Machine Vision and Applications (2015) 26:723–740
DOI 10.1007/s00138-015-0696-8

ORIGINAL PAPER

Detection of foreground in dynamic scene via two-step background subtraction

K. L. Chan[1]

Received: 27 February 2014 / Revised: 21 April 2015 / Accepted: 12 June 2015 / Published online: 3 July 2015

Abstract Various computer vision applications such as video surveillance and gait analysis have to perform human detection. This is usually done via background modeling and subtraction. It is a challenging problem when the image sequence captures the human activities in a dynamic scene. This paper presents a method for foreground detection via a two-step background subtraction. Background frame is first generated from the initial image frames of the image sequence and continuously updated based on the background subtraction results. The background is modeled as non-overlapping blocks of background frame pixel colors. In the first step of background subtraction, the current image frame is compared with the background model via a similarity measure. The potential foregrounds are separated from the static background and most of the dynamic background pixels. In the second step, if a potential foreground is sufficiently large, the enclosing region is compared with the background model again to obtain a refined shape of the foreground. We compare our method with various existing background subtraction methods using image sequences containing dynamic background elements such as trees and water. We show through the quantitative measures the superiority of our method.

Keywords Video surveillance · Background subtraction · Dynamic scene · Foreground detection · Dynamic texture

✉ K. L. Chan
itklchan@cityu.edu.hk

1 Department of Electronic Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong

1 Introduction

Many computer vision applications, such as video surveillance [1], gait analysis [2], video segmentation and retrieval [3], demand the detection of moving objects such as humans and vehicles. To detect moving targets, one common approach is to model the background scene and in each image frame finds out those pixels that are similar to the background and those that are not. This is background subtraction. An alternative approach is to detect the moving targets by finding those pixels that are certainly not similar to the background. There have been many researches in detecting foreground targets in static scene while detecting foreground targets in dynamic scene is more challenging. In the former category, the background may be stationary or moderately dynamic. In the latter category, the background scene may exhibit vigorous motions. Bouwmans [4] presented a survey on statistical background modeling. The methods were classified in term of category, strategies used, and critical situation to be tackled. Sobral and Vacavant [5] also presented a recent review and evaluation of 29 background subtraction methods.

This paper reports on the detection of foreground targets in a dynamic environment. Our proposed method is robust so that it can detect targets in a scene containing moving elements such as water and trees. The detection process is complicated due to the presence of various defects in the image sequence. For instance, to detect swimmers in the swimming pool, the water surface can exhibit random motions and ripples. Other objects such as lane dividers are swaying. The wet body of swimmers and splashes can give rise to specular reflections in the captured video frames. Traditional background modeling/subtraction methods are not designed for tackling these problems. For instance, Ning et al. [6] used the least median of squares method for modeling individual pixels of the background scenes which are mainly static. Li et al. [7] proposed a method for modeling background by the principal features and their statistics. The principal features for static background were color and gradient, while dynamic background was characterized by color co-occurrence. Foreground and background were classified and detected by a Bayesian framework. Pixelwise background color has also been modeled by a single Gaussian distribution. It would be better to employ multiple Gaussian distributions when pixels may exhibit multiple background colors due to repetitive motions and illumination changes. Stauffer and Grimson [8] proposed the modeling of background colors using mixture of Gaussian distributions. Pixel values that did not match any of the background distributions were regarded as foreground. However, random motions and sudden changes in the background are very difficult to model. Since conventional background subtraction methods do not assume the existence of a vigorously moving background, they will not work on dynamic scene. We therefore model the moving background as dynamic texture and propose a method for foreground detection in dynamic scene.

The organization of the paper is as follows. Section 2 presents a review of relevant work on background modeling and subtraction. We describe the challenges and our approach to achieve accurate foreground detection in a dynamic environment. Section 3 describes the generation of background frame which is used as the initial background model in our foreground detection framework. Section 4 describes our two-step dynamic background subtraction and the updating of background model. We briefly introduce the block-based mixture of Gaussians model, codebook method, self-organizing artificial neural networks, and two sample-based background subtraction methods (visual background extractor, pixel-based adaptive segmenter), which are selected for experimentation. All the methods are evaluated using image sequences containing illumination changes or moving background such as trees and water. Selected image frames and numeric results are presented in Sect. 5. We examine the experimental results and discuss the strengths and limitations of our two-step background subtraction method with respect to other methods in Sect. 6. Finally, we conclude the paper in Sect. 7.

2 Related work

Automatic video surveillance systems for monitoring moving objects typically consist of the background subtraction/foreground detection, tracking of targets along the image sequence, and inference of the motion. Typical targets are humans or vehicles and they are the foregrounds in each image frame. Their detection can be achieved via the background subtraction process. One common assumption is that the background is stationary or changes slowly. Some recent methods have been proposed that can tackle background movements. In [9], moving objects were first detected by subtracting the static background and the current image frame. Pixels of moving objects were labeled as dynamic background and foreground. Finally, Markov Random Field model was used to infer the coherence over the labels for a more accurate segmentation of background and foreground. Liao et al. [10] proposed a pattern kernel density estimation technique to model the probability distribution of local patterns in the pixel. A multimodal background models and multi-scale fusion scheme were developed for subtracting the dynamic background. In [11], the background was modeled as a set of warping layers to account for the intensity changes of pixels due to background motions. Foreground regions were then segmented as they cannot be modeled by composition of the background layers. El Baf et al. [12] proposed to model the dynamic background using a type-2 fuzzy mixture of Gaussians model to tackle critical situations such as waving trees and moving water.

To detect and recognize human motion in the aquatic environment is very challenging. Most of the background is non-stationary. Lu and Tan [13] modeled the entire background (the swimming pool) with a mixture of Gaussian distributions. The method has the advantages of minimum memory required to store the background model and is easier to update. Kam et al. [14] developed a video surveillance system capable of detecting drowning incidents in swimming pool. The background was modeled by incorporating prior knowledge about movements of water and lane dividers. Swimmers were detected based on the similarity of colors to the background model. In [15,16], the surveillance system was changed to employ a block-based background model and swimmers were detected by hysteresis thresholding. Eng et al. [16,17] proposed to model the background as a composition of homogeneous blob movements. The method can adapt to change of background regions over time which is good for modeling the dynamic aquatic environment. The problem of specular reflection at nighttime was also tackled by a filtering module.

Many background subtraction methods use intensity (color or gray value) information and assume that the salient features are still stable when the background exhibits quasi-rigid motions. They may not work on scene where the background elements can move vigorously. The intensity patterns of the background can be greatly distorted spatially and temporally. Methods have been proposed to use motion feature. Wang et al. [18] proposed measuring motion frequency in a pre-processing step. The per pixel motion frequency was estimated by calculating the average number of significant interframe difference. Motion frequency was classified as low, medium and high by thresholding. Pixel color of the latter two classes was changed to remove specular reflections, sensor noise, etc. Swimmer detection was finally carried out on the transformed video frame using block-based background model. In [19], a modified mixture of Gaussians method was used to subtract background (detect foreground). Hierarchical optical flow field was computed and only those optical flow vectors of large temporal intensity difference were retained. Finally, the initial foreground region was pruned by considering the amplitude and orientation of the optical flow vectors. In [20], each foreground feature point was selected if its motion (estimated by temporal difference of Gaussian) was similar to its neighbors. Foreground model was generated by clustering of 2D spatial coordinates and motion vectors (estimated by optical flow computation) of feature points.

Doretto et al. [21] considered the image sequence of moving scene as dynamic texture. They proposed a method to model, recognize and synthesize the visual signals close to moving scene. Chetverikov et al. [22] addressed two related problems: detect regions of dynamic texture and detect targets in a dynamic texture. Dynamic texture was modeled as optical flow residual which was a simplified and faster version of their previous method [23] using level-set function. Traver et al. [24] proposed an optical-flow-based dynamic texture detection method. Dynamic texture was detected when there was violation of the brightness constancy assumption.

Our foreground detection method, as shown in Fig. 1, is composed of background modeling and dynamic background subtraction. Background frame is generated from the original image sequence. The background is modeled by partitioning the background frame into non-overlapping blocks of pixels. We consider the moving background elements (e.g., waving trees, rippling water, swaying lane dividers) as dynamic texture. Some background elements may exhibit changes due to illumination changes (e.g., windows) or camera jitter. The remaining background elements (e.g., road, swimming pool walkway) are considered as stationary. All these elements must be handled by our background subtraction method. To speed up the background modeling/subtraction process, we employ pixel colors rather than extracting motion feature from the image sequence. We compare each pixel of the current image frame with the background model in a two-step dynamic background subtraction framework.

The idea of two-step background subtraction was first proposed in [25]. A background frame is generated from a short image sequence by vector median filtering The background is modeled as block-based color Gaussian mixture model by k-means clustering. In the first step of background subtraction, the current image frame is compared with the background model via a spatial similarity measure. The potential targets are separated from most of the background pixels. In the second step, if a potential target is sufficiently large, the enclosing block is compared with the background model again to obtain a refined shape of the foreground. The parameters of the two-step background subtraction are set empirically. There is no background model updating step. In that work, test was carried out on only one swimming image sequence.

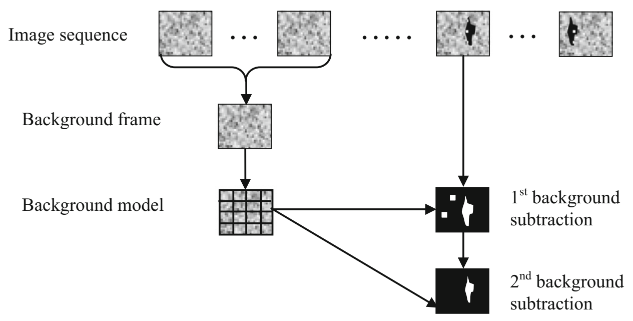

Fig. 1 Overview of the two-step background subtraction (TSBS)

In the present work, initial background model can be generated quickly using only one original image frame. The background scene is modeled by the real observed colors in the image sequence, not synthetic values. We sampled some image frames and employed ROC analysis to determine the two critical parameters of the two-step background subtraction. To speed up the background subtraction, once the new pixel matches with a background pixel, the search will stop. The background model is updated periodically based on the previous background subtraction results. A thorough test was carried out on many image sequences using the same set of parameters.

Stauffer and Grimson [8] claimed the pixelwise mixture of Gaussians can deal with repetitive motions of scene elements. However, the results in [9] and [12] indicate that the pixelwise mixture of Gaussians is not effective in modeling dynamic background such as swaying trees and waving water. Eng et al. [16,17] partitioned the background frame into blocks and model each block of background colors using hierarchical k-means clustering. They implemented a spatial searching process to detect the displaced background colors. The detected swimmer is refined via the foreground modeling step. Our method also models the background scene by a block-based scheme since it is a better approach to tackle background motions than the pixelwise scheme. Haque and Murshed [26] proposed a background subtraction method by measuring the similarity of the current pixel with all recently observed background colors instead of using the mean values as the believed-to-be background colors. We also find that k-means clusters are not effective in modeling highly textured background such as swaying trees. Therefore, the background scene is modeled as blocks of background frame pixel colors. We define the background color similarity measure which is different from Eng et al. [16,17]. As foreground colors can be wide ranging, our method has no foreground modeling step. We implement a two-step approach to detect and subtract background colors in a coarse-to-fine manner.

Zha et al. [9] considered the background/foreground segmentation as a labeling problem. Multi-scale images were formed. While dynamic background may be wrongly detected as foreground in small scale, it will be detected as background in large scale. True foreground will always be detected in all scales. They proposed a Markov Random Field model to infer the spatial and temporal coherence over the background/foreground labels. Our method compares each frame of the image sequence with the background model via a spatial similarity measure. It needs no temporal coherence. Therefore, stationary foregrounds can be detected immediately. Some methods, e.g., [9] and [11], need to set up multiple images for the subtraction of dynamic background in each video frame. Our method demands only one background frame. Some methods employ optical-flow-based techniques to model background motions. There is no guarantee that the motion vectors of foregrounds and backgrounds are separable. Moreover, to compute motion feature is more time consuming. Eng et al. [16,17] implemented a filtering module to tackle the problem of specular reflection of the water surface at nighttime. We find that the use of appropriate color model can tackle the temporal changes of the background colors.

3 Initialization of the background model

The background subtraction methods chosen for comparison usually have the background modeling step included. We generate a background frame which contains no foreground objects for our foreground detection method. If the image sequence contains image frames with no foreground objects, we can select one as the background frame. If the training image frames contain moving objects, the background frame is generated from the image sequence by vector median filtering to remove the foreground colors. The only limitation is that if slow moving objects or stationary objects are present in the initial training frames, the background model is not correct. That will take some time for the updating step to correct the background model. Fortunately, we did not encounter this situation in the testing image sequences.

Let $V_{x,y}^{\tau}$ be an array of colors over τ number of sampled image frames, $V_{x,y}^{\tau} = \{v_{x,y}^{t} | t = 1, \ldots, \tau\}$, where $v_{x,y}^{t}$ is the color vector of the t^{th} sampled image frame at spatial coordinate (x, y) having color components $v_{x,y,i}^{t}$, $i = 1, \ldots, k$. Each background pixel is obtained by

$$b_{x,y} = \arg\min_{v_{x,y}^{q}} \sum_{r=1}^{\tau} \left(\sum_{i=1}^{k} |v_{x,y,i}^{r} - v_{x,y,i}^{q}| \right) \qquad (1)$$

where q is the index of the sampled image frame containing the most likely background color, k is the number of color components. In most cases, τ is set to 10. We sample 10 frames in a training sequence of 100 frames at an interval of 10. Therefore, the background frame pixel colors are real background colors, not artificial values computed by statistical measure. Figure 2 shows the background frames generated from the testing image sequences. For PETS 2001 and Campus image sequences, we select one image frame as background frame. For the two swimming pools image sequences, all image frames contain swimmers. We generate the background frames by vector median filtering. It can be seen that the foreground swimmers (see Fig. 14) are completely removed in the background frames as shown in Fig. 2c, d. The background frame is partitioned into a number of non-overlapping blocks for use in the following foreground detection.

4 Foreground detection

In the following sub-section, we propose a method for foreground detection via a two-step background subtraction (TSBS). Finally, we present an algorithm for background model updating.

4.1 Two-step background subtraction (TSBS)

We are inspired by the research works in dynamic texture detection. Dynamic textures, usually considered as spatio-temporally varying visual patterns, are modeled in various approaches such as Markov random field and incoherent optical flow field [24]. In our study, we need to subtract both static and dynamic backgrounds. Take the swimming pool scene as an example. The foreground swimmers may be moving fast or slow or even motionless. Static background elements contain the walkway surrounding the swimming pool, lifeguard stand, etc. However, when there is camera movement (e.g., jitter) or illumination changes, these background elements are not purely stationary. Dynamic background elements are the water and swaying lane dividers. There are various kinds of background motion—sinusoidal movements of ripples, random motions of water, unidirectional motion of splashes, undulating motion of lane dividers, etc. To tackle illumination changes or specular reflections in the background scene, we adopt the HSV color model for the image data. HSV is better than RGB since the base color is separated from the brightness. In the first step of background subtraction, the dynamic and static background colors are detected by a spatial similarity measure. Finally, in the second step, the initial foreground regions are refined by the subtraction of background colors via a close-range similarity measure.

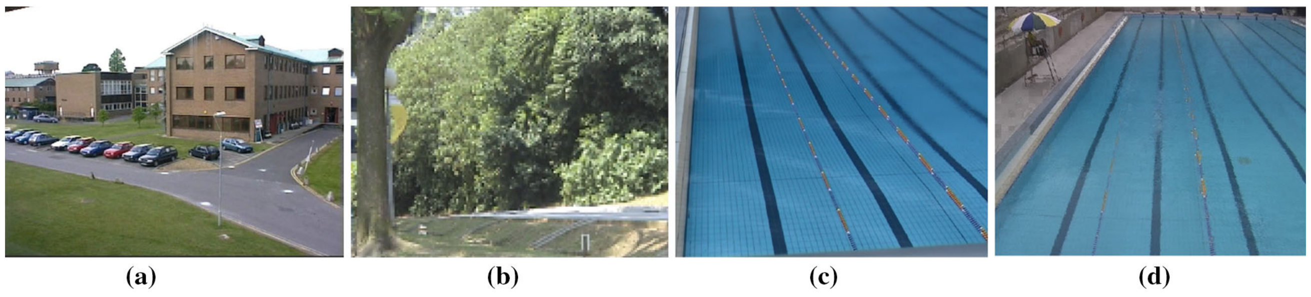

Fig. 2 Background frames generated from: **a** PETS 2001, **b** Campus, **c** swimming pool 1, **d** swimming pool 2

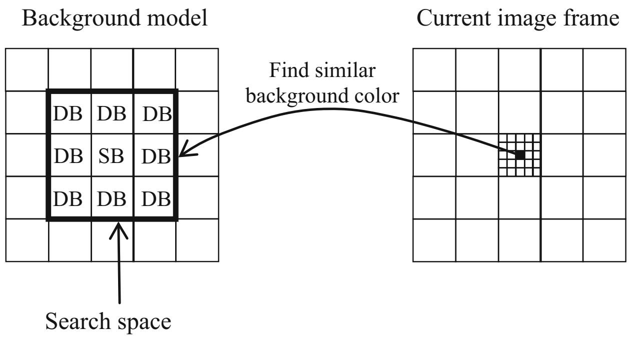

Fig. 3 First step of background subtraction (*DB* block containing dynamic background colors, *SB* block containing static background colors)

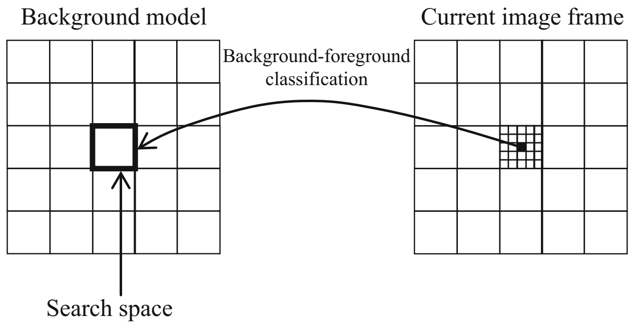

Fig. 4 Second step of background subtraction

We consider the problem of background subtraction as a searching problem in the background model such that there are colors in the background model very similar to the colors in the current image frame. With the use of a stationary camera in the acquisition of the image sequence, it is reasonable to assume that the background elements do not move over a long distance. Therefore, a dynamic background can be found in nearby regions of the background model. The limited search space, besides reducing the computation time, also reduces the chance of foreground colors being wrongly regarded as background. A static background can be easily detected at close proximity. Figure 3 shows this spatial similarity search strategy. The current image frame is partitioned into non-overlapping blocks. Each pixel of a block is tentatively labeled as background or potential foreground by computing the similarity with each pixel of the search space in the background model. If the current pixel belongs to a static background, it is likely to find a match in the center block. If it belongs to a dynamic background, it may find a match in neighboring blocks. If it belongs to foreground, probably it never finds a match in the search space.

Due to complex background scene and wide ranging foreground colors, there are two problems in the first step of background subtraction. First, some background colors in the current image frame cannot find a match in the background model due to large change of colors resulted from motion or illumination change. Second, some foreground colors in the current image frame are wrongly regarded as background. The larger the search space, the higher chance the errors exist in the first step of background subtraction. To obtain a refined foreground region, we implement the second step of background subtraction. To reject the false positive errors, we examine the potential foregrounds detected in the first step. If the foreground pixels can cluster to form a sufficiently large region, the block of pixels enclosing that potential foreground is allowed to proceed to the second step of background subtraction. Therefore, scattered and small foreground regions can be eliminated. To reject the false negative errors, we limit the search space in the second step of background subtraction. As shown in Fig. 4, each pixel of the foreground containing block is classified as background or foreground by searching the background model at close proximity. This is to avoid the foreground color to find a false match in neighboring blocks of the background model. The algorithms of our foreground detection method are shown below. Detail of the implementation of TSBS is described in the following paragraphs.

Each image frame is partitioned into $n_1 \times n_2$ non-overlapping blocks $B_{a,b}$, where $1 \leq a \leq n_1$ and $1 \leq b \leq n_2$. The block size is the same as in the background model. In step 1, pixels are classified as potential target or background. For each pixel p in $B_{a,b}$, background models of the enclosing block and neighboring blocks are used in the similarity measure

Algorithm 1 – Step 1 of TSBS

```
Partition current image frame into blocks
For each pixel
    Label pixel as potential foreground
    Calculate similarity of pixel with each pixel of the search space in background model
    If similarity >= $DT_{far}$
        Label pixel as background
        Step over to the next pixel
```

Algorithm 2 – Step 2 of TSBS

```
For each block in current image frame
    If size of potential foreground > $N_{target}$
        For each pixel
            Label pixel as foreground
            Calculate similarity of pixel with each pixel of the same block in background model
            If similarity >= $DT_{near}$
                Label pixel as background
                Step over to the next pixel
```

$$S_{p,m,n} = \sum_{i=1}^{k} \omega_i \exp\left(-\left(\frac{p_i}{b_{m,n,i}} - 1\right)^2\right) \quad (2)$$

where p_i is the ith color component, ω_i is the weight on the ith color component, k is the number of color components, and $b_{m,n,i}$ is the ith component of a background color in block $B_{m,n}$, $m = a - 1{:}a + 1, n = b - 1{:}b + 1$. The maximum distance of neighboring blocks depends on how vigorous the background motion is. In our method, we set the distance to one block. The non-linear similarity measure makes the identification of true background pixels easier. If the pixel belongs to background, at least one background color is close and the corresponding similarity measure is large (near to 1). If the pixel is not a background, no background colors are close and all similarity measures are low. Our similarity measure is similar to the one used in [24]. In [24], the weights on color components depend on local variance. We have studied, using the Campus image sequence, the effect of the weights on color components. A stronger weight on H will produce better background subtraction result. Therefore, we fix the weights on HSV to 0.9, 0.05, 0.05, respectively. Other similarity measure, e.g., Eq. (6) in this paper, exhibits a linear relationship with the ratio of current pixel color to background color. If the current pixel color has 10 % deviation from the corresponding background color, our method can result in 10 % higher similarity than the linear method. Therefore, our similarity measure is more tolerable to change of background color which is common in dynamic scene. Thresholding of the similarity measure is governed by the parameter $\mathrm{DT}_{\mathrm{far}}$. p is first labeled as a potential foreground pixel. If $S_{p,m,n} \geq \mathrm{DT}_{\mathrm{far}}$, p is re-labeled as a background pixel and step over to the next pixel.

In step 2, each potential foreground region will be refined to produce the final foreground region. For each block, if the number of potential foreground pixels is sufficiently large, a genuine target may be present in this block. Otherwise, this block belongs to background. This filtering process aims to remove small and scattered foreground regions. Each pixel of the foreground containing block is compared with the background model of that block by a close-range similarity measure

$$S_p = \sum_{i=1}^{k} \omega_i \exp\left(-\left(\frac{p_i}{b_{a,b,i}} - 1\right)^2\right) \quad (3)$$

where $b_{a,b,i}$ is the ith component of a background color in the enclosing block $B_{a,b}$. k, p_i and ω_i are specified as in Eq. (2). If the pixel belongs to background, at least one background color is close and the corresponding similarity measure is large (near to 1). If the pixel is not a background, no background colors are close and all similarity measures are low. Thresholding of the similarity measure is governed by the parameter $\mathrm{DT}_{\mathrm{near}}$. p is first labeled as a foreground pixel. If $S_p \geq \mathrm{DT}_{\mathrm{near}}$, p is re-labeled as a background pixel and step over to the next pixel.

Traditional background modeling methods capture the movement of background elements via statistical measure of the background colors in the temporal domain. The adoption of a block-based scheme and the search for similar background colors in neighboring blocks in the first step of TSBS help to accommodate background motions as the reference background model is spatially extended. To tackle the temporal change of background colors caused by illumination change and specular reflection, we employ background model updating and HSV color space. To ensure a refined foreground detection, the background–foreground segregation is carried out in pixelwise manner. Also, as will be shown in the ROC analysis in Sect. 5.1, $\mathrm{DT}_{\mathrm{near}}$ is set a higher value to relax the foreground detection in the second step of TSBS.

4.2 Updating of background model

To perform background subtraction along a long image sequence, we need a background model updating process. An updated background frame is generated based on the background subtraction results. For each pixel location, we perform a backward search. From the background subtraction results and the original image frames, if the latest observed background color is similar to the existing color of the background frame by the following similarity measure, the new background color will replace the existing background color.

$$S_{x,y} = \sum_{i=1}^{k} \omega_i \exp\left(-\left(\frac{o_{x,y,i}}{b_{x,y,i}} - 1\right)^2\right) \tag{4}$$

where $b_{x,y,i}$ is the ith component of the existing background color at location (x, y), $o_{x,y,i}$ is the ith component of the latest observed background color at location (x, y). k and ω_i are specified as in Eq. (2). Otherwise, the existing color of the background frame remains unchanged. In our experimentation, the updating process is performed once every 10 image frames. The algorithm of background model updating is shown below.

In summary, TSBS can subtract the background, no matter whether it is static or dynamic. Assume the background frame contains no target colors, TSBS is able to detect stationary foregrounds, while other simple background subtraction methods such as interframe difference cannot. Compared with many other methods, TSBS can recover and identify covered background scene immediately. Since we adopt a block-based framework, TSBS can tolerate camera jitter. One limitation is that our method does not cater for large camera movements such as pan, tilt and zoom. TSBS employs no temporal coherence measure. Therefore, it can tackle sudden changes, such as entry and exit of targets.

5 Experimental results

Two parameters,$\mathrm{DT}_{\mathrm{far}}$ and $\mathrm{DT}_{\mathrm{near}}$, are critical to the performance of the foreground detection. Image frames are sampled from four testing image sequences. As will be explained in the following section, ROC analysis is carried out to determine the optimal values of these parameters. We test our method using image sequences containing dynamic background elements and compare with various background subtraction methods. Computation time and complexity of our method are presented.

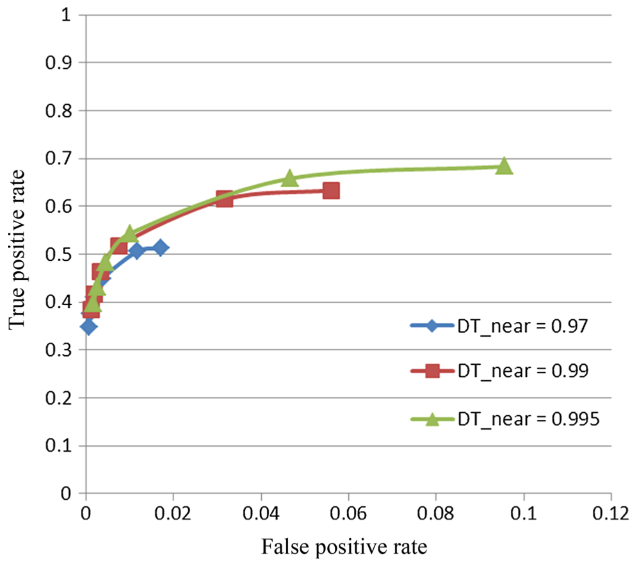

Fig. 5 ROC for various $\mathrm{DT}_{\mathrm{far}}$ and $\mathrm{DT}_{\mathrm{near}}$

5.1 Choice of parameter values

We would like to employ the same parameter values in our method on all testing image sequences. The block size should equal to a certain fraction of the image size. For large image frame size of 720×576 pixels or 768×576 pixels, we set the block size to 24×24. For small image frame size of 160×128 pixels, we set the block size to 8×4. N_{target}, as introduced in Sect. 4.1, is set between 10 and 30 % of the block size. For large block, N_{target} is set to 10 % of the block size as the actual size of the potential target is large enough. For small block, N_{target} is set to 30 % of the block size to allow a sufficiently large target to proceed to the second step of TSBS. To determine the optimal $\mathrm{DT}_{\mathrm{far}}$ and $\mathrm{DT}_{\mathrm{near}}$, we perform ROC analysis. 35 image frames are sampled from four image sequences. These image sequences are the same being used in the experimentation as described in Sect. 5.2. $\mathrm{DT}_{\mathrm{far}}$ is allowed to vary from 0.91 to 0.995. $\mathrm{DT}_{\mathrm{near}}$ is allowed to vary from 0.97 to 0.995. The ground truths of the sampled image frames are manually segmented. The average true positive rate and average false positive rate are computed. Figure 5 shows the ROC for various $\mathrm{DT}_{\mathrm{far}}$ and $\mathrm{DT}_{\mathrm{near}}$. We employ piecewise cubic interpolation to determine the knee point at $\mathrm{DT}_{\mathrm{far}} = 0.985$, $\mathrm{DT}_{\mathrm{near}} = 0.995$.

```
Algorithm 3 – Background model updating
Repeat
    Input the latest background subtraction result and the corresponding image frame
    For each pixel
        If it is classified as background and not updated
            Calculate similarity of observed background with existing background model
            If similarity > $DT_{far}$
                Update background model
```

5.2 Analyzing image sequences containing trees and water

In the first part of the evaluation, three background modeling algorithms: block-based mixture of Gaussians model, codebook and self-organizing artificial neural networks, are employed. We adopt and simplify the method of Eng et al. [17] for modeling local color information as block-based mixture of Gaussians (MoG). Each image frame is partitioned into $n_1 \times n_2$ non-overlapping blocks $B_{a,b}$, where $1 \le a \le n_1$ and $1 \le b \le n_2$. All pixels of the same block (a, b) over a short duration are collected and clustered via k-means into H homogeneous regions. Each homogeneous region of that block position is characterized by the mean and standard deviation of the Gaussian distribution, $\mu_{B^h_{a,b}} = \left\{ \mu^R_{B^h_{a,b}} \ \mu^G_{B^h_{a,b}} \ \mu^B_{B^h_{a,b}} \right\}^{\mathrm{T}}$ and $\sigma_{B^h_{a,b}} = \left\{ \sigma^R_{B^h_{a,b}} \ \sigma^G_{B^h_{a,b}} \ \sigma^B_{B^h_{a,b}} \right\}^{\mathrm{T}}$, respectively, where $1 \le h \le H$. Each block is clustered hierarchically until all regions have the compactness

$$K_{c=R,G,B} = \frac{\sigma^c_{B^h_{a,b}}}{\mu^c_{B^h_{a,b}}} \tag{5}$$

less than the tolerance or the number of regions reaches the allowable maximum. Clean background frames are generated from the original image sequence. Initial background model is generated from these clean background frames, also via k-means. In contrast to the method of Eng et al. [17], we do not model the foreground region and there is no special consideration for filtering specular reflection. The similarity of a pixel with respect to a homogeneous background region can be computed using the distance measure:

$$D(I_{x,y}|\mu_{B^h_{a,b}}, \sigma_{B^h_{a,b}}) = \sqrt{\sum_{c=R,G,B} \frac{\left(I^c_{x,y} - \mu^c_{B^h_{a,b}}\right)^2}{\left(\sigma^c_{B^h_{a,b}}\right)^2}} \tag{6}$$

If the distance of the current pixel with respect to any homogeneous background region is below a certain threshold, it is regarded as a background pixel. MoG, like TSBS, is a block-based background modeling plus a pixel-based background subtraction method. The difference in the modeling step is that MoG models the background colors as statistical parameters while TSBS uses the original background colors. In background subtraction MoG, only searches similar background colors within the enclosing block while TSBS performs the search first in a larger space and then refines the background subtraction result within the enclosing block.

Programs of the other two background subtraction methods are available on-line. The first one uses codebook (CB) [27] for modeling background.[1] At each pixel, samples collected from an image sequence are quantized into a set of codewords based on color distortion measure and brightness bounds. The model can handle illumination variations and moving backgrounds. To obtain good results, the two parameters Ep1 and Ep2 are automatically estimated from the image sequence and the parameter period is also tuned. CB is a pixel-based background modeling and subtraction method. It models temporal background changes while TSBS, with a spatially extended search space, allows for more vigorous background motions.

The last method uses self-organizing artificial neural networks (SOBS) [28].[2] Each pixel has a neuronal map with a set of weight vectors. The background model, as represented by the weight vectors, is trained by updating the weight vectors of the pixel and its neighbors according to a distance measure. The method can handle gradual illumination variations, moving backgrounds, camouflage and cast shadows. The method is also adjusted instead of using the default setting to obtain good background subtraction result in each image sequence. SOBS, like TSBS, also uses HSV color model. The weigh vectors, after updating, are synthetic colors while TSBS always uses original background colors. SOBS relies on a suitable neighborhood to tackle background motions while TSBS uses a fixed block distance to deal with background motions.

[1] http://www.umiacs.umd.edu/~knkim/UMD-BGS/index.html.

[2] http://www.na.icar.cnr.it/~maddalena.l/MODLab/SoftwareSOBS.html.

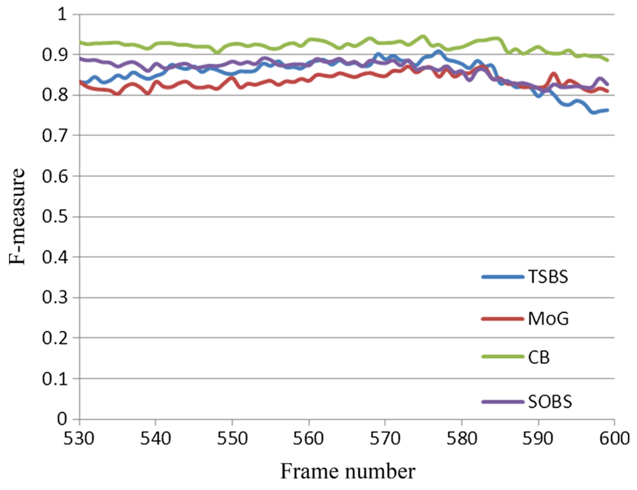

Fig. 6 F-measure versus image frame number of PETS 2001 image sequence

All testing methods are evaluated using the image sequence PETS 2001 dataset 1 camera 1,[3] Campus with wavering tree branches,[4] and two swimming pool image sequences. All methods are run on a 2.1 GHz PC with 1 Gbyte memory. Block-based MoG and TSBS are developed using MATLAB. We evaluate the methods quantitatively in terms of recall (Re), precision (Pr) and F-measure (F1). Recall gives the ratio of detected true positive pixels (TP) to total number of foreground pixels present in the ground truth which is the sum of true positive and false negative pixels (FN).

$$\mathrm{Re} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{7}$$

Precision gives the ratio of detected true positive pixels to total number of foreground pixels detected by the method which is the sum of true positive and false positive pixels (FP).

$$\mathrm{Pr} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{8}$$

F-Measure is the weighted harmonic mean of precision and recall. It can be used to rank different methods.

$$\mathrm{F1} = \frac{2 \times \mathrm{Pr} \times \mathrm{Re}}{\mathrm{Pr}+\mathrm{Re}} \tag{9}$$

The higher the value of the quantitative measure, the better is the accuracy.

In the first experiment, frames 530 to 599 are selected from the image sequence PETS 2001 dataset 1 camera 1. The image frame size is 768 × 576 pixels. It is a fairly easy image sequence and there is not much background motion. The main challenge is illumination change. Figure 6 shows the F-measure versus image frame number obtained by MoG, CB, SOBS and TSBS. Basically, all methods can detect the walker and car. CB produces false positive errors in the region of a window. MoG and SOBS produce more false negative errors. Table 1 shows the average values of the quantitative measures. TSBS achieves the highest precision and CB achieves the highest F-measure value.

Table 1 Average quantitative measures of the PETS 2001 image sequence

Method	Re	Pr	F1
MoG [17]	0.893	0.785	0.835
CB [27]	**0.953**	0.892	**0.922**
SOBS [28]	0.792	0.952	0.864
TSBS	0.756	**0.982**	0.853

The bold entries are the best results

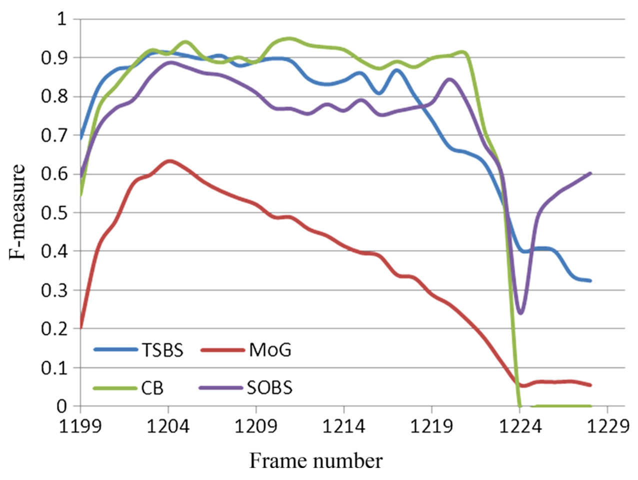

Fig. 7 F-measure versus image frame number of Campus image sequence part 1

The second image sequence Campus contains wavering tree branches. The image frame size is 160 × 128 pixels. In the first section, a van moves from right to left. Figure 7 shows the F-measure versus the image frame number from frames 1199 to 1228. MoG produces many false positive errors which are similar to the results presented in [9,12]. CB can tackle the trees but it cannot detect the foreground target in the last few frames. SOBS produces more false negative errors. TSBS obtains a fairly balance result. Table 2 shows the average values of the quantitative measures. Among all the testing methods, TSBS achieves the highest recall and F-measure.

In the second section, a man walks from right to left and a motorcycle moves from left to right. Figure 8 shows the F-measure versus the image frame number from frames 1304 to 1403. MoG produces more scattered false positive errors. CB produces some clustered regions of false positive errors

[3] ftp.pets.rdg.ac.uk/PETS2001.

[4] http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html.

Table 2 Average quantitative measures of the Campus image sequence part 1

Method	Re	Pr	F1
MoG [17]	0.747	0.271	0.360
CB [27]	0.669	0.781	0.719
SOBS [28]	0.639	**0.863**	0.730
TSBS	**0.852**	0.700	**0.743**

The bold entries are the best results

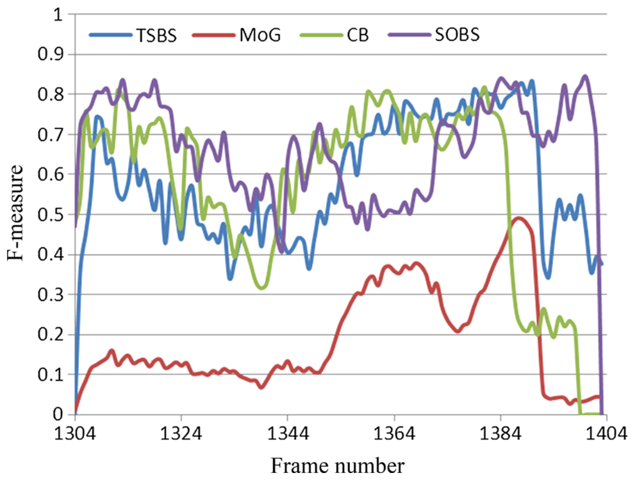

Fig. 8 F-measure versus image frame number of Campus image sequence part 2

Table 3 Average quantitative measures of the Campus image sequence part 2

Method	Re	Pr	F1
MoG [17]	0.624	0.121	0.190
CB [27]	**0.779**	0.477	0.563
SOBS [28]	0.554	**0.865**	**0.663**
TSBS	0.683	0.526	0.584

The bold entries are the best results

in the trees and banner. It cannot detect the foreground target in the last few frames. SOBS produces more false negative errors. TSBS obtains a fairly balance result. Table 3 shows the average values of the quantitative measures.

The first swimming pool image sequence captures some swimmers. The image frame size is 720 × 576 pixels. In the first section, the swimmer at the center swims away from the camera in freestyle. Another swimmer on the right swims slowly towards the camera in freestyle and appears very small in each image frame. Figure 9 shows the F-measure versus the image frame number from frames 1201 to 1300. Many background points, especially in the water ripples and splashes nearby the center swimmer, are erroneously regarded as foreground by MoG. CB produces more false negative errors. SOBS produces more false positive errors. TSBS can detect the center swimmer but not the right swimmer. There are

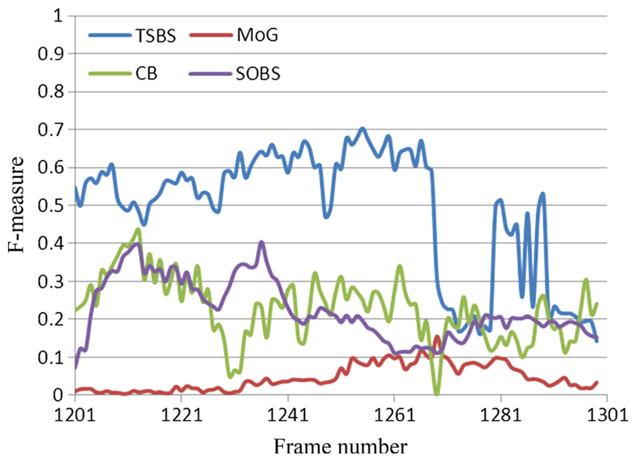

Fig. 9 F-measure versus image frame number of the first swimming pool image sequence part 1

Table 4 Average quantitative measures of the first swimming pool image sequence part 1

Method	Re	Pr	F1
MoG [17]	0.224	0.025	0.045
CB [27]	0.205	0.347	0.229
SOBS [28]	0.287	0.189	0.225
TSBS	**0.431**	**0.661**	**0.487**

The bold entries are the best results

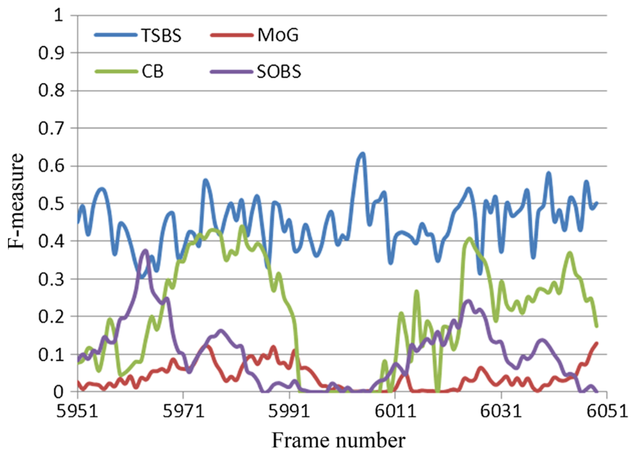

Fig. 10 F-measure versus image frame number of the first swimming pool image sequence part 2

much fewer false positive errors. Table 4 shows the average values of the quantitative measures. Among all the testing methods, TSBS achieves the highest recall, precision and F-measure.

In the second section, there is only one swimmer moving away from the camera in freestyle. Figure 10 shows the F-measure versus the image frame number from frames 5951 to 6050. MoG can only detect a very small part of the swimmer. CB cannot detect the swimmer in many image frames. SOBS

Table 5 Average quantitative measures of the first swimming pool image sequence part 2

Method	Re	Pr	F1
MoG [17]	0.038	0.047	0.040
CB [27]	0.159	0.346	0.203
SOBS [28]	0.061	**0.386**	0.101
TSBS	**0.893**	0.306	**0.451**

The bold entries are the best results

Table 6 Average quantitative measures of the second swimming pool image sequence part 1

Method	Re	Pr	F1
MoG [17]	0.347	0.262	0.290
CB [27]	0.169	0.369	0.188
SOBS [28]	0.131	0.240	0.161
TSBS	**0.383**	**0.952**	**0.517**

The bold entries are the best results

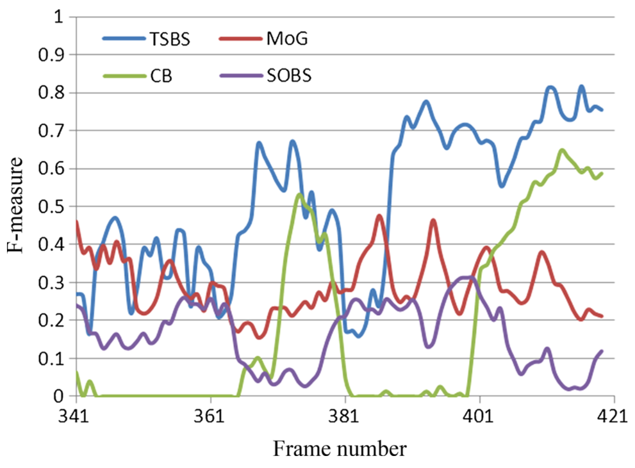

Fig. 11 F-measure versus image frame number of the second swimming pool image sequence part 1

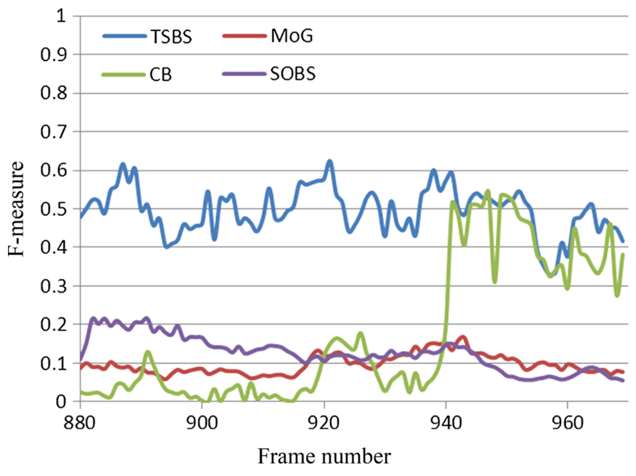

Fig. 12 F-measure versus image frame number of the second swimming pool image sequence part 2

Table 7 Average quantitative measures of the second swimming pool image sequence part 2

Method	Re	Pr	F1
MoG [17]	0.235	0.061	0.097
CB [27]	0.173	0.647	0.175
SOBS [28]	0.214	0.093	0.126
TSBS	**0.334**	**0.966**	**0.494**

The bold entries are the best results

produces fewer false positive errors but the detected swimmer region is also small. TSBS can detect a very good shape of the swimmer. Table 5 shows the average values of the quantitative measures. Among all the testing methods, TSBS achieves the highest recall and F-measure.

The second swimming pool image sequence captures some swimmers and the stationary lifeguard. The image frame size is 720 × 576 pixels. There is camera jitter problem in this image sequence. In the first section, the swimmer at the center swims quickly towards the camera in butterfly. The swimmer on the right swims slowly in freestyle towards the camera. Figure 11 shows the F-measure versus the image frame number from frames 341 to 420. MoG can only detect a small part of the center swimmer and the right swimmer is basically undetected. CB cannot detect the center swimmer in many frames. SOBS detects part of the center swimmer and there are many false positive errors in the sunshade and lane dividers. TSBS can detect a fairly good shape of both swimmers. Table 6 shows the average values of the quantitative measures. Among all the testing methods, TSBS achieves the highest recall, precision and F-measure.

In the second section, the swimmer at the center swims away from the camera in backstroke. The swimmer on the right swims slowly in freestyle towards the camera. Figure 12 shows the F-measure versus the image frame number from frames 880 to 970. MoG can only detect a small part of both swimmers and there are many false positive errors. CB also detects a small part of the swimmers. SOBS can detect the swimmers but there are many false positive errors in the sunshade, swimming pool walkway and lane dividers. TSBS can detect a fairly good shape of both swimmers. Table 7 shows the average values of the quantitative measures. Among all the testing methods, TSBS achieves the highest recall, precision and F-measure.

Figures 13 and 14 show some visual results from the four image sequences. The first row shows some original image frames. The second row shows the corresponding hand segmented ground truths. Rows 3 to 5 are the background subtraction results obtained by MoG, CB, and SOBS, respectively. The last row shows the background subtraction results obtained by our method TSBS.

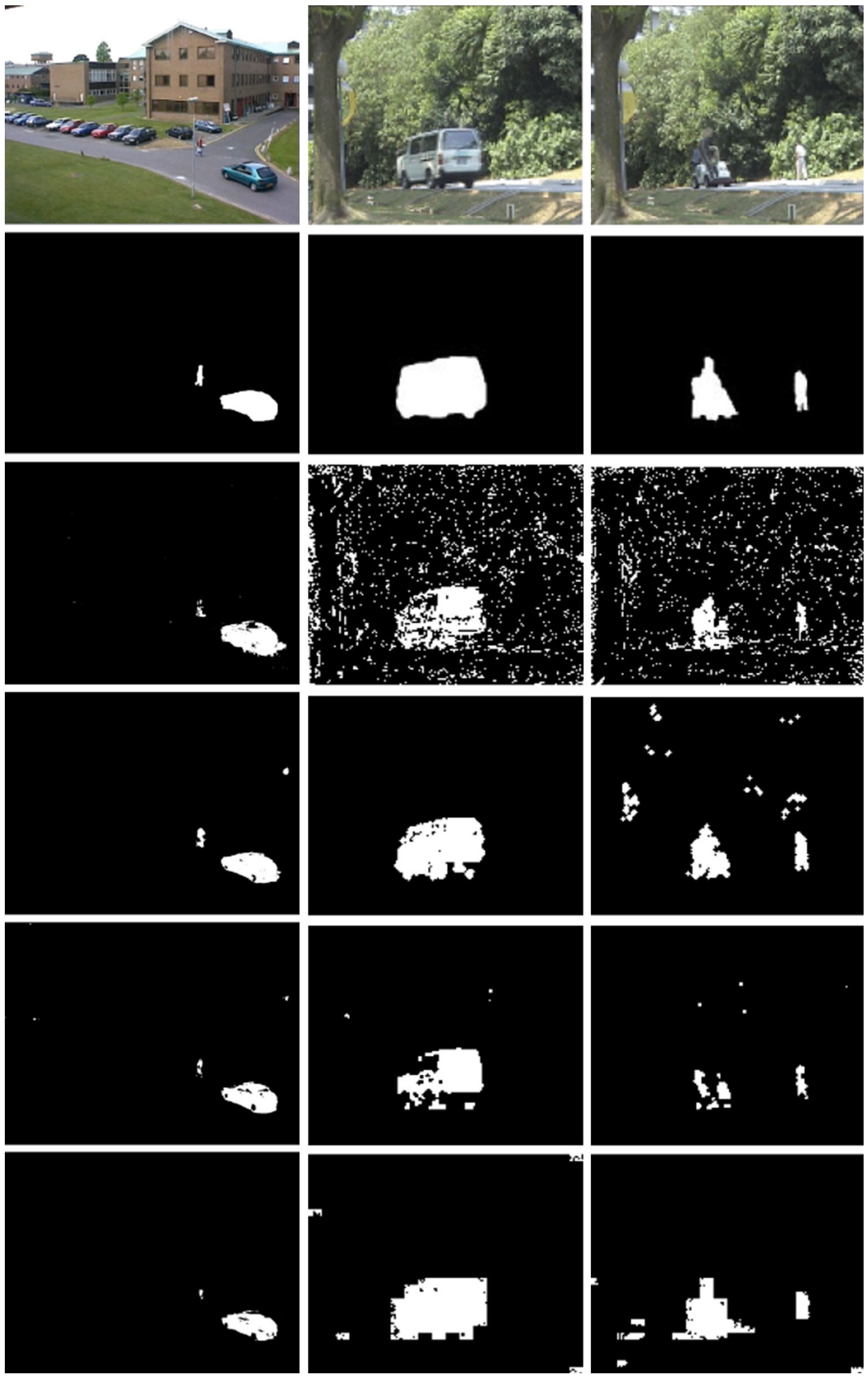

Fig. 13 Background subtraction results from PETS 2001 (*first column*), Campus part 1 (*second column*), Campus part 2 (*third column*), original image frames (*top row*), ground truths (*second row*), results obtained by block-based MoG (*third row*), results obtained by CB (*fourth row*), results obtained by SOBS (*fifth row*), results obtained by TSBS (*last row*)

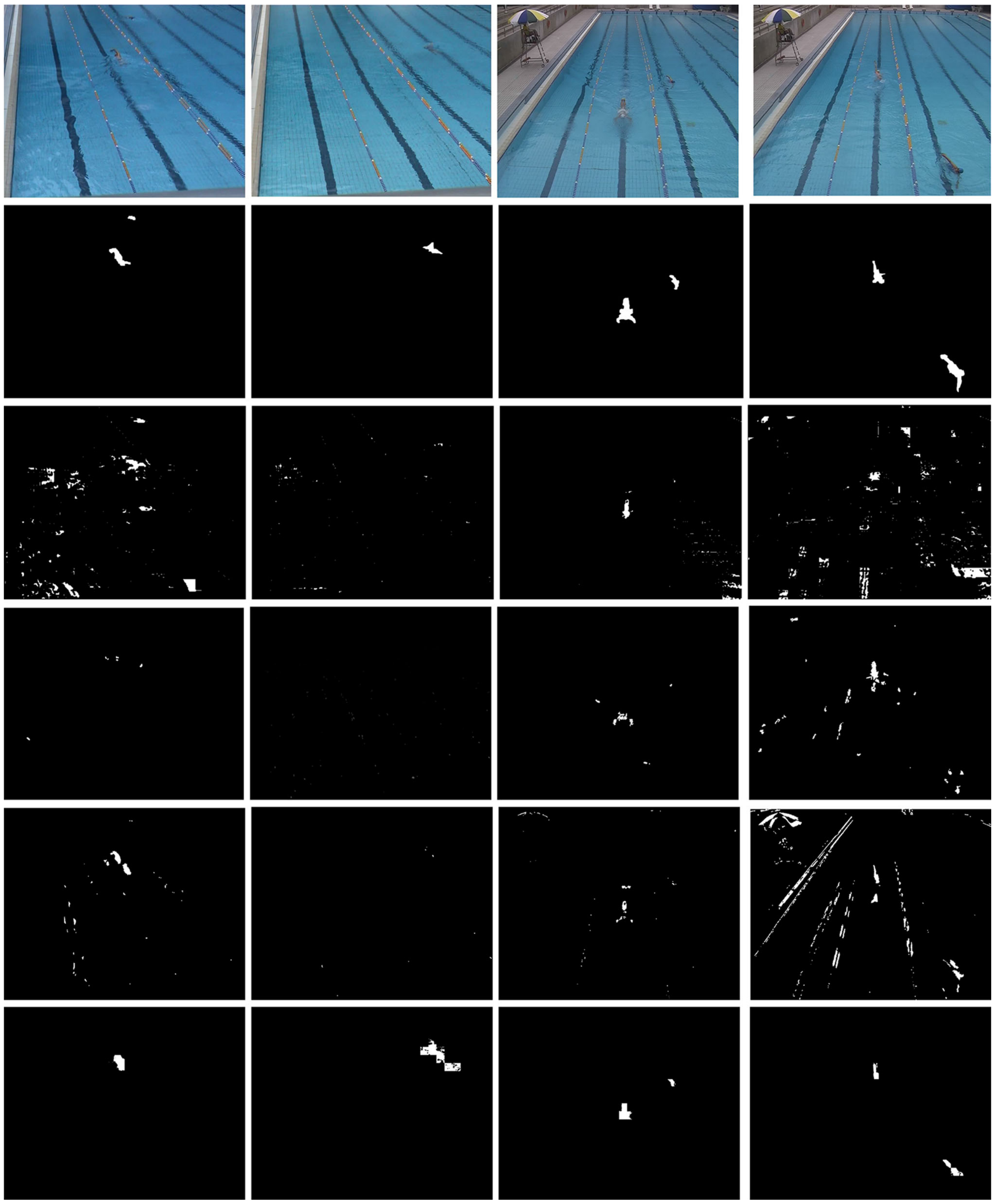

Fig. 14 Background subtraction results from the first swimming pool image sequence part 1 (*first column*), part 2 (*second column*), the second swimming pool image sequence part 1 (*third column*), part 2 (*fourth column*), original image frames (*top row*), ground truths (*second row*), results obtained by block-based MoG (*third row*), results obtained by CB (*fourth row*), results obtained by SOBS (*fifth row*), results obtained by TSBS (*last row*)

Table 8 Results of TSBS on the dynamic background image sequences

Image sequence	Re	Pr	F1	FPR	FNR	PWC	Sp
Boats	0.5441	0.5056	**0.5241**	0.0034	0.4559	0.6196	0.9966
Canoe	0.9402	0.8936	**0.9163**	0.0041	0.0598	0.6082	0.9959
Fall	0.4992	0.6845	**0.5773**	0.0042	0.5008	1.2949	0.9958
Fountain01	0.1312	0.0391	0.0603	0.0027	0.8688	0.3396	0.9973
Fountain02	0.8459	0.6263	**0.7197**	0.0011	0.1541	0.1417	0.9989
Overpass	0.8723	0.3149	0.4628	0.0258	0.1277	2.7133	0.9742

The bold entries are the best results

Table 9 Results of VIBE on the dynamic background image sequences

Image sequence	Re	Pr	F1	FPR	FNR	PWC	Sp
Boats	0.5277	0.1074	0.1784	0.0196	0.4723	2.1594	0.9804
Canoe	0.8974	0.6940	0.7827	0.0138	0.1026	1.6782	0.9862
Fall	0.8325	0.3415	0.4844	0.0355	0.1675	3.8338	0.9645
Fountain01	0.5803	0.0323	**0.0613**	0.0076	0.4197	0.7795	0.9924
Fountain02	0.8215	0.4282	0.5629	0.0015	0.1785	0.1691	0.9985
Overpass	0.7983	0.6003	**0.6853**	0.0053	0.2017	0.7254	0.9947

The bold entries are the best results

Table 10 Average results of TSBS and other background subtraction methods

Method	Re	Pr	F1	FPR	FNR	PWC	Sp
SOBS [28]	**0.8798**	0.5856	0.6439	0.0157	**0.1202**	1.6367	0.9843
VIBE [29]	0.7430	0.3673	0.4592	0.0139	0.2571	1.5576	0.9861
PBAS [30]	0.6955	**0.8326**	**0.6829**	**0.0011**	0.3045	**0.5394**	**0.9989**
TSBS	0.6388	0.5107	0.5434	0.0069	0.3612	0.9529	0.9931

The bold entries are the best results

5.3 Analyzing image sequences of change detection challenge

In the second part of the evaluation, two more background subtraction methods are employed. Barnich and Van Droogenbroeck [29] proposed a random policy to select pixels for background modeling. They called the method visual background extractor (VIBE). Hofmann et al. [30] proposed a similar background subtraction method. They called the method pixel-based adaptive segmenter (PBAS). In contrast to VIBE which has fixed values for randomness parameters and decision threshold, PBAS allows parameters to be adaptively adjusted at runtime.

All testing methods are evaluated using the change detection[5] 2012 dataset, category of dynamic background [31]. There are six image sequences (boats, canoe, fall, fountain01, fountain02, overpass). The ground truth images contain five labels (static, hard shadow, outside region of interest, unknown motion, motion). When we evaluate TSBS and VIBE, we count TP, TN, FP, FN in the way as suggested in the website of change detection. Besides the three quantitative measures as mentioned in Sect. 5.2, we also calculate specificity (Sp), false positive rate (FPR), false negative rate (FNR), percentage of wrong classifications (PWC). The numeric results of PBAS and SOBS are obtained from the website of change detection. As for our method, we use the same values of DT_{far}, DT_{near}, N_{target} as mentioned in Sect. 5.1. The block size is set equal to a certain fraction of the image size. For image frame size of 720×480 pixels, we set the block size to 24×24. For image frame size of 432×288 pixels, we set the block size to 24×8. For image frame size of 320×240 pixels, we set the block size to 16×8.

Table 8 shows the results of TSBS on the six image sequences of dynamic background. Table 9 shows the results of VIBE on the six image sequences of dynamic background. Table 10 shows the average results of TSBS and other background subtraction methods on the dynamic background image sequences. Figure 15 shows some visual results from the six image sequences. The first column shows some original image frames and the results obtained by VIBE. The second column shows the results obtained by our method TSBS. The third column shows the corresponding ground truths.

[5] http://www.changedetection.net.

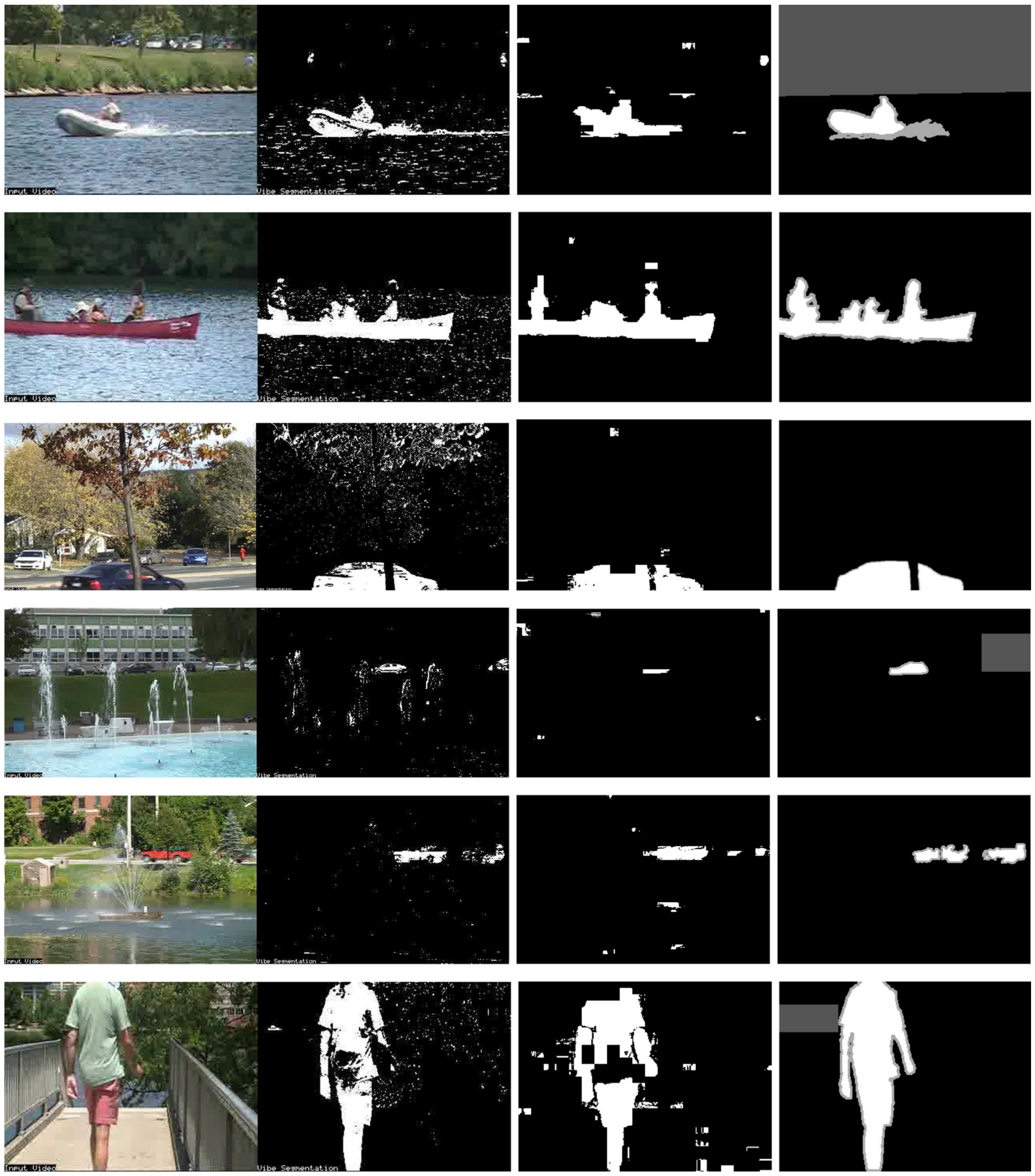

Fig. 15 Background subtraction results from the change detection dataset category of dynamic background—original image frames and results obtained by VIBE (*first column*), results obtained by TSBS (*second column*), ground truths (*last column*)

5.4 Computation time and complexity analysis of TSBS

The computation time depends on computing platform, software implementation, image frame size, block size, as well as the contents of the image. We present the number of operations involved in background model initialization, background subtraction and background model updating. In background model initialization, if the background frame is generated by vector median filtering, 10 image frames are used and so complexity is $O(10)$. In most image sequence, we can select one image frame containing no foreground objects as background frame. The computation complexity is only $O(1)$. In the first step of background subtraction, each pixel searches for the similar background color in the search space in background frame. To speed up the background subtraction, once the new pixel matches with a background pixel, the search will stop. For static background pixel, a minimal one similarity calculation may be required. For foreground pixel, it searches the whole search space and the number of similarity calculations is 9 × block size. The complexity ranges from $O(1)$ to $O(9b_1b_2)$ where $b_1 \times b_2$ is the block size. In the second step of background subtraction, each pixel of the foreground block searches for the similar background color within the block in background frame. Static background pixel may require one similarity calculation. For foreground pixel, the number of similarity calculations equals to block size. The complexity ranges from $O(1)$ to $O(b_1b_2)$. The background model updating process is performed once every 10 image frames. It involves a backward search and similarity calculation. At each pixel location, if all 10 background subtraction results are labeled as foreground, there is no updating. If the latest result is labeled as background and its color is similar to the existing color of the background frame, one similarity calculation is required. If all 10 results are labeled as background but only the earliest pixel color is similar to the existing background model color, 10 similarity calculations are involved. The complexity ranges from $O(1)$ to $O(10)$.

TSBS is developed and run on a 2.1 GHz PC with 1 Gbyte memory using MATLAB. For a low-resolution image of 320 × 240 pixels and block size of 16 × 8 (set in relation to image size), the computation time per image frame is about 17.7 s. The computation time is reduced to about 3.8 s using a smaller block size of 4 × 4. For an image of 432 × 288 pixels and block size of 24 × 8, the computation time is about 83.6 s. For an image of 720 × 480 pixels and block size of 24 × 24, the computation time is about 191.2 s. To reduce the computation time, the method will be implemented in C language and executed on a GPU board in the future.

6 Discussion

In the experimentation, we use the same set-up for TSBS on all image sequences to illustrate the generality of our method. The block size and N_{target} are automatically fixed with respect to the image frame size. The remaining two critical parameters DT_{far} and DT_{near} are estimated by ROC analysis. We also test our method using a fixed block size of 4 × 4. We observe that with large block size, scattered false positive errors are reduced but the shape of the detected foreground is not as good as that obtained with small block size. With block size fixed with respect to the image frame size, recall is lower, precision and F-measure are higher than a small block size of 4 × 4. For the reference methods, MoG has the same set-up but CB and SOBS can tune their parameters automatically for each image sequence. The results obtained by TSBS are very good. In many cases, it achieves the highest quantitative measure values. TSBS only demands one background frame for modeling the background motions in the spatial domain. To tackle the temporal change of background colors, we employ the HSV color space. This is in contrast with other methods that model the dynamic background in the temporal domain, e.g., the pixelwise Gaussian mixture model. Our background model is easy to compute and update. In addition, our background frame contains original background colors which are better than the believed-to-be background statistical parameters.

The PETS 2001 image sequence captures a very stable background scene and is much easier to process than other image sequences in our experimentation. CB produces false positive errors which may be due to illumination changes (in the windows of buildings) and camera jitter. SOBS produces more false negative errors, probably because windows of the moving car have similar colors to the road. As shown in Table 1, all methods achieve similarly high quantitative measure values. The visual results shown in Fig. 13 illustrate that TSBS has very few false positive errors that lead to its highest precision among all methods.

The Campus with wavering tree branches image sequence is more complicated. Besides the waving banner, the large background region of trees is very difficult to tackle. MoG certainly demands some post-processing to obtain a clean foreground. In the first section, CB also cannot detect the reappearance of the van when it moves behind the tree. In the second section, it produces more false positive errors. SOBS generally produces more false negative errors. The detected foreground is the smallest among all methods. TSBS obtains a fairly balance result. It never misses the targets and they are detected in fairly good shape in every image frame.

We test the methods using two swimming pool image sequences capturing different swimming styles at different viewpoints. It can be seen that MoG, CB and SOBS may erroneously identify water ripples, lane dividers, sunshade, etc., as foregrounds or detect only part of the swimmer. The false positive errors are due to the moving background scene, camera jitter and specular reflection. The false negative errors are due to the existence of colors similar to the foreground in the background model. They will lead to a lower precision or recall. Either one can result in a lower F-measure. In some image frames, in particular when CB is used, the swimmers are totally undetected. TSBS can always detect the swimmers, especially when the body is above the water surface. It achieves the highest F-measure in all swimming pool image sequences, indicating that the method can strike a good balance between true foreground and errors. One limitation of TSBS is that when the potential target region found in the first step spans across blocks, the final foreground detected will be small because of the small size of the potential target within a block. In the future, rectification can be implemented to allow neighboring blocks containing the same potential target to proceed to the second step of foreground detection.

Finally, we compare our method with VIBE using the change detection dynamic background image sequences. We also quote the numeric results of PBAS and SOBS. Our method is not optimized for the dataset. Compared with VIBE, TSBS still can achieve higher F-measure in four out of six dynamic background image sequences. Both methods achieve very low precision in fountain01 image sequence due to small size of the moving car. TSBS also achieve low recall due to large block size. We find that with a block size of 4×4, TSBS can achieve higher recall, precision and F-measure than VIBE. TSBS achieves lower precision than VIBE in overpass image sequence. It is due to some foreground blocks are excluded in the second step of background subtraction and more false positive errors in the fence. We use the method as suggested in the website of change detection to calculate the average ranking. TSBS achieves 26.57 which is better than VIBE of 31. According to the ranking table in the website of change detection, the average rankings of PBAS and SOBS are 15.43 and 23.00, respectively. The performance of TSBS is close to SOBS and better than the nonparametric method of kernel density estimation [32] of 27.29.

7 Conclusion

We develop a method for the detection of foreground in a scene containing vigorous motions via a two-step dynamic background subtraction. A background frame is generated from the original image sequence, either by selecting one image frame containing no foreground or by vector median filtering. The background frame will be updated along the image sequence based on the background subtraction results. In the first step of background subtraction, static and dynamic background pixels are rejected while the potential targets are identified. This initial segregation is achieved by referencing all the background colors within a certain range. In the second step of background subtraction, each sufficiently large target is checked with the background model proximally to obtain a refined shape of the true foreground. After performing ROC analysis, we can use the same set- up for our method on all testing image sequences to illustrate the generality of our method. We compare our method with various background subtraction methods using our image sequences as well as publicly available change detection dataset. Quantitative measures are employed to evaluate the methods. Our method can detect the foreground, no matter whether the background scene has illumination changes, or exhibits vigorous motions such as waving trees and moving water.

Acknowledgments The author would like to thank the reviewers for their comments and suggestions.

References

1. Hsieh, J.-W., Hsu, Y.-T., Liao, H.-Y.M., Chen, C.-C.: Video-based human movement analysis and its application to surveillance systems. IEEE Trans. Multimed. **10**(3), 372–384 (2008)
2. Cunado, D., Nixon, M.S., Carter, J.N.: Automatic extraction and description of human gait models for recognition purposes. Comput. Vis. Image Underst. **90**, 1–41 (2003)
3. Lu, C.M., Ferrier, N.J.: Repetitive motion analysis: segmentation and event classification. IEEE Trans. Pattern Anal. Mach. Intell. **26**(2), 258–263 (2004)
4. Bouwmans, T.: Recent advanced statistical background modeling for foreground detection: a systematic survey. Recent Pat. Comput. Sci. **4**(3), 147–176 (2011)
5. Sobral, A., Vacavant, A.: A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. Comput. Vis. Image Underst. **122**, 4–21 (2014)
6. Ning, H., Tan, T., Wang, L., Hu, W.: Kinematics-based tracking of human walking in monocular video sequences. Image Vis. Comput. **22**, 429–441 (2004)
7. Li, L., Huang, W., Gu, I.Y.-H., Tian, Q.: Statistical modeling of complex backgrounds for foreground object detection. IEEE Trans. Image Process. **13**(11), 1459–1472 (2004)
8. Stauffer, C., Grimson, W.E.L.: Learning patterns of activity using real-time tracking. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 747–757 (2000)
9. Zha, Y., Bi, D., Yang, Y.: Learning complex background by multiscale discriminative model. Pattern Recognit. Lett. **30**, 1003–1014 (2009)
10. Liao, S., Zhao, G., Kellokumpu, V., Pietikäinen, M., Li, S.Z.: Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1301–1306 (2010)
11. Ko, T., Soatto, S., Estrin, D.: Warping background subtraction. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1331–1338 (2010)
12. El Baf, F., Bouwmans, T., Vachon, B.: Type-2 fuzzy mixture of Gaussians model: application to background modeling. In: Proceedings of International Symposium on Visual Computing. LNCS, vol. 5358, Part I, pp. 772–781 (2008)
13. Lu, W., Tan, Y.P.: A vision-based approach to early detection of drowning incidents in swimming pools. IEEE Trans. Circuits Syst. Video Technol. **14**(2), 159–178 (2004)
14. Kam, A.H., Lu, W., Yau, W.-Y.: A video-based drowning detection system. In: Proceedings of European Conference on Computer Vision. LNCS, vol. 2353, pp. 297–311 (2002)
15. Eng, H.-L., Toh, K.-A., Kam, A.H., Wang, J., Yau, W.-Y.: An automatic drowning detection surveillance system for challenging outdoor pool environments. In: Proceedings of IEEE International Conference on Computer Vision (2003)
16. Eng, H.-L., Wang, J., Kam, A.H., Yau, W.-Y.: Novel region-based modeling for human detection within highly dynamic aquatic environment. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2004)

17. Eng, H.-L., Wang, J., Kam, A.H., Yau, W.-Y.: Robust human detection within a highly dynamic aquatic environment in real time. IEEE Trans. Image Process. **15**(6), 1583–1600 (2006)
18. Wang, J., Eng, H.-L., Kam, A.H., Yau, W.-Y.: Integrating color and motion to enhance human detection within aquatic environment. In: Proceedings of IEEE International Conference on Multimedia and Expo, pp. 1179–1182 (2004)
19. Zhou, D., Zhang, H.: Modified GMM background modeling and optical flow for detection of moving objects. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, pp. 2224–2229 (2005)
20. Tang, P., Gao, L., Liu, Z.: Salient moving object detection using stochastic approach filtering. In: Proceedings of International Conference on Image and Graphics, pp. 530–535 (2007)
21. Doretto, G., Chiuso, A., Wu, Y.N., Soatto, S.: Dynamic textures. Int. J. Comput. Vis. **51**(2), 91–109 (2003)
22. Chetverikov, D., Fazekas, S., Haindl, M.: Dynamic texture as foreground and background. Mach. Vis. Appl. **22**(5), 741–750 (2011)
23. Fazekas, S., Amiaz, T., Chetverikov, D., Kiryati, N.: Dynamic texture detection based on motion analysis. Int. J. Comput. Vis. **82**, 48–63 (2009)
24. Traver, V.J., Mirmehdi, M., Xie, X., Montoliu, R.: Fast dynamic texture detection. In: Proceedings of European Conference on Computer Vision. LNCS, vol. 6314, pp. 680–693 (2010)
25. Chan, K.L.: Detection and decomposition of foreground target from image sequence. In: Proceedings of the 13th IAPR Conference on Machine Vision Applications, pp. 459–462 (2013)
26. Haque, M., Murshed, M.: Robust background subtraction based on perceptual mixture-of-Gaussians with dynamic adaptation speed. In: Proceedings of IEEE International Conference on Multimedia and Expo Workshops, pp. 396–401 (2012)
27. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.S.: Real-time foreground–background segmentation using codebook model. Real-Time Imaging **11**, 172–185 (2005)
28. Maddalena, L., Petrosino, A.: A self-organizing approach to background subtraction for visual surveillance applications. IEEE Trans. Image Process. **17**(7), 1168–1177 (2008)
29. Barnich, O., Van Droogenbroeck, M.: VIBE: a powerful random technique to estimate the background in video sequences. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 945–948 (2009)
30. Hofmann, M., Tiefenbacher, P., Rigoll, G.: Background segmentation with feedback: the pixel-based adaptive segmenter. In: Proceedings of IEEE Workshop on Change Detection at CVPR-2012, pp. 38–43 (2012)
31. Goyette, N., Jodoin, P.-M., Porikli, F., Konrad, J., Ishwar, P.: changedetection.net: a new change detection benchmark dataset. In: Proceedings of IEEE Workshop on Change Detection at CVPR-2012, pp. 16–21 (2012)
32. Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.S.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proc. IEEE **90**(7), 1151–1163 (2002)

K. L. Chan received his M.Sc. degree in Electronics from the University of Wales Institute of Science and Technology, UK and Ph.D. degree from the University of Wales College of Medicine, UK. He is currently an Assistant Professor of the Department of Electronic Engineering, the City University of Hong Kong. His research interests include image processing and computer vision.