symmetry

MDPI

Article

WisenetMD: Motion Detection Using Dynamic Background Region Analysis

Sang-ha Lee [1], Gyu-cheol Lee [1], Jisang Yoo [1] and Soonchul Kwon [2,*]

[1] Department of Electrical Engineering, Kwangwoon University, 20 Kwangwoon-ro, Nowon-gu, Seoul 01897, Korea; kcv456@kw.ac.kr (S.-h.L.); gyucheol0116@gmail.com (G.-c.L.); jsyoo@kw.ac.kr (J.Y.)

[2] Department of Smart Convergence, Kwangwoon University, 20 Kwangwoon-ro, Nowon-gu, Seoul 01897, Korea

* Correspondence: ksc0226@kw.ac.kr; Tel.: +82-2-940-8637

Received: 4 March 2019; Accepted: 26 April 2019; Published: 3 May 2019



Abstract: In this paper, we propose a method for calculating the dynamic background region in a video and removing false positives in order to overcome the problems of false positives that occur due to the dynamic background and frame drop at slow speeds. Therefore, we need an efficient algorithm with a robust performance value including processing speed. The foreground is separated from the background by comparing the similarities between false positives and the foreground. In order to improve the processing speed, the median filter was optimized for the binary image. The proposed method was based on a CDnet 2012/2014 dataset and we achieved precision of 76.68%, FPR of 0.90%, FNR of 18.02%, and an F-measure of 75.35%. The average ranking across categories is 14.36, which is superior to the background subtraction method. The proposed method was operated at 45 fps (CPU), 150 fps (GPU) at 320 × 240 resolution. Therefore, we expect that the proposed method can be applied to current commercialized CCTV without any hardware upgrades.

Keywords: motion detection; background subtraction; foreground segmentation; surveillance; video signal processing

1. Introduction

Motion detection algorithms that can be applied to surveillance cameras, such as Closed-Circuit Television (CCTV), have been studied extensively and should have robust performance results; processing speed is also an important factor. If the processing speed is slow, the performance degradation can occur due to frame drop. According to CDnet 2014 [1], in low frame rate video experiments, the performance is affected. In the case of a video stream, if the processing speed is slow, the frame is delayed to receive a non-continuous image. As a result, the performance of the motion detection algorithm that uses the information of the previous frame is degraded. Another important issue in motion detection algorithms is the elimination of false positives in dynamic backgrounds, such as leaves, rivers, and so on.

The purpose of this paper is to propose an algorithm with robust performance and fast processing speed. We propose a false positives elimination algorithm and a speed optimization method. The proposed method is compared with conventional algorithms based on the CDnet 2012/2014 dataset [1,2]. The proposed method uses RGB color values to remove false positives in the dynamic background region. We define dynamic background samples to collect false positives and generate a dynamic background region by analyzing the video scene for this purpose. When the foreground is detected in the dynamic background region, it is removed by re-checking for similarities with false positives and the foreground. In addition, a median filter optimization for binary images to improve processing speed is also presented. Generally, the median filter requires a significant

amount of computation because it has to be aligned for the values in the mask for every pixel. In the proposed method, using the integral image of the binary image, a median filter with efficient processing regardless of mask size is applied. In this paper, we describe the related works and the proposed method in Sections 2 and 3, respectively. In Sections 4 and 5, comparative performance values and conclusions will be presented.

2. Related Work

Various techniques have been proposed in relation to motion detection algorithms. Recently, techniques based on deep learning have been proposed and achieved good performance [3–6]. In FgSegNet [3], features are extracted using triplet CNN encoder. Feature pooling module is used to extract features at multiple scales for the features extracted from the triplet CNN encoder. Learning Multi-scale Features for Foreground Segmentation (FgSegNet_v2) [4] uses a CNN-based approach of flexible encoder-decoder type. In addition, it improved the structure of the Feature Pooling Module proposed in FgSegNet [3]. The structure of Feature Pooling Module was improved in FgSegNet_v2 [4] as well; this can extract features at multiple scales with wider receptive fields than the Feature Pooling Module, which makes it possible to find everything from small to large. However, due to hardware dependency and dataset overfitting, motion algorithms are still unable to run on an embedded board equipped with a CCTV system.

Motion detection algorithms generally approach background models by modeling background subtraction. There are many ways to model background information. Among these are parametric probability density functions (such as the Gaussian mixture background model [7]), the RPCA method [8–10], and the non-parametric approach. SharedModel [11] is based on Gaussian mixture modeling, which takes the relationship between pixels and background samples into account. Recently, Spatiotemporal GMM for Background Subtraction with Superpixel Hierarchy (STSHBM) [12] has been used for expanding into the temporal domain as well as the spatial domain using hierarchical super-pixel segmentation, spanning tree, and optical flow. Total Variation Regularized Robust Principal Component Analysis (RPCA) for Irregularly Moving Object Detection Under Dynamic Background (TV-RPCA) [13] analyzes in the temporal domain and the spatial domain using RPCA. The method of extending to the spatiotemporal domain shows robust performance but requires a considerable amount of computation. In addition, there are many ways [14–16] to extend to the spatiotemporal domain.

The non-parametric approaches typically use color components and local textures as background information. Multimode Background Subtraction (MBS) [17] classifies backgrounds based on pixel sets using RGB and YCbCr channels. Various local pattern methods have been proposed that are similar to Local Binary Patterns (LBP) [18,19]. Among them, Bilodeau has revealed that LBSP performs well when it is used with a background subtraction algorithm [20]. Self-Balanced SENsitivity SEgmenter (SuBSENSE) [21] uses a background modeling method using color/LBSP information based on ViBe+ [22] and PBAS [23]. When using a background subtraction algorithm, it is important to prevent false positives for dynamic backgrounds such as leaves and rivers. Various methods have been proposed to remove false positives in previous studies. Spatiotemporal Low-Rank Modeling for Complex Scene Background Initialization (SRPCA) [24] generates a motion-compensated binary matrix using optical flow to remove redundant data. Then, a set of dynamic frames is generated from the input image to prevent the detection of false positives. Universal Background Subtraction Using Word Consensus Models (PAWCS) [25] uses persistence, which represents the value of the background, to keep the true information in the dynamic background, and to remove the false information to prevent the detection of false positives in the dynamic background. Similarly, a Weight-Sample-Based Method for Background Subtraction (WeSamBE) [26] uses weights on collected background samples and updates the background according to these weights. A sliding window and self-regulated learning-based background updating method for change detection in videos (SWCD) [27] uses a sliding window approach combined with dynamic control of update parameters for updating background frames, which we call sliding window-based change detection.

Also, other approaches to motion detection in surveillance cameras have been proposed. To perform motion detection on a moving camera, Wu et al. [28] proposed a method to reconstruct BG motion through a fast inpainting algorithm for pixels in a coarse foreground area. Sparse and low-rank representation with contextual regularization (SLRC) [29] proposed the use of dedicated backgrounds for multiple scenarios using dictionary learning-based sparse representation in order to perform motion detection for multiple cameras.

3. Proposed Method

The proposed method is a motion detection algorithm based on background subtraction. A flowchart of the method is shown in Figure 1. The method consists of five modules: background samples, Background (BG)/Foreground (FG) classification, False Positive (FP) re-check, feedback process, and post-processing. The background sample module is a module that saves and collects background samples used to classify between the foreground and background. The BG/FG classification module calculates the distance between the current input image and the background samples, to determine whether it is foreground or background. This is described in more detail in Sections 3.1 and 3.2. The FP re-check module performs a re-check to prevent the false positives that can occur when background items such as trees or rivers are moving. In this paper, we define a candidate region in which a dynamic background can exist, and use dynamic background samples to collect false positives. When the foreground is detected in the dynamic background region, it is removed by re-checking for false positives in the dynamic background samples. The feedback process module updates the parameters used in the background samples and the BG/FG classification module. The post-processing module is a module for processing the filter operation and morphology operation, to improve the quality of the resulting image. This is described in more detail in Sections 3.3–3.5.

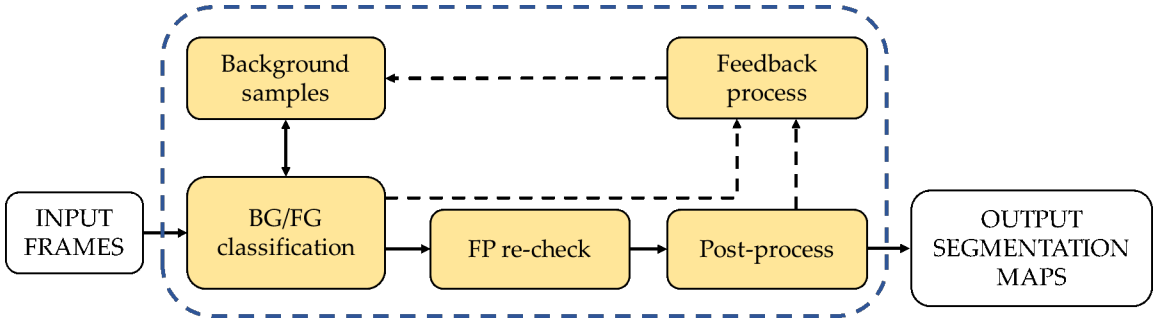


Figure 1. Flowchart of the proposed method.

3.1. Background Samples

Since the proposed algorithm operates based on background subtraction, it is essential to collect background samples. In this paper, we used the sample consensus method used in SuBSENSE [21]. The background samples have the same resolution as the input image. The number of samples is fixed at 50.

$$B_n^t(x) \in \{B_1^t(x), B_2^t(x)..., B_N^t(x)\} \quad (1)$$

where t is the index of the frame, n is the index of the background samples, and x is the index of the pixel. The components of background samples include color and texture information. Color information uses RGB pixel values, while texture information uses LBSP [19] values. The LBSP is a texture feature similar to the LBP. Research has shown that using the LBSP [19] feature in the background subtraction algorithm achieves good performance [20]. The mask shapes of the LBP and LBSP are shown in Figure 2.

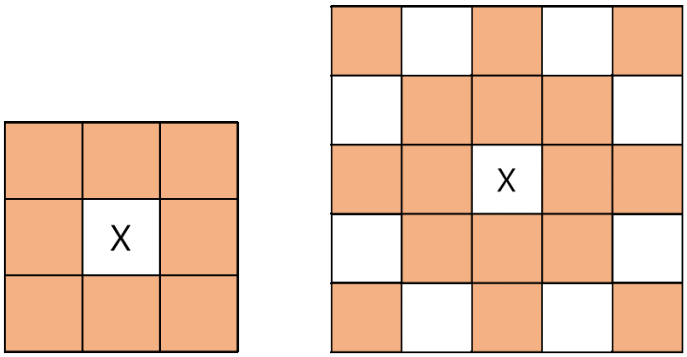


Figure 2. Mask shape of the LBP and LBSP: (**a**) LBP and (**b**) LBSP.

The LBP [18] uses the pixel values of eight marked areas in a 3 × 3 mask, while the LBSP [19] uses pixel values of 16 marked areas in a 5 × 5 mask. Both texture features are calculated using the pixel values of the marked area in the mask, and the reference pixel value. In this paper, we used the LBSP [19] calculation method used in SuBSENSE [21]. Equations (2) and (3) are used to calculate the mask values of the LBP [18] and LBSP [19], respectively:

$$m_{LBP}(i_p, i_x) = \begin{cases} 1, & if\ \|i_p - i_x\| \leq T_d \\ 0, & otherwise \end{cases} \tag{2}$$

$$m_{LBSP}(i_p, i_x) = \begin{cases} 1, & if\ \|i_p - i_x\| \leq T_d \cdot i_x \\ 0, & otherwise \end{cases} \tag{3}$$

where i_p is the pixel value of the area of the colored part in the mask, i_x is the reference pixel value, T_d is the threshold value in the LBP [18], and T_r (≈0.3) is the threshold ratio used in the LBSP [19]. In Equation (3), the LBSP [19] uses the reference pixel value and the threshold ratio to calculate the mask value. The mask value of the LBSP [19] is calculated as 0 or 1 by comparing the absolute difference of i_p and i_x with the calculated threshold value. This method allows the LBSP feature values to adaptively respond to the pixel distribution for the contrast. After calculating the value of the LBSP mask, the process of encoding these values into a 16-bit binary string is performed. Equation (4) represents a formula for converting the calculated LBSP mask values into a 16-bit binary string:

$$LBSP(x) = \sum_{p=0}^{15} m_{LBSP}(i_p, i_x) \cdot 2^p \tag{4}$$

Since this process cannot individually store 16 binary numbers (1 bit) based on the characteristics of computer structure, it is essential to have effective data capacity when it is expressed as one 16-bit number. Due to the characteristics of a video, scenes will change as time passes. Therefore, it is essential that the collected background samples are properly updated. In this study, the background samples were updated using the method of Barnich [30] and Van Droogenbroeck [22]. For each frame, the color component of the current input image and the LBSP [19] component are collected with a probability of $1/T_t(x)$. Components to be updated are randomly selected and updated among 50 background samples. This update method has the advantage of using both previous and current information. The background update parameter $T_t(x)$ is calculated for each frame in the feedback process module.

3.2. BG/FG Classification

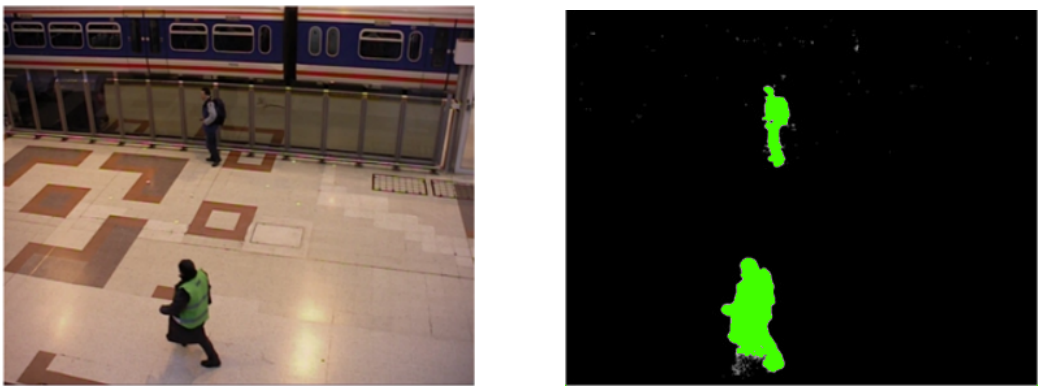
The BG/FG classification module classifies the foreground and background in the input image based on background sample information in the background samples module. Equation (5) represents

a formula for calculating the foreground/background in the input image. This equation is the same as in SuBSENSE [21]:

$$S^t(x) = \begin{cases} 1, & if\ N\{dist(I^t(x), B_n^t(x)) < R_(dist)^(t-1)(x)\} < 2 \\ 0, & otherwise \end{cases} \quad (5)$$

In the above equation, It(x) is the input image, while $S^t(x)$ is the binary image with the foreground (1) and background (0) separated. $dist(I^t(x), B_n^t(x))$ returns the L1 distance and hamming distance of the input image $I^t(x)$ and the background sample $B_n^t(x)$. The example of the FG/BG classification is shown in Figure 3. Equation (6) shows how to calculate the L1 distance. Figure 4 shows how to calculate the hamming distance.

$$L1\ distance = \|I^t(x) - B_n^t(x)\| \quad (6)$$



(**a**) (**b**)

Figure 3. FG/BG classification. (**a**) Input image $I^t(x)$ and (**b**) binary image $S^t(x)$.

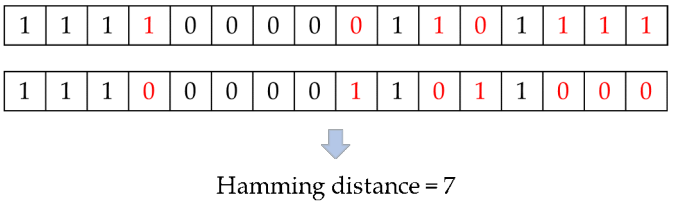


Figure 4. Examples of hamming distance calculation.

As shown in the above equation, the L1 distance is a value obtained by calculating an absolute difference between two values. As shown in Figure 4, the hamming distance is represented by the number of parts (red) with the same position, but with different values for two arrays of the same length. Since the LBSP [19] has 16 encoded binary arrays, the hamming distance can efficiently represent the difference between two LBSP [19] values. $R_{dist}^{t-1}(x)$ returns the color distance threshold $R_{color}^{t-1}(x)$ and LBSP distance threshold $R_{LBSP}^{t-1}(x)$. Equations (7) and (8) are formulas for calculating $R_{color}^{t-1}(x)$ and $R_{LBSP}^{t-1}(x)$:

$$R_{color}^t(x) = R_{color}^0 \cdot R^t(x) \quad (7)$$

$$R_{LBSP}^t(x) = 2^{R^t(x)} + R_{LBSP}^0 \quad (8)$$

where R_{color}^0 and R_{LBSP}^0 are initial values of the color distance threshold and LBSP distance threshold, respectively. $R^t(x)$ is a parameter that is updated in the feedback process. It is used to calculate the distance threshold. It is modeled to have a large value on a dynamic background, and a value close

to 1 on a normal background. It can prevent detection of false positives that can occur on dynamic backgrounds, such as rivers and leaves.

3.3. FP Re-Check

FP re-check is a module that detects and removes false positives from dynamic backgrounds in binary image $S^t(x)$. In this paper, we defined a parameter representing the dynamic region. Equations (9)–(11) are formulas for calculating the dynamic region:

$$DR^t(x) = \begin{cases} 1, & if\ BR^t(x) > blinkthreshold \\ 0, & otherwise \end{cases} \tag{9}$$

$$BR^t(x) = TB^t(x)/t \tag{10}$$

$$TB^t(x) = TB^{t-1}(x) + 1, if\ S^t(x)\ xor\ S^{t-1}(x) = 1 \tag{11}$$

In ViBe+ [22], the foreground and background tend to be periodically repeated in the dynamic background. This case is called "blinking". It can be expressed as the condition of Equation (11). In the above equation, $DR^t(x)$ is a parameter indicating the part with a dynamic background in the input image, $BR^t(x)$ is the blinking rate per frame, and $TB^t(x)$ is the total number of blinking pixels. When a value of $BR^t(x)$ is higher than the blink threshold the pixel is regarded as a dynamic background region. Figure 5 shows the dynamic background region calculated from Equations (9)–(11).

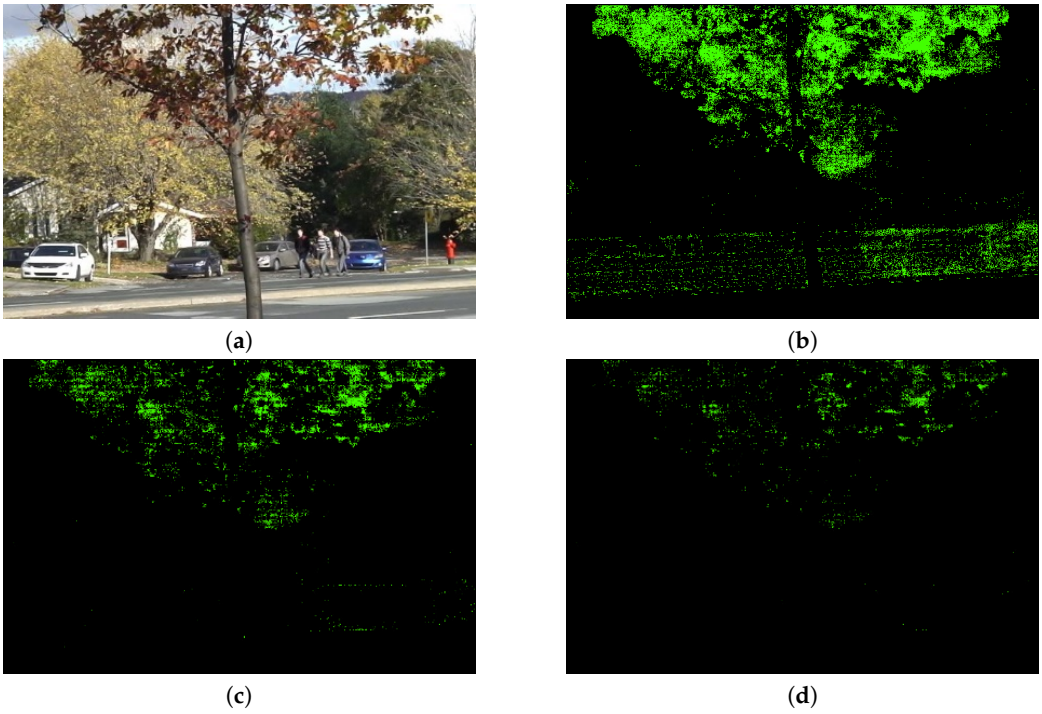


Figure 5. The dynamic background for the blink threshold value. (**a**) Color image, (**b**) $DR^t(x)$ (*blinkthreshold*: 0.01), (**c**) $DR^t(x)$ (*blinkthreshold*: 0.025), and (**d**) $DR^t(x)$ (*blinkthreshold*: 0.05).

The input image shown in Figure 5 is a video image in which a tree shakes in the wind and people and vehicles pass by. When the blink threshold value is 0.01, the dynamic background is relatively well extracted. However, blinking occurs more frequently when the object passes. When the *blinkthreshold* is 0.05, the dynamic background is not extracted sufficiently. Experimental results showed that the *blinkthreshold* was 0.025.

We also defined dynamic background samples for collecting false positives. The dynamic background sample has the same resolution as the input image, like the background samples. The number of samples is

fixed at 30. The dynamic background sample collects the color information of false detection components occurring in the dynamic background. Figure 6 represents conditions for collecting false positives.

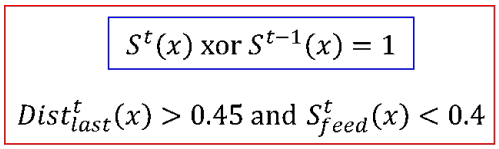


Figure 6. Conditions for collecting false positives.

The blue region in Figure 6 is a feature of the false positive component used in ViBe+ [22]. This indicates a case when the previous result and the current result are different. If this happens frequently, it is likely to be a dynamic background [22]. This is called "blinking". However, this feature tends to occur frequently even when an object passes, or when noise is severe. In this paper, we used additional conditions included in the red region to better represent the false positive component. Equations (12) and (13) show the parameters used in Figure 6:

$$Dist^t_{last}(x) = \left(\frac{L1Dist(I^t(x), I^{t-1}(x))}{255 \times 3} + \frac{hdist(LBSP^t(x), LBSP^{t-1}(x))}{16 \times 3}\right)/2 \tag{12}$$

$$S^t_{feed}(x) = (1 - \alpha) \cdot S^{t-1}_{feed}(x) + \frac{\alpha}{255} \cdot S^{t-1}(x) \tag{13}$$

where $Dist^t_{last}(x)$ is a value obtained by normalizing the color/LBSP distance between the current frame and the previous frame to a range between [0, 1]. $S^t_{feed}(x)$ is a parameter that expresses the trajectory of the object. It is composed of feedback of the previous result $S^{t-1}(x)$. $Dist^t_{last}(x) > 0.45$ can prevent blinking due to image noise, and $S^t_{feed}(x) < 0.4$ can prevent blinking caused by the passing of an object through the image. When the condition of Figure 6 is satisfied, the pixel is regarded as a false positive component and the color component is stored in the dynamic background samples. This process selects and updates the sample randomly, as in the background sample update. When an object is detected on a dynamic background region, the color distance between the collected false positive and the pixel is calculated. If the value is smaller than the dynamic color threshold, it is regarded as a background. This is because the false positives that occur in the dynamic background tend to have distinct brightness differences from the background, such that it is possible to distinguish false positives from color components alone. Through this process, it is possible to efficiently remove the false positives that may occur in a dynamic background.

Figure 7 shows the comparison before and after using FP re-check. This video sequence provides a situation in which an object passing under a tree shaking in the wind to be detected. Figure 7a,b are the input image and ground truth, respectively. Figure 7c shows the detection of a lot of FPs in the shaking tree, and Figure 7d shows the removal of most of the FPs detected in the shaking tree.

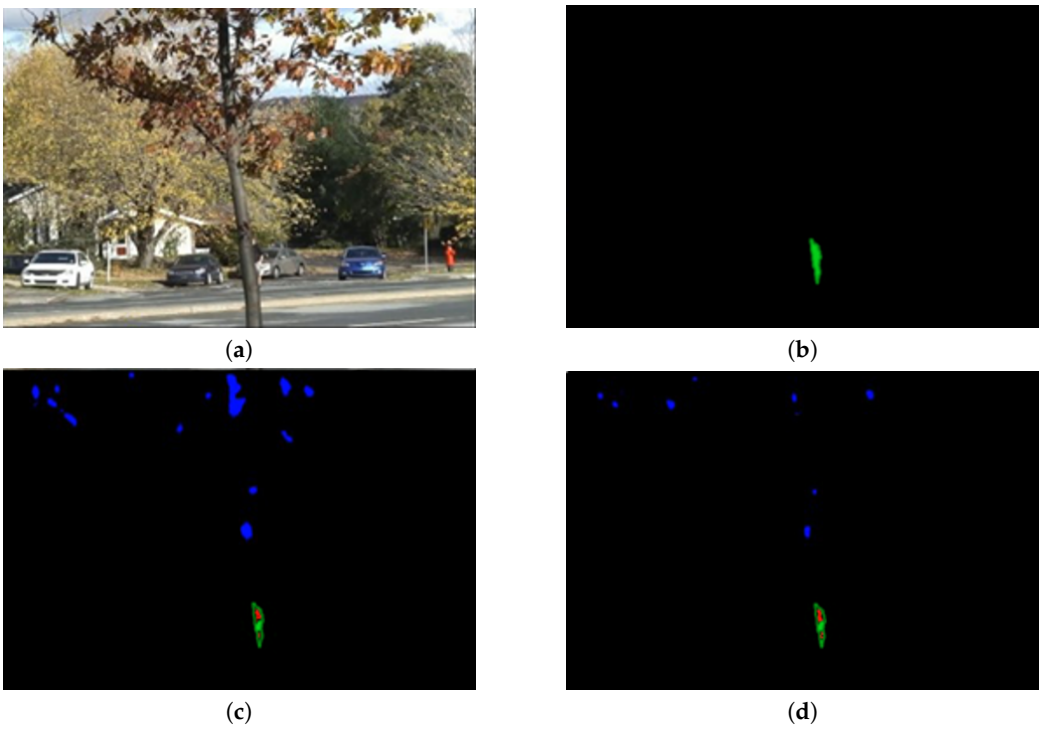


Figure 7. A comparison of before and after using FP re-check. (**a**) Input image, (**b**) ground truth, (**c**) output before using the FP re-check, (**d**) output after using the FP re-check. (Background: black, Foreground: green, False positive: blue, False negative: red.).

3.4. Feedback Process

The feedback process is a module that calculates the $R^t(x)$ and $T^t(x)$ parameters used in the background samples and BG/FG classification module. Equations (14) and (15) are formulas used to calculate the parameters:

$$D^t_{min}(x) = D^{t-1}_{min}(x) \cdot (1-\alpha) + d^t(x) \cdot \alpha \tag{14}$$

$$v^t(x) = \begin{cases} v^{t-1} + w^t(x), & if\ S^t(x) xor S^{t-1}(x) = 1 \\ v^{t-1}(x) - v_{decr}, & otherwise \end{cases} \tag{15}$$

$$w^t(x) = \begin{cases} 1.0, & if\ DR^t(x) = 0 \\ 1.5, & if\ Dist^t_{last}(x) > 0.45,\ S^t_{feed}(x) < 0.4 \\ 0.8, & otherwise \end{cases} \tag{16}$$

where α is the learning rate and $d^t(x)$ is the minimum value of all color/LBSP distances normalized between the input image and the background sample. A smaller value of $d^t(x)$ is often a background, while a larger value is a background or an object. $D^t_{min}(x)$ is fed back to $d^t(x)$ for every frame. This feedback method increases the reliability of the value. We also used two learning rates to vary the feedback rate. These two values are 0.04 (short term) and 0.01 (long term). Short term contains more recent values, while long term contains many older values. This update method allows the algorithm to be robust to the rate at which the environment changes.

$v^t(x)$ is a parameter that quantifies the degree of "blinking" mentioned in the FP re-check module. In this paper, we defined another parameter, $w^t(x)$, to update this parameter. $w^t(x)$ has a large value when motion is large within a dynamic background. In this case, we used the condition to distinguish false positives used in Figure 6. If the motion is not large, $w^t(x)$ has a small value. This has the following advantages. First, false positive detection can be prevented for large motion scenes within a dynamic background. Second, a false negative that can occur when an object passes over a dynamic

background can be removed. Figure 8 is a video scene that should detect an object passing through a river without detecting river flow. Figure 8c shows that this parameter is not used. It does not detect false positives for dynamic backgrounds, but detects false negatives. On the other hand, Figure 8d shows robustness for false positives and false negatives.

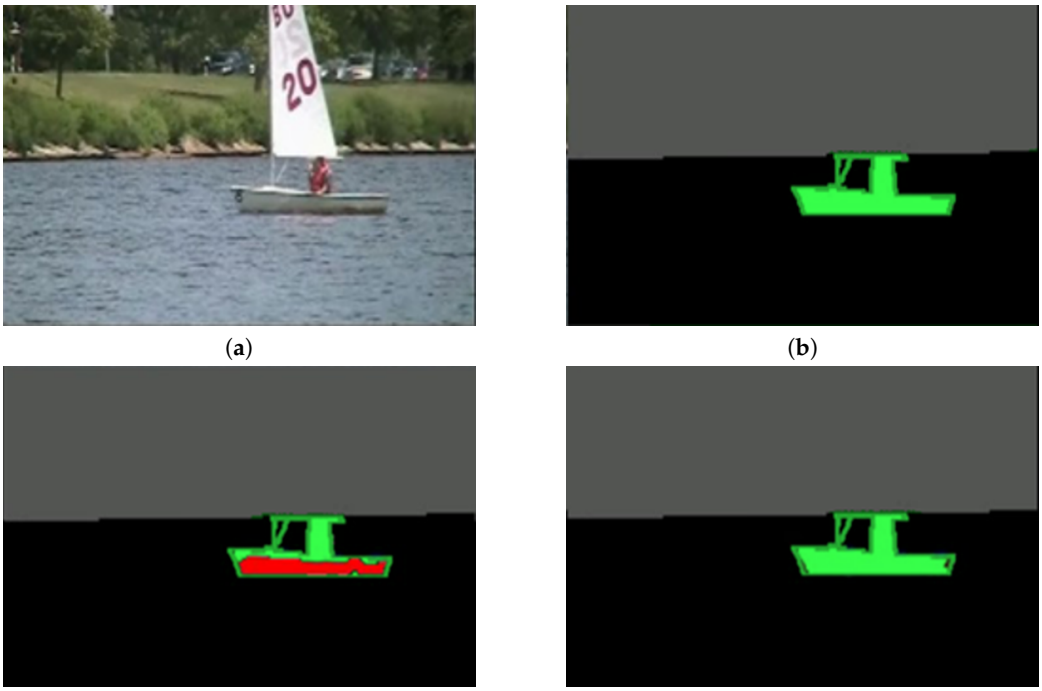


Figure 8. A comparison of before and after using the $w(x)$ parameter. (**a**) Input image, (**b**) ground truth, (**c**) output before using the FP re-check, (**d**) output after using the FP re-check. (Background: black, Foreground: green, False positive: blue, False negative: red.).

The decreasing constant v_{decr} uses a value of 0.1. Equations (17) and (18) are used for calculating the $R^t(x)$ and $T^t(x)$ parameters:

$$R^t(x) = \begin{cases} R^{t-1} + v^t(x), & if\ R^{t-1}(x) < (1 + D^t_{min}(x) \cdot 2)^2 \\ R^{t-1}(x) - \frac{1}{v^t(x)}, & otherwise \end{cases} \tag{17}$$

$$T^t(x) = \begin{cases} T^{t-1} + \frac{1}{v^t(x) \cdot D^t_{min}(x)}, & if\ S^t(x) = 1 \\ T^{t-1}(x) - \frac{v^t(x)}{D^t_{min}(x)}, & otherwise \end{cases} \tag{18}$$

The updating method of $R^t(x)$ and $T^t(x)$ is the same as that of SuBSENSE [21]. $R^t(x)$ is a parameter that is updated by $v^t(x)$. It has a high value for noisy or dynamic backgrounds. The condition of $R^{t-1}(x) < (1 + D^t_{min}(x) \cdot 2)^2$ is to have an exponential relationship when increasing the value of $R^t(x)$. This exponential relationship can reduce false positives by returning higher values of $R^t(x)$ for severely shaken trees.

$T^t(x)$ is a parameter used to update background samples. This parameter is used to determine whether to update background samples with a $1/T^t(x)$ probability in every frame. In other words, the smaller the value, the more frequently updates will occur. It has a large value in an area where an object is detected, and a small value in an area where the background is detected. It also has a small value within dynamic backgrounds. This is because the background is dynamic, thus, requiring more color/LBSP components than a normal background. Figure 9 shows the parameter visualization used in the proposed algorithm.

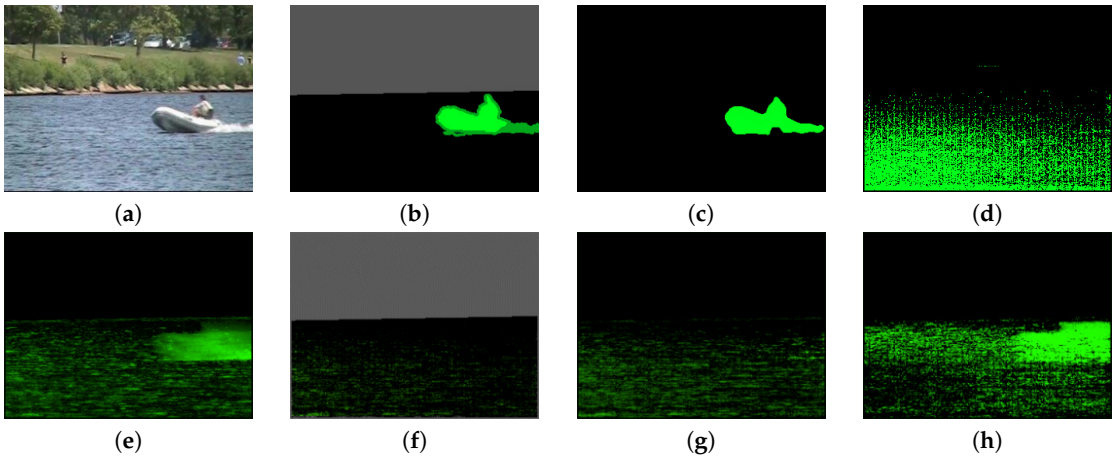


Figure 9. Parameter visualization used in the proposed algorithm. (**a**) $I^t(x)$, (**b**) ground truth, (**c**) S^t_{final}, (**d**) $DR^t(x)$, (**e**) $D^t_{min}(x)$, (**f**) $v^t(x)$, (**g**) $R^t(x)$, and (**h**) $T^t(x)$.

3.5. Post-Processing

The post process module performs post-processing operations based on results obtained from the FP re-check module. To improve the quality of results, we proceeded with morphological operations and binary median filtering operations for the resulting images. This process not only removes noise components in the resulting image but also allows the foreground silhouette to be better represented. However, median filtering requires considerable computation because it requires sorting for every pixel. The sorting algorithms have a common feature that shows the throughput exponentially increases as the amount of data to be sorted increases. In this paper, we proposed a speed optimization method of a median filter for a binary image using an integral image. The feature is a binary image that is consistent with 0 and 1 values. Using this feature, instead of using a sorting algorithm, the median value can be calculated by using the sum of the values in the mask region using the integral image. The integral image is a data structure for quickly and efficiently calculating the sum of values in a rectangular region. Figure 10 shows the example of calculating the sum of the values of the rectangular region using an integral image for the binary image.

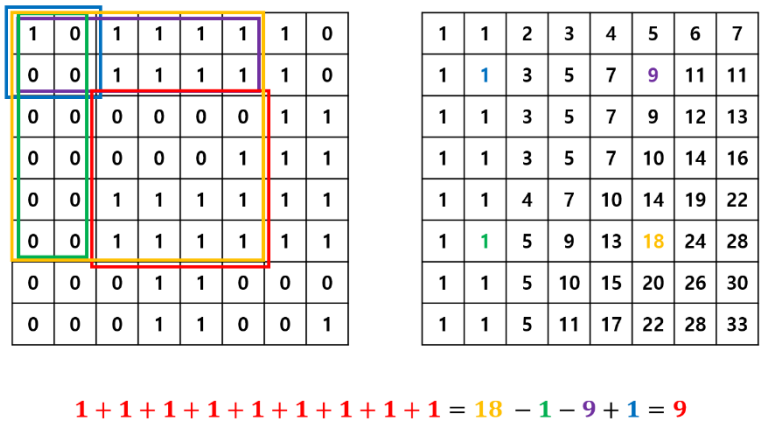


Figure 10. An example of calculating the sum of the values of the rectangular region.

The sum of the values in the rectangular region can be processed quickly regardless of the size of the rectangle. Instead of a sorting algorithm that requires a considerable amount of computation, it computes efficiently by calculating the sum of mask region values. If the value is less than half of the mask size, it is calculated as a value of 0. Conversely, when the value is larger than half of the mask size, it is calculated as 1. Therefore, it is possible to perform fast median filtering regardless of the

mask size. Table 1 shows the conventional method and the proposed method, with a comparison of the processing speed according to the mask size at the VGA resolution.

Table 1. Comparison of the processing speed according to the mask size at the VGA.

Mask Size	Conventional Method	Proposed Method
7×7	10 msec	1 msec
9×9	11 msec	1 msec
11×11	13 msec	1 msec
13×13	15 msec	1 msec

4. Experimental Results

In this paper, we evaluated and compared the proposed algorithm with changedetection.net. The site of changedetection.net provides a CDnet dataset for evaluating and comparing motion detection algorithms. The CDnet dataset consists of two datasets: the CDnet 2012 dataset [2] and CDnet 2014 dataset [1]. It includes various environments, ranging from sequences suitable for motion detection algorithms to sequences taken under harsh environments. In this paper, we evaluated the proposed algorithm for the CDnet 2012/2014 dataset [1,2], and compared it with other background subtraction algorithms. The proposed algorithm consists of C ++ language and the OpenCV library. The experimental environment consists of a 7th-generation Intel® Core™ i7 at 3.60 GHz, and a Nvidia GeForce GTX 1080 Graphics Card, using C++, CUDA, OpenCV, and OpenMP.

SuBSENSE [21] is one of these comparison algorithms. It is an algorithm that proposes a background modeling method using color/LBSP information based on ViBe+ [22] and PBAS [23]. PAWCS [25] proposes persistence of background samples based on the SuBSENSE [21] algorithm. It has an improved performance compared to SuBSENSE [21]. Multimode Background Subtraction (MBS) [17] is an algorithm that classifies backgrounds based on pixel sets using RGB and YCbCr channels. It classifies them using clustering. The proposed algorithm showed a lower performance than PAWCS [25], but better performance than the other algorithms.

The CDnet 2014 dataset [1] has 11 categories, including bad weather, low frame rate, night video, camera zoom in/out and rotating PTZ, turbulence video, and six categories from the CDnet 2012 dataset [2]. Each of these categories also has four to six sequences. Tables 2 and 3 shows a comparison of other background subtraction algorithms and the proposed algorithm for the CDnet 2012/2014 dataset [1,2]. The F-Measure is computed for both the precision P and recall R by $F = 2 \cdot P \cdot R/(P + R)$, representing the tradeoff between the two values.

WeSamBE [26] uses a method similar to PAWCS [25]. It sets weights on collected background samples and updates the background according to these weights. SharedModel [11] is a Gaussian Mixture Modeling (GMM) algorithm that takes the relationship between pixels and background samples into account. SWCD [27] is a sliding window approach combined with dynamic control of update parameters for updating background frames, which we called sliding window-based change detection.

Table 2. Comparison of different background subtraction algorithms and the proposed algorithm on the CDnet 2012.

Method	Precision	FPR	FNR	FPS (QVGA)
Proposed method	0.8650	0.0058	0.1584	45 fps
PAWCS [25]	0.8746	0.0051	0.1453	9 fps
SuBSENSE [21]	0.8576	0.0062	0.1719	22 fps
MBS [17]	0.8480	0.0069	0.1897	8 fps
Spectral-360[31]	0.8461	0.0080	0.2230	12 fps
STBM [32]	0.8210	0.0089	0.1650	12 fps
SGMM-SOD [33]	0.8339	0.0062	0.2303	34 fps

Table 3. Comparison of different background subtraction algorithms and the proposed algorithm on the CDnet 2014.

Method	Precision	FPR	FNR	F-Measure	Average Ranking across Categories	FPS (QVGA)
Proposed method	0.7668	0.0090	0.1802	0.7535	14.36	45 fps
SWCD [27]	0.7527	0.0070	0.2161	0.7583	18.36	10 fps
PAWCS [25]	0.7857	0.0051	0.2282	0.7403	14.64	9 fps
SuBSENSE [21]	0.7509	0.0096	0.1876	0.7408	16.73	22 fps
MBS [17]	0.7382	0.0073	0.2611	0.7288	21.09	8 fps
WeSamBE [26]	0.7679	0.0076	0.2045	0.7446	16.27	2 fps
SharedModel [11]	0.7503	0.0088	0.1902	0.7474	17.27	35 fps

The proposed algorithm showed lower performance than PAWCS [25] in terms of precision and FPR. However, the proposed method showed better performance than these algorithms in terms of FNR and F-measure. Compared to SuBSENSE [21], it showed better performance. Also, the proposed algorithm showed the highest rank in terms of the average ranking across categories. The average ranking across categories means the average of the sum of ranks for all categories of CDnet 2014 dataset [1].

Table 4 shows the F-measure and processing speed comparisons for the 11 categories of the CDnet 2014 dataset [1]. The 11 categories are Baseline, Dynamic Background, Bad Weather, Shadow, Night Videos, Low Framerate, PTZ, Turbulence, Camera Jitter, Intermittent Object Motion, and Thermal. Baseline category represents a mixture of mild challenges typical of the next four categories. Some videos have subtle background motion, others have isolated shadows; some have an abandoned object, and others have pedestrians that stop for a short while and then move away. These videos are fairly easy to process, and are provided mainly as a reference. The Dynamic Background category includes scenes with strong (parasitic) background motion: boats on shimmering water, cars passing next to a fountain, or pedestrians, cars, and trucks passing in front of a tree shaken by the wind. The Bad Weather category includes outdoor videos captured in challenging winter weather conditions. Shadows category consists of indoor and outdoor videos exhibiting strong as well as faint shadows. The Night Videos category includes videos captured at night. The Low Frame Rate category contains videos captured at varying frame rates between 0.17 fps and 1 fps. The PTZ category contains video footage captured by pan–tilt–zoom cameras in slow continuous pan mode, intermittent pan mode, two-position patrol-mode PTZ, or zooming-in/zooming-out. The Turbulence category includes outdoor videos showing air turbulence caused by rising heat. The Camera Jitter category contains indoor and outdoor videos captured by unstable (e.g., vibrating) cameras. The jitter magnitude varies from one video to another. The Intermittent Object Motion category includes videos with scenarios known for causing "ghosting" artifacts in the detected motion, i.e., objects move, then stop for a short while, after which they start moving again. Some videos include still objects that suddenly start moving, e.g., a parked vehicle driving away, and also abandoned objects. This category is intended for testing how various algorithms adapt to background changes. The Thermal category includes videos that have been captured by far-infrared cameras. These videos contain typical thermal artifacts such as heat stamps (e.g., bright spots left on a seat after a person gets up and leaves), heat reflection on floors and windows, and camouflage effects, when a moving object has the same temperature as the surrounding regions.

STSHBM [12], PAWCS [25], and SWCD [27] have robust performance but slow processing speed. In this case, a frame drop occurs in the actual video stream. Since the algorithms use the information from the previous frame, the performance can be degraded. The proposed algorithm shows overall robust results for the 11 categories. The proposed algorithm has a fast processing speed of 45 fps at QVGA resolution. Also, the proposed algorithm runs at 150 fps at the same resolution on a Nvidia GeForce GTX 1080 GPU.

Table 4. Comparison of F-measure and processing speed with the proposed method.

Method	F-Measure											FPS (QVGA)
	Baseline	Dynamic Background	Bad Weather	Shadow	Night Videos	Low Frame Rate	PTZ	Turbulence	Camera Jitter	Intermittent Object	Thermal	
Proposed method	0.9487	0.8376	0.8616	0.8984	0.5701	0.6445	0.3367	0.8304	0.8228	0.7264	0.8152	45 fps
SRPCA [24]	-	0.8866	-	-	-	-	-	-	-	-	-	4 fps
STSHBM [12]	0.9534	0.9120	-	0.8930	-	-	-	-	-	-	-	0.98 fps
PAWCS [25]	0.9397	0.8938	0.8152	0.8710	0.4152	0.6588	0.4615	0.6450	0.8137	0.7764	0.8324	9 fps
SuBSENSE [21]	0.9503	0.8177	0.8619	0.8646	0.5599	0.6445	0.3476	0.7792	0.8152	0.6569	0.8171	22 fps
TVRPCA [13]	-	0.5516	-	-	-	-	-	-	-	-	-	0.27 fps
SWCD [27]	0.9214	0.8645	0.8233	0.8779	0.5807	0.7374	0.4545	0.7735	0.7411	0.7092	0.8581	10 fps
SharedModel [11]	0.9522	0.8222	0.8480	0.8898	0.5419	0.7286	0.3860	0.7339	0.8141	0.6727	0.8319	35 fps

5. Conclusions

In this paper, we defined a dynamic background region, searched for dynamic backgrounds, and newly defined dynamic background samples to collect false positive components that might occur in the dynamic background region, then re-checked false positives. In this way, it is possible to remove false detection components that often occur in a dynamic background. However, since the dynamic background region of this paper is a parameter based on the assumption that "when blinking occurs, the region is likely to be a dynamic background region", there is a limitation when searching for a perfect dynamic background region. To improve this, it is necessary to find the unique tendency of the dynamic background, and model it mathematically. We also performed the speed optimization for median filtering for the binary image using an integral image. The proposed method can efficiently perform median filtering regardless of mask size. It is feasible to use the proposed algorithms for real-time CCTV video monitoring with superior motion detection performance and fast processing speed.

Author Contributions: Conceptualization, S.-h.L.; Methodology, S.K.; Software, S.-h.L.; Investigation, G.-c.L.; Writing—Original Draft Preparation, S.-h.L.; Writing—Review & Editing, J.Y., S.K. and G. -c.L.; Supervision, S.K.; Project Administration, S.K.

Acknowledgments: This work was supported by an Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korean government (MSIT) (No. R0132-15-1005, Content visual browsing technology in online and offline environments).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, Y.; Jodoin, P.M.; Porikli, F.; Konrad, J.; Benezeth, Y.; Ishwar, P. CDnet 2014: An expanded change detection benchmark dataset. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Columbus, OH, USA, 23–28 June 2014; pp. 393–400.
2. Goyette, N.; Jodoin, P.M.; Porikli, F.; Konrad, J.; Ishwar, P. Changedetection. net: A New Change Detection Benchmark Dataset. In Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Providence, RI, USA, 16–21 June 2012; pp. 1–8.
3. Lim, L.A.; Keles, H.Y. Foreground segmentation using convolutional neural networks for multiscale feature encoding. *Pattern Recognit. Lett.* **2018**, *112*, 256–262. [CrossRef]
4. Lim, L.A.; Keles, H.Y. Learning Multi-scale Features for Foreground Segmentation. *arXiv* **2018**, arXiv:1808.01477.
5. Lim, L.A.; Keles, H.Y. Foreground segmentation using a triplet convolutional neural network for multiscale feature encoding. *arXiv* **2018**, arXiv:1801.02225.
6. Zheng, W.; Wang K.; Wang F.Y. A novel background subtraction algorithm based on parallel vision and Bayesian GANs. *Neurocomputing* **2017**, *30.*
7. Stauffer, C.; Grimson, W.E.L. Adaptive background mixture models for real-time tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, CA, USA, 25 June 1998; pp. 246–252.
8. Wright, J.; Ganesh, A.; Rao, S.; Ma, Y. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. *arXiv* **2009**, arXiv:0905.0233.

9. Lin, Z.; Ganesh, A.; Wright, J.; Wu, L.; Chen, M.; Ma, Y. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. In Proceedings of the Third International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, Aruba, The Netherlands, 13–16 December 2009; pp. 213–216.
10. Candès, E.J.; Li, X.; Ma, Y.; Wright, J. Robust principal component analysis? *J. ACM* **2011**, *58*, 11. [CrossRef]
11. Chen, Y.; Wang, J.; Lu, H. Learning Sharable Models for Robust Background Subtraction. In Proceedings of the 2015 IEEE International Conference on Multimedia and Expo (ICME), Turin, Italy, 29 June–3 July 2015; pp. 1–6.
12. Chen, M.; Wei, X.; Yang, Q.; Li, Q.; Wang, G.; Yang, M.H. Spatiotemporal GMM for background subtraction with superpixel hierarchy. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1518–1525. [CrossRef] [PubMed]
13. Cao, X.; Yang, L.; Guo, X. Total variation regularized rpca for irregularly moving object detection under dynamic background. *IEEE Trans. Cybern.* **2016**, *46*, 1014–1027. [CrossRef] [PubMed]
14. Mallat, S.G. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **1989**, *2*, 674–693. [CrossRef]
15. Guido, R.C. Practical and useful tips on discrete wavelet transforms. *IEEE Signal Process. Mag.* **2015**, *32*, 162–166 [CrossRef]
16. Guido, R.C.; Addison, P.; Walker, J. Introducing wavelets and time-frequency analysis. *IEEE Eng. Biol. Med. Mag.* **2009**, *28*, 13. [CrossRef] [PubMed]
17. Sajid, H.; Cheung, S.C.S. Universal Multimode Background Subtraction. *IEEE Trans. Image Process.* **2017**, 26, 3249–3260. [CrossRef] [PubMed]
18. Heikkila, M.; Pietikainen, M. A texture-based method for modeling the background and detecting moving objects. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 657–662. [CrossRef] [PubMed]
19. Bilodeau, G.A.; Jodoin, J.P.; Saunier, N. Change detection in feature space using local binary similarity patterns. In Proceedings of the 2013 International Conference on Computer and Robot Vision (CRV), Regina, SK, Canada, 28–31 May 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 106–112.
20. St-Charles, P.L.; Bilodeau, G.A. Improving background subtraction using local binary similarity patterns. In Proceedings of the 2014 IEEE Winter Conference on Applications of Computer Vision (WACV), Steamboat Springs, CO, USA, 24–26 March 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 509–515.
21. St-Charles, P.L.; Bilodeau, G.A.; Bergevin, R. Subsense: A universal change detection method with local adaptive sensitivity. *IEEE Trans. Image Process.* **2015**, *24*, 359–373. [CrossRef] [PubMed]
22. Droogenbroeck, M.; Paquot, O. Background Subtraction: Experiments and Improvements for ViBe. In Proceedings of the 25th IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 32–37.
23. Hofmann, M.; Tiefenbacher, P.; Rigoll, G. Background segmentation with feedback: The pixel-based adaptive segmenter. In Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Providence, RI, USA, 16–21 June 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 38–43.
24. Javed, S.; Mahmood, A.; Bouwmans, T.; Jung, S.K. Spatiotemporal low-rank modeling for complex scene background initialization. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 1315–1329. [CrossRef]
25. St-Charles, P.-L.; Bilodeau, G.-A.; Bergevin, R. A self-adjusting approach to change detection based on background word consensus. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 6–9 January 2015.
26. Jiang, S.; Lu, X. WeSamBE: A Weight-Sample-Based Method for Background Subtraction. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *28*, 2105–2115. [CrossRef]
27. Işık, Ş.; Özkan, K.; Günal, S.; Gerek, Ö.N. SWCD: A sliding window and self-regulated learning-based background updating method for change detection in videos. *J. Electron. Imaging* **2018**, *27*, 023002. [CrossRef]
28. Wu, Y.; He, X.; Nguyen, T.Q. Moving Object Detection With a Freely Moving Camera via Background Motion Subtraction. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *27*, 236–248. [CrossRef]
29. Chen, B.H.; Shi, L.F.; Ke, X. A Robust Moving Object Detection in Multi-Scenario Big Data for Video Surveillance. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 982–995. [CrossRef]
30. Barnich, O.; Van Droogenbroeck, M. ViBe: A universal background subtraction algorithm for video sequences. *IEEE Trans. Image Process.* **2011**, *20*, 1709–1724. [CrossRef] [PubMed]
31. Sedky, M.; Moniri, M.; Chibelushi, C. Object Segmentation Using Full-Spectrum Matching of Albedo Derived from Colour Images. U.S. Patent 2,374,109, 12 October 2011.

32. Chen, M.; Yang, Q.; Li, Q.; Wang, G.; Yang, M.H. Spatiotemporal Background Subtraction using Minimum Spanning Tree and Optical Flow. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 521–534.
33. Heras, R.; Sikora, T. Complementary background models for the detection of static and moving objects in crowded environments. In Proceedings of the IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS), Klagenfurt, Austria, 30 August–2 September 2011; pp. 71–76.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).