Computer Vision and Image Understanding 152 (2016) 58–66

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Adaptive maintenance scheme for codebook-based dynamic background subtraction

Zhi Zeng*, Jianyuan Jia, Zhaofei Zhu, Dalin Yu

School of Mechano-electronics Engineering, Xidian University, Taibai Road, Xi'an 710071, China

ARTICLE INFO

Article history:
Received 18 December 2015
Revised 18 August 2016
Accepted 19 August 2016
Available online 21 August 2016

Keywords:
Background subtraction
Arbitrary cylindrical color model
Codebook model
Adaptive scheme

ABSTRACT

We propose a novel adaptive maintenance scheme for the codebook-based background subtraction algorithm. With this technique, the accuracy and efficiency of the model are significantly improved. In the proposed method, we develop an equal-qualification updating strategy to replace the maximum-negative-run-length-based filtering strategy. Further, we substitute the cache-based foreground learning process with a random updating scheme. These modifications not only preserve the accuracy of the codebook model but also significantly reduce the number of parameters used in the maintenance scheme. In the modified framework, parameters that are scenario-sensitive are identified through extensive experiments and analysis. Then, adaptive methods are proposed for them. The proposed method ensures the best performance of the system across a variety of complex scenarios. In our experiments, comparisons are provided to confirm that the performance of the codebook model is significantly improved owing to the adaptive technique. The overall performance of the proposed method is evaluated against more than 20 state-of-the-art methods using several modern datasets. It is demonstrated that, despite using only color information, the proposed method outperforms the majority of the solutions by a significant margin.

1. Introduction

Background subtraction is a fundamental step in the majority of computer vision applications including intelligent visual surveillance (Unzueta et al., 2012), intelligent visual observation of animals and insects (Mashak and Hosseini, 2012), optical motion capture (Guerra-filho, 2005), human-machine interaction (Senior et al., 2010), and content-based video coding (Paul et al., 2013). The purpose of a background subtraction algorithm is to distinguish moving objects from the scene.

Background subtraction is a challenging task because a real background in nature is not static. For example, a real background may consist of objects with repetitive motion, such as waving trees or grasses. Such a background has multiple appearances (Stauffer and Grimson, 1999). In this example, if we monitor a pixel of the charge-coupled device (CCD) sensor, we may observe several clusters of samples locate at different regions in the red, green, and blue (RGB) color space. Therefore, a basic background modeling technique such as running average (Lee and Hedley, 2002) or single Gaussian model (Wren et al., 1997) cannot model such a background effectively. To solve this problem, the Gaussian mixture model (GMM) (Stauffer and Grimson, 1999) was proposed where a mixture of Gaussian probability density functions is used for modeling each pixel. This method has been widely used and improved in terms of accuracy and robustness (Lee, 2005; Porikli and Tuzel, 2005; Zivkovic and Van Der Heijden, 2006). Many other nonparametric models have been developed to address this challenge, including Kernel density estimation (KDE) (Elgammal et al., 2002), the codebook model (Kim et al., 2005), and consensus-based model (Barnich and Van Droogenbroeck, 2011).

A real background may also contain surfaces with complex illumination variations such as shadows, highlights, and rippling water. For such a background, both direct and diffuse light contribute to the background illumination and the intensity of illumination may change quickly and drastically (Kim et al., 2005). To address this challenge, background features that tolerate complex illumination variations have been developed, including the gradient (Javed et al., 2002), edge (Jain et al., 2007), cylindrical color model (Kim et al., 2005), and local binary pattern (Heikklä et al., 2006).

Another challenge is that the dynamics of a background are not the same in different applications or even different regions of the same scene. This leads to inconsistent performance of a segmentation algorithm across different scenes or across different regions of the same scene (St-Charles et al., 2015a). Therefore, the majority of the methods can provide acceptable results on a specific sequence when adjusted accordingly. However, such adjustments lack flexibility and cannot address backgrounds consisting of regions of different dynamics effectively. To address this problem, several attempts have been made to adaptively adjust parameters

* Corresponding author.
E-mail address: zengzhi_2012@hotmail.com (Z. Zeng).

http://dx.doi.org/10.1016/j.cviu.2016.08.009
1077-3142/© 2016 Elsevier Inc. All rights reserved.

according to the dynamics of the background (St-Charles et al., 2015a; Hofmann et al., 2012).

Even after many attempts, existing background segmentation techniques continue to have difficulty with these problems (Sobral and Vacavant, 2014; Bouwmans, 2014). A statistical report from a recent benchmark dataset indicates that dynamic background subtraction remains one of the most challenging tasks (Goyette et al., 2014). Therefore, how to develop an accurate, robust, and concise background model and its maintenance scheme is an open problem and attracts considerable attention. For a summary of related works, one can refer to several surveys (Sobral and Vacavant, 2014; Bouwmans, 2014; Piccardi, 2004; Radke et al., 2005; Nascimento and Marques, 2006; Brutzer et al., 2011).

As a well-known cluster model, the codebook-based background segmentation approach offers many advantages over traditional methods, such as its ability to work with illumination changes and process multi-appearance backgrounds; it also incurs less memory cost. The classical codebook model (Kim et al., 2005) has several distinguishing features including the cylindrical color model, codebook structure, and maximum negative run length-controlled filter process. During the past decade, many works have been dedicated to the improvement of the codebook model (Doshi and Trivedi, 2006; Guo et al., 2011; Zaharescu and Jamieson, 2011; Hu et al., 2012; Guo et al., 2013; Zeng and Jia, 2014). The classical codebook model uses a cylindrical color model that is developed based on the observation that under lightening variations, pixel values are almost totally distributed in an elongated shape along the axis going toward the origin of the RGB color space. This color model uses cylinders whose axes point to the origin of the RGB space to model the distribution of the pixel value. The geometrical parameters of this color model are the center, color distortion, and intensity bounds. A low-pass filter is used to update the center and the color distortion. The bound is updated using the maximum criteria. Incorrectly learned backgrounds are deleted using a maximum-run-length technique. This technique is developed under the assumption that an appearance that disappears for a long time should not be used for background modeling. Finally, new backgrounds are learned using a cache mechanism. When a foreground model stays in the cache sufficiently long, it is moved from the cache to the background model.

The codebook model involves many parameters, especially when multi-feature and multi-scale methods are used (Guo et al., 2011; Zaharescu and Jamieson; 2011). A common approach to tune these parameters is to optimize them according to an estimation function with respect to several training sequences. However, because some of these parameters are scenario sensitive, one must choose training sequences according to the application carefully (Kim et al., 2005; Guo et al., 2011). Further, such a video-level parameter-tuning scheme cannot address a background consisting of regions with highly different dynamics. Therefore, a robust and accurate pixel-level adaptive parameter-adjusting approach for the maintenance scheme is desired. Unfortunately, to our knowledge, there remains a deficiency of effective adaptive mechanisms to adjust the maintenance parameters for the codebook model owing to the complexity of its maintenance scheme.

To solve this problem, in this paper, a novel adaptive maintenance scheme is proposed for the codebook model in three steps. First, an equal-qualification updating strategy is developed to replace the maximum-run-length strategy. Second, a random updating scheme is used to replace the cache mechanism. Finally, parameters that are scenario sensitive are identified through extensive experiments and analysis. Then, adaptive methods are proposed for them. Experimental results indicate that these modifications not only preserve the accuracy of the codebook model but also significantly reduce the amount of scenario sensitive parameters used in the maintenance scheme.

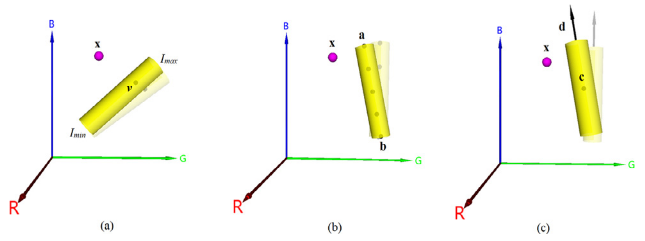

Fig. 1. Cylindrical color model, the arbitrary-cylindrical color model, and the modified arbitrary-cylindrical color model. Yellow cylinders are new color models and transparent cylinders are previous color models. Pink balls are new incoming samples. (a) Classical cylindrical color model. This color model is controlled by its center, illumination bounds, and the color distortion bound. Illumination bounds are updated according to the maximum criterion. Other parameters are updated through a low-pass filter. (b) Arbitrary cylindrical color model. This color model is controlled by its two ending points and the color distortion bound. It is updated by solving a linear regression problem. Approximations of previous samples are drawn in gray dots along the axis of the cylinder. (c) Modified arbitrary-cylindrical color model. This color model is controlled by its center, direction, and two bounds. All parameters are updated through a low-pass filter. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In the proposed model, two sets of parameters are scenario sensitive: the geometry of the new added code word and the learning probability. The geometry of a new added code is set to be the weighted average of code words in the codebook. The learning probability is determined according to the contrast between the misclassified foreground and background color model and the spacial-temporal consistency of the segmentation results. Experiments demonstrate that this adaptive scheme is effective in addressing camouflage-moving objects under dynamic backgrounds.

In our experiments, we compare the proposed method with the classical codebook model. It is verified that the performance of the codebook model is significantly improved using the proposed adaptive maintenance scheme. Moreover, the proposed method ensures the best performance of the system across a variety of complex scenarios without manual tuning. The overall performance of the proposed method is also evaluated against more than 20 state-of-the-art methods on a modern benchmarks dataset (Goyette et al., 2014). For a more comprehensive comparison, we also evaluate the proposed method against six states-of-the-art methods on eight dynamic scenes collected from three other datasets (Li et al., Nov. 2004; Bloisi et al., 2011; Toyama et al., 1999). Experimental results confirm that despite using only color information, the proposed method outperforms the majority of the solutions. Please note that our program is available on line at https://github.com/zengzhi2015/AAC.

The codebook model and its modifications are introduced in Section 2. The proposed adaptive scheme is presented in Section 3. Implementation details are explained in Section 4. In Section 5, experiments are conducted to evaluate the proposed method. Finally, the paper is concluded in Section 6.

2. Codebook model and its modifications

2.1. Classical codebook model

In the classical codebook model, each pixel is modelled by a codebook containing one or more code words. Each code word is represented by an RGB vector $\mathbf{v}_i=\{R_i, G_i, B_i\}$ and a 6-tuple $\mathbf{aux}_i=\{I_{\min}, I_{\max}, f_i, \lambda_i, p_i, q_i\}$, where $I_{\min}$ and $I_{\max}$ are the minimum and the maximum brightness, respectively. f_i is the frequency with which the code word has occurred. λ_i is the maximum-negative-run-length, meaning the longest time interval during which this code word has not been accessed. p_i and q_i are the first and last access times of the code word, respectively. The updating process of a code word is illustrated in Fig. 1(a). The working procedure of the classical codebook is given as follows in pseudo code:

```
If in the learning stage
    For each code word in the codebook
        Calculate the color distortion and intensity difference
        If both the color distortion and intensity difference are in bounds
            Update the code word
        End If
    End For
    If no match is found
      Add a new code word
    End If
End If
If right after the learning stage
    Delete those code words in the codebook whose
        maximum-negative-run-length is overly large
End If
If in the segmentation stage
    For each code word in the codebook
      Calculate the color distortion and intensity difference between the
        new sample and the code
      If both the color distortion and intensity difference are in bounds
        Label this pixel as background
        Update the code word
      End If
    End For
    If no match is found
        Label this pixel as foreground
        For each code word in the cache
            Calculate the color distortion and intensity difference
            If both the color distortion and intensity difference are in
        bounds
                Update the code word
            End If
        End For
        If no match is found
            Add a new code word to the cache
        End If
    End If
    Delete those code words in the codebook whose
        maximum-negative-run-length is overly large
    Delete those code words in the cache whose
        maximum-negative-run-length is overly large
    Move frequently observed code words from the cache to the codebook
End If
```

2.2. Arbitrary cylindrical color model

It is found that in the majority of multi-source illumination cases, the direction of the illumination variations is not toward the origin of the RGB color space (Zeng and Jia, 2014) and hence, the performance of the cylindrical color model (Kim et al., 2005) is degraded significantly. Because multi-source illuminations are common in nature, the arbitrary-cylindrical color model (Zeng and Jia, 2014) is proposed to solve this problem. This model represents pixel value distributions using cylinders whose axes need not pass through the origin, which considerably enhances the accuracy of the codebook model. Furthermore, the classical cylindrical color model is a special case of the arbitrary-cylindrical color model. Therefore, in the present work, we use this color model for the codebook model.

The arbitrary cylindrical color model represents pixel value distributions using cylinders whose axes need not pass through the origin. As discussed above, this model is more accurate in addressing multi-source illumination variations. As illustrated in Fig. 1(b), the model is represented by its two ending points $\mathbf{a}=\{a_r, a_g, a_b\}$ and $\mathbf{b}=\{b_r, b_g, b_b\}$. It is updated by solving a linear regression problem under the approximation that previous pixel values are evenly distributed along the line segment **ab**. Because the new line segment must lie in the plane expanded by the incoming pixel and the line **ab**, the three-dimensional linear regression problem can be simplified to a two dimensional problem and therefore, can be further simplified by a checking-up-table problem.

However, it is difficult to develop a real-time adaptive scheme for such a model because changing the table is time costly. Therefore, in the proposed method, we modify its geometrical descriptions in Section 2.3. This modification preserves the accuracy of the arbitrary-cylindrical color model and significantly simplifies the updating scheme.

2.3. Modified codebook model

In this sub-section, we propose the modified codebook model in two steps. First, we describe the modified arbitrary cylindrical color model. Then, we introduce the modified maintenance scheme.

The modification of the representation of the arbitrary-cylindrical color model is illustrated in Fig. 1(c). In the modified color model, each code word is represented by four geometrical parameters: the center $\mathbf{c}=\{c_R, c_G, c_B\}$, direction $\mathbf{d}=\{d_R, d_G, d_B\}$, radius r, and half-length l. The center and direction capture the color and tone of the illumination variation, respectively. The radius and length are thresholds.

An incoming pixel $\mathbf{x}=\{r, g, b\}$ is considered to be within a code word $\mathbf{cw}_i$, if and only if it meets the following condition:

$$\begin{cases} u_i = \langle \mathbf{x}-\mathbf{c}_i,\ \mathbf{d}_i\rangle \le l_i \\ v_i^2 = \langle \mathbf{x}-\mathbf{c}_i,\ \mathbf{x}-\mathbf{c}_i\rangle^2 - \langle \mathbf{x}-\mathbf{c}_i,\ \mathbf{d}_i\rangle^2 \le r_i^2 \end{cases}, \tag{1}$$

The expression $\langle,\rangle$ represents the inner product. The pixel $\mathbf{x}$ is said to be a background pixel if it is within at least one code word of the codebook.

When we updated a code word according to a pixel $\mathbf{x}$, the center $\mathbf{c}_i$ is updated as follows:

$$\mathbf{c}_i(k+1) = \mathbf{c}_i(k) + \alpha[\mathbf{x}(k) - \mathbf{c}_i(k)], \tag{2}$$

where α is a learning rate and k indicates the number of the captured frame. The direction $\mathbf{d}_i$ is updated in a similar manner:

$$\mathbf{d}_i(k+1) = \begin{cases} \mathbf{d}_i(k)l(k) + \alpha[\mathbf{x} - \mathbf{c}_i(k) - \mathbf{d}_i(k)l(k)] & u_i(k) \ge 0 \\ \mathbf{d}_i(k)l(k) + \alpha[\mathbf{c}_i(k) - \mathbf{x} - \mathbf{d}_i(k)l(k)] & u_i(k) < 0 \end{cases}, \tag{3}$$

and

$$\mathbf{d}_i(k+1) = \frac{\mathbf{d}_i(k)}{\|\mathbf{d}_i(k)\|}, \tag{4}$$

where the expression ||.|| represents the norm operation. The width r_i and the length l_i are updated as follows:

$$r_i^2(k+1) = r_i^2(k) + \alpha\left(9v_i(k)^2 - r_i^2(k)\right), \tag{5}$$

and

$$l_i^2(k+1) = l_i^2(k) + \alpha\left(9u_i(k)^2 - l_i^2(k)\right), \tag{6}$$

The geometrical parameters for the modified color model are independent. Therefore, the updating process of the color model can be described by a simple analytical expression with a few parameters and it is significantly easier to develop an adaptive scheme for this updating process.

Now, we introduce the modified maintenance scheme. It can be seen from the first sub-section that the maintenance structure of the codebook is complex owing to the maximum-run-length-controlled filtering process and the cache-based foreground learning mechanism. Studies (St-Charles et al., 2015a; Guo et al., 2013) have attempted to simplify the maintenance scheme by applying the weighting scheme used in GMM (Stauffer and Grimson, 1999). However, we found that the weighting scheme used in GMM has the following problem in its maintenance scheme. The Gaussian

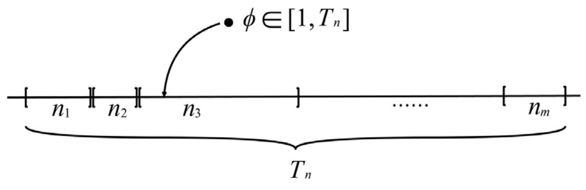

Fig. 2. Equal-qualification-maintenance scheme. We match n_i in each code word to a line segment whose length is n_i. Then, we arrange these segments as illustrated in this graph. The total length of all line segments is T_n. We randomly choose a point ϕ on the line segment T_n. Suppose this point is located in the interval n_3. We decrease n_3 by one. Obviously, the probability of an n_i being chosen is proportional to the length of the line segment n_i.

distribution with the least weight is always replaced by a new incoming Gaussian distribution. That is, the maintenance scheme favors persistent Gaussian distributions over infrequent Gaussians distributions. This treatment makes the algorithm intolerable to low frequency repetitive motions. Therefore, when addressing dynamic backgrounds, it is more appropriate to assume that a different background appearance should have the same qualification for modelling the background (Barnich and Van Droogenbroeck, 2011). Following this assumption, we developed a novel maintenance scheme called equal-qualification-updating strategy. In this strategy, each code word is viewed as a cluster of samples and the total number of samples of all code words is fixed. When a new incoming sample is going to be added to the codebook, we randomly choose a cluster and reduce its number of samples by one. The probability of the cluster being chosen is proportional to its number of samples. The proposed strategy implies that when a new sample is added, we randomly discard an old sample from the existing population with equal probability.

In detail, the proposed model uses only one parameter n_i to control the maintenance process. The parameter n_i represents the number of samples contained in each cluster (or code word). We denote a code word as $\mathbf{cw}_i = \{\mathbf{c}_i, \mathbf{d}_i, r_i, l_i, n_i\}$. Each pixel is modelled by a codebook containing one or more code words. For each code word, n_i should be a positive integer and the other four components should be real numbers. The direction $\mathbf{d}_i$ consists of directional cosines. For all the code words, the parameter n_i must meet the following constraint:

$$\sum n_i = T_n, \tag{7}$$

where T_n is a positive integer. Eq. (7) indicates that the total number of samples contained in all clusters is fixed. When we add a new code word to the codebook according to an observation $\mathbf{x}$, we must reduce the parameter n_i of a randomly chosen code word according to a randomly generated integer $\phi \in [1, T_n]$. That is,

$$n_i(k+1) = \begin{cases} n_i(k) - 1 & \text{if} \sum_{j=1}^{i-1} n_j(k) < \varphi \text{ and } \sum_{j=1}^{i} n_j(k) \geq \varphi \\ n_i(k) & \text{otherwise} \end{cases}, \tag{8}$$

Eq. (8) can be easily understood by the following figure.

If the parameter n_i of a code word is zero, we delete the code word. We add the new code word as follows:

$$\begin{cases} \mathbf{c}_i = \mathbf{x} \\ \mathbf{d}_i = \mathbf{x}/\|\mathbf{x}\| \\ r_i = r_{init} \\ l_i = l_{init} \\ n_i = 1 \end{cases}, \tag{9}$$

where r_{init} and l_{init} are real numbers. Similarly, when we update an existing code word, its parameter n_i is updated by:

$$n_i(k+1) = n_i(k) + 1. \tag{10}$$

Eq. (10) means a new sample is added to the cluster. Considering Eq. (7), we must remove an existing sample in the manner described by Eq. (8). The proposed algorithm indicates that the total number of clusters is a variable, however, the total number of samples is fixed.

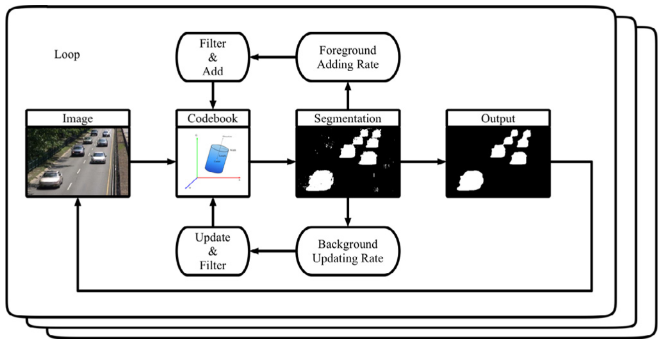

Fig. 3. Block diagram of the modified codebook model.

The classical codebook model uses a cache structure to learn new incoming backgrounds. Code words staying in the cache for a long time are moved into the codebook model. However, this scheme doubles the memory and computation costs and involves additional parameters. Therefore, we use a different foreground learning strategy following the idea of (Barnich and Van Droogenbroeck, 2011). In detail, if a pixel is judged to be the background, we update the background model with a probability of ψ_b. Similarly, if a pixel is judged to be the foreground, we add it to the background model with another probability ψ_f. The parameters ψ_b and ψ_f can be regarded as the weights for the model modification rate. The difference is that these two parameters are adaptively determined in the proposed method. The flow chart of the proposed codebook model is given in Fig. 3.

Compared with the maximum-run-length scheme and the cache strategy in (Kim et al., 2005), the proposed method is not required to filter out all code words for every iteration and does not have any additional hidden model. Therefore, the proposed method not only reduces the memory costs but also significantly increases the computation efficiency. Furthermore, the proposed method involves substantially fewer parameters and is easier for adaptive control.

3. Adaptive parameter adjustment

The variations of pixel-values are different at different surfaces and under different illumination conditions. Therefore, the parameters of the color model must be adjusted adaptively for each pixel. Experiments indicate that in the modified color model, these parameters are sensitive to scenarios: the initial width r_{init}, initial length l_{init}, and learning probabilities ψ_f and ψ_b. In the following, we will describe how to adjust these parameters adaptively.

To our knowledge, few studies concern the adaptive tuning of the initial width r_{init} and the initial length l_{init}. However, these two parameters do influence the segmentation results. As illustrated in Fig. 4, we made comparisons between pixel models *using* and *without using* adaptive tuning of the initial width and initial length. Graphs (a) and (d) are screen shots. The observed pixels are rounded by red boxes. Graphs (b) and (e) are pixel models *without using* adaptive tuning. Graphs (c) and (f) are pixel models *using* adaptive tuning.

For the observed pixel in graph (a), its distribution is mainly covered by the large cylinder indicated in graphs (b) and (c). Because the background has double appearances, additional cylinders are required. If the sizes of the added cylinders are overly small, several small cylinders are required as illustrated in graph (b). This would cost additional memory usage and increase the complexity.

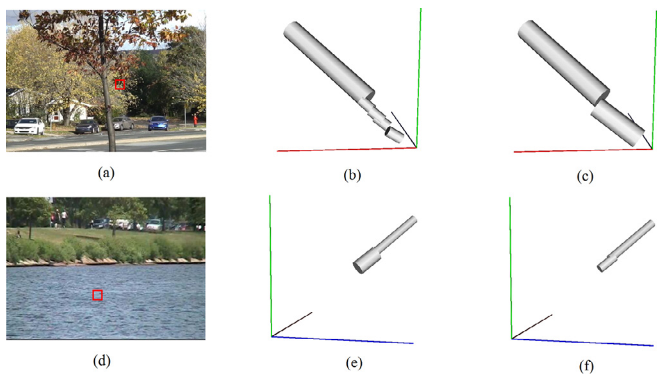

Fig. 4. Comparisons between pixel models *using* and *without using* adaptive tuning of the initial width r_{init} and initial length l_{init}. Graphs (a) and (d) are screen shots. The observed pixels are rounded by red boxes. Graphs (b) and (e) are pixel models *without using* adaptive tuning. Graphs (c) and (f) are pixel models *using* adaptive tuning. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Therefore, it is advantageous to use only one additional cylinder whose size matches the range of the distribution as illustrated in graph (c).

For the observed pixel in graph (b), its distribution is mainly covered by the long and slim cylinder as indicated in graphs (e) and (f). Because the background has multiple appearances, additional cylinders must be added to cover the distribution. If the added cylinder is overly large as illustrated in graph (e), the detection rate will decrease. Therefore, it is more efficient to add a slimmer cylinder as illustrated in graph (f).

From the above discussion, it is obvious that the size of the added cylinder should match the range of the distribution. Therefore, we adaptively adjust the size of the added code as follows:

$$r_{init} = \frac{1}{T_n}\sum_{i=1}^{m} r_i n_i, \tag{11}$$

and

$$l_{init} = \frac{1}{T_n}\sum_{i=1}^{m} l_i n_i, \tag{12}$$

where m is the total number of code words in the codebook. Eqs. (11) and (12) indicate that the size of the added code is the weighted average of all the cylinders in the codebook. In practice, one can set upper and lower bounds for both r_{init} and l_{init}; hence, they can be neither overly large nor small. Empirically, we set $r_{init} \in [10, 20]$ and $l_{init} \in [10, 20]$.

The foreground adding probability ψ_f also has a considerable influence on the segmentation quality. As illustrated in the third row of Fig. 5, low-contrast foregrounds are more likely to be absorbed into dynamic backgrounds. To address this problem, we decrease the foreground adding probability ψ_f according to the contrast between the misclassified foreground and the background model. In the proposed work, we define the contrast C_{st} between the misclassified foreground pixel and the background model as follows:

$$C_{st} = \min_i \left[\max\left(\frac{u_i}{l_i}, 1\right) \times \max\left(\frac{v_i^2}{r_i^2}, 1\right) \right]. \tag{13}$$

Empirically, the foreground adding probability ψ_f is given as follows:

$$\psi_f = \min\left(\frac{C_{st}-1}{2}, 0.1\right). \tag{14}$$

It can be seen that the greater the contrast, the greater the foreground adding probability. A comparison between the classification results *using* and *without using* this adaptive tuning approach is illustrated in Fig. 5. In this scene, the contrast between the boat and the water is extremely small. For both approaches, the dynamic background is absorbed into the pixel model reasonably well. However, the low contrast foreground is more likely to be absorbed without the proposed adaptive approach.

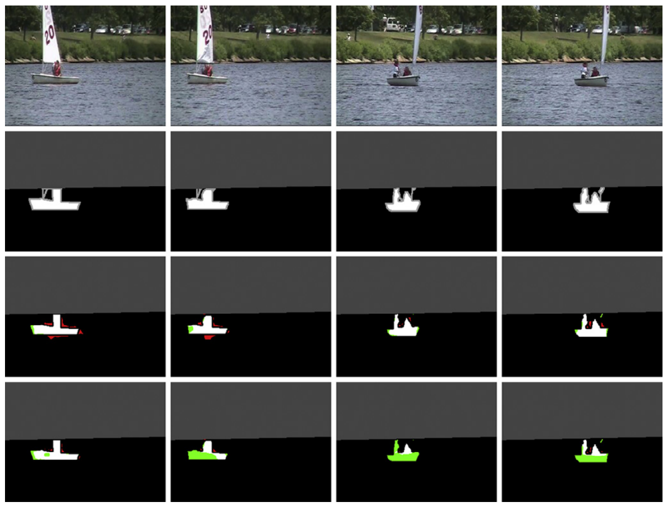

Fig. 5. Comparisons between classification results using and without considering foreground contrast. Graphs in the first row are screen shots. The ground truths are given in the second row. Classification results for *using* and *without using* adaptive tuning of the foreground adding probability are illustrated in the third and fourth rows respectively. Note that we use a color format which adopting the following specifications: true-positive pixels in white, true-negative pixels in black, false-positive pixels in red, and false-negative pixels in green. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The raw segmentation result always contains considerable salt-and-pepper noise. Updating the background model without suppressing such noise would inevitably decrease the convergence speed and contaminate the background model. Therefore, in ViBe+ (Van Droogenbroeck and Paquot, 2012), the authors suggest using connected component analysis to modify the updating mask. In PBAS (Hofmann et al., 2012) and SuBSENSE (St-Charles et al., 2015b), temporal consistency is considered to suppress blinking pixels. Following these ideas, we use special-temporal consistency in the determination of the weights ψ_b and ψ_f.

In detail, for the raw segmentation mask, we remove a connected foreground region whose area is less than T_{cn} pixels and fill a connected background region whose area is less than $2T_{cn}$ pixels. Then, we compare the raw segmentation mask with the modified mask. For each pixel, the latest l_c comparison results c_p are stored in a first-in-first-out buffer $\mathbf{B}_f = \{c_p(t), c_p(t-1), \ldots, c_p(t+1-l_c)\}$, where t is the time. Each comparison result c_p can assume the following four different values: "0" (if the pixel is labeled as background in both masks); "1" (if the pixel is labeled as foreground in the raw mask and labeled as background in the modified mask); "2" (if the pixel is labeled as background in the raw mask and labeled as foreground in the modified mask); "3" (if the pixel is labeled as foreground in both masks). Finally, ψ_b and ψ_f are adjusted according to $\mathbf{B}_f$ as follows:

$$\psi_b = \frac{\exp[-U(B_f, 2)]}{100}\xi_b \tag{15}$$

and

$$\psi_f = C_{st}\exp[-U(B_f, 2)]\left\{1 + 2\left[\frac{U(B_f, 1)}{l_c}\right]\right\}\xi_f, \tag{16}$$

where $\xi_b = 1$, only if $c_p(t) = 0$ and $\xi_f = 1$, only if $c_p(t) = 1$; Otherwise, $\xi_b = \xi_f = 0$. The function $U(\mathbf{B}, n)$ in Eqs. (15) and (16) indicates the number of times that the value n appears in the buffer $\mathbf{B}$. It can be seen from Eqs. (15) and (16) that the greater the value $U(\mathbf{B}_f, 2)$, the smaller the value ψ_b. Conversely, the greater the value $U(\mathbf{B}_f, 1)$, the greater the value ψ_f.

Because the parameter T_{cn} should be different for different scenarios, we propose an adaptive method to adjust this parameter. Given a raw segmentation mask, we store the areas of all connected regions in an array $\mathbf{A}$. Through a large number of statistical analyses, it has been found that the greater the area, the less the number of the regions. The majority of connected regions in the segmentation mask have an area of one pixel, which is excessively small to indicate a real object in the scene. Therefore, we set T_{cn} to a value such that 99.5% of the connected regions in the segmentation mask are eliminated. In detail, we sort the values in $\mathbf{A}$ in ascending order. Then, T_{cn} is set to A_T, where A_T is the Tth component of $\mathbf{A}$ and $T = 0.995\ V(\mathbf{A})$. The function $V(\mathbf{A})$ indicates the size of $\mathbf{A}$.

4. Implementation details

The proposed approach requires an initialization stage. In our experiments, we used 250 frames for initialization on average. In the proposed approach, there are also three other parameters: T_n, α, and l_c. As indicated in the next section, these parameters are not scenario sensitive. In the experimental section, only one optimal set of values is used: $T_n = 15$, $\alpha = 0.1$, and $l_c = 20$. We summarize the proposed adaptive background subtraction algorithm as follows:

```
1) In the training stage:
  a) For each incoming pixel, verify if it meets the condition given by Eq. (1);
    i. If it meets the condition, update the pixel model according to Eqs. (2)–(6), (8), and (10);
    ii. If it does not meet the condition, add the pixel to the model according to Eqs. (8) and (9);
2) In the segmentation stage:
  a) For each incoming pixel, verify if it meets the condition given by Eq. (1);
    i. If it meets the condition, label the pixel as background;
    ii. If it does not meet the condition, label the pixel as foreground;
  b) For each pixel that has been labeled as background:
    i. Calculate the background updating probability $\psi_b$ according to Eq. (15);
    ii. Update the pixel model with a probability of $\psi_b$;
  c) For each pixel that has been labeled as foreground:
    i. Calculate the foreground adding probability $\psi_f$ according to Eq. (16);
    ii. Add the pixel to the model with a probability of $\psi_f$.
```

The proposed method was implemented in the C language on a 2.0 GHz i7 processor with 8 GB RAM, running the Windows 7 operating system. Though the program was not optimized, the processing speed was approximately 20 fps on average. This is because we do not update the color model each time it is matched. Because ψ_b is small, we update the color model with a small probability when it is matched. Only the checking procedure is performed each turn. However, calculating Eq. (1) is much less time consuming. Further, for the majority of background regions, only one or two code words were sufficient to represent the background. Therefore, one or two verifications were adequate for classifying a pixel value.

5. Experiments

In Test One, we first compared the performance of the proposed model between *using* and *without using* the adaptive approach. Second, we analyzed the relation between parameter adjustments and F-measure for typical scenarios. Third, we compared the proposed approach with more than 20 state-of-the-art methods listed on the CDnet dataset (Goyette et al., 2014). Finally, we evaluated the proposed method with eight testing sequences selected from three different datasets (Li et al., 2004; Bloisi et al., 2011; Toyama et al., 1999). Seven metrics are used to evaluate their performances. These metrics are Recall (RE), Specificity (SP), False Positive Rate (FPR), False Negative Rate (FNR), Percentage of Wrong Classifications (PWC), Precision (PR), and F-Measure (FM). These metrics are defined as $RE = TP/(TP + FN)$, $SP = TN/(FP + TN)$, $FPR = FP/(FP + TN)$, $FNR = FN/(FN + TP)$, $PWC = (FN + FP)/(TP + FN + FP + TN)$, $PR = TP/(TP + FP)$, and $FM = 2TP/(2TP + FP + FN)$. Parameters TP, FP, FN, and TN are the true positive, the false positive, the false negative, and the true negative, respectively.

5.1. Test One

We used all the test sequences in the dynamic background category for comparisons. This category contained six different videos captured from the real world. Each testing sequence consisted of thousands of frames. These frames had previously been segmented manually and were available on line. Segmentation results obtained using the classical codebook model, proposed method without adaptive scheme, and proposed method are presented in Table 1. It can be seen that owing to the adaptive maintenance scheme, the overall performance of the proposed method was considerably superior.

5.2. Test Two

In Fig. 6, the relations between parameter adjustments and F-measure for typical scenarios are indicated. Parameter l_c is the size of the temporal window. This parameter determines how fast the model responds to segmentation temporal inconsistency. It can be seen that the F-measure is not sensitive to this parameter. The parameter T_n controls the maximum number of background appear-

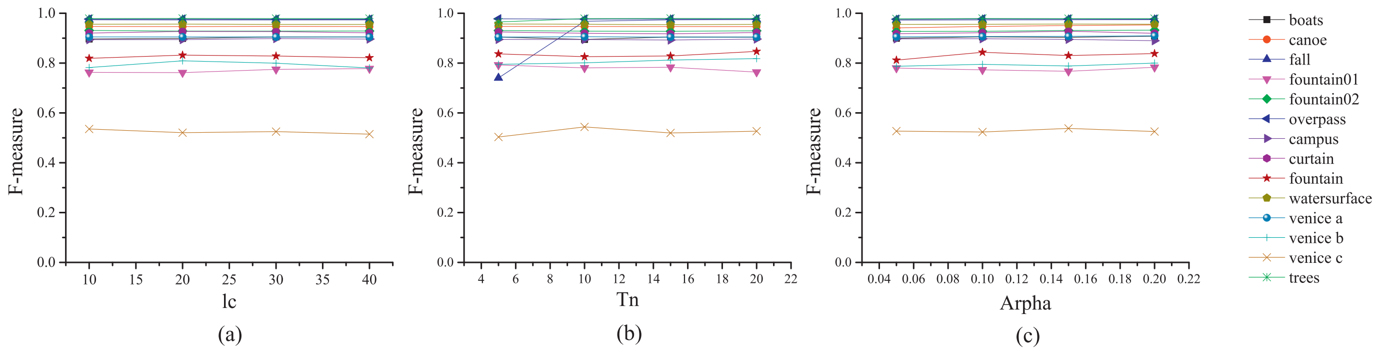

Fig. 6. F-measure of the proposed method on each category with changing parameter values. (a) Relation between l_c and *F-measure* for each testing sequence. (b) Relation between T_n and *F-measure* for each testing sequence. (c) Relation between α and *F-measure* for each testing sequence.

Table 1
Quantitative comparisons between the proposed method, the non-adaptive method, and the classical codebook model.

	RE	SP	FPR	FNR	PWC	PR	FM
Adaptive							
Boats	**0.9373**	0.9992	0.0008	**0.0627**	**0.1129**	0.8723	**0.9036**
Canoe	**0.9742**	0.9972	0.0028	**0.0258**	0.3599	0.9212	0.9470
Fall	**0.9851**	**0.9993**	**0.0007**	**0.0149**	**0.0954**	**0.9612**	**0.9730**
Fountain01	**0.6832**	0.9999	0.0001	**0.3168**	**0.0290**	0.8861	**0.7716**
Fountain02	**0.9573**	0.9998	0.0002	**0.0427**	0.0286	0.9007	0.9281
Overpass	**0.9868**	0.9996	0.0004	**0.0132**	**0.0582**	0.9674	**0.9770**
Overall	**0.9207**	0.9992	0.0008	**0.0793**	**0.1140**	0.9182	**0.9167**
Non-adaptive							
Boats	0.4328	**0.9999**	**0.0001**	0.5672	0.3673	**0.9590**	0.5965
Canoe	0.9432	**0.9989**	**0.0011**	0.0568	**0.3041**	**0.9701**	**0.9565**
Fall	0.9505	0.9979	0.0021	0.0495	0.2972	0.8894	0.9189
Fountain01	0.6466	**1.0000**	**0.0000**	0.3534	0.0343	**0.9153**	0.7579
Fountain02	0.9205	**0.9999**	**0.0001**	0.0795	**0.0223**	**0.9747**	**0.9468**
Overpass	0.9479	**0.9998**	**0.0002**	0.0521	0.0923	**0.9826**	0.9649
Overall	0.8069	**0.9994**	**0.0006**	0.1931	0.1862	**0.9485**	0.8569
Classical codebook							
Boats	0.8680	0.9995	0.0005	0.1320	0.1295	0.9210	0.8937
Canoe	0.9154	0.9987	0.0013	0.0846	0.4259	0.9625	0.9384
Fall	0.8874	0.9664	0.0336	0.1126	3.4974	0.3228	0.4734
Fountain01	0.6755	0.9955	0.0045	0.3245	0.4721	0.1119	0.1919
Fountain02	0.8821	0.9992	0.0008	0.1179	0.1053	0.7035	0.7827
Overpass	0.8632	0.9966	0.0034	0.1368	0.5197	0.7747	0.8166
Overall	0.8486	0.9927	0.0073	0.1514	0.8583	0.6327	0.6828

ances. For most backgrounds, when T_n is greater than five, the F-measure is not sensitive to T_n. However, for a complex scene such as "fall", a greater T_n may be required. The parameter α controls the updating rate. Fig. 6 also indicates that the F-measure is not sensitive to this parameter because ψ_b and ψ_f also influence the model modification rate and these two parameters have been adjusted automatically.

5.3. Test Three

We compared the proposed method with more than 20 state-of-the-art methods listed on the CDnet dataset. We choose all testing sequences in the dynamic background category for comparisons. Examples of the classification results are displayed in Fig. 7. Screenshots and segmentation results for each scenario are illustrated from top to bottom. Pictures in the first row are screenshots. Pictures in the second row are ground truths. The segmentation results using the proposed method, SuBSENSE (St-Charles et al., 2015b), Spectral-360 (Sedky and Chibelushi, 2014), SOBS (Maddalena and Petrosino, 2012), and GMM (Zivkovic, 2004) are displayed in the third to the seventh rows, respectively. It can be seen that the proposed algorithm considerably outperformed the others.

Quantitative comparison results are listed in Table 2. For all methods, the greater the RE, SP, PR, and FM and the less the FPR, FNR, and PWC, the better the segmentation results. The best scores are highlighted in bold letters. It is clear that despite using only color information, the performance of the proposed method was significantly superior. The F-measure is a good indicator of overall performance. The F-measure of the proposed method was significantly greater than that of the others. Because both the Recall and Precision of the proposed method is high, it can be concluded that the proposed method performed well in both detection rate and noise suppression ability. The proposed method could achieve superior results if texture information was incorporated. The comparison results reinforce the power of the proposed adaptive parameter tuning approach.

To increase the comprehensiveness of the test, we also evaluated the proposed method against six states-of-the-art methods on an additional eight dynamic scenes collected from three different datasets (Li et al., 2004; Bloisi et al., 2011; Toyama et al., 1999). A visual comparison is provided in Fig. 8 and a quantitative comparison is given in Table 3.

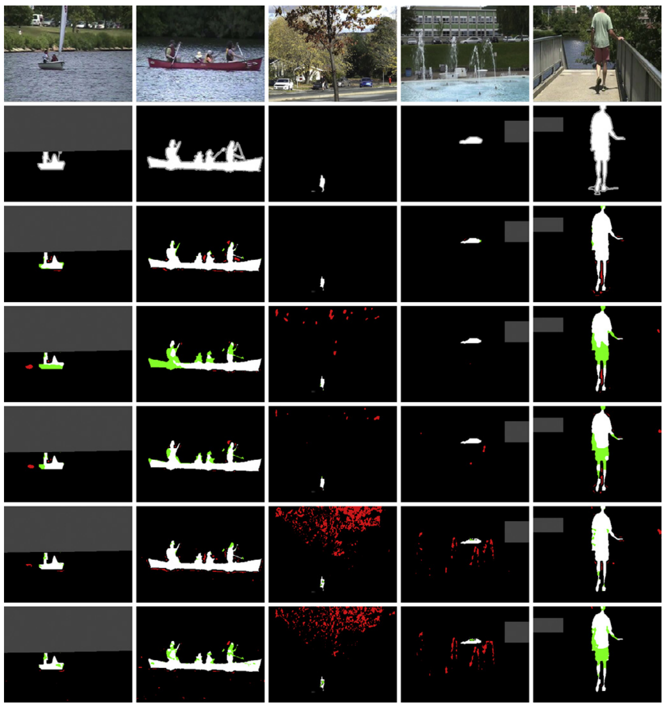
Fig. 7. Comparative background segmentation for typical frames captured from five sequences. Pictures in the first row are screenshots. Pictures in the second row are ground truths. The segmentation results using the proposed method, SuBSENSE, Spectral-360, SOBS, and GMM are presented in the third to the seventh rows, respectively. Note that we use a color format which adopting the following specifications: true-positive pixels in white, true-negative pixels in black, false-positive pixels in red, and false-negative pixels in green. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 2
Quantitative comparisons on the CDnet dataset.

	RE	SP	FPR	FNR	PWC	PR	FM
Proposed	**0.9207**	0.9992	0.0008	**0.0793**	**0.1140**	**0.9182**	**0.9167**
FTSG (Wang et al., 2014b)	0.8691	0.9993	0.0007	0.1309	0.1887	0.9129	0.8792
Bin Wang (Wang et al., 2014a)	0.9177	0.9956	0.0044	0.0823	0.4837	0.799	0.8436
CwisarDH (De Gregorio and Giordano, 2014)	0.8144	0.9985	0.0015	0.1856	0.3270	0.8499	0.8274
SuBSENSE (St-Charles et al., 2015b)	0.7768	**0.9994**	**0.0006**	0.2232	0.4042	0.8915	0.8177
SaliencySubsense	0.7676	0.9994	0.0006	0.2324	0.4106	0.9	0.8157
PBAS (Hofmann et al., 2012)	0.6955	0.9989	0.0011	0.3045	0.5394	0.8326	0.6829
Spectral-360 (Sedky and Chibelushi, 2014)	0.7819	0.9992	0.0008	0.2181	0.3513	0.8456	0.7766
RMoG (Varadarajan et al., 2013)	0.7892	0.9978	0.0022	0.2108	0.4238	0.7288	0.7352
KNN	0.8047	0.9937	0.0063	0.1953	0.8059	0.6931	0.6865
ViBe (Barnich and Van Droogenbroeck, 2011)	0.7616	0.9980	0.0020	0.2384	0.3838	0.7291	0.7197
SC_SOBS (Maddalena and Petrosino, 2012)	0.8918	0.9836	0.0164	0.1082	1.6899	0.6283	0.6686
SOBS_CF (Maddalena and Petrosino, 2009)	0.9014	0.9820	0.0180	0.0986	1.8391	0.5953	0.6519
GMM | S & G (Stauffer and Grimson, 1999)	0.8344	0.9896	0.0104	0.1656	1.2083	0.5989	0.6330
GMM | Zivkovic (Zivkovic, 2004)	0.8019	0.9903	0.0097	0.1981	1.1725	0.6213	0.6328
CP3-online (Liang and Kaneko, 2014)	0.726	0.9963	0.0037	0.274	0.6613	0.6122	0.6111
KDE – ElGammal (Elgammal et al., 2002)	0.8012	0.9856	0.0144	0.1988	1.6393	0.5732	0.5961
Multiscale BG (Lu, 2014)	0.7392	0.9905	0.0095	0.2608	1.1365	0.5515	0.5953
EFIC (Allebosch et al., 2015)	0.6929	0.9956	0.0044	0.3071	0.9927	0.644	0.5662
GraphCutDiff	0.7693	0.9063	0.0937	0.2307	9.2106	0.5357	0.5391
Euclidean distance (Benezeth et al., 2010)	0.7757	0.9714	0.0286	0.2243	3.0095	0.4487	0.5081
Mahalanobis distance (Benezeth et al., 2010)	0.1237	0.9988	0.0012	0.8763	1.1753	0.7451	0.1798

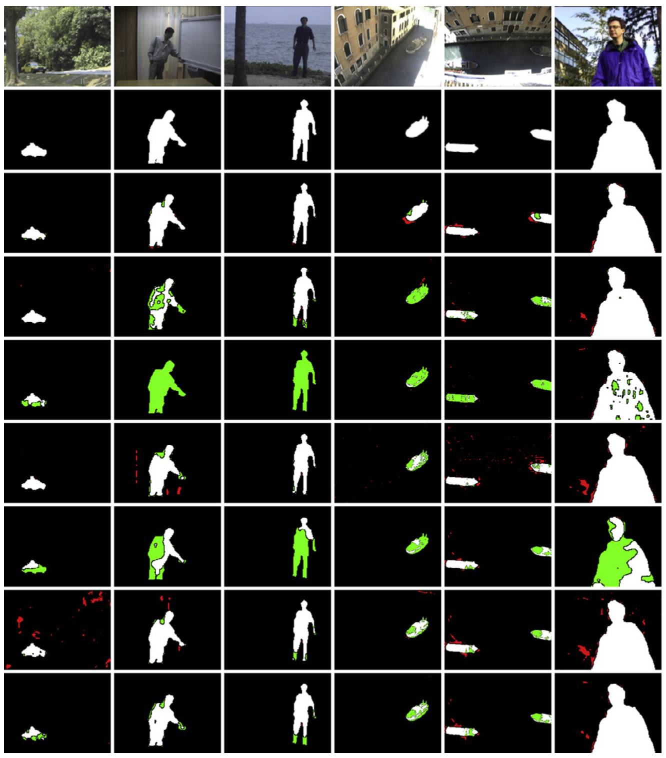

Fig. 8. Background segmentation results for typical sequences. Pictures in the first row are screenshots. Pictures in the second row are ground truths. Segmentation results using the proposed method, codebook model (Kim et al., 2005), GMM (Zivkovic, 2004), KDE (Elgammal et al., 2002), PBAS (Hofmann et al., 2012), SOBS (Maddalena and Petrosino, 2012), and ViBe (Barnich and Van Droogenbroeck, 2011) are in the third to the ninth rows, respectively. Note that we use a color format which adopting the following specifications: true-positive pixels in white, true-negative pixels in black, false-positive pixels in red, and false-negative pixels in green. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Visually, it appears that the segmentation quality of the proposed method is considerably greater than the others, particularly in scenes with camouflage moving objects under dynamic backgrounds (see the fourth and fifth column). From the quantitative comparison results, it can be seen that the proposed method outperformed the others in six out of eight testing sequences by a significant margin. The proposed method was only slightly surpassed by two other approaches on the campus and Venice c videos.

Table 3
Quantitative comparison results for eight testing sequences in terms of F-measure.

	Proposed	Codebook	GMM	KDE	PBAS	SOBS	ViBe
Campus	0.8922	0.8799	0.6992	**0.8948**	0.4482	0.5984	0.8243
Curtain	**0.9235**	0.7585	0.1619	0.8443	0.4485	0.9168	0.8919
Fountain	**0.8198**	0.6316	0.3929	0.6240	0.4493	0.6189	0.6063
Venice a	**0.9052**	0.4109	0.2032	0.6671	0.7147	0.8069	0.7857
Venice b	**0.7985**	0.6691	0.1852	0.7037	0.6996	0.6603	0.6660
Venice c	0.5218	**0.5497**	0.2875	0.1595	0.4278	0.3596	0.5343
Watersurface	**0.9561**	0.8928	0.0801	0.9499	0.4533	0.9246	0.8815
Wavingtrees	**0.9782**	0.9704	0.9336	0.9509	0.5926	0.9390	0.9706

6. Conclusions and future works

We proposed an adaptive maintenance scheme for the codebook background subtraction algorithm. To improve the accuracy of the codebook model and make it suitable for developing an adaptive maintenance scheme, we modified the color model, developed an equal qualification updating procedure, and introduced a random updating scheme with feedback. In the proposed framework, several scenario sensitive parameters were obtained. These parameters were adjusted adaptively to ensure the best performance of the system.

The proposed method was tested using the CDnet dataset. We compared the performance of the proposed model between *using* and *without using* the adaptive technique. It was verified that the performance of the model was significantly improved using the adaptive technique. Furthermore, we compared the proposed approach with more than 20 state-of-the-art methods listed on the CDnet dataset. It was demonstrated that despite using only color information, the proposed method outperformed the majority of the listed solutions in dynamic background subtraction.

The proposed method can be further improved by adding rapid global illumination detection. It can also be improved by incorporating more powerful texture features. Finally, the learning process of the proposed method can be accelerated by incorporating a moment term.

Acknowledgements

We are grateful to the reviewers for their constructive suggestions. This work was supported by the Fundamental Research Funds for the Central Universities (No. K5051399028).

References

Allebosch, G., Deboeverie, F., Veelaert, P., Philips, W., 2015. EFIC: edge based foreground background segmentation and interior classification for dynamic camera viewpoints. Adv. Concepts Intell. Vis. Syst.

Barnich, O., Van Droogenbroeck, M., 2011. ViBe: a universal background subtraction algorithm for video sequences. IEEE Trans. Image Process 20, 1709–1724. doi:10.1109/TIP.2010.2101613.

Benezeth, Y., Jodoin, P.-M., Emile, B., Laurent, H., Rosenberger, C., 2010. Comparative study of background subtraction algorithms. J. Electron. Imaging. 19, 033003. doi:10.1117/1.3456695.

Bloisi, D., Iocchi, L., Fiorini, M., Graziano, G., 2011. Automatic maritime surveillance with visual target detection. In: Proceedings of the International Defense and Homeland Security Simulation Workshop (DHSS). Rome, Italy, pp. 141–145.

Bouwmans, T., 2014. Traditional and recent approaches in background modeling for foreground detection: an overview. Comput. Sci. Rev. 11-12, 31–66. doi:10.1016/j.cosrev.2014.04.001.

Brutzer, S., Höferlin, B., Heidemann, G., 2011. Evaluation of background subtraction techniques for video surveillance. In: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit, pp. 1937–1944. doi:10.1109/CVPR.2011.5995508.

De Gregorio, M., Giordano, M., 2014. Change detection with weightless neural networks. In: Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Work, pp. 409–413. doi:10.1109/CVPRW.2014.66.

Doshi, A., Trivedi, M., 2006. Hybrid cone-cylinder codebook model for foreground detection with shadow and highlight suppression. IEEE Int'f Conf. Video Signal Based Surveill.

Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.S., 2002. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proc. IEEE. 90, 1151–1163. doi:10.1109/JPROC.2002.801448.

Goyette, N., Jodoin, P.-M., Porikli, F., Konrad, J., Ishwar, P., 2014. A novel video dataset for change detection benchmarking. IEEE Trans. Image Process 23, 4663–4679. doi:10.1109/TIP.2014.2346013.

Guerra-filho, G.B., 2005. Optical motion capture: theory and implementation. J. Theor. Appl. Informatics. 12, 61–89. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.7248.

Guo, J.M., Hsia, C.H., Liu, Y.F., Shih, M.H., Chang, C.H., Wu, J.Y., 2013. Fast background subtraction based on a multilayer codebook model for moving object detection. IEEE Trans. Circuits Syst. Video Technol 23, 1809–1821. doi:10.1109/TCSVT.2013.2269011.

Guo, J.M., Liu, Y.F., Hsia, C.H., Shih, M.H., Hsu, C.S., 2011. Hierarchical method for foreground detection using codebook model. IEEE Trans. Circuits Syst. Video Technol 21, 804–815. doi:10.1109/TCSVT.2011.2133270.

Heikklä, M., Pietikäinen, M., Member, S., Pietika, M., Heikklä, M., Pietikäinen, M., 2006. A texture-based method for modeling the background and detecting moving objects. IEEE Trans. Pattern Anal. Mach. Intell. 28, 657–662. doi:10.1109/TPAMI.2006.68.

Hofmann, M., Tiefenbacher, P., Rigoll, G., 2012. Background segmentation with feedback: the pixel-based adaptive segmenter. In: IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work, pp. 38–43. doi:10.1109/CVPRW.2012.6238925.

Hu, H., Xu, L., Zhao, H., 2012. A spherical codebook in YUV color space for moving object detection. Sens. Lett. 10, 177–189.

Jain, V., Kimia, B., Mundy, J., 2007. Background modeling based on subpixel edges. In: Image Process. 2007. ICIP, pp. 1–4. doi:10.1109/ICIP.2007.4379586.

Javed, O., Shafique, K., Shah, M., 2002. A hierarchical approach to robust background subtraction using color and gradient information. In: Work. Motion Video Comput. 2002. Proceedings, pp. 22–27. doi:10.1109/MOTION.2002.1182209.

Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L., 2005. Real-time foreground-background segmentation using codebook model. Real-Time Imaging 11, 172–185. doi:10.1016/j.rti.2004.12.004.

Lee, B., Hedley, M., 2002. Background estimation for video surveillance. Image Vis. Comput. New Zeal 315–320.

Lee, D.S., 2005. Effective Gaussian mixture learning for video background subtraction. IEEE Trans. Pattern Anal. Mach. Intell. 27, 827–832. doi:10.1109/TPAMI.2005.102.

Li, L., Huang, W., Gu, I.Y.-H., Tian, Q., 2004. Statistical modeling of complex backgrounds for foreground object detection. IEEE Trans. Image Process 13 (Nov. (11)), 1459–1472.

D. Liang, S. Kaneko, Improvements and experiments of a compact statistical background model, 2014.

Lu, X.Q., 2014. A multiscale spatial-temporal background model for motion detection. In: Proc. IEEE Int. Conf. Image Process, pp. 3268–3271.

M.Sedky, M.Moniri, Chibelushi, C.C., 2014. Spectral-360: a physical-based technique for change detection. In: Proc. IEEE Work. Chang. Detect., pp. 405–408.

Maddalena, L., Petrosino, A., 2009. A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection. Neural Comput. Appl. 19, 179–186. doi:10.1007/s00521-009-0285-8.

Maddalena, L., Petrosino, A., 2012. The SOBS algorithm: what are the limits? In: IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work, pp. 21–26. doi:10.1109/CVPRW.2012.6238922.

Mashak, S.A.B.S., Hosseini, B., 2012. Real-time bird detection based on background subtraction. In: World Congr. Intell. Control Autom, pp. 4507–4510.

Nascimento, J.C., Marques, J.S., 2006. Performance evaluation of object detection algorithms for video surveillance. IEEE Trans. Multimed. 8, 1–27. doi:10.1109/TMM.2006.876287.

Paul, M., Lin, W., Lau, C., Lee, B., 2013. Video coding with dynamic background. EURASIP J. Adv. Signal Process. 2013 1–17.

Piccardi, M., 2004. Background subtraction techniques: a review. In: 2004 IEEE Int. Conf. Syst. Man Cybern. (IEEE Cat. No.04CH37583), 4, pp. 3099–3104. doi:10.1109/ICSMC.2004.1400815.

Porikli, F., Tuzel, O., 2005. Bayesian background modeling for foreground detection. In: Proc. Third ACM Int. Work. Video Surveill. Sens. Networks - VSSN '05, 55 doi:10.1145/1099396.1099407.

Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B., 2005. Image change detection algorithms: a systematic survey. IEEE Trans. Image Process 14, 294–307 doi:10.1.1.5.291.

Senior, A., Tian, Y., Lu, M., 2010. Interactive motion analysis for video surveillance and long term scene monitoring. In: Asian Conf. Comput. Vis., pp. 164–174.

Sobral, A., Vacavant, A., 2014. A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. Comput. Vis. Image Underst. 122, 4–21. doi:10.1016/j.cviu.2013.12.005.

Stauffer, C., Grimson, W.E.L., 1999. Adaptive background mixture models for real-time tracking. In: Proc. 1999 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Cat No PR00149, 2, pp. 246–252. doi:10.1109/CVPR.1999.784637.

St-Charles, P.L., Bilodeau, G.A., Bergevin, R., 2015. A self-adjusting approach to change detection based on background word consensus. In: IEEE Winter Conf. Appl. Comput. Vis., pp. 990–997.

St-Charles, P.L., Bilodeau, G.A., Bergevin, R., 2015. SuBSENSE: a universal change detection method with local adaptive sensitivity. IEEE Trans. Image Process 24, 359–373. doi:10.1109/TIP.2014.2378053.

Toyama, K., Krumm, J., Brumitt, B., Meyers, B., Sep. 1999. Wallflower: Principles and practice of background maintenance. In: Proc. 7th IEEE Int. Conf. Comput. Vis., 1, pp. 255–261.

Unzueta, L., Nieto, M., Cortés, A., Barandiaran, J., Otaegui, O., Sánchez, P., 2012. Adaptive multicue background subtraction for robust vehicle counting and classification. IEEE Trans. Intell. Transp. Syst. 13, 527–540. doi:10.1109/TITS.2011.2174358.

Van Droogenbroeck, M., Paquot, O., 2012. Background subtraction: experiments and improvements for ViBe. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 32–37.

Varadarajan, S., Miller, P., Zhou, H.Y., 2013. Spatial mixture of Gaussians for dynamic background modelling. In: Proc. IEEE Int. Conf. Adv. Video Signal Based Surveill, pp. 27–30.

Wang, B., Dudek, P., Fast Self, A, 2014. tuning Background Subtraction Algorithm. In: IEEE Conf. Comput. Vis. Pattern Recognit, pp. 4321–4324. doi:10.1109/CVPRW.2014.64.

Wang, R., Bunyak, F., Seetharaman, G., Palaniappan, K., 2014. Static and moving object detection using flux tensor with split Gaussian models. In: IEEE Conf. Comput. Vis. Pattern Recognit. Work, pp. 414–418. doi:10.1109/CVPRW.2014.68.

Wren, C.R., Azarbayejani, a., Darrell, T., Pentland, A.P., 1997. Pfinder: real-time tracking of the human body. IEEE Trans. Pattern Anal. Mach. Intell. 19, 780–785. doi:10.1109/34.598236.

Zaharescu, A., Jamieson, M., 2011. Multi-scale multi-feature codebook-based background subtraction. In: Comput. Vis. Work. (ICCV Work. 2011 IEEE Int. Conf., pp. 1753–1760. doi:10.1109/ICCVW.2011.6130461.

Zeng, Z., Jia, J.Y., 2014. Arbitrary cylinder color model for the codebook based background subtraction. Opt. Express. 22, 21577–21588.

Zivkovic, Z., 2004. Improved adaptive Gaussian mixture model for background subtraction. In: Proc. 17th Int. Conf. Pattern Recognition, 2004. ICPR 2004, 2, pp. 28–31. doi:10.1109/ICPR.2004.1333992.

Zivkovic, Z., Van Der Heijden, F., 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognit. Lett. 27, 773–780. doi:10.1016/j.patrec.2005.11.005.