Electronic Letters on Computer Vision and Image Analysis 13(1):50-67, 2014

A Novel Interest-Point-Based Background Subtraction Algorithm

Alireza Dehghani and Alistair Sutherland

School of Computing, Dublin City University, Dublin, Ireland
Received 10^{th} Mar 2014; accepted 12^{th} Jun 2014

Abstract

Current Background Subtraction (BGS) algorithms are mostly pixel-based methods. We propose an Interest-Point(IP)-based BGS algorithm applicable in IP-based Computer Vision applications. Based on a block-wise processing strategy, the frames are divided into blocks of the same size. IPs inside each block are together Events. Throughout the frame sequence, the algorithm stores the Events in each block as well as the numbers of their occurrences (Repetition Index (RI)) in a Binary Tree. The RI is used to classify Events as either background or foreground. The background Events appear significantly more often than foreground Events. Events with an RI greater than a certain threshold are classified as background, the rest as foreground. This Event classification is used to label IPs of frames into the foreground and background IPs. Experimental results quantitatively show that the proposed algorithm delivers a good subtraction rate in comparison with other BGS approaches. Moreover, it creates a map of the background usable for further processing, it is robust to changes in illumination and can keep itself updated to changes in the background.

Key Words: Background Subtraction; Foreground Detection; Interest Points.

1 Introduction

The segmentation of areas of an image related to moving objects, foreground, from the areas related to static objects in the scene, background, is called Background Subtraction (BGS) when the processed image is captured by static cameras [1]. This is the earliest stage of many Computer Vision (CV) applications such as human motion analysis, automated surveillance, video indexing, and vehicle navigation. Therefore, it exhibits a strong influence on further processing [2]. Accordingly, a great deal of research has been conducted on BGS over the past few years and many algorithms have been proposed, most of them pixel-based. They rely on the difference between pixels either *individually* [3], using the pixels' illumination, or *regionally*, using the texture of a group of pixels in the form of blocks [4] or clusters [5], to model and update the background [6].

In contrast, Interest Points (IPs) have not so far been used for BGS. As the most lightweight way of object representation [7], they represent well-defined features of the image such as corners. This makes them superior to other object representation methods, such as Kernel and Silhouette, in terms of the speed, accuracy and robustness [8]. IPs have delivered a high level of descriptive power and robustness to illumination changes [9]. Many remarkable IP detector and local feature descriptors such as the Scale Invariant Feature Transform (SIFT)

Correspondence to: <alireza.dehghani2@mail.dcu.ie>

Recommended for acceptance by <Frèdèric Lerasle>
ELCVIA ISSN: 1577-5097
Published by Computer Vision Center / Universitat Autònoma de Barcelona, Barcelona, Spain

[10], Speeded Up Robust Features (SURF) [11], and Gradient Location and Orientation Histogram (GLOH) [12] have been proposed by now. The ORB (Oriented FAST and Rotated BRIEF) [13], which is rotation-invariant and resistant to noise, performs the same as SIFT and better than SURF, while being twice as fast. The different feature descriptors have been compared in the literature [12].

Following these, we propose an IP-based BGS algorithm in this paper, applicable in IP-based CV applications. More precisely, in our approach the images are firstly divided into blocks of the same size. IPs inside blocks are combined together to create Events. Throughout the frame sequence, the algorithm stores Events for each block as well as the number of their occurrences (Repetition Index (RI)) in a Binary Tree (BT). Then, the RI is used to classify Events into background and foreground Events. If any Event has RI more than a threshold, it is classified as background Event. Otherwise it is considered as foreground Event. This Event classification is used to label the IPs of each frame into the foreground and background IPs. The main *contributions* of our proposed approach lie in its difference from pixel-based algorithms in the following respects:

- The number of IPs is much less than the number of pixels in the image. So they reduce the computational cost greatly.
- IP-based BGS methods are robust to illumination changes due to the robustness of IPs to environmental changes [9]. In contrast, pixel-based BGS methods face a significant difficulty regarding illumination changes. Although they compensate for the weakness of a single frame by using the temporal information computed from a sequence of frames [14], it increases the computation cost by itself.
- Our algorithm can work widely in daylight, twilight or night-time as well as different scene conditions. In contrast, the pixel-based BGS methods need to impose constraints such as the background colour, background contents, and the environmental light conditions in order to make BGS more feasible [8].
- IP-based BGS approach delivers more information to CV applications than pixel-based approaches. To put it another way, some of the foreground IPs of the moving object can still be observed during partial occlusion [15]. This may be used to overcome the occlusion problems in tracking applications. In comparison, although some pixels of the foreground may also be visible in the pixel-based approaches, they are not as individually meaningful as the IPs.
- This approach works relatively well when the foreground objects do not move (although in this case it seems the IPs of the foreground should behave like the background and be still, they have movement enough to be recognized as the foreground IPs). In contrast, the traditional pixel-based approaches only subtract the background when the foreground moves.

1.1 Related Works

Owing to the importance of BGS [16] as the earliest step of algorithms such as tracking, recognition, and behaviour analysis approaches, a huge amount of research has been conducted on this field. On this basis, several surveys can be found in the literature which have studied and classified the proposed algorithms from different points of view [17, 18, 19]. In terms of image measurement and segmentation methods, background segmentation is divided into motion-based, appearance-based, shape-based, and depth-data-based [2]. Motion-based BGS methods (the scope of this paper) are classified, based on the way they model the background, into: the *traditional models*, which are basic, simple to implement, but have limitations; and the *recent models*, which are more sophisticated, capable of addressing more complicated challenges, and require improvements to become real-time [20].

The traditional models are briefly classified into: *(i) Basic Models*, which model the background using techniques such as average [21], median [22] or histogram analysis over time [23]; *(ii) Statistical Models*, which, for instance, take into account statistically the history of pixel brightness (Gaussian methods [24]), or model the background using supervised learning methods such as SVM [25] (Support Vector Models); *(iii) Cluster Models*, which cluster pixels in each frame using K-means [26], Codebooks [27], or basic sequential clustering approaches [28]; *(iv) Neural Networks*, where the background is modelled by a learnt neural network; *(v)*

Estimation Models, which estimate the background using filters such as Wiener [29], Kalman, and Chebychev [30].

In contrast, the recent models are classified into: *(i) Advanced Statistical background Models*, which for example use new distributions in Mixture Models, or fuse different distributions in Hybrid Models; *(ii) Fuzzy background Models*, which use fuzzy concepts to deal with imprecisions and uncertainties in BGS [19]; *(iii) Discriminative Subspace Learning Models*, where discriminative methods are used to provide supervised modelling of the background in contrast with the former re-constructive subspace learning models [31]; *(iv) Robust Subspace Models*, which separate the background and foreground using a robust subspace model based on a low-rank and sparse decomposition ; *(v) Sparse Models*, where sparse models such as structure sparsity models [32], dynamic group sparsity models [33], and dictionary models [34] are used; and *(vi) Transform Domain Models*, where the background and foreground are discriminated in a different domain using different transformation such as Fast Fourier, Discrete Cosine [35], Walsh [36], Wavelet [37], and Hadamard [38].

Beside the above, the size of the image element in background modelling, which could be pixel [3], block of pixels [4], or cluster of pixels [5], is another important issue and determines the precision and robustness to noise. The bigger the size of element, the higher the precision of the algorithm and the lower its robustness to noise. The type of feature, moreover, is another important factor in BGS, which is classified into: spectral features (color features); spatial features (edge features, texture features); and temporal features (motion features) [38]. Although the IPs are locations in the image, corresponding to pixels, and their descriptors have a combination of spectral and spatial properties, they have not been used for BGS yet. This motivates us to introduce a novel IP-based BGS algorithm which: (i) separates the foreground IPs from the background ones; (ii) creates a map of the background usable for further processing; and (iii) is robust to changes in illumination. The BGS approaches operate under either the static or moving camera scenarios [39]. The proposed algorithm works in static camera conditions. Nonetheless, it can be extended to the motion camera by updating the positions of IPs using the structure from motion, as we have aimed to do in future work.

The rest of this paper is outlined as follow: Section 2 presents the proposed algorithm. In Section 3 our algorithm is analysed qualitatively and quantitatively by conducting some experiments. Section 4 discusses the conclusions and introduces feature works.

2 IP-Based BGS Algorithm

In this section, we first discuss the motivation and the idea behind the proposed algorithm. Then, the technical description of the algorithm is presented.

2.1 Motivation

All the objects in the frames of a video are divided into two groups: the background objects (static objects in consecutive frames); and foreground objects (ones moving around the image in different frames). This fact inspires us to propose a novel IP-based BGS approach, which is used to discriminate the moving IPs from the stationary ones (IP-based BGS). On this basis, the extracted IPs from the background areas of image should remain stationary throughout the frames in contrast to the moving IPs of foreground. Although this seems to imply that all the frames should be processed to decide, if any IP belongs to the background IPs group, this is not the case in practice. As will be shown in Section 3, the background IPs need to stay motionless for more than a specific number of consecutive frames, not necessarily all the frames. More importantly, this enables the algorithm to update the background model following changes (e.g., when something is added to or removed from the scene).

To fulfil this idea, suppose an index $I(x, y)$ is assigned to any location (x, y) of the image plane. For any frame k, the (x, y) coordinate of extracted IPs are dealt with one by one to update the $I(x, y)$ index of corresponding locations. That is to say, for any IP i of frame k, the index $I(x_i, y_i)$, corresponding to the coordinate of IP, is increased by one. This implicitly means that an IP has located at that position of image one

more time. In this way, the index $I(x, y)$ records and updates the number of times the location (x, y) of image has met an IP. Since the background IPs are extracted from the stationary areas of image, they hit the specific locations of image repeatedly over the frames and so those locations have a higher value of I. In contrast, the foreground IPs are moving around the image throughout the frames. Therefore, their corresponding locations (x, y) have lower values of $I(x, y)$. Even when the foreground objects try to stand still, they have enough movement to cause changes in the location of their extracted IPs.

Although this idea is simple and seems to be efficient, it looks too severe and suffers from the problem of spurious background IPs which are actually foreground IPs. This happens when a location in the image is occupied repeatedly by foreground IPs of different parts of a foreground object, when it moves around the image and hits that location. To overcome this problem, the neighbouring locations in the image are dealt with together as a group using a **block-wise** processing approach, which divides the image into blocks of the same size. On this basis, the combination of locations in a block, hit by IPs inside that block, are considered together as an **Event**. In other words, an Event is a simultaneous observation of a group of IPs, specific pattern, inside a block.

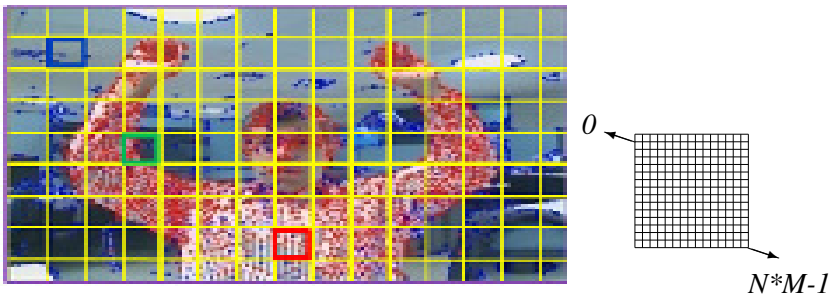
Accordingly, instead of counting the number of times a location is met by any IP, the number of times a combination of locations (an Event) is met inside a block is recorded. This strategy guaranties that only the specific persistent Events inside blocks,the Event with high level of repetition, are related to the static background IPs, because foreground IPs are unlikely to create exactly these patterns. Therefore, counting the number of repetitions of the Events instead of the individual IPs across the frames gives us a correct interpretation of the real background IPs. Fig. 2 shows repetition of Events for a block of image. Based on this definition, the blocks of the image are divided into three different types:

Background Blocks: These blocks are completely occupied by background IPs (areas of image with no moving objects). The numbers of repetitions of the Events in these blocks are significantly greater than in the other blocks. These Events are labelled as *dominant* Events. It is expected that the Events inside these blocks are seen in all of the frames. Nevertheless, the number of IPs and their position in each block can be changed because of changes in illumination, noise in the image, image resolution, and weakness of the IP detector by itself and so on.

Foreground Blocks: These blocks, areas where moving objects are present, are occupied by foreground IPs. The IPs of these blocks are unlikely to create the same Events throughout all the frames. This implies that foreground IPs increasingly create new Events for their associated blocks, which cannot appear dominantly. The block-wise processing strategy ensure that they less likely can create spurious background Events.

Background-Foreground Blocks: These are the blocks occupied by the outer boundary of the foreground object. Therefore, some IPs in these blocks belong to the background while the others belong to the foreground. Although the background IPs in any block tend to create dominant Events, it will not happen here because either some background IPs have been eliminated by the foreground or the foreground has added some new IPs. Consequently, new Event are added to the record of each block, which will never be dominant from the repetition point of view because they do not persist long enough.

According to the concept of motion detection, which is all about determining if a pixel (an IP here) at time t is associated to a moving object or not, the proposed algorithm detects the motion by classifying all the IPs into background IPs and foreground IPs. When the algorithm is run, it starts to model the background by finding the location of the background IPs. It learns quickly for blocks which are not covered by moving objects. For the covered blocks, it needs only a few frames to find the location of background IPs. Consequently, the algorithm models the background quickly over the frames. In addition, it is able to follow any changes in background (adding to or removing from the background) and adapts itself to these changes. Fig. 1 shows an image with its foreground IPs and background IPs, in red and blue colours respectively. The mentioned three different type of blocks can be seen in blue, red and green colours, respectively.



(a) Image, IPs, and blocks; blue: background block, red: foreground block, green: background-foreground block.

(b) Structure of a block.

Figure 1: The block and its different types.

2.2 Technical Description of Algorithm:

For the purpose of block-wise processing, the image is divided into blocks of $N \times M$ pixels (Fig. 1). In terms of hardware implementation and efficiency, it would be better to select N and M as a power of 2 because the memory blocks are counted based on powers of 2 too. Fig. 1 shows a frame of 320×240 pixels along side its blocks in yellow colour and the structure of a block. The extracted IPs of any frame are assigned to appropriate blocks based on their location (x, y) in the image plane. Consequently, the blocks related to the smoother areas of the image have no IPs, some blocks have a few IPs, and those associated with the highly textured areas of image have higher number of IPs.

To store the Events of blocks for consecutive frames, firstly we need to define a unique tag for any Event using its composing IPs. This tag is composed of some numbers, corresponding to the location of IP in the block, separated by commas e.g. $\{9, 21, 39\}$. To do this, the 2D coordinates of pixels in the block is mapped into a 1D coordinate by numbering pixels from 0 at the top left corner of the block, and then counting along each row from left to right to $N \times M\text{-}1$ at the bottom right corner (Fig. 1(b)).

Now, the Events as well as their number of repetitions should be stored. To preserve the speed and efficiency, the Binary Search Tree (BST) [40], which is a fast way of storing, sorting and searching, is used. On this basis, a Binary Tree (BT) is created for each block and its Events throughout the frames are stored in the tree as well as the numbers of their occurrences (Repetition Index (RI)). If any Event is happening for the first time, a new node with RI of 1 is created in the tree. Otherwise, the node corresponding to Event is found and its RI is increased by 1. So, the BTs of blocks summarize the Events and RIs of blocks.

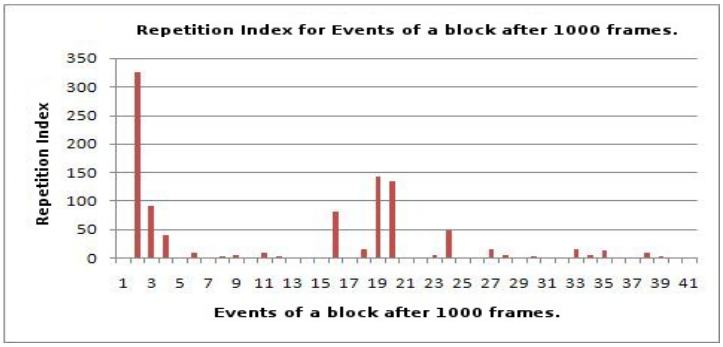


Figure 2: Repetition Index for Events of a block after 1000 frames.

As stated before, the background IPs appear as specific Events regularly at fixed blocks. Regardless the rare changes in the position of these IPs in blocks, due to changes in illumination and low resolution, they appear significantly more often than a threshold criterion. As can be seen in Table 1, the non-dominant Events created

by foreground IPs, have RI value less than the threshold. Therefore, the algorithm classifies any Event as dominant if its RI is equal to or greater than a threshold. The algorithm is summarized in Algorithm 1.

Algorithm 1 IP-Based BGS Algorithm

Input: IPs of frames $k = 1, \ldots, K$.
Output: background IPs & foreground IPs of each frame.
Initialization:
for frame $k = 1$ **do**
  Divide image frame into B blocks.
  Create a Binary Tree (BT) for each block.
  Create the lists of background events (BG-E-list) and background IPs (BG-IP-list).
end for
Background Modelling & Subtraction:
for every frame $k > 1$ **do**
  Background Modelling:
  for each IP i **do**
    Assign IP i to the corresponding block based on its (x, y) coordinate.
  end for
  for each block j **do**
    Create an Event E_m from its assigned IPs.
    Search Binary Tree BT_j for Event E_m.
    if $E_m \in BT_j$ **then**
      Increase Repetition Index of Event E_m by 1 ($RI_m = RI_m + 1$).
      if $RI_m > threshold$ **then**
        Add Event E_m to BG-E-list.
        Add IPs of Event E_m to BG-IP-list.
      end if
    else
      Add a new node in BT_j and set its RI to 0.
    end if
  end for
  Background Subtraction:
  for each IP i **do**
    if IP $i \in$ BG-IP-list **then**
      IP $i \Rightarrow$ background IP.
    else
      IP $i \Rightarrow$ foreground IP.
    end if
  end for
  $k = k + 1$
end for

Table 1 and Fig. 2 show the stored Events for block 102 (7^{th} row and column) in its BT after 1000 frames. As the algorithm assumes, only a few Events appear dominant. On the other hand, the non-dominant Events have been created by the foreground IPs when they have met this block. The dominant Events have been marked with the "$\surd$" sign in the "D" column of the table. IPs which create these dominant Events are classified as background IPs (bold numbers in "Event" columns of table). They are stored in a list of background IPs.

Fig. 3 shows the real image with its extracted IPs (FAST corner IPs in this case), foreground and background IPs in red and blue, respectively.

Table 1: The stored Events of block 102 if image for 1000 frames.

#	Event	RI	D	#	Event	RI	D	#	Event	RI	D
1	(363,110)	1		15	(280,110,353)	1		29	(303,171)	1	
2	**(303,14)**	325	√	16	**(285,110)**	82	√	30	(303,353,14)	2	
3	**(285,14)**	92	√	17	(285,11)	1		31	(303,353)	1	
4	**(285)**	40	√	18	**(285,12)**	16	√	32	(303,394)	1	
5	(125,333)	1		19	**(303,110)**	143	√	33	**(323,14)**	13	√
6	(110,353)	9		20	**(303)**	134	√	34	(323)	6	
7	(110)	1		21	(285,9)	1		35	**(305,14)**	14	√
8	(124,353,14)	2		22	(285,14,394)	1		36	(305,12)	1	
9	(144,353)	4		23	(303,11)	5		37	(323,110,353)	1	
10	(144,12,353)	1		24	**(303,12)**	49	√	38	**(323,110)**	10	√
11	**(144,353,14)**	10	√	25	(303,110,353)	1		39	(343,14)	3	
12	(280,110)	2		26	(303,110,394)	1		40	(323,353)	1	
13	(222,110,353)	1		27	**(305,110)**	16	√	41	(353,14)	1	
14	(280,171)	1		28	(305)	4					

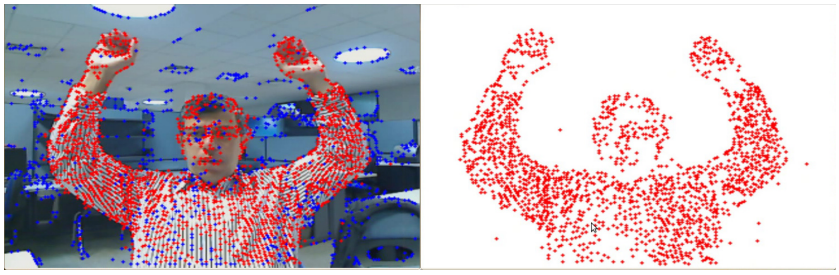


Figure 3: The real image with its IPs, red: foreground IPs, blue: background IPs.

2.3 Post-Processing

To improve the performance of the proposed IP-based BGS algorithm, two post-processing tasks are applied which try to decrease the False Negative (FN) and False Positive (FP) rates.

As can be seen in Table 1, the positions of some IPs have been changed because of changes in illumination, noise, and the quality of the images. For instance, the Events $(285, 14)$ and $(285, 12)$ have happened 92 and 16 times throughout 1000 frames, respectively. This means that IPs 12 and 14 should presumably be the same, their position having been changed across the frames. Since IP 14 has a greater repetition in its corresponding Event than IP 12, it could be said that IP 14 has been mislabelled as 12 in some frames because of noise. Although after 1000 frames the RI of Event $(285, 12)$ has satisfied the threshold criterion and so IP 12 has been added to the BG-IP-list, it has been misclassified as a foreground IP for a while before satisfying the threshold condition. To overcome this type of error, which increases the False Positive (FP) rate, a simple comparison between such non-dominant and dominant Events of the block ($(285, 12)$ and $(285, 14)$ for example) in terms of Euclidean distance is performed. This saves the IPs of these non-dominant Events from being misidentified as foreground IPs (another example: Events $(303, 12)$ and $(303, 14)$).

1	1	1
1	0	1
1	1	1

Figure 4: The 2D kernel of 3×3 filter.

Another post-processing task is to decide about the foreground blocks with no foreground blocks in their

neighbourhood. To do this, an image with the same size as the number of blocks (foreground block binary image) is constituted. The value of each pixel in this image is 1 if the corresponding block has at least one foreground IP. Otherwise it is 0. Then a $2D$ 3×3 filter with the kernel shown in Fig. 4 is applied to this image to calculate how many foreground blocks each block has around itself. By applying this filter, the foreground blocks with no foreground block in its neighbourhood are identified and considered as spurious and then the status of their IPs are changed to background. Also, background blocks with no foreground IPs which have more than a threshold foreground blocks in their neighbourhood are reclassified as foreground blocks. Fig. 5 shows the result of the post-processing procedures.

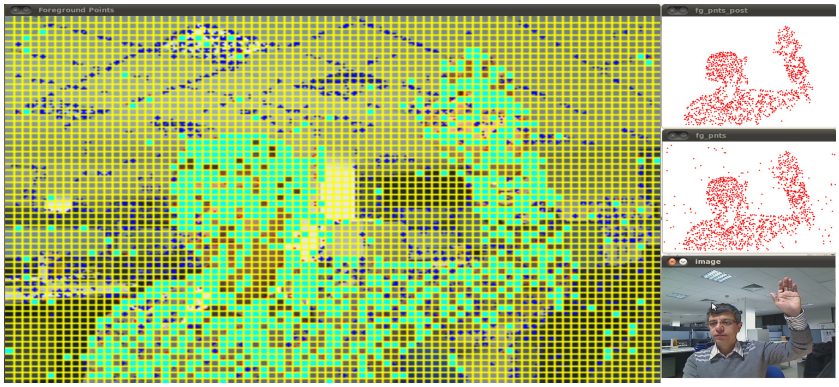


Figure 5: Left: foreground blocks, Right from down to top: the real image, foreground IPs before, and after post-processing.

3 Experimental Results

To evaluate the performance of the proposed IP-based BGS algorithm and also to compare it with the other BGS approaches, we present the results of several experiments in this section.

3.1 Evaluation Factors

The False Negatives (FN) and False Positives (FP) are the evaluation factors considered in these experiments. On this basis, the Precision and Recall are defined as [41]:

$$Precision = \frac{TP}{TP + FP} \quad , \quad Recall = \frac{TP}{TP + FN} \tag{1}$$

Specificity and Accuracy, as the other two measures used to measure the performance of binary classification task, are used as complementary evaluation factors in some experiments. They are defined as following [41]

$$Specificity = \frac{TN}{TN + FP} \quad , \quad Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

The F-measure, which is the harmonic mean of precision and recall values, is another factor for taking into account the precision and recall factors concurrently. The F_β measure for non-negative real values of β is:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} \tag{3}$$

β provides the user the flexibility to give the recall β times importance as precision. The special case of this formula for β equal to 1 is known as the F_1 measure weights the precision and recall evenly.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4}$$

F_2 and $F_{0.5}$ are two other commonly used F measures, which put more emphasis on recall and precision respectively. We use this measure to determine the best value for the threshold, in this paper.

3.2 Experiment 1: Effect of Block Sizes on the Precision and Recall

Through this experiment, the effect of block size on the efficiency of the algorithm is evaluated according to the above mentioned evaluation factors (Fig. 6, 7, 8). As Fig. 6 shows, the blocks of sizes of 4×4 and 8×8 deliver higher performances than the other sizes (particularly than the larger sizes of 16×16 and 32×32). It is because:

- The smaller block sizes contain lower numbers of IPs. This decreases the risk of losing the background Events in cases where some background IPs of these Events disappear due to: covering by foreground objects; changing the position due to noise; and so on. This situation is more drastic around the outer contour of foreground object (in Background-Foreground blocks).
- The foreground objects involve less blocks, which decreases the FN & FP errors, even more.
- The smaller is the block size, the lower number of Events and faster BT it has. This affects the run time speed of the algorithm as is verified in Fig. 8.

On the other hand, the smallest sizes of 1×1 and 2×2 cannot deliver the same performance as 4×4 and 8×8 because they implicitly ignore the block-wise processing strategy and behave as when the IPs are dealt with individually. This justifies the superiority of block-wise processing over individual processing of IPs.

According to the concept of evaluation factors, the higher value of Precision is equal to the higher number of retrieved foreground IPs out of the whole retrieved IPs. On the other hand, Recall shows the percentage of correctly retrieved foreground IPs out of the ground truth foreground IPs. Obviously, the block of 8×8 offers higher Precision and Recall, particularly better than the 1×1 and 2×2 sizes (Fig. 6). In addition, Specificity determines the percentage of the ground truth background IPs has been truly rejected (the algorithm tries to identify the foreground IPs and reject the background ones). As can be seen, the higher block sizes deliver higher Specificity than (or at least equal to) the lower sizes when the algorithm aims to identify the background IPs and reject the foreground ones.

This experiment has been conducted on images with 512×256 pixels. Although the block size is dependent on the image resolution, the result of this experiment can be generalized to any image resolution. To do this, simply the block size can be scaled in proportion to the image resolution.

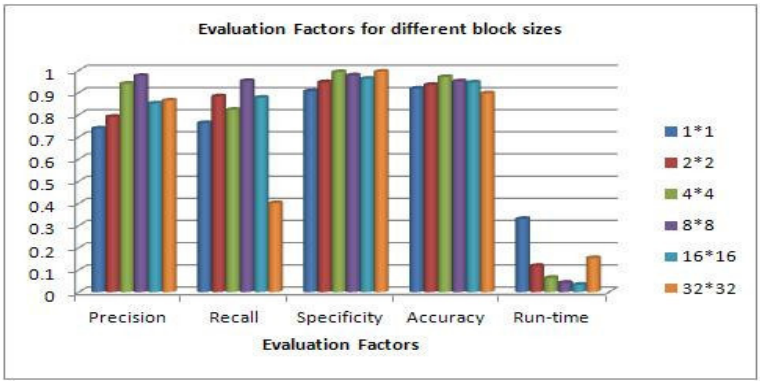


Figure 6: Evaluation Factors versus block sizes for resolution 512×256.

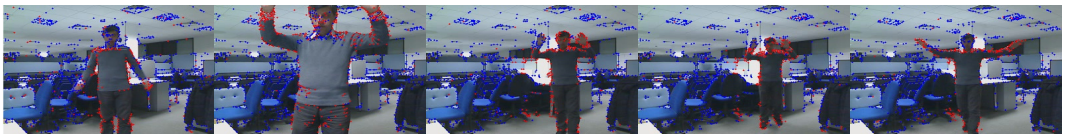
Fig. 7 compares visually the effect of block size on the result of the algorithm on 5 random frames of a video with 450 frames. It is obvious from this figure that the algorithm has a faster and more accurate background Modelling and Update for the block sizes of 4×4 and 8×8 than the other sizes. As can be seen in figures 7(a), 7(b), 7(e), and 7(f), the FP (red IPs in background areas) and FN (blue IPs in foreground areas) rates are higher than 7(c) and 7(d). It means that the algorithm has incorrectly classified background IPs as foreground IPs in the former case and conversely the foreground IPs as background IPs in the later ones. Again, it confirms that the block sizes of the 4×4 and 8×8 outperform than the other sizes.

(a) Block size: 1×1

(b) Block size: 2×2

(c) Block size: 4×4

(d) Block size: 8×8



(e) Block size: 16×16

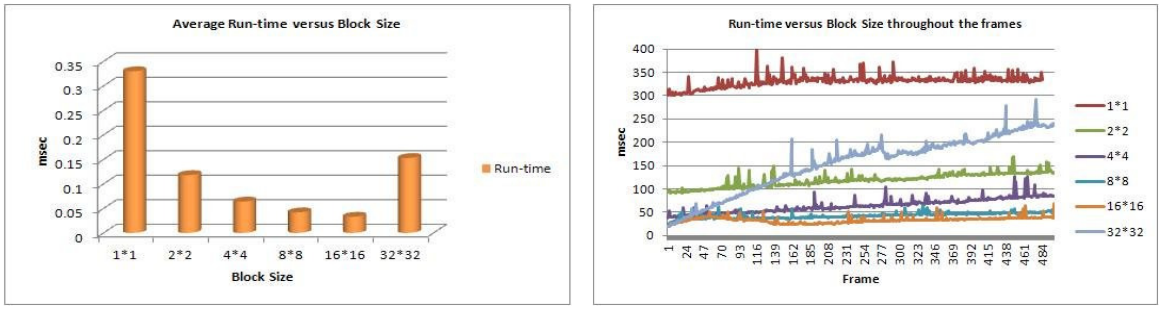
(f) Block size: 32×32

Figure 7: 5 frames vs different block sizes of 1×1, 2×2, 4×4, 8×8, 16×16, and 32×32.

Fig. 8 compares the run-time speed of the algorithm against the block size. A can be seen (Fig. 8(b)), the sizes of 4×4, 8×8, and 16×16 are faster. Among these, although the 16×16 has the fastest execution time, it offers no efficiency in terms of evaluation factors. Consequently, the size 4×4 and then 8×8 look to be the best sizes for the blocks in the proposed algorithm.

3.3 Experiment 2: Robustness Under Low-Light conditions

In this experiment, the proposed algorithm is examined under low-light conditions. The pixel-based BGS algorithms produce the white and black pixels for the foreground and background areas of image, respectively.

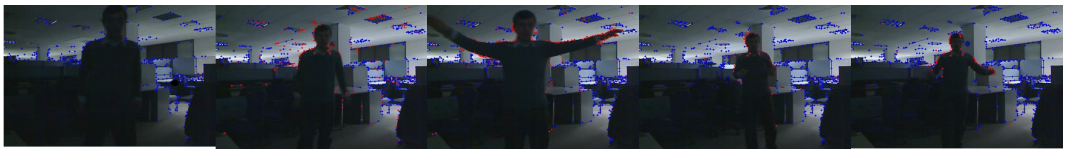


(a) Average run-time versus block size.

(b) Run-time versus block size.

Figure 8: Run-time versus block size

In low-light conditions, the foreground white pixels look like separated islands which cannot be interpreted to localize the foreground areas (9(b)). Whereas, the proposed approach effectively subtracts the background IPs and presents the foreground IPs, which can be used meaningfully. To get some intuition for why this point works, here is an example. In applications like IP-based motion tracking, the IPs of the object should be tracked across the frames. In this sense, if the IPs of the moving object (foreground IPs) can be discriminated first from the background IPs, similar to what the proposed algorithm does, it makes the tracking process easier. Not only the foreground IPs, but also the background IPs can significantly be used for IP matching and tracking purposes. In contrast, the traditional pixel-based BGS algorithms just delete the background pixel of the image On. the other hand, IPs are more robust to the low-light conditions. Consequently, the superposition of these properties of IPs delivers more robustness to the low-light conditions rather than pixels. Fig. 9 shows this comparison over 5 random frames of a video with 450 frames.



(a) IP-based BGS algorithm with block size of 8×8.

(b) Pixel-based BGS algorithm.

Figure 9: Comparison between IP-based and pixel-based BGS methods under low-light conditions over 5 random frames of a video with 450 frames.

3.4 Experiment 3: Comparison of the IP-Based BGS with Different BGS Algorithms

For the purpose of performance evaluation, in this section we compare the efficiency of the proposed algorithm against some of the well-known and state-of-the-art BGS algorithms on the Wallflower dataset [29] based on the total number of error as the sum of FN and FP, in a similar manner to state-of-the-art BGS papers. This dataset has widely been used in this context and consists of seven different scenarios with a diverse range of difficulty that might happen in practice. Table 2 summarizes these scenarios [29].

Although our algorithm is IP-based and looks a little different from the pixel-based BGS algorithms, it similarly subtracts the foreground IPs from the background ones. Nevertheless, the number of IPs in the image

Table 2: The Wallflower dataset scenarios.

Scenario	Description
Moved Object	A person enters into a room, makes a phone call, and leaves.
Time of Day	A person enters and sits down in a room where the light is changing gradually from dark.
Light Switch	A person enters a lighted room and turns off the lights. Afterwards, person walks in the room, turns on the light, and moves the chair.
Waving Trees	A person walking in front of a swaying tree.
Camouflage	A person walking in front of a monitor with rolling interference bars with similar color to the person's clothing.
Boostrapping	A busy cafe containing several people.
Foreground Aperture	A person with a uniformly coloured shirt waking up and moving slowly.

is much less than the number of pixels. Therefore, the total number of erroneous IPs would not be comparable with the total number of erroneous pixels. To compensate for this, we define the Error Ratio as the ratio of the total number of erroneous IPs (pixels) to the total number of IPs (pixels). Tables 3 and 4 summarize the results for this performance comparison, numerically and visually respectively.

$$ErrorRatio = \frac{Total\ erroneous\ IPs\ (pixels)}{Total\ IPs\ (pixels)} \tag{5}$$

As Table 3 shows, our proposed approach delivers a lower Error Rate, and so better performance, in comparison with most of the existing algorithms. Although some of the algorithms in the MoG family work better than ours, those cannot deliver the same advantages in terms of the scope of this paper. Due to the intrinsic properties of the IPs and also the block processing structure of our proposed approach, it delivers an interpretation of the scene in the image plane, which can be used for further processing such as occlusion-detection usable in tracking applications. Some of the information, which provides this interpretation are: classifying the blocks into foreground, background, and Background-Foreground blocks; hiding and disclosing the background IPs with foreground IPs; geometrical relationship between the IPs in the same and the adjacent blocks; identifying the background IPs in low-light conditions.

In contrast, the pixel-based BGS algorithms only represent the foreground pixels in form of white pixels without providing any information about the relationships among these pixels and the background pixels. Besides, they are not suitable for IP-based Computer Vision algorithms, unless the output foreground mask of a BGS stage is applied to the IPs to subtract the foreground IPs from the background ones.

Table 4 compares the test frame of 7 different scenarios in the Wallflower Dataset for several BGS algorithms alongside the ground truth. The algorithms perform non-uniformly for different scenarios, i.e. they work well for some scenarios while the result for other scenarios is not so good. For example, KDE shows the best and worst results for "Waving Trees" and "Time of Day" scenarios respectively, while SL-IRT has the best and worst results for "Camouflage" and "Waving Trees". In this regard, our approach works relatively well for all of the scenarios, except for the "Moved Object". Even more, the number of FN errors is less than the number of FP errors. This is an important point because it demonstrates that the algorithm can find the foreground IPs over the moving object and it has a low level of foreground misclassification in foreground areas of the image. The FP errors, i.e. the misclassified foreground IPs in background areas, can be compensated for by using some additional post-processing procedures.

3.5 Experiment 4: Effect of Threshold on the Results

As described in Section 2.2, a threshold value is used to discriminate the background and foreground Events. To evaluate the effect of threshold on the results and find the best value for it, the performance of algorithm for a range of thresholds from 1 to 100 is evaluated using the F-measure factor, described in Section 3.1. Fig. 10

Table 3: Comparison on the Wallflower dataset for different BGS algorithms.

Algorithm	Error	Moved Object	Time Day	Light Switch	Waving Tree	Camouflage	Bootstrap	FG Aper	Total Error	Error Ratio
SG [42]	FN	0	945	1857	3110	4101	2215	3464	35133	0.2614
	FP	0	535	15123	357	2040	92	1290		
MOG [3]	FN	0	1008	1633	1323	398	1874	2442	27053	0.2012
	FP	0	20	14169	341	3098	217	530		
MoG-PSO [43]	FN	0	807	1716	43	2386	1551	2392	13916	0.1035
	FP	0	6	722	1689	1463	519	572		
MoG-IHLS [44]	FN	0	379	1146	31	188	1647	2327	9739	0.0724
	FP	0	99	2298	270	467	333	554		
Improved MOG [45]	FN	0	597	1481	44	106	1176	1274	7081	0.0526
	FP	0	358	669	288	413	134	541		
MoG-MRF [46]	FN	0	47	204	15	16	1060	34	3808	0.0283
	FP	0	402	546	3011	467	102	604		
S-TAPP-MOG [47]	FN	-	-	-	153	643	1414	1912	7844	0.1021
	FP	-	-	-	1152	1382	811	377		
ASTNA [48]	FN	-	-	-	253	823	2349	1900	7031	0.0915
	FP	-	-	-	100	1173	73	360		
KDE [49]	FN	0	1298	760	170	238	1755	2413	26450	0.1968
	FP	0	125	14153	589	3392	933	624		
SL-PCA [50]	FN	0	579	963	1027	350	304	2441	17677	0.1315
	FP	1065	16	632	2057	1548	6129	537		
SL-ICA [51]	FN	0	1199	1557	33720	3054	2560	2721	15308	0.1138
	FP	0	0	210	148	43	16	428		
SL-INMF [52]	FN	0	724	1593	3317	6624	1401	3412	19098	0.1420
	FP	0	481	303	652	234	190	165		
SL-IRT [53]	FN	0	1282	2822	4527	1491	1734	2438	17053	0.1268
	FP	0	159	389	7	114	2080	12		
Our Method	FN	0	20	25	208	18	172	17	649	0.0938
	FP	0	15	42	57	39	2	34		

shows the F_1, $F_{0.5}$, F_2, as well as the total number of errors for the Wallflower dataset. As can be seen, the threshold values between 20 to 25 is where the lowest total number of error and highest F_1, $F_{0.5}$, F_2 are gained.

On one hand, the threshold values lower than the optimal value increase the FN error, while keep FP roughly constant. On the other hand, the higher threshold values, higher than the optimal value, cause a greater FP and keep the FN approximately constant. Both cases yield an increase in the total number of error. So, to keep the total error at the lowest possible value, a balance between these two source of errors is established by selecting a value between 20 to 25 for threshold.

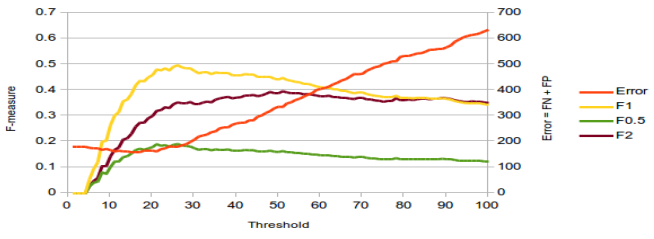
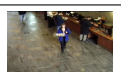
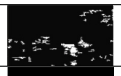
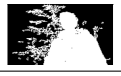
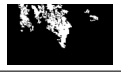


Figure 10: F-measure and total number of error versus threshold.

Table 4: Results on the Wallflower dataset for different BGS algorithms.

Methods	Moved Object	Time of Day	Light Switch	Waving Trees	Camo-uflage	Boot-strap	FG Aper
Test Image							
Ground Truth							
SG [42]							
MOG [3]							
MOG-PSO [43]							
MOG-IHLS [44]							
Improved MOG [45]							
MOG-MRF [46]							
S-TAP-PMOG [47]	-	-	-				
ASTNA [48]	-	-	-				
KDE [49]							
Sl-PCA [50]							
SL-ICA [51]							
SL-INMF [52]							
SL-IRT [53]							
Our Method							

4 Conclusions

In this paper, we have proposed a completely new IP-based BGS algorithm which can be used in any IP-based CV application. The key characteristic of our approach is its robustness to illumination changes. According to a block-wise processing strategy, the algorithm divides images into blocks of the same size. IPs inside blocks are dealt with together as Events (Event)s across the frames, by storing the Events as well as their number of occurrences (RI). Meanwhile, RI is used to classify Events into the background and foreground Events. If any Event has RI more than a threshold, it is classified as background Event; otherwise it is classified as foreground Event. The Event classification is used to label the IPs of frames into the foreground and background IPs. In comparison with the traditional BGS algorithms, the proposed IP-based algorithm has the following key characteristics:

- Is real-time. As the first stage of any CV system, it is fast enough and adds no latency to the system.

- Is adaptive to changes in background. It needs only a few frames to find out the location of the background IPs as well as any changes in background (adding or removing any object from the background).
- Works well in low-light indoor environments (as a need for the gaming tools used in home locations).
- Not only subtracts the foreground and background objects, but also creates an interpretation of scene, which can be used for any further processing in CV systems.
- Works relatively well when the foreground objects do not move

We applied our algorithm to BGS under different circumstances. Also, experiments on a public dataset, Wallflower, show that our approach outperforms most of state-of-the-art methods, while it delivers some valuable advantages against those which perform better than our proposed algorithm. The proposed algorithm works in static camera conditions. For the future work, we plan to make the current algorithm to work in moving camera scenarios, by updating the positions of IPs using the structure from motion.

Acknowledgements

The proposed work was supported by the Irish Research Council (IRC) under their Enterprise Partnership Scheme in partnership with Movidius plc.

References

[1] Herrero, S., Bescós, J.: Background subtraction techniques: systematic evaluation and comparative analysis. In: Advanced Concepts for Intelligent Vision Systems, Springer (2009) 33–42

[2] Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Computer vision and image understanding **104**(2) (2006) 90–126

[3] Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on. Volume 2., IEEE (1999)

[4] Fang, X., Xiong, W., Hu, B., Wang, L.: A moving object detection algorithm based on color information. In: Journal of Physics: Conference Series. Volume 48., IOP Publishing (2006) 384

[5] Bhaskar, H., Mihaylova, L., Achim, A.: Video foreground detection based on symmetric alpha-stable mixture models. Circuits and Systems for Video Technology, IEEE Transactions on **20**(8) (2010) 1133–1138

[6] Varcheie, P.D.Z., Sills-Lavoie, M., Bilodeau, G.A.: An efficient region-based background subtraction technique. In: Computer and Robot Vision, 2008. CRV'08. Canadian Conference on, IEEE (2008) 71–78

[7] Aanæs, H., Dahl, A.L., Pedersen, K.S.: Interesting interest points. International Journal of Computer Vision **97**(1) (2012) 18–35

[8] Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. Acm Computing Surveys (CSUR) **38**(4) (2006) 13

[9] Leutenegger, S., Chli, M., Siegwart, R.Y.: Brisk: Binary robust invariant scalable keypoints. In: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE (2011) 2548–2555

[10] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2) (2004) 91–110

[11] Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (SURF). Computer vision and image understanding **110**(3) (2008) 346–359

[12] Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. Pattern Analysis and Machine Intelligence, IEEE Transactions on **27**(10) (2005) 1615–1630

[13] Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE (2011) 2564–2571

[14] Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.S.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proceedings of the IEEE **90**(7) (2002) 1151–1163

[15] Musa, Z., Watada, J.: Video tracking system: A survey. An international journal of research and surveys (ICIC express letters) **2**(1) (2008) 65–72

[16] Varona, J., Gonzàlez, J., Rius, I., Villanueva, J.J.: Importance of detection for video surveillance applications. Optical Engineering **47**(8) (2008) 087201–087201

[17] Cristani, M., Farenzena, M., Bloisi, D., Murino, V.: Background subtraction for automated multisensor surveillance: a comprehensive review. EURASIP Journal on Advances in signal Processing **2010** (2010) 43

[18] Bouwmans, T.: Recent advanced statistical background modeling for foreground detection-a systematic survey. Recent Patents on Computer Science **4**(3) (2011) 147–176

[19] Bouwmans, T.: Background subtraction for visual surveillance: A fuzzy approach. Handbook on Soft Computing for Video Surveillance (2012) 103–134

[20] Bouwmans, T., Porikli, F., Hferlin, B., Vacavant, A.: Traditional and Recent Approaches in Background Modeling for Foreground Detection: An Overview. CRC Press (2014)

[21] Lee, B., Hedley, M.: Background estimation for video surveillance. In: Image and vision computing New Zealand. (2002) 315–320

[22] McFarlane, N.J., Schofield, C.P.: Segmentation and tracking of piglets in images. Machine Vision and Applications **8**(3) (1995) 187–193

[23] Zheng, J., Wang, Y., Nihan, N.L., Hallenbeck, M.E.: Extracting roadway background image: Mode-based approach. Transportation Research Record: Journal of the Transportation Research Board **1944**(1) (2006) 82–88

[24] Bouwmans, T., El Baf, F., Vachon, B., et al.: Background modeling using mixture of gaussians for foreground detection-a survey. Recent Patents on Computer Science **1**(3) (2008) 219–237

[25] Lin, H.H., Liu, T.L., Chuang, J.H.: A probabilistic svm approach for background scene initialization. In: Image Processing. 2002. Proceedings. 2002 International Conference on. Volume 3., IEEE (2002) 893–896

[26] Butler, D.E., Bove, V.M., Sridharan, S.: Real-time adaptive foreground/background segmentation. EURASIP Journal on Advances in Signal Processing **2005**(14) (1900) 2292–2304

[27] Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Background modeling and subtraction by codebook construction. In: Image Processing, 2004. ICIP'04. 2004 International Conference on. Volume 5., IEEE (2004) 3061–3064

[28] Xiao, M., Han, C., Kang, X.: A background reconstruction for dynamic scenes. In: Information Fusion, 2006 9th International Conference on, IEEE (2006) 1–7

[29] Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: Principles and practice of background maintenance. In: Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on. Volume 1., IEEE (1999) 255–261

[30] Chang, R., Gandhi, T., Trivedi, M.M.: Vision modules for a multi-sensory bridge monitoring approach. In: Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on, IEEE (2004) 971–976

[31] Skocaj, D., Uray, M., Leonardis, A., Bischof, H.: Why to combine reconstructive and discriminative information for incremental subspace learning. In: Proc. Computer Vision Winter Workshop. (2006)

[32] Huang, J., Zhang, T., Metaxas, D.: Learning with structured sparsity. The Journal of Machine Learning Research **12** (2011) 3371–3412

[33] Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) (1996) 267–288

[34] Tropp, J.A., Gilbert, A.C.: Signal recovery from random measurements via orthogonal matching pursuit. Information Theory, IEEE Transactions on **53**(12) (2007) 4655–4666

[35] Porikli, F., Wren, C.: Change detection by frequency decomposition: Wave-back. In: Proc. of Workshop on Image Analysis for Multimedia Interactive Services. (2005)

[36] Tezuka, H., Nishitani, T.: A precise and stable foreground segmentation using fine-to-coarse approach in transform domain. In: Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, IEEE (2008) 2732–2735

[37] Gao, T., Liu, Z.g., Gao, W.c., Zhang, J.: A robust technique for background subtraction in traffic video. In: Advances in Neuro-Information Processing. Springer (2009) 736–744

[38] Baltieri, D., Vezzani, R., Cucchiara, R.: Fast background initialization with recursive hadamard transform. In: Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on, IEEE (2010) 165–171

[39] Elqursh, A., Elgammal, A.: Online moving camera background subtraction. In: Computer Vision–ECCV 2012. Springer (2012) 228–241

[40] Gilberg, R.F., Forouzan, B.A.: Data structures: a pseudocode approach with c++, brooks (2001)

[41] Olson, D.L., Delen, D.: Advanced data mining techniques [electronic resource]. Springer (2008)

[42] Pande, A., Verma, A., Mittal, A.: Network aware optimal resource allocation for e-learning videos. In: 6th International Conference on Mobile Learning. (2007)

[43] White, B., Shah, M.: Automatically tuning background subtraction parameters using particle swarm optimization. In: Multimedia and Expo, 2007 IEEE International Conference on, IEEE (2007) 1826–1829

[44] Setiawan, N.A., Seok-Ju, H., Jang-Woon, K., Chil-Woo, L.: Gaussian mixture model in improved hls color space for human silhouette extraction. In: Advances in Artificial Reality and Tele-Existence. Springer (2006) 732–741

[45] Wang, H., Suter, D.: A re-evaluation of mixture of gaussian background modeling [video signal processing applications]. In: Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on. Volume 2., IEEE (2005) ii–1017

[46] Schindler, K., Wang, H.: Smooth foreground-background segmentation for video processing. In: Computer Vision–ACCV 2006. Springer (2006) 581–590

[47] Cristani, M., Murino, V.: A spatial sampling mechanism for effective background subtraction. In: VISAPP (2). (2007) 403–412

[48] Cristani, M., Murino, V.: Background subtraction with adaptive spatio-temporal neighborhood analysis. In: VISAPP (2). (2008) 484–489

[49] Elgammal, A., Harwood, D., Davis, L.: Non-parametric model for background subtraction. In: Computer VisionECCV 2000. Springer (2000) 751–767

[50] Oliver, N.M., Rosario, B., Pentland, A.P.: A bayesian computer vision system for modeling human interactions. Pattern Analysis and Machine Intelligence, IEEE Transactions on **22**(8) (2000) 831–843

[51] Tsai, D.M., Lai, S.C.: Independent component analysis-based background subtraction for indoor surveillance. Image Processing, IEEE Transactions on **18**(1) (2009) 158–167

[52] Bucak, S.S., Günsel, B., Gursoy, O.: Incremental non-negative matrix factorization for dynamic background modelling. In: PRIS. (2007) 107–116

[53] Li, X., Hu, W., Zhang, Z., Zhang, X.: Robust foreground segmentation based on two effective background models. In: Proceedings of the 1st ACM international conference on Multimedia information retrieval, ACM (2008) 223–228