      | 0.8184   | 0.7924 | 0.7824  | 0.6887  | 0.5727  | 0.6624  | 0.4354 | 0.5311 | 0.3472 | –        | –        | –       |
| SH    | 0.9304      | 0.8890   | 0.8623 | 0.8770  | 0.7732  | 0.7769  | 0.7476  | 0.6766 | 0.7570 | 0.6907 | –        | –        | –       |
| DB    | 0.8761      | 0.8138   | 0.8832 | 0.5792  | 0.6080  | 0.6464  | 0.6216  | 0.6855 | 0.4776 | 0.1676 | –        | –        | –       |
| IM    | 0.6098      | 0.6523   | 0.6450 | 0.5752  | 0.3618  | 0.5209  | 0.3757  | 0.3373 | 0.4587 | 0.4851 | –        | –        | –       |
| CJ    | 0.8990      | 0.7694   | 0.7588 | 0.7482  | 0.5600  | 0.7083  | 0.6038  | 0.6316 | 0.4330 | 0.2752 | –        | –        | –       |
| CD-2012 [41] | 0.8386 | 0.8152   | 0.8088 | 0.7490  | 0.6280  | 0.6791  | 0.6345  | 0.5681 | 0.5754 | 0.4378 | .7283    | 0.7197   | –       |
| BW    | 0.8301      | 0.8528   | 0.8517 | 0.6378  | 0.7863  | 0.7963  | 0.6212  | 0.6512 | 0.7720 | 0.6094 | –        | –        | –       |
| LF    | 0.6002      | 0.6437   | 0.5982 | 0.6738  | 0.5366  | 0.6107  | 0.4298  | 0.6110 | 0.4897 | 0.3683 | –        | –        | –       |
| NV    | 0.5835      | 0.5390   | 0.4615 | 0.4293  | 0.3795  | 0.3777  | 0.3816  | 0.3748 | 0.4016 | 0.3020 | –        | –        | –       |
| PTZ   | 0.3133      | 0.3185   | 0.3564 | 0.0875  | 0.1130  | 0.2347  | 0.1042  | 0.2623 | 0.1313 | 0.1663 | –        | –        | –       |
| TB    | 0.8455      | 0.8197   | 0.8317 | 0.5299  | 0.7236  | 0.6892  | 0.6629  | 0.7650 | 0.3030 | 0.5274 | –        | –        | –       |
| CD-2014 [42] | 0.7548 | 0.7331   | 0.7230 | 0.6230  | 0.5734  | 0.6167  | 0.5461  | 0.5521 | 0.5045 | 0.4237 | .6732    | .5961    | –       |

The bold face values mean the best performances and the average values.

### TABLE 15  
Comparison of the methods’ complexity

| Method      | bpp | fps (Average) |
|-------------|-----|---------------|
| FBGS        | 6   | 42.78         |
| PBAS [5]    | 15  | 46.51         |
| LOB [2]     | 15  | 35.68         |
| SuBS [3]    | 15  | 25.02         |
| ViBe+ [24]  | 4   | 15.22         |

The bold face values mean the best performances and the average values.

Finally, by comparing our proposal and the existing approaches, we make the following observations:

1. The intensity variation of a pixel is one of the main reasons for false detection. To address this issue, we propose a novel FI that can correct intensity variations (Figure 1). Therefore, using the FI, our proposed BGS provides correct background/foreground segmentation, as shown in experimental results (Table 12).

2. The boundary edges of objects can increase false positive due to illumination variations. However, the proposed FI can smooth the edge information to reduce the false positive. Table 11 shows that our colour + FI-based BGS increases 1.59% average F1-score compared with the colour + GM-based BGS.

3. Accurate segmentation is very difficult to perform for challenging video cases such as dynamic background, thermal effect, intermittent object motion, camera jitters, bad weather, night videos, pan-tilt-zooming, and turbulence. However, our FI-based BGS approach can improve the classification accuracy for these challenging video cases. In Table 14, our proposal achieves the best segmentation accuracy for dynamic background and pan-tilt-zooming video cases.

4. Real-time moving object detection can be possible more easily by using our FI-based BGS approach. Our proposal reduces the time complexity and memory space complexity significantly to use for real-time applications (Table 15).

### CONCLUSIONS AND FUTURE WORKS

The FI is a combination of a colour/grey image and a gradient image. Only the GM is not error-free. Therefore, we created an FI that reduces noise due to illumination changes, thermal effects, shadows, and so on. Therefore, the FI can also be applied in other research areas where image noise reduction is necessary. On the other hand, dynamic threshold determination, background model updates, and post-processing also play a key role in correct background/foreground segmentation.

The proposed FBGS method shows competitive effectiveness. More importantly, our proposed algorithm performs well for the most challenging cases, such as dynamic backgrounds, PTZ, turbulence, camera jitter, thermal images, intermittent object motion, bad weather, shadows, and so on. Also, our FBGS can be utilized in real time.

In the future, we will use deep CNN to extract salient features in addition to FI feature in our method in order to improve detection accuracy. Also, we will create a condition-dependent sample update strategy instead of a random sample update policy for the background sample update. Moreover, we will reduce the method’s complexity by considering fewer features.
and fewer parameters when determining the dynamics of the background. In addition, because of the many parameters used to model dynamic backgrounds, it is hard to analyse them to find the optimal parameters. In the future, we will study how to design an optimal hyperparameter problem.

ACKNOWLEDGEMENTS
This work was supported by Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00294, Service mobility support distributed cloud technology).

ORCID
Md Alamgir Hossain  https://orcid.org/0000-0002-6865-6650
Van Dung Nguyen  https://orcid.org/0000-0002-3940-3929

REFERENCES
1. Liao, S., et al.: Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1301–1306. IEEE (2010)
2. St.Charles, P.L., Bilodeau, G.A.: Improving background subtraction using local binary similarity patterns. In: IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 509–515. IEEE (2014)
3. St. Charles, P.L., et al.: Subsense: A universal change detection method with local adaptive sensitivity. IEEE Trans. Image Process. 24(1), 359–373 (2015)
4. Guo, L., et al.: Background subtraction using local SVD binary pattern. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 86–94 (2016)
5. Hofmann, M., et al.: Background segmentation with feedback: The pixel-based adaptive segmenter. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 38–43. IEEE (2012)
6. Bouwmans, T., Garcia-Garcia, B: Background subtraction in real applications: Challenges, current models and future directions. arXiv preprint arXiv:190103577 (2019)
7. Jain, N.K., et al.: A review on traffic monitoring system techniques. In: Soft Computing: Theories and Applications, pp. 569–577. Springer, Singapore (2019)
8. Aggarwal, A., et al.: Object tracking using background subtraction and motion estimation in MPEG videos. In: Asian Conference on Computer Vision, pp. 121–130. Springer (2006)
9. Kalavani, P., Vimala, D.: Human action recognition using background subtraction method. Int. Res. J. Eng. Technol. 2(3), 1032–1035 (2015)
10. Liu, Y., et al.: Edge-to-fog computing for color-assisted moving object detection. In: Big Data: Learning, Analytics, and Applications, Vol. 10989, p. 1098903. International Society for Optics and Photonics (2019)
11. Bouwmans, T.: Traditional and recent approaches in background modeling for foreground detection: An overview. Comput. Sci. Rev. 11, 31–66 (2014)
12. Li, Q.L., He, J.F.: Vehicles detection based on three-frame difference method and cross-entropy threshold method. Comput. Eng. 37(4), 172–174 (2011)
13. Zhang, Y., et al.: Three-frame difference algorithm research based on mathematical morphology. Procedia Eng. 29, 2705–2709 (2012)
14. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: CVPR, p. 2246. IEEE (1999)
15. Elgammal, A., et al.: Non-parametric model for background subtraction. In: European Conference on Computer Vision, pp. 751–767. Springer (2000)
16. Lee, D.S.: Effective Gaussian mixture learning for video background subtraction. IEEE Trans. Pattern Anal. Mach. Intell. 27(5), 827–832 (2005)
17. Zikovic, Z., van der Heijden, F.: Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognit. Lett. 27(7), 773–780 (2006)
18. Schick, A., et al.: Improving foreground segmentations with probabilistic superpixel Markov random fields. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 27–31. IEEE (2012)
19. Wang, R., et al.: Static and moving object detection using flux tensor with split Gaussian models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 414–418 (2014)
20. Chen, Y.Q., et al.: An effective sub-superpixel-based approach for foreground subtraction. IEEE Trans. Ind. Electron. 67(1), 601–609 (2019)
21. Kim, K., et al.: Real-time foreground-background segmentation using codebook model. Real-Time Imaging 11(3), 172–185 (2005)
22. Wang, H., Suter, D.: A consensus-based method for tracking: Modelling background scenario and foreground appearance. Pattern Recognit. 40(3), 1091–1105 (2007)
23. Barnich, O., Van Droogenbroeck, M.: ViBe: A universal background subtraction algorithm for video sequences. IEEE Trans. Image Process. 20(6), 1709–1724 (2011)
24. Van Droogenbroeck, M., Paquot, O.: Background subtraction: Experiments and improvements for ViBe. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 32–37. IEEE (2012)
25. Tiefenbacher, P., et al.: PID-based regulation of background dynamics for foreground segmentation. In: IEEE International Conference on Image Processing (ICIP), pp. 3282–3286. IEEE (2014)
26. Babaei, M., et al.: A deep convolutional neural network for video sequence background subtraction. Pattern Recognit. 76, 635–649 (2018)
27. Heikkila, M., Pietikainen, M.: A texture-based method for modeling the background and detecting moving objects. IEEE Trans. Pattern Anal. Mach. Intell. 28(4), 657–662 (2006)
28. Bilodeau, G.A., et al.: Change detection in feature space using local binary similarity patterns. In: International Conference on Computer and Robot Vision, pp. 106–112. IEEE (2013)
29. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
30. Ren, S., et al.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
31. Redmon, J., et al.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
32. He, K., et al.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
33. De Gregorio, M., Giordano, M.: Change detection with weightless neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 403–407 (2014)
34. Braham, M., Van Droogenbroeck, M.: Deep background subtraction with scene-specific convolutional neural networks. In: IEEE International Conference on Systems, Signals and Image Processing (IWSSIP), Bratislava, 23–25 May, pp. 1–4. IEEE (2016)
35. Wei, H., Peng, Q.: A block-wise frame difference method for real-time video motion detection. Int. J. Adv. Rob. Syst. 15(4), 1–13 (2018)
36. Sedky, M., et al.: Spectral-360: A physics-based technique for change detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 399–402 (2014)
37. KaewTraKulPong, P., Bowden, R.: An improved adaptive background mixture model for real-time tracking with shadow detection. In: Video-Based Surveillance Systems, pp. 135–144. Boston, MA: Springer (2002)
38. Maddalena, L., Petrosino, A.: The SOBS algorithm: What are the limits? In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 21–26. IEEE (2012)
39. Laugraud, B., et al.: Simple median-based method for stationary background generation using background subtraction algorithms. In: International Conference on Image Analysis and Processing, pp. 477–484. Springer (2015)
40. Hossain, M.A., et al.: An analysis of post-processing for background subtraction. In: Proceedings of the KIISE Korea Computer
41. Goyette, N., et al.: Changedetection.net: A new change detection benchmark dataset. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 1–8. IEEE (2012)

42. Wang, Y., et al.: CDnet 2014: An expanded change detection benchmark dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 387–394 (2014)

43. Sheikh, Y., Shah, M.: Bayesian modeling of dynamic scenes for object detection. IEEE Trans. Pattern Anal. Mach. Intell. 27(11), 1778–1792 (2005)

44. Elgammal, A., et al.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proc. IEEE 90(7), 1151–1163 (2002)

45. Friedman, N., Russell, S.: Image segmentation in video sequences: A probabilistic approach. In: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, pp. 175–181. Morgan Kaufmann Publishers Inc. (1997)

How to cite this article: Hossain MA, Nguyen V, Huh E-N. The trade-off between accuracy and the complexity of real-time background subtraction. IET Image Process. 2021;15:350–368.

https://doi.org/10.1049/ipr2.12026