# Real-Time Adaptive Histogram Min-Max Bucket (HMMB) Model for Background Subtraction

Sujoy Madhab Roy, *Student Member, IEEE*, and Ashish Ghosh, *Senior Member, IEEE*

***Abstract*— This paper proposes an efficient real-time background subtraction algorithm, which is essential in many computer vision applications. Initially, histograms of the intensity values for each channel of a pixel position in a set of training frames are constructed. A background model, histogram min-max bucket, is constructed from the minimum and maximum values of contiguous non-zero frequencies of the temporal intensity histogram. A novel feature of this algorithm is the use of a single sliding window to update the system adaptively, capturing the proper background even under sudden and/or gradual illumination changes in the scene. We incorporate both local and global information with the help of the current and the previously visited pixel values for binary classification of a pixel into foreground and background. This algorithm is compared with several state-of-the-art techniques and experimental studies show that the proposed method outperforms all these methods in terms of accurate binary classification.**

***Index Terms*— Background subtraction, computer vision, illumination invariance, binary pixel classification, sliding window, real-time analysis.**

## I. Introduction

BACKGROUND subtraction (BS) is a popular technique to identify the moving objects (foreground) within a video stream. Foreground contains valuable information to apply in various real life applications. BS is widely used in real-time video processing and it is a decisive step in many computer vision systems. The main steps of the BS procedure are background initialization, foreground detection and background maintenance [1]. In general, the initial step of BS method is to construct a static model of the background information, and then use this model over the incoming frames to identify foreground regions. Extraction of moving objects accurately from a video scene becomes difficult when the scene contains non-static background like waving tree branches, different height and spread of water fountain, moving escalator, etc. Various illumination changes and appearance of new background (car is starting from a parking lot, box moving from one place to another) in the scene adds to the complexity of the problem. To overcome the above difficulties, the background model must be a representation of the scene with no moving objects and must be dynamically updated.

In the last few decades, researchers are trying to solve this problem from various aspects. Wren [2], Stauffer and Grimson [3], and Zivkovic and Van der Heijden [4] modeled the background by taking one or more Gaussian functions. The authors of [5] modeled the background by constructing a codebook for each pixel using cylindrical colour model. A local binary similarity pattern (LBSP) is used to incorporate the local information into the background model by St-Charles and Bilodeau [6] and St-Charles *et al.* [7]. Oliver *et al.* [8] extracted important features using principal component analysis (PCA) from the video scene and used them for accurate separation of foreground. A tensor based method [9] integrated the temporal and spatial information together for classification. BS can also be called as an intrinsic problem of foreground object segmentation in videos [10]–[12]. The state-of-the-art methods [2]–[5], [8], [9] can extract moving objects from complex video scenes; but they may fail to detect moving objects accurately under sudden illumination changes.

In this article, a new BS method is proposed in which a background model is built, for each pixel separately, from the histogram of the training frames. This background model consists of a number of 'buckets', which are clusters of contiguous histogram bins with non-zero frequency. This model is updated continuously as new frames arrive in order to identify the foregrounds accurately. In the proposed method each pixel of a newly arrived frame is classified as foreground or background after testing in three stages. A pixel's value is searched in the initial background model in the first step. In the second step local (short term) and global (long term) illumination variations are identified if the search procedure is unsuccessful in the first step. Abrupt illumination changes are identified in the third step if first two steps fail to classify the pixel value.

The proposed model is more general and more flexible than a mixture model [3], [13], as it does not consider the shape and/or nature of the underlying probability density function (*pdf*) of the background. Novelty of this method is that it adaptively updates the system with the help of a single sliding window (described by a pair of integers corresponding to the lower and upper limits of the window) which is capable of identifying foreground objects in videos even with dynamic background and witch can also detect the foreground even in the presence of abrupt illumination change in the whole frame (e.g., switch on/off of the lights in an indoor scene).

Manuscript received September 12, 2016; revised December 9, 2016 and January 3, 2017; accepted February 4, 2017. Date of publication February 15, 2017; date of current version July 2, 2018. This paper was recommended by Associate Editor M. Wang.

The authors are with the Machine Intelligence Unit, Indian Statistical Institute, Kolkata 108, India (e-mail: sujoyroy_r@isical.ac.in; ash@isical.ac.in).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2017.2669362

The proposed method gives equal priority to the local and global illumination changes to adaptively update the model and shows at least at par performance. It can even produce better result if there is an abrupt illumination changes in the video scene, with respect to the state-of-the-art methods.

This article is organized as follows. Section 2 gives a brief review of the existing BS techniques. Model construction, foreground detection and model updating procedure is discussed in Section 3. Section 4, discusses the performance evaluation methods. Experimental results are shown in Section 5 and conclusions are drawn in Section 6.

## II. Review of Existing Backgroud Subtraction Methods

BS is widely used in computer vision to identify moving objects in a video scene taken by a static camera. Background models mainly consist of three different approaches: pixel based methods [2]–[5], [13]–[19], small region (consisting of a few pixels) based methods [20]–[23] and large region (consisting of a frame itself) based methods [8], [9], [24], [25]. Toyama *et al.* combined the pixel, small and large region based information in Wallflower algorithm[1] [26].

Among all these methods, pixel based methods are most popular in the BS literature and their effective applicability in computer vision applications. It also opens a way to parallelize the method over multiple processing units or over a GPU and helps real-time video processing. Wren *et al.* introduced a intensity based simple model [2] where they calculate the arithmetic mean of the pixel values of the corresponding position from last $n$ frames, independently for each position. The authors then fitted a Gaussian *pdf* over the last $n$ visited pixel values at the same pixel position. Lo and Velastin [18], Cucchiara *et al.* [19], and McFarlane and Schofield [27], proposed to take the median value instead of the mean value for the construction of the background model as median is a robust statistic. Although these methods are simple and effective, they cannot efficiently handle complex background. To model multiple backgrounds, in [3] and [13], Stauffer and Grimson proposed Gaussian mixture model (GMM) which model each pixel by the sum of weighted Gaussian distributions. Though GMM model has shown improvement for scenes having non-static background, such as, with moving tree branches, it has some disadvantages also. It cannot take care of certain illumination variation with a predefined fixed number of Gaussians. To control certain illumination changes, Zivkovic and Van Der Heijden [4] introduced a technique to adaptively count the exact number of Gaussians for more accurate evaluation. In [15] and [28], a non-parametric model, based on kernel density estimation (KDE) is considered by Elgammal *et al.* to avoid the difficulty of finding the proper shape of the *pdf* with limited number of training frames. This pixel based model is constructed from the histogram of the previously buffered background values. To update the model, they used a short-term and one long-term memory model at each pixel position by maintaining a first-in-first-out order, i.e., oldest pixel values are replaced by the current pixel values. A non-linear pixel level algorithm for BS proposed by Manzanera [16], [29] is based on sigma-delta ($\Sigma - \Delta$) [29] motion detection. This model is based on the use of Zipf's law [30] and involves comparison and elementary increment or decrement operations. If the foreground object is static for a long time in the scene, this method misclassifies the object as background because of small or no motion of the foreground. A real-time pixel based codebook model is proposed by Kim *et al.* [5], [17] which is also model multiple backgrounds. For a large number of sequences of training frames, a codebook is created for all pixels to represent a compressed form of the background model. This model is capable of identifying periodic motion of the background objects in the scene. They used a cylindrical colour model to measure the colour distortion, for background modeling as well as foreground detection. The codebook model is very computationally intensive and no new codewords are added after the training phase. A pixel based probabilistic BS technique, Visual Background extractor (ViBe) [14], is introduced by Barnich *et al.* For each pixel, a set of values of the same location or of its neighbourhood from the previous frames are considered to determine the belongingness of the pixel to the background. Initialization of the background model is done with only the first frame of the video. To classify a pixel according to its model, they searched for its closest values within the set of samples by defining a sphere (in 3D colour space) centered at that pixel location. A pixel is classified as background if the cardinality of the set intersection of this sphere and the collection of model samples is not less than a given threshold value. This algorithm produces good results but one drawback is that the foreground gets gradually transformed into background when the object moves slow or it has no motion for a long time. A BS method based on Dirichlet process [31] has been proposed by Haines *et al*. The authors use a probabilistic regularisation method to collect the neighbourhood pixels information and remove the noise produced by the Gibbs sampling. This algorithm shows reasonable results on different datasets, but processing speed is on the lower side. St-Charles *et al.* introduced a BS algorithm [6] which is based on ViBe [14] and LBSP. This method shows an improvement over the ViBe method in terms of statistical measurements but its per frame execution time is a constraint for real-time image processing. St-Charles *et al.* also proposed [7] a non-parametric method by combining LBSP and ViBe. This method produced some impressive quantitative results, but execution time is still a concern for real-time processing. In [32], background estimation is formulated using convolutional neural network (ConvNet) trained with a scene-specific dataset by Braham *et al*. Hierarchical features are extracted using a deep learning method and applied for BS but this method requires very high computational resources.

Pixel based BS methods classify every pixel into foreground or background independently of one another, but

[1] we use the words method and algorithm interchangeably.

region based methods use region level features for binary classification of the entire regions or blobs of the frame. Shah *et al.* have constructed a self-adaptive codebook model [23] with pixel level information as well as block or region based information. This method integrates some advantages into the codebook model [5], like adding new codewords in the initial model and removing noisy codewords from the model. Noh *et al.* consider multiple cues (pixel colour, pixel texture and region appearance) for BS framework [33]. To model the background the authors use a variant of codebook model [5] for each pixel in the image. They also combine a region-level process to detect foreground blob and refine it using partial directed Hausdorff distance. Ahn *et al.* have proposed a BS algorithm [21] using regions as visual primitives rather than pixels. This method initially divides an image into small homogeneous regions and then finds the effective region object likelihood with the help of colour and motion information. A region level background/foreground decision function is used to classify an image to extract the human silhouette. This algorithm showed very good results on real sequences but could not achieve real-time processing speed. Varcheie *et al.* [22] divide an image into rectangular parts at various scales and calculate the histogram and also the variance for each individual region to construct a BS method. A model is built using the texture information and GMM [3] is applied to refine the result. As the method uses GMM as a sub-part, it requires even more time to execute than GMM.

A frame based BS method considers a frame or a set of frames for backgroud model construction, for separating foreground and background. Oliver *et al.* [8] applied eigenvalue decomposition over a set of initial frames accumulated over time. Modeling is done on a frame using PCA which extracts important features (i.e., principal components of the data) with large variances from correlated features. The algorithm can handle static parts of the frames efficiently, but it is not so good for detecting foreground objects with small amount of motion; as PCA usually identifies the portion of the frame where the objects contain large amount of motion as foreground. A tensor based on-line subspace learning algorithm [9] is proposed by Hu *et al.*. They collected a group of frames to construct a 3D matrix (called 3rd-order tensor) and unfolded it into three directions to make three 2D matrices (called projection subspaces). The algorithm used an incremental singular value decomposition (SVD) [34] method to discover the dominant projection subspaces of a 3rd-order tensor and adaptively updated the sample mean and an eigenbasis when new frames arrive. The authors assigned greater priority to recent observations by giving them higher weights than the past ones by using a parameter called forgetting factor. This method takes spatial as well as temporal information to find the foreground objects by unfolding a third order matrix into three 2D matrices and as a result it consumes much computational time and memory. In [24], a low-rank model (representing variables of interest as low-rank matrices) is formulated for BS by Zhou *et al*. The underlying concept is that high-dimensional data (e.g., a set of images) often can be represented by a low-dimensional subspace without any/little loss of useful information. A low-rank and structured sparse decomposition method is proposed by Liu *et al.* [25], in which they considered the background as a low-rank matrix and the foreground by a structured sparse outlier matrix. A drawback of low-rank matrix modeling method it that they always assumes that any variation in the background appearance can be captured by the low-rank matrix; which may not always be true.

All the methods discussed in the above section can handle various complex scenarios like dynamic background, different types of object motions, small illumination changes in the scene, etc. But these methods cannot cope with video scenes containing sudden illumination changes. In this article we propose a BS method that can efficiently handle these complex video scenes even under sudden illumination changes. The proposed method employs a continuously adaptive update policy of the background model (i.e., buckets), even during testing phase of the video sequence. One notable contribution of the present work is the use of a sliding window, with variable lower and upper bounds, to adaptively update the initial background model. Most of the current state-of-the-art methods (except a few, e.g., SuBSENSE [7]) update the model only when a pixel classified as background but not when a pixel is classified as foreground. In contrast, the proposed algorithm adaptively updates the model in both situations in order to capture the foreground even under local, global and abrupt illumination changes, by shifting the sliding window every single time when the new pixel is classified as foreground or as background. Even in the case of SuBSENSE, the model is updated only after the current pixel has already been classified. Therefore the effect of the updated model is only seen in the subsequent frames. Our proposed method, on the other hand, updates the model first, before classifying the current pixel, because of which the updated model is effective right from the current frame. The proposed algorithm also gives equal priority to local and global illumination changes and updates the background model instantaneously to account for those illumination changes. This helps our method in producing better foreground detection even when an abrupt illumination change occurs in a video scene. To the best of our knowledge, in the literature there is no BS algorithm which has these properties. A robust analysis using statistical measurements, applied on various complex video sequences, is reported in this article. Comparisons with recent state-of-the-art techniques along with qualitative and quantitative analyses of results have also been made in this article. The execution time of the proposed method is also impressive for real-time BS.

## III. The Proposed Method

In the proposed pixel based BS algorithm, foreground is identified by searching all pixel intensity values of the test-frames in their corresponding initial models created using the training frames (see Section III-B) or adaptively updated model (see Section III-E). The key features of the proposed algorithm are:

- It can model any number of backgrounds. Algorithm 1 and Figure 2 describe the way the proposed model identifies multiple backgrounds by constructing multiple buckets.

**Algorithm 1** Initial HMMB Model Construction

**Input:** $n$ training frames.

I: **for** each pixel position $x$ in the frame of size $N$ **do**
II: **for** a channel $C \in \{R, G, B\}$ **do**
III: Construct the histogram by taking all intensity values of $n$ initial input frames.
IV: Create arrays $minBucket_C(x)$ and $maxBucket_C(x)$ to store the minimum and maximum values, respectively for all contiguous non-zero frequencies.
V: Accumulate the total frequency enclosed by a pair of corresponding minimum and maximum values.
VI: Sort the accumulated frequencies in non-increasing order and modify the arrays $minBucket_C(x)$ and $maxBucket_C(x)$ accordingly.
VII: **end for**
VIII: **end for**

**Output:** $minBucket_C(x) = \{l_1^C(x), l_2^C(x), \ldots, l_{K_C(x)}^C(x)\}$ and $maxBucket_C(x) = \{r_1^C(x), r_2^C(x), \ldots, r_{K_C(x)}^C(x)\}$ for all pixel positions and for all channels, where $K_C(x)$ is the number of buckets at pixel position $x$ of channel $C$.

- It is able to adaptively update the model to identify the foreground efficiently. Section III-C illustrates how the initial model (see Algorithm 1) adaptively updates under different illumination conditions with the help of a sliding window.
- It can handle videos having both local and global illumination changes. Local change is the difference between the current and previous test frames (see Eqn. 1); and global change is the difference between the current and first test frames (see Eqn. 2 and Appendix VI). We assign equal priority to both these changes. Elgammal *et al.* [15], [28] picked the frames in the first-in-first-out manner to update the model by losing some valuable information. On the otherhand the authors of [14] probabilistically collected the valuable information present in the previously visited frames; but it may end up by not including important features to upgrade the model. In Sections III-E.1 and III-E.2, we describe the procedure to handle both local and global illumination changes and balance the model more accurately by assigning the same priority.
- At each iteration, it can handle the gradual and/or abrupt illumination changes with high accuracy. Section III-E.1 describes the procedure to manage gradual illumination changes. Third phase of the three fold detection procedure (see Section III-C) and Section III-E.2 demonstrates the novelty of the proposed method (accurate object identification when abrupt illumination changes occur in a video scene).

### A. Overview of the Proposed Algorithm

A pipeline of proposed pixel based algorithm is presented in Figure 1. For every pixel of every channel of the video sequence, we construct an initial background model, using

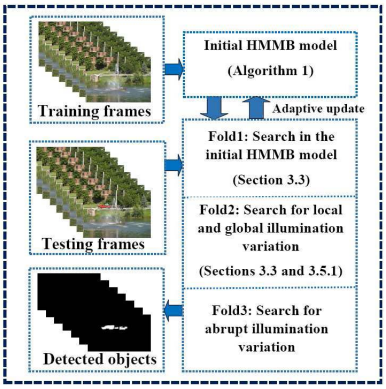
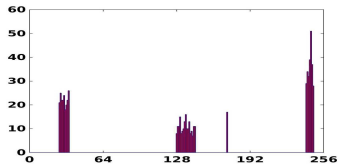



Fig. 1. Pipeline of the proposed algorithm.

the first few frames of the video sequence as initial training frames. The initial background model consists of a set of 'buckets', which are clusters of consecutive pixel values that occur with non-zero frequency at that particular pixel position, within the training frames (see Algorithm 1). The background model at a particular pixel position may consists of several such buckets, separated by regions of pixel values that did not occur at that position. These buckets of pixel values are represented in our algorithm by the minimum and maximum pixel values occupied by those buckets. After the initial background model construction we move into the testing phase. In the testing phase, for each new frame, we compare the pixel value at any particular pixel position of the new frame with the already constructed background model at that pixel position. For each new frame, depending upon the video sequence, the following cases may occur. Fold1: The new pixel may fall inside the existing background model. Fold2: The new pixel value may fall outside the existing background model due to gradual illumination change (either local or global). Fold3: The new pixel may fall outside the existing background model due to abrupt illumination change. For the purpose of detecting gradual and abrupt illumination changes, for each new frame, we compute a temporally local difference image and also a temporally global difference image. From these local and global difference images we decide whether a gradual or abrupt illumination change is occurring or not. If a gradual or abrupt illumination change is detected, we then update our background model in such a way so as to include the new pixel value within the updated model. Otherwise, if none of the above three cases occur and the new pixel value is still falls outside the existing background model, then we classify that pixel as foreground.

### B. Background Model Construction

Let $P_C^t(x)$ be the pixel intensity value at location $x$ of channel $C$ ($\in \{R, G, B\}$) in the $t$-th frame of a video scene (in general $P_C(x)$ is the intensity at pixel location $x$ of channel $C$). Background information is gathered by accumulating some initial training frames in which the foreground information is absent. For each pixel position $x$ of a channel $C$, let $I_C^n(x) = \{P_C^1(x), P_C^2(x), \ldots, P_C^n(x)\}$ be the collection of intensity values of the training frames, i.e., $t = 1, 2, \ldots, n$. A histogram is built for each channel $C$ at pixel




(a)


(b)

| frequency | minBucket | maxBucket |
|---|---|---|
| 150 | 240 | 248 |
| 100 | 25 | 35 |
| 80 | 127 | 145 |
| 7 | 171 | 173 |

(c)

Fig. 2. Histogram and bucket construction. (a) Histogram. (b) Buckets. (c) Frequency and bucket boundaries.

location $x$ by accumulating the intensity values $I_C^n(x)$. The proposed BS model is built using these histograms (see Figure 2). All contiguous bins of non-zero frequencies (only one non-zero frequency for a particular intensity value is also accepted) into different clusters, denoted by the intervals $[l_1^C(x),\ r_1^C(x)]$, $[l_2^C(x), r_2^C(x)], \ldots, [l_{K_C(x)}^C(x), r_{K_C(x)}^C(x)]$ and collect the minimum and maximum values of all the clusters in $minBucket_C(x) = \{l_1^C(x), l_2^C(x), \ldots, l_{K_C(x)}^C(x)\}$ and $maxBucket_C(x) = \{r_1^C(x), r_2^C(x), \ldots, r_{K_C(x)}^C(x)\}$. We consider a bucket by taking one pair of corresponding minimum and maximum intensity values, denoted by $LSW_C(x)$ and $RSW_C(x)$, at pixel $x$, for channel $C$. Note that, there exists an equal number of minimum and maximum values which are the starting and ending points of each cluster (see Figures 2a and 2b). Let there be $K_C(x)$ number of minimum values (i.e., $K_C(x)$ number of buckets). Now we calculate the total frequency inside each bucket (starting from the minimum value to the maximum value), arrange them in non-increasing order and organize the minimum and maximum boundary values accordingly (see Figure 2c). In this way, for every pixel location, multiple buckets are constructed which constitute the initial BS model (see Algorithm 1). We do not make any assumption about the shape or nature of the underlying probability distribution of the multiple background model (e.g., Gaussian) except that the distribution has a finite support, i.e., every bucket in the multiple background model lies within an interval of minimum and maximum values. Also we do not have to estimate statistical parameters like mean, standard deviation, skewness etc. For each pixel position $x$ of each channel $C$ we create a separate background model.

Employing the set of $n$ training frames as input, we construct the initial BS model which we call the histogram min-max bucket (HMMB) model consisting of two arrays, given as $minBucket_C(x)$ and $maxBucket_C(x)$ for minimum and maximum values respectively for all pixel locations $x$ where $l_k^C(x)$ and $r_k^C(x)$ are respectively the minimum and maximum boundary of the $k$-th bucket of channel $C$.

Before going to the algorithmic construction of initial HMMB model, one small annotated example is given in Figure 2. In Figure 2a, we have shown one histogram containing 4 contiguous non-zero frequencies in the range $26-34, 128-144, 172$ and $241-247$. In Figure 2b, we created 4 buckets (i.e., $K_C(x) = 4$) with left and right boundaries by two sets $minBucket_C = \{25, 127, 171, 240\}$ and $maxBucket_C = \{35, 145, 173, 248\}$, respectively. Accumulated frequency values of the buckets are $100, 80, 7, 150$, respectively and are shown inside each bucket (see Figure 2b). Figure 2c shows the bucket frequencies in non-increasing sorted order along with their corresponding minimum and maximum bucket boundary values.

HMMB model contains the left and right boundary values of all the buckets (like the last two columns of the table given in Figure 2c) for each pixel location $x$ for all $C$ channels. We present the HMMB model building procedure in Algorithm 1. Here we consider only the $RGB$ colour frames for model construction. This algorithm can also be used for gray level images having one channel (cardinality of $C$ is 1). The rest of the procedure remains unchanged throughout.

### C. Three Fold Detection Procedure

Following the algorithm concept discussed in Section III-A, at the testing or object detection phase of the algorithm, in order to verify a pixel's belongingness to background, we divide the detection algorithm into three phases - called three fold detection procedure.

In the first phase, we search the pixel value, $P_C(x)$, within the buckets of the initial HMMB model (see Algorithm 1) until a match is found. If a match is found, we call $x$ as a *probable background* (we classify a pixel location $x$ as a background if $x$ is *probable background* for all $C$ channels). For the first test frame, we search $P_C(x)$ only in the buckets of the initial HMMB model. In the adaptive update stage, initial buckets can be extended on either side with the help of left and right bucket boundaries (called a sliding window), but the buckets can not be reduced from its initial size on either sides.

If the search procedure fails in the first phase, we enter into the second phase of the algorithm. In this phase, we try to identify gradual illumination changes by considering both temporally local and temporally global illumination changes in a video scene. Gradual changes usually occur in a small neighbourhood of a bucket. We search for positive (negative) gradual change in the left (right) region of a bucket. If the search is successful for the $k$-th bucket, we call the pixel position $x$ as a *probable background*.

Unsuccessful search in the first two phases leads the procedure into the third phase. In this phase, we search for abrupt illumination change in a video scene, by computing median value of the temporally local difference (between current and previous frame) image. If this median value suggests (sec Algorithm 2 and Section III-E.2) that there is an abrupt illumination change in the neighbourhood of the $k$-th bucket, then we modify that bucket (detailed in Section III-E.2) and call $x$ as a *probable background*.

TABLE I

ADJUSTMENT OF THE LEFT OR RIGHT BOUNDARIES OF THE SLIDING WINDOW FOR $k$-th BUCKET AT PIXEL POSITION $x$ IN CHANNEL $C$

| MOVEMENT | SLIDING WINDOW POSITION | ACTION TAKEN | RESET PARAMETER |
|---|---|---|---|
| right | $\text{RSW}_C(x) > r_k^C(x)$ and $\text{LSW}_C(x) = l_k^C(x)$ | increment $\text{RSW}_C(x)$ | $maxChange_C(x) = 0$ |
| right | $\text{RSW}_C(x) = r_k^C(x)$ and $\text{LSW}_C(x) < l_k^C(x)$ | increment $\text{LSW}_C(x)$ and if $\text{LSW}_C(x) > l_k^C(x)$ increment $\text{RSW}_C(x)$ with the same amount | $maxChange_C(x) = 0$ and $\text{LSW}_C(x) = l_k^C(x)$ |
| left | $\text{LSW}_C(x) < l_k^C(x)$ and $\text{RSW}_C(x) = r_k^C(x)$ | decrement $\text{LSW}_C(x)$ | $minChange_C(x) = 0$ |
| left | $\text{LSW}_C(x) = l_k^C(x)$ and $\text{RSW}_C(x) > r_k^C(x)$ | decrement $\text{RSW}_C(x)$ and if $\text{RSW}_C(x) < r_k^C(x)$ decrement $\text{LSW}_C(x)$ with the same amount | $minChange_C(x) = 0$ and $\text{RSW}_C(x) = r_k^C(x)$ |

Failure in all the above three phases implies that $x$ belongs to the foreground (see Eqn. 3).

### D. Local and Global Pixel Intensity Changes

If illumination changes occur in a video scene, the background model must be updated adaptively to identify the foreground objects precisely. Proper modification of the model is required to adapt to the local and global intensity changes in the video sequence. Stauffer and Grimson, Hu *et al.*, in [3] and [9], used different weights (i.e., different priorities) to interpolate the local and global intensity changes for each pixel of a frame. This priority upgrading procedure required some mathematical operations, which requires a substantial amount of time to update the model. On the other hand, Elgammal *et al.* [15] and Monari and Pasqual [35] considered two models (one a short-term and one long-term model) to identify illumination change in the video scene. These methods require a large amount of memory for model construction. Proper selection of buffer size is also a crucial choice.

In the proposed method, same priority is given to local and global changes. For each pixel position $x$ of channel $C$ we maintain four parameters. The first two parameters are used to capture small variation in the vicinity of the initial buckets due to illumination changes. The first parameter $newDiffold_C^t(x)$ ($newDiffold_C^t$ is used when all pixel positions in a frame is considered) is the difference between two consecutive test frames' ($(t-1)$-th and $t$-th) pixel intensity values determined from the following equation

$$newDiffold_C^t(x) = P_C^t(x) - P_C^{t-1}(x); \tag{1}$$

to capture local illumination changes and the second parameter $diffSum_C^t(x)$ is used to store the cumulative sum of these differences (found in Eqn. 1), computed as

$$diffSum_C^t(x) = \sum_{i=2}^{t} newDiffold_C^i(x), \tag{2}$$

to capture global illumination changes, where $t = 2, 3, \ldots, n$. The third and fourth parameters are $minChange_C(x)$ and $maxChange_C(x)$ respectively for large (or abrupt) negative and positive illumination changes in the video scene. We discuss more about these parameters later in this section. To capture the local illumination change, we use both the parameters $newDiffold_C^t(x)$ and $diffSum_C^t(x)$, but abrupt illumination change is identified by using parameters $newDiffold_C^t(x)$, $maxChange_C(x)$ and $minChange_C(x)$.

### E. Adaptive Update Procedure of HMMB Model

The initial HMMB model is good enough to identify the foreground efficiently when the background is static for rest of the test frames. But in many video sequences, background changes due to illumination change or multi-modal behaviour of the scene. One can divide the illumination variation into two types as gradual and sudden. To overcome the problem of detecting foreground objects even with these types of changes in a scene, for pixel position $x$, we update the initial model adaptively with the help of a sliding window, denoted by $[LSW_C(x),\ RSW_C(x)] \in \{[l_1^C(x), r_1^C(x)], [l_2^C(x), r_2^C(x)], \ldots, [l_{K_C(x)}^C(x), r_{K_C(x)}^C(x)]\}$ to expand the buckets boundaries $minBucket_C(x)$ and $maxBucket_C(x)$, respectively for channel $C$. Table I describes under which conditions the left and/or right boundaries of the sliding window move and by how much.

*1) Update Model for Gradual Changes:* If a test pixel does not belong to the initial HMMB model, then either the pixel belongs to the foreground or there is an illumination change in that pixel position.

If a test pixel's intensity value $P_C^t(x)$ does not belongs to any of the buckets enclosed by the boundaries $[l_k^C(x), r_k^C(x)]$, $\forall k = 1, 2, \ldots, K_C(x)$ in the initial HMMB model, we go for either of the following two choices. Choice ($i$): when both the parameters $newDiffold_C^t(x)$ and $diffSum_C^t(x)$ are positive, then we test if $newDiffold_C^t(x) \in [0, \alpha]$, $(\alpha > 0)$ and also test if $diffSum_C^t(x) \in [\beta, \gamma]$, $(\gamma > \beta > 0)$, where $(\alpha, \beta, \gamma)$ is a triplet of parameter values. This triplet determines how much an initial bucket needs to expand in either side to capture both local and global illumination changes. If both the above tests are successful for the $k$-th bucket, we add $diffSum_C^t(x)$ to either $l_k^C(x)$ or $r_k^C(x)$ or both (see Table I) for adaptive update. Choice ($ii$): when parameters $newDiffold_C^t(x)$ and $diffSum_C^t(x)$ are both negative, we test if $newDiffold_C^t(x) \in [-\alpha, 0]$, $(\alpha > 0)$ and also test if $diffSum_C^t(x) \in [-\gamma, -\beta]$, $(\gamma > \beta > 0)$. If both the tests are successful, to update the model adaptively, we add absolute value of $diffSum_C^t(x)$ with $l_k^C(x)$ or $r_k^C(x)$ or both (see Table I). Using the parameter $diffSum_C^t(x)$, we track the pixel difference between the current and the first test frames, i.e., $diffSum_C^t(x)$ holds the global change of the intensity values at pixel position $x$ (see Appendix VI).

If we can not find any gradual illumination change in the video scene, then we try to find abrupt illumination change. To capture the abrupt change in the scene, we modify $maxChange_C(x)$ or $minChange_C(x)$ by adding

$newDiffold^t_C(x)$ when it is positive or negative, respectively for channel $C$ of the $t$-th test frame.

*2) Update Model for Abrupt Changes:* When there is an abrupt illumination change (e.g., light off/on in an indoor scene, sudden heavy rain in a driveway etc.), the absolute value of $newDiffold^t_C$ is very high for majority of the pixels. We also consider the fact that if a sudden illumination change occurs in a video scene, it affects a major portion of the frame. To identify abrupt illumination changes, we find the median value, $m$, of difference image $newDiffold^t_C$ for each channel $C$ of $t$-th test frame. Median is a robust statistic for summarizing the entire set of values. The median value, $m$, indicates that 50% of the pixel changes are less than $m$ and the other 50% are more than $m$. If the median value, $m$, is more than a user specified parameter $\zeta$, we can conclude that more than 50% values are greater than $\zeta$. This indicates that there is an abrupt illumination change in the scene. If an abrupt change occurs in the neighbourhood of the $k$-th bucket, then modify the $k$-th bucket adaptively using the sliding window. If $median(newDiffold^t_C) \geq \zeta$, then add $maxChange_C(x)$ with $r^C_k(x)$ or $l^C_k(x)$ or both, and after that reset $maxChange_C(x)$ to 0 (see Table I). Again if $median(newDiffold^t_C) \leq -\zeta$, then add the absolute value of $minChange_C(x)$ with $l^C_k(x)$ or $r^C_k(x)$ or both, and reset $minChange_C(x)$ to 0 (see Table I).

*3) Finding the Median:* To identify sudden illumination changes in the $t$-th frame of a video scene, we find the median of $newDiffold^t_C(x)$ for all $x$ positions of channel $C$. But finding the median for a big frame (e.g., 720 × 576 pixels) is a hurdle for real-time background subtraction. In conventional sorting methods, time complexity of median finding algorithm is $O(N \log N)$ for input size $N$ [36]. Although there exists linear-time algorithm for finding the median, it is doing some recursive mathematical calculations and takes more memory space for those operations [36]. To overcome the above shortcomings, a median finding algorithm is proposed (see Algorithm 2). This algorithm is useful when we know the range of the input data-set beforehand. To develop the proposed model, we deal with the $RGB$ colour images whose intensity values are 8-bit unsigned integers. Thus the parameter $newDiffold^t_C(x) \in [-255, 255]$ for all $t$, $C$ and $x$, i.e., it takes integer values among 511 consecutive integers.

In Algorithm 2, we count the frequency of each integer in the range $[-255, 255]$ and store them in an array of size 511. Then add up the frequency values starting from the first cell until the sum reaches half of the total frequency value (i.e., half of the frame size), the cell where the algorithm stops will indicate the median value. Median finding algorithm for the $t$-th test frame is given in Algorithm 2.

### F. Binary Classification for Foreground-Background Partition

Binary classification is required to classify a pixel position $x$ into foreground or background in the testing phase of the HMMB algorithm. If search is successful at pixel position $x$ for all $C$ channels in the three fold detection procedure (Section III-C), then $x$ belongs to the background.

**Algorithm 2** Median Finding Algorithm

**Input:** $newDiffold^t_C$ of size $N$ pixels.
I: **for** each channel $C \in \{R, G, B\}$ **do**
II: Take an integer array of size 511 initialized with zero values to count integers in the range $[-255, 255]$
III: **for** each value of $newDiffold^t_C(x)$ at position $x$ **do**
VI: increment the $i$-th cell of the array if $i = newDiffold^t_C(x)+255$ where $newDiffold^t_C(x)$ is obtained from Eqn. 1. (This step is nothing but the histogram of $newDiffold^t_C(x)$ for all $x$ positions after adding 255).
V: **end for**
VI: Add up the frequency values (cumulative sum) starting from $1^{st}$ cell of the array until the sum reaches $N/2$ and let $j$ be the last visited cell. If $N$ is even, then two possible cases are:
(a) $N/2$ and $(N-1)/2$ both are in $j$-th cell.
(b) $N/2$ in $j$-th cell and $(N-1)/2$ in $(j-1)$-th cell.
VII: **end for**
**Output:** If $N$ is even, then for case (a) return $j-255$, and for case (b) return $((j-1-255)+(j-255))/2 = j-255.5$. If $N$ is odd, then return $(j-255)$.

In all other possible cases $x$ belongs to the foreground. i.e.,

$$x \in \begin{cases} \textbf{background,} & \text{if successful search for all } C \text{ channels} \\ \textbf{foreground} & \text{otherwise.} \end{cases} \quad (3)$$

### G. Computational Complexity

Algorithm 1 is applied over $n$ training frames each of size $N$, visiting each pixel of a training frame $C$ times. Therefore the time complexity of this algorithm is $O(NCn) = O(N)$ (as $C = 1$ or 3 and $n$ ($<< N$) is a small constant).

In Algorithm 2, we visit each pixel position of a testing frame of size $N$ for $C$ ($C = 1$ or 3) times. Then we visit an array of constant size 511 where $N >> 511$. Time complexity of this algorithm is $O(C * (N + 511)) = O(N)$.

So both the algorithms have linear time complexity, which helps for real-time execution.

## IV. Performance Evaluation Accessories

We compared the proposed algorithm with some state-of-the-art methods. We used several datasets, some algorithms from the BGSLibrary [2] and some standard evaluation metrics, all of which are described below.

### A. Datasets

We used Wallflower [3] and CDnet [4] datasets for the experiments. These datasets contain many real life indoor and outdoor scenarios and many of them have the ground-truth frames along with the video frames. The Wallflower dataset contains 7 different challenging image categories (like moving background object, waving trees, shadow, camouflage etc.) for

[2] https://github.com/andrewssobral/bgslibrary/

[3] http://research.microsoft.com/en-us/um/people/jckrumm/WallFlower/TestImages.htm/

[4] http://changedetection.net/

TABLE II

$\alpha$, $\beta$ AND $\gamma$ PARAMETER VALUES OF HMMB METHOD FOR DIFFERENT VIDEO SCENES

| | PEDESTRIANS | FOUNTAIN02 | BUSSTATION | OVERPASS | DININGROOM | OFFICE | CANOE |
|---|---|---|---|---|---|---|---|
| $\alpha$ | 25 | 20 | 25 | 15 | 10 | 25 | 10 |
| $\beta$ | 0 | 0 | 0 | 5 | 5 | 0 | 0 |
| $\gamma$ | 45 | 35 | 45 | 25 | 15 | 45 | 15 |

background maintenance. All frames sizes are $160 \times 120$ pixels and average frame number is about 2290. The CDnet 2014 dataset contains 11 test categories and this is the extended version of the CDnet 2012 dataset. These datasets contain challenging video sequences including dynamic background, shadow, night low-visibility, air turbulence etc. In each category 3 to 6 video sequences are available which total to 53 sequences. Each video sequence contains 900 to 7000 frames amounting to about 160000 frames in total. Frame sizes vary from $320 \times 240$ to $720 \times 576$ pixels. Among all the datasets, we select those video sequences where some initial frames are dedicated for background construction with no foreground objects being present in those frames.

### B. Library

To judge the performance of HMMB with respect to the state-of-the-art algorithms, we borrow few algorithms from the open source BGSLibrary, which requires a C++ framework and the OpenCV library to run.

### C. Evaluation Metrics

To evaluate the performance of the proposed HMMB method by finding the foreground detected images, we used four metrics: Precision, Recall, F-measure and Accuracy [37], [38], defined as, Precision $= \frac{TP}{TP+FP}$, Recall $= \frac{TP}{TP+FN}$, F-measure $= \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$, and Accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$; where TP = true positives, FP = false positives, TN = true negatives and FN = false negatives.

## V. EXPERIMENTAL RESULTS

This section analyses the performance of the proposed method. To this end we compared the HMMB method (HMMB object-code[5]) with several existing methods. In this section, we stated the parameter values of HMMB along with the qualitative and quantitative results.

### A. Parameter Settings

We used few parameters to update the model and classify each pixel into foreground or background. These are

i) $\alpha$ for local illumination changes;
ii) $\beta$ and $\gamma$ for global illumination changes;
iii) $\zeta$ for abrupt illumination changes.

In the experimental stage, we set $\alpha \in \{5, 10, 15, 20, 25\}$, $\beta \in \{0, 5\}$, $\gamma \in \{15, 25, 35, 45\}$ and $\zeta \in \{20\}$. Here, actually we take 5 sets of triplets $(\alpha, \beta, \gamma)$ from the parameter set $PS = \{(5, 0, 15), (10, 0, 15), (15, 5, 25), (20, 0, 35), (25, 0, 45)\}$ for our experimental purpose (see Table II). $\alpha$ is used to threshold the local illumination changes whereas $\beta$ and $\gamma$ are set to threshold the global illumination changes (see Section III-E.1). The triplet $(\alpha, \beta, \gamma)$ control the

[5]HMMB object-code: www.isical.ac.in/~sujoyroy_r.

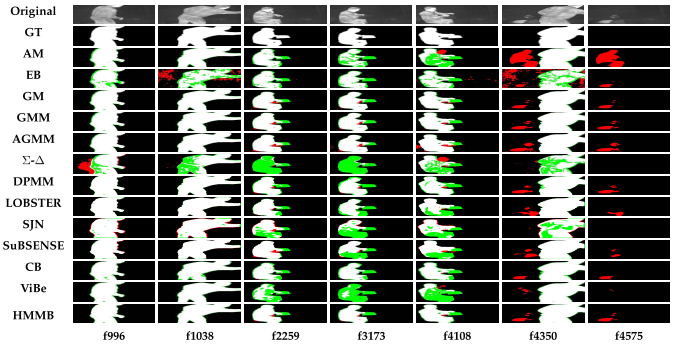



Fig. 3. Foreground detected results of *library* data sequence using different methods. White, red, green and black colours represent true positives, false positives, false negatives and true negatives, respectively.

conditions under which we allow the initial HMMB model to be adaptively updated due to temporally local and global changes.

The HMMB model is continuously updated for each pixel position $x$, each channel $C$ and each frame $t$. The HMMB model is updated either if an abrupt illumination change occurs or if there is a gradual change in the background of the video sequence. In case of gradual change, the model is updated only if both of the following two conditions are satisfied. The first condition is that the current rate of change of pixel value (i.e. $|newDiffOld^t_C(x)|$, or temporally local change, or short term change, where $|.|$ denotes the magnitude of a value) lies in the interval $[0, \alpha]$. The second condition is that the total change from the first test frame to the current frame ($|diffSum^t_C(x)|$, i.e. temporally global change, or long term change) lies in the interval $[\beta, \gamma]$. If the current rate of change is very large ($|newDiffOld^t_C(x)| > \alpha$), it is unlikely to have been caused by gradual change in the background and more likely to be because of the presence of foreground. Next, even when the current rate of change is small (i.e. $|newDiffOld^t_C(x)| < \alpha$), but if the total change is very large ($|diffSum^t_C(x)| > \gamma$), then also it is more likely to have been caused by foreground. But even if $|diffSum^t_C(x)| < \gamma$, we cannot be sure that this is because of changing background. So we may not want to start updating the background model immediately, instead we may want to wait for the changes to accumulate to a certain minimum threshold $\beta$ before we begin updating our background model. From the above considerations it is clear that, firstly the parameter $\alpha$ is a threshold that controls whether the rate of change of the background pixel values is small enough, for background model updates to occur. Secondly, the parameter $\beta$ is assigned a small value (typically less than 5) when we want to wait for the changes to accumulate upto $\beta$ before we start updating our background model. Thirdly, the parameter $\gamma$ is a threshold that controls whether the total change of pixel value (global change) is small enough, for background model updates to occur. When the triplet $(\alpha, \beta, \gamma)$ are close to their

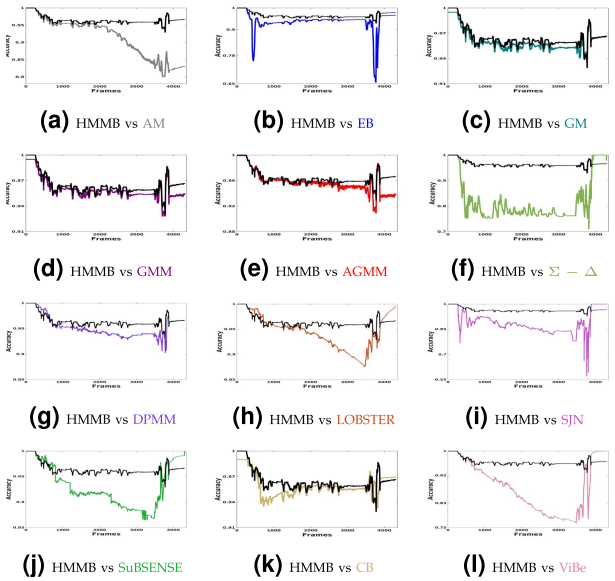



Fig. 4. Comparison of accuracy values for different methods with HMMB method for all test frames.

optimal values, for a given video sequence, small changes to $(\alpha, \beta, \gamma)$ is not likely to abruptly break the aforementioned conditions. Therefore small changes to $(\alpha, \beta, \gamma)$ near their optimal values will not produce significantly different results, although drastically varying the parameter values may degrade the performance. We have experimentally found that for most video sequences at least one of the $(\alpha, \beta, \gamma)$ triplets out of the parameter set $PS$, produces close to optimal results.

The parameter $\zeta$ is used for detecting abrupt illumination changes. If for a given frame, the median value of the difference image $newDiffOld^t_C$ is greater than $\zeta$, we mark that frame as an abrupt change. We have found from our experimental studies that any value of $\zeta$ in the interval [14, 58], can be used to successfully detect all the abrupt changes in all the tested video sequences. Thus our method is completely insensitive to the parameter $\zeta$ as long as it lies within the above range. For the experimental results given in this article, we have used $\zeta = 20$, which could properly detect all the abrupt changes.

### *B. Experimental Results and Comparison With Existing Techniques*

We compared the performance of HMMB with state-of-the-art BS algorithms like: $(i)$ Adaptive median (AM) [27], $(ii)$ Single Gaussian model (GM) [2], $(iii)$ Gaussian mixture model (GMM) [3], $(iv)$ Adaptive Gaussian mixture model (AGMM) [4], $(v)$ Eigenbackground (EB) method [8], $(vi)$ Zipfian $\Sigma - \Delta$ method [16], $(vii)$ Local binary similarity patterns (LOBSTER) method [6], $(viii)$ Multiple cues BS method (SJN) [33], $(ix)$ Self-balanced sensitivity segmenter (SuBSENSE) method [7], $(x)$ Dirichlet process mixture models (DPMM) [31], $(xi)$ Codebook (CB) model [17] and $(xii)$ ViBe universal BS method [14].

We have used the BGSLibrary for algorithms $(i)$–$(ix)$. We have implemented the CB model [17] and DPMM model [31]; and the authors of ViBe [14] have provided their ViBe code for our experimental purpose. After processing a frame, we used a $5 \times 5$ median filter to refine the results. For all foreground detected images, white, red, green and black colours are used to represent the true positives (foreground), false positives (background detect as foreground), false negatives (foreground detect as background) and true negatives (background), respectively for better visualization.

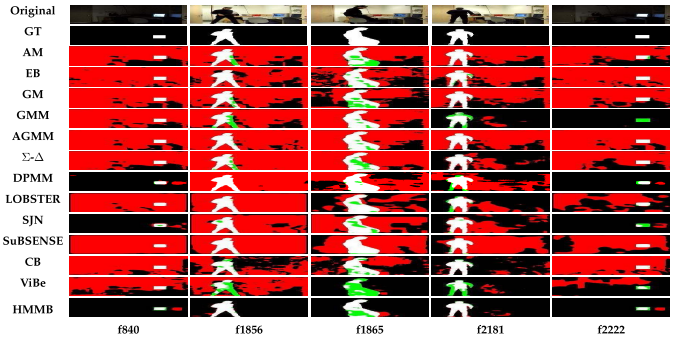



Fig. 5. Foreground detected results of *light switch* data sequence using different methods. White, red, green and black colours represent true positives, false positives, false negatives and true negatives, respectively.

### *C. Qualitative Evaluations*

In the first experiment, we considered the *library* video clips of "thermal" category from CDnet dataset, where the object stayed static for a long time in the same place and someone can think of it as a background by watching only those frames. Three colour channels of all the frames in this category holds the same intensity value for a particular pixel position. Foreground detected results of a few frames using different methods are shown in Figure 3. Original frames (Original), ground-thuth (GT), symbols corresponding to different state-of-the-art methods and the proposed HMMB method are shown on the left side of each sequence (each row). Frame numbers f996, f1038, f2259, f3173, f4108, f4350, f4575 are shown in the bottom of each column. From the figure, it is evident that GM, GMM, AGMM, DPMM, CB and HMMB are detecting the object properly. The AM method falsely identified the person in frames f4350 and f4575; and as a result accuracy degraded. SJN produce false positives in the major portion of the video frames along with many false negatives in frames f3173, f4108 and f4350. LOBSTER and SuBSENSE start with better accuracy but in frames f3173, f4108 produce some false negatives. The $\Sigma - \Delta$ method mainly concerns the motion of object and detects the object as a background (see frames f2259 and f3173) where the object stops its movement. In frames f2259 and f3173 ViBe also produced false negatives like $\Sigma - \Delta$ method and degrading the accuracy.

In the second experiment, we chose the *light switch* video scene from Wallflower repository where sudden illumination change occurs. In this indoor scene, sudden illumination change occurred due to switch on/off of the light for several times. In this dataset, we manually generate some ground-truths with the help of some experts for better understanding of the results. HMMB can identify the moving object with this sudden illumination change in the video scene. But all other considered BS methods (listed in Section V-B) could not handle this type of complex scenario. The detected foreground

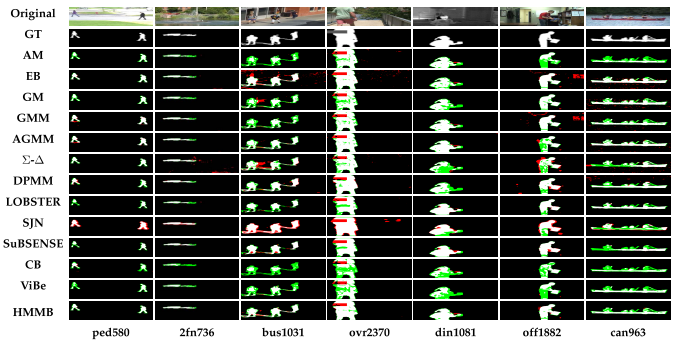



Fig. 6. Foreground detection on 7 video clips taken from CDnet dataset. Each row depicts figures of original input images (Original), ground-truth images (GT), results corresponding to AM [27], EB [8], GM [2], GMM [3], AGMM [4], $\Sigma - \Delta$ [16], DPMM [31], LOBSTER [6], SJN [33], SuBSENSE [7], CB [17], ViBe [14] and the proposed method HMMB. Columns represent 580th frame of *Pedestrians* (ped580), 736th frame of *Fountain*02 (2fn736), 1031th frame of *BusStation* (bus1031), 2370th frame of *Overpass* (ovr2370), 1081th frame of *DiningRoom* (din1081), 1882th frame of *Office* (off1882) and 963th frame of *Canoe* (can963) from left to right. White, red, green and black colours represent the true positives, false positives, false negatives and true negatives, respectively.

of a few frames (f840, f1856, f1865, f2181, f2222) for different methods are shown in Figure 5. The figure indicates that most of the methods (AM, EB, GM, GMM, AGMM, $\Sigma - \Delta$, LOBSTER, SuBSENSE, CB and ViBe) falsely detected the major portion of the background as object due to sudden illumination changes in the scene and downgrade the detection accuracy. The output emphasizes that the proposed HMMB method outperforms the above mentioned existing methods for video scenes with sudden illumination changes. Accuracy diagram for video scene with sudden illumination changes (see Figure 8) is another evidence in support of the claim.

In the third experiment, we chose 7 video scenes of different category from CDnet dataset. These scenes are *Fountain*02, *Canoe* and *Overpass* from "dynamic background", *Pedestrians* and *Office* from "baseline", *Bus Station* from "shadow" and *Dinning Room* from "thermal" category. This dataset contains different challenging indoor and outdoor video scenes with lighting variation, dynamic backgrounds, shadows and different object motions, which are helpful to establish the robustness and effectiveness of the proposed HMMB algorithm. We tested our algorithm on many other video sequences and the results are available in the supplementary materials[6]. Figure 6 highlights a few frames from original images, GT images and foreground detected images after applying different state-of-the-art methods and the proposed method. This figure also suggests that the proposed method can handle different complex video scenes having dynamic background with sprayed water in *Fountain*02, waving trees and moving water in both *Overpass* and *Canoe*, shadow effect in *Bus Station*, thermal effect in *Dinning Room*, a mixture of mild challenges like dynamic background, shadow effect and object motion in *Pedestrians* and *Office*, efficiently.

In Figure 6, second column shows a frame of *Fountain*02 video scene, where a car is moving behind the fountain. It is clear that GMM, AGMM, DPMM, LOBSTER, SuBSENSE and HMMB can better detect the car with very less amount of false negatives. On the other hand AM, GM, CB and ViBe produce more false negatives, SJN produces more false positives but EB and $\Sigma - \Delta$ produce both false positives along with false negatives. SJN makes false positives in almost all the frames (see row 11). AM produces lots of false negatives, whereas EB produces plenty of false positives in *Bus Station, Overpass, Office, Canoe* (see third and fourth rows) because the methods are unable to handle the movements of tree branches, water surface and lighting changes in the video scenes. In the *Pedestrians* scene, where persons are moving in front of a simple and steady background (see first column), GMM and AGMM produce little amount of false positives in between the legs of the walking person by misclassifying the shadow. CB and ViBe generate some false negatives, but HMMB detects more accurate foreground. Many false negatives are produced by DPMM in *Dinning Room* scene, SuBSENSE in *Canoe* and LOBSTER in both the *Bus Station* and *Pedestrians* scenes. In *Office* scene, a person enters in a room where lighting intensity varies frequently, EB and GMM exhibit false positives near the right top corner because of non adaptation of the lighting variations. GM, GMM, AGMM, $\Sigma - \Delta$ and DPMM methods produce false positives near the water surface and false negatives in body of the persons in the boat, in the *Canoe* scene. In the *Overpass* scene, all the methods (except EB and SJN) handle the movement of the tree branches efficiently but AM, GM, GMM, LOBSTER, SuBSENSE, CB and ViBe could not properly capture the water motion and produce false negatives near the hip area of the person. In the scenes *Bus Station*, *Dinning Room*, *Office* and *Canoe*, CB and ViBe produce huge amount of false negatives. HMMB can produce relatively better object detection results in all the frames but in *Canoe* scene it produces more false negatives near the body of the persons in the boat.

Figures 3, 4, 5 and 6 display that HMMB gives better results than the state-of-the-art methods. More results of full video sequences are available at www.isical.ac.in/~sujoyroy_r. The results suggest that HMMB can handle complex dynamic background as well as little or more object motions. Quantitative evaluations also ensure the superior performance of HMMB.

### *D. Quantitative Evaluations*

In the first experiment, the thresholds taken for the proposed method are $\alpha = 15, \beta = 5, \gamma = 25$ and $\delta = 20$. In Figure 4a, we see AM starts with good accuracy but degrades gradually (after 2000 frames) because of the shadow effect [14]. The method EB (Figure 4b) gives good performance but suddenly jumps near the region of frame numbers 400 and 3750 due to small motion of the object. HMMB performs better than SJN (Figure 4i) for major portion of the video. LOBSTER (Figure 4h) and SuBSENSE (in Figure 4j) start with higher accuracy than HMMB but when the object is static the accuracy degrades. On the other hand Figure 4f (for $\Sigma - \Delta$) and Figure 4l (for ViBe) display that they perform well when there is some movement of the object in the scene, but when the object is stationary for sometime (around frame numbers 500 to 3900), it gradually merges that region with the background and the accuracy comes down.

[6]www.isical.ac.in/~sujoyroy_r.

TABLE III

PRECISION (PRE), RECALL (REC) AND F1 VALUES OF DIFFERENT METHODS. IN EACH COLUMN, RED, GREEN AND BLUE COLOUR REPRESENT THE BEST, THE SECOND BEST AND THE THIRD BEST MEASUREMENTS

| | PEDESTRIANS | | | FOUNTAIN02 | | | BUSSTATION | | | OVERPASS | | | DININGROOM | | | OFFICE | | | CANOE | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Pre | Rec | F1 | Pre | Rec | F1 | Pre | Rec | F1 | Pre | Rec | F1 | Pre | Rec | F1 | Pre | Rec | F1 | Pre | Rec | F1 |
| **AM** | 0.9123 | 0.5307 | 0.6710 | 0.6873 | 0.4846 | 0.5607 | 0.9018 | 0.2734 | 0.4196 | 0.7431 | 0.8329 | 0.7854 | 0.7852 | 0.7815 | 0.7833 | 0.8190 | 0.6035 | 0.6949 | 0.7914 | 0.5071 | 0.6181 |
| **EB** | 0.8815 | 0.7972 | 0.8372 | 0.7184 | 0.6967 | 0.7023 | 0.8031 | 0.7091 | 0.7532 | 0.4891 | 0.8937 | 0.6322 | 0.8573 | 0.7106 | 0.7771 | 0.6480 | 0.8933 | 0.7511 | 0.5015 | 0.8903 | 0.6416 |
| **GM** | 0.8935 | 0.6030 | 0.7201 | 0.9287 | 0.5092 | 0.6578 | 0.8281 | 0.4837 | 0.6107 | 0.7583 | 0.7813 | 0.7628 | 0.8388 | 0.7181 | 0.7738 | 0.5356 | 0.8314 | 0.6515 | 0.7835 | 0.5961 | 0.6771 |
| **GMM** | 0.8922 | 0.7703 | 0.8268 | 0.9318 | 0.7021 | 0.8008 | 0.8417 | 0.4971 | 0.6251 | 0.7923 | 0.7958 | 0.7896 | 0.9027 | 0.8418 | 0.8712 | 0.6291 | 0.8531 | 0.7241 | 0.8725 | 0.6754 | 0.7614 |
| **AGMM** | 0.8755 | 0.8102 | 0.8415 | 0.9346 | 0.7101 | 0.8048 | 0.8312 | 0.6096 | 0.7034 | 0.7581 | 0.8592 | 0.8055 | 0.8391 | 0.8732 | 0.8558 | 0.6371 | 0.8377 | 0.7237 | 0.8828 | 0.7037 | 0.7831 |
| **SD** | 0.9038 | 0.7381 | 0.8126 | 0.4010 | 0.8026 | 0.5348 | 0.8739 | 0.3370 | 0.4864 | 0.5951 | 0.8093 | 0.6859 | 0.8493 | 0.3712 | 0.5145 | 0.8119 | 0.4628 | 0.5708 | 0.8112 | 0.5374 | 0.6465 |
| **DPMM** | 0.8453 | 0.7353 | 0.7857 | 0.8251 | 0.7490 | 0.7852 | 0.7834 | 0.7214 | 0.7511 | 0.7622 | 0.6290 | 0.6892 | 0.8152 | 0.7514 | 0.7820 | 0.8393 | 0.7885 | 0.8131 | 0.7329 | 0.7043 | 0.7183 |
| **LOBSTER** | 0.8371 | 0.8020 | 0.8192 | 0.8055 | 0.5607 | 0.6612 | 0.8716 | 0.6265 | 0.7290 | 0.7991 | 0.5281 | 0.6359 | 0.9032 | 0.7608 | 0.8259 | 0.9130 | 0.7467 | 0.8215 | 0.7803 | 0.6065 | 0.6825 |
| **SJN** | 0.8905 | 0.6726 | 0.7655 | 0.7552 | 0.6626 | 0.7059 | 0.7325 | 0.6738 | 0.7019 | 0.5851 | 0.4976 | 0.5378 | 0.8125 | 0.6169 | 0.7013 | 0.7415 | 0.4633 | 0.5703 | 0.8014 | 0.4785 | 0.5985 |
| **SuBSENSE** | 0.9021 | 0.7784 | 0.8357 | 0.8511 | 0.7758 | 0.8117 | 0.8839 | 0.6988 | 0.7805 | 0.7296 | 0.8946 | 0.8037 | 0.8902 | 0.7822 | 0.8327 | 0.9395 | 0.7915 | 0.8592 | 0.9362 | 0.6059 | 0.7357 |
| **CB** | 0.9271 | 0.6143 | 0.7390 | 0.9507 | 0.4417 | 0.6032 | 0.9251 | 0.4181 | 0.5759 | 0.8171 | 0.6037 | 0.6944 | 0.9050 | 0.5861 | 0.7108 | 0.9017 | 0.5956 | 0.7173 | 0.9375 | 0.5960 | 0.7287 |
| **ViBe** | 0.9435 | 0.5871 | 0.7238 | 0.9517 | 0.4872 | 0.6445 | 0.9208 | 0.3805 | 0.5385 | 0.7517 | 0.6391 | 0.6908 | 0.9503 | 0.4287 | 0.5909 | 0.8479 | 0.6718 | 0.7496 | 0.9255 | 0.5992 | 0.7274 |
| **HMMB** | 0.9139 | 0.7990 | 0.8526 | 0.9411 | 0.7530 | 0.8366 | 0.8391 | 0.7332 | 0.7826 | 0.8029 | 0.8431 | 0.8225 | 0.9286 | 0.8504 | 0.8874 | 0.9230 | 0.8184 | 0.8676 | 0.9193 | 0.6921 | 0.7897 |

We compared the accuracy of each of the state-of-the-art methods with HMMB in Figure 4. Here $y$-axis represents the accuracy of different methods throughout the testing frames ($x$-axis) compared with HMMB in different sub-figures. Here, we take different $y$-axis range for better understanding of the accuracy values for existing methods with respect to the proposed method. From these figures, we found that, results obtained by the proposed method are comparable with GM, GMM, AGMM, DPMM and CB methods.

The thresholds used for the second experiment for different methods are shown in Table IV. Thresholds taken for the proposed method are $\alpha = 25, \beta = 0, \gamma = 45$ and $\delta = 20$. Figure 8 demonstrates the accuracy values of all the frames, where we found that HMMB method achieves the highest accuracy values by a big margin with respect to the other state-of-the-art methods. This figure shows the median accuracy of the proposed method as 0.9 and the variance is also very small compared to other methods, which proves the robustness of the proposed method.

Table II shows the parameter values of HMMB method for different video scenes and for the state-of-the-art methods, we consider the same parameter settings provided by the authors to generate the comparative results. In Table III, we present the Precision (Pre), Recall (Rec) and F-measure (F1) values for the overall performance of HMMB method with other methods by considering all frames of each category. We use red, green and blue colour to mark the best, the second best and the third best measurements, respectively in each column of the table. In *Pedestrians* and *Overpass* video secnes, all the methods performed well but in *Fountain*02, F1 values obtained by AM and SD methods, decreased as they could not adapt to the dynamic background properly. In *Bus Station*, where few persons are talking to each other, the shadow effect makes the scene complex to identify the foreground accurately. Here smaller Recall values of AM, SD, CB and ViBe methods indicate that they create many false negatives and as a result F1 values are smaller despite high Precision values. The F1 value of the proposed method is higher than the other methods and this suggests that the robustness of the proposed method in complex environments. As SJN method produces smaller Recall values in most of the sequences, F1 values are also reduced. In *Dinning Room* scene, ViBe and SD produce many false negatives and both the Recall and F1 values are reduced. DPMM and SuBSENSE produce higher Precision, Recall and F1 values in majority

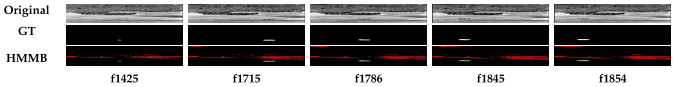



Fig. 7. Foreground detection of the video scene *turbulence0* of "turbulence" category taken from CDnet dataset. Each row depicts figures of original input images (Original), ground-truth images (GT) and the proposed method HMMB. Columns represent the frame numbers f1425, f1715, f1786, f1845, f1854 from left to right. White, red, green and black colours represent the true positives, false positives, false negatives and true negatives, respectively.

TABLE IV

PARAMETERS OF DIFFERENT BS METHODS USED FOR *Light Switch* SEQUENCE

| ALGORITHM | PARAMETERS |
|---|---|
| AM [27] | T = 25 |
| EB [8] | T = 250, ED = 10 |
| GM [2] | T = 15, LR = 0.0001 |
| GMM [3] | T = 10, LR = 0.0001, n = 5 |
| AGMM [4] | T = 25, LR = 0.0001, n = 5 |
| $\Sigma - \Delta$ [16] | AF = 1, $V_{min} = 10, V_{min} = 245$ |
| DPMM [31] | wt = 0.0001, sd mul = 0.6, cap = 512, com = 8, con = 0.01 |
| LOBSTER [6] | $T_r = 0.365$, N = 35, $\sharp_{min} = 2, T_d = 30, T_{desc} = 12$ |
| SJN [33] | $\alpha = 0.05, \tau_T = 1, \tau_A = 10, \varepsilon_1 = 15, \varepsilon_2 = 20$ |
| SuBSENSE [7] | N = 50, $\sharp_{min} = 2, T_r = 0.3, T_d = 30, \alpha^{LT} = 100$ |
| CB [17] | $\alpha = 0.4, \beta = 1.1, \epsilon_1 = 0.2, \epsilon_2 = 40$ |
| ViBe [14] | N = 20, R = 20, $\sharp_{min} = 2, \phi = 16$ |

T = threshold, ED = embedded dimensions, LR = learning rate, n = number of Gaussians, and other parameters are the same as given in the corresponding papers.

of the video sequences. In *Fountain*02, *Overpass* and *Canoe*, LOBSTER produces relatively lower F1 values than the other video scenes. As *Canoe* video scene contains dynamic background due to frequent changes of both the movement of tree branches and streaming water in the lake, CB method produces highest Precision value but high false negatives reduced the F1 value. Table III shows that the proposed method produces better balance of Precision and Recall values and achieves the highest F1 values for all the 7 video sequences. This statistical measurements display that HMMB gives better performance than the other methods.

### *E. System Settings and Execution Time*

The proposed method has been proved to be efficient in foreground classification for different challenging video sequences. Both qualitative and quantitative evaluations highlight the superiority of HMMB over other existing methods. We implemented HMMB in C++ using the OpenCV library. We executed all the algorithms on an Intel i7 machine with 3.40 GHz CPU, 8 GB RAM and Linux operating system.

In Figure 9, we compared the execution time of HMMB method with all other methods using box plot. Here we

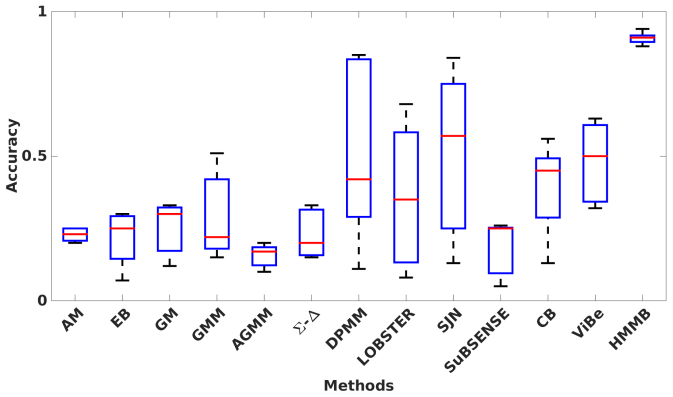



Fig. 8. Box plot showing Accuracy values of different methods for frames with sudden illumination changes in the "light switch" sequence. In this box plot, the middle line (red line) represents the median accuracy. The upper and lower lines (parallel to horizon) of the rectangle covering the red line are third and first quartile respectively. The upper and lower lines (parallel to horizon) outside the rectangle are respectively maximum and minimum accuracy.

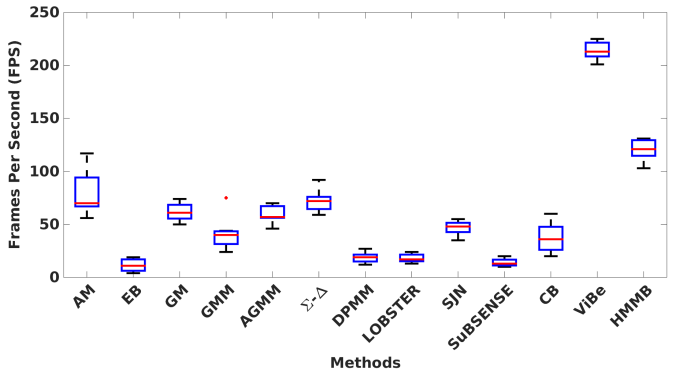



Fig. 9. Box plot showing processing speed of the videos taken from CDnet dataset used in Figure 6 in terms of frames per second (FPS).

found that except ViBe, HMMB performs better than the best execution time shown by other methods. GM, GMM, AGMM and CB use many floating point operations and/or transcendental functions to classify a pixel and update the background model. On the other hand LOBSTER, SuBSENSE calculate LBSP for the neighbourhood of every pixel. These algorithms therefore run at lower speeds. The ViBe uses only simple integer operations and some tricks for fast searching (e.g., swapping pixel values into the first place into the memory that is more likely to occur in the near future). Proposed HMMB method, like ViBe, also uses integer operations, but unlike ViBe, it repeats the same procedure for all three colour channels, whereas ViBe does not run separately for every channel, which makes it faster than HMMB.

### *F. Discussion and Future Works*

Though the proposed method is able to detect foregrounds in several complex environments having dynamic background motions, presence of shadow and sudden illumination variations, the method is unable to produce good foreground detection results for video sequences where constant distortion occurs and the object size is very small.

We consider a video scene *turbulence0* of "turbulence" category from CDnet dataset. The video is captured with a telephoto lens where air turbulence distorted the frames throughout the video. All three channels of each frame in this video contain the same intensity value for a pixel position. A few original images, GT and foreground detected images (f1425, f1715, f1786, f1845, f1854) using HMMB are given in Figure 7. The foreground detection results (see row 3) show plenty of false positives due to constant unsteady movement of air in the scene. Therefore, HMMB fails to detect accurate foreground for such sequences. Another weak point (although occurring with very low probability of $1/(2^{256}-1)$) of the proposed method appears when a pixel contains every intensity value in the range $[0, 255]$ in the training phase of the algorithm. In this situation the pixel position belongs to the background for the rest of the testing frames.

In future, to make the proposed algorithm more robust against more complex sequences, we wish to consider low-rank model. The low-rank concept helps us to efficiently analyses the portions where the objects might be present. Thereafter, we will consider a motion saliency map to remove unwanted background motions and use a regularizing parameter to idenify the foreground regions accurately.

## VI. Conclusion

In this article, a real-time HMMB model is proposed for foreground-background classification from a video scene. Major advantages of the proposed method are: a self adaptive model for bucket adjustment, gives equal priority to local and global information, and can cope with sudden illumination changes in the scene. The proposed method gave higher accuracy in most of the video sequences used for experimentation and execution time is also less than most of the existing state-of-the-art methods. Experimental results show that HMMB model is resilient to different types of object motions and also various lighting conditions.

The proposed method could further be improved with model construction by taking less number of initial frames and memory, an adaptive parameter settings to increase accuracy, dealing with moving camera scenes, capture more complex motion models and camouflaged object detection.

## Appendix A

The parameter $diffSum_C^t(x)$ is the difference between the intensity values of the current ($t$-th) and the first test frames.

*Proof:* Let $a_1, a_2, \ldots, a_t$ be the pixel intensity values of the first $t$ test frames of channel $C$ at position $x$ and $d_i = a_{i+1} - a_i$ be the difference between the intensity values of two consecutive frames $i+1$ and $i$, for all $i = 1, 2, \ldots, t-1$. Now, $diffSum_C^t(x) = \sum_{i=1}^{t-1} d_i = d_1 + d_2 + \ldots + d_{t-1} = (a_2 - a_1) + (a_3 - a_2) + \ldots + (a_t - a_{t-1}) = a_t - a_1$. ■

## References

[1] A. Sobral and A. Vacavant, "A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos," *Comput. Vis. Image Understand.*, vol. 122, pp. 4–21, May 2014.

[2] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, Jul. 1997.

[3] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2. Jun. 1999, pp. 246–252.

[4] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 773–780, 2006.

[5] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Background modeling and subtraction by codebook construction," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, vol. 5. Oct. 2004, pp. 3061–3064.

[6] P.-L. St-Charles and G.-A. Bilodeau, "Improving background subtraction using local binary similarity patterns," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2014, pp. 509–515.

[7] P. L. St-Charles, G. A. Bilodeau, and R. Bergevin, "SuBSENSE: A universal change detection method with local adaptive sensitivity," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 359–373, Jan. 2015.

[8] N. M. Oliver, B. Rosario, and A. P. Pentland, "A Bayesian computer vision system for modeling human interactions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 831–843, Aug. 2000.

[9] W. Hu, X. Li, X. Zhang, X. Shi, S. Maybank, and Z. Zhang, "Incremental tensor subspace learning and its applications to foreground segmentation and tracking," *Int. J. Comput. Vis.*, vol. 91, no. 3, pp. 303–327, 2011.

[10] W. Wang, J. Shen, X. Li, and F. Porikli, "Robust video object cosegmentation," *IEEE Trans. Image Process.*, vol. 24, no. 10, pp. 3137–3148, Oct. 2015.

[11] W. Wang, J. Shen, and F. Porikli, "Saliency-aware geodesic video object segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3395–3402.

[12] L. Chen, J. Shen, W. Wang, and B. Ni, "Video object segmentation via dense trajectories," *IEEE Trans. Multimedia*, vol. 17, no. 12, pp. 2225–2234, Dec. 2015.

[13] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 747–757, Aug. 2000.

[14] O. Barnich and M. Van Droogenbroeck, "ViBe: A universal background subtraction algorithm for video sequences," *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, Jun. 2011.

[15] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2000, pp. 751–767.

[16] A. Manzanera, "Σ-Δ background subtraction and the Zipf's law," in *Proc. Prog. Pattern Recognit., Image Anal. Appl. (CIARP)*, 2007, pp. 42–51.

[17] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imag.*, vol. 11, no. 3, pp. 172–185, Jun. 2005.

[18] B. P. L. Lo and S. A. Velastin, "Automatic congestion detection system for underground platforms," in *Proc. Int. Symp. Intell. Multimedia, Video Speech Process. (ISPACS)*, May 2001, pp. 158–161.

[19] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts and shadows in video streams," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.

[20] S. S. Huang, L. C. Fu, and P. Y. Hsiao, "Region-level motion-based background modeling and subtraction using MRFs," *IEEE Trans. Image Process.*, vol. 16, no. 5, pp. 1446–1456, May 2007.

[21] J.-H. Ahn and H. Byun, "Human silhouette extraction method using region based background subtraction," in *Proc. Int. Conf. Comput. Vis./Comput. Graph. Collaboration Techn. Appl. (MIRAGE)*, 2007, pp. 412–420.

[22] P. D. Z. Varcheie, M. Sills-Lavoie, and G.-A. Bilodeau, "An efficient region-based background subtraction technique," in *Proc. IEEE Can. Conf. Comput. Robot Vis. (CRV)*, May 2008, pp. 71–78.

[23] M. Shah, J. Deng, and B. Woodford, "Enhanced codebook model for real-time background subtraction," in *Proc. 18th Int. Conf. Neural Inf. Process. (NIPS)*, 2011, pp. 449–458.

[24] X. Zhou, C. Yang, H. Zhao, and W. Yu, "Low-rank modeling and its applications in image analysis," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 36:1–36:33, Jan. 2014.

[25] X. Liu, G. Zhao, J. Yao, and C. Qi, "Background subtraction based on low-rank and structured sparse decomposition," *IEEE Trans. Image Process.*, vol. 24, no. 8, pp. 2502–2514, Aug. 2015.

[26] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Proc. 7th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 1. Sep. 1999, pp. 255–261.

[27] N. J. B. McFarlane and C. P. Schofield, "Segmentation and tracking of piglets in images," *Brit. Mach. Vis. Appl.*, vol. 8, no. 3, pp. 187–193, 1995.

[28] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proc. IEEE*, vol. 90, no. 7, pp. 1151–1163, Jul. 2002.

[29] A. Manzanera and J. C. Richefeu, "A robust and computationally efficient motion detection algorithm based on Σ-Δ background estimation," in *Proc. Indian Conf. Comput. Vis., Graph. Image Process. (ICVGIP)*, 2004, pp. 46–51.

[30] D. M. W. Powers, "Applications and explanations of Zipf's law," in *Proc. Joint Conf. New Methods Lang. Process. Comput. Natural Lang. Learn.*, 1998, pp. 151–160.

[31] T. S. F. Haines and T. Xiang, "Background subtraction with Dirichlet process mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 670–683, Apr. 2014.

[32] M. Braham and M. Van Droogenbroeck, "Deep background subtraction with scene-specific convolutional neural networks," in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, May 2016, pp. 1–4.

[33] S. Noh and M. Jeon, "A new framework for background subtraction using multiple cues," in *Proc. Asian Conf. Comput. Vis. (ACCV)*, 2012, pp. 493–506.

[34] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, 2008.

[35] E. Monari and C. Pasqual, "Fusion of background estimation approaches for motion detection in non-static backgrounds," in *Proc. IEEE Conf. Adv. Video Signal Based Surveill. (AVSS)*, Sep. 2007, pp. 347–352.

[36] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.

[37] M. Piccardi, "Background subtraction techniques: A review," in *Proc. IEEE Int. Conf. Syst., Man Cybern. (SMC)*, vol. 4. Oct. 2004, pp. 3099–3104.

[38] C. E. Metz, "Basic principles of ROC analysis," *Seminars Nucl. Med.*, vol. 8, no. 4, pp. 283–298, Oct. 1978.

**Sujoy Madhab Roy** received the B.Tech. degree in computer science and engineering from Maulana Abul Kalam Azad University of Technology, India, in 2005 and the M.Tech. degree in computer science from Indian Statistical Institute, Kolkata, India, in 2013. He is currently a Senior Research Scholar with the Machine Intelligence Unit, Indian Statistical Institute. His research interests include image and video processing, pattern recognition, machine learning.

**Ashish Ghosh** is a Professor with the Machine Intelligence Unit, Indian Statistical Institute. He has already published over 200 research papers in internationally reputed journals and refereed conferences, and has edited eight books. His research interests include pattern recognition and machine learning, data mining, image and video analysis, soft computing, neural networks, evolutionary computation, and bioinformatics. He received the prestigious Young Scientists Award in Engineering Sciences from the Indian National Science Academy in 1995 and the prestigious Young Scientists Award in computer science from the Indian Science Congress Association in 1992.