Pattern Recognition 84 (2018) 28–38

Contents lists available at ScienceDirect

Pattern Recognition

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

A novel framework for background subtraction and foreground detection

Guian Zhang, Zhiyong Yuan*, Qianqian Tong, Mianlun Zheng, Jianhui Zhao**

School of Computer Science, Wuhan University, Wuhan 430072, China

ARTICLE INFO

Article history:
Received 21 November 2017
Revised 19 June 2018
Accepted 1 July 2018
Available online 3 July 2018

Keywords:
Background modeling
Mino vector
Dynamic nature
KDE
Denoising
Tetris update scheme

ABSTRACT

In this paper, we propose a novel image background subtraction framework based on KDE. Firstly a new data structure called Mino Vector (MV) is designed for each pixel; we define dynamic nature (DN) for pixels of a scene and rank them in terms of DN for getting quantized results named dynamic rank (DR). Then, the varying KDE is adopted and implemented which significantly improves the estimation accuracy. Unlike using a global threshold in literature, we adaptively set a threshold for each pixel according to its DR. Inspired by the popular computer game Tetris, we present a Tetris update scheme (TUS) to update the background model in which the bottom row will be cleared, so do noises when the update condition is met. In experiments, we evaluate our framework on a well-known video dataset, CDnet 2012. Our results indicate that our framework achieves competitive results when compared with the state-of-the-art methods.

1. Introduction

A fundamental step for many image processing tasks is the separation of foreground and background. In this paper, we propose a novel framework (depicted in Fig. 1) where several new approaches are presented for pursuing more robust and accurate background subtraction. Because the quality of reference data seriously affects the accuracy of outcomes, we design an ingenious data structure for each pixel[1], called Mino Vector (MV), which has a fixed length of 263 and it is functionally divided into two parts: The first part contains subscripts from 1 to 256 for storing the number of corresponding intensity values (0 to 255), and the second part contains statistical information, such as spans, maximum and minimum intensity values, main locations these intensity values located, along with other parameters. We note that MV keeps purified pixel values for each pixel in background scene over time, which is substantially different from conventional methods which preserve a fixed length (i.e., for a time window) of reference data.

Based on the MV, we for the first time define a dynamic nature (DN) which is referred to the span of the intensity value expanding in each pixel of a scene during a period of time, and then associate each pixel with DN as an attribute. Generally, scenes regarded as dynamic are associated with a certain motion, such as swaying trees, rippling water, etc. In most cases, dynamic areas are only part of the scene while the remaining areas are static. From this point of view, the difference between a dynamic scene and a static scene is that whether there exists at least one dynamic area. With consideration of DN in a dynamic scene, dynamic areas must be composed of pixels with large values of DN, while static areas include pixels with small values of DN (and the same is true for a static scene). This is a process of quantizing the criterion of the subjective judgment and certainly this quantized criterion is still subjective. It implies that the boundaries between dynamic and static areas are relative. For further weakening the effect of the subjective factor, we rank DN values of a scene and generate a second-order attribute, called dynamic rank (DR), which indicates ranks of those DNs. In this way, an area with relatively large DN values in a static scene may own a DR that is the same as a dynamic area in a dynamic scene. Therefore, we assert that there is no need to distinguish between the dynamic scene and the static scene. By this assertion, we model the background scene, whatever dynamic or static, in a unified way in terms of DN and DR.

When samples are categorized into MVs, we use them to study the optimal bandwidth. Too small a value of bandwidth will cause under-smoothing problems such as an arising of spurious features, but too large a selection of the bandwidth may lead to over-smoothing with a certain loss of underlying structures. The best choice is to let the data appeared before determine the value. Those data with a high frequency of appearance should own more

* Corresponding author.
** Co-corresponding author.
E-mail addresses: zhiyongyuan@whu.edu.cn (Z. Yuan), jianhuizhao@whu.edu.cn (J. Zhao).

[1] unless otherwise specified, a pixel is the operating unit in this paper, for color image, reducing to each channel.

https://doi.org/10.1016/j.patcog.2018.07.006
0031-3203/

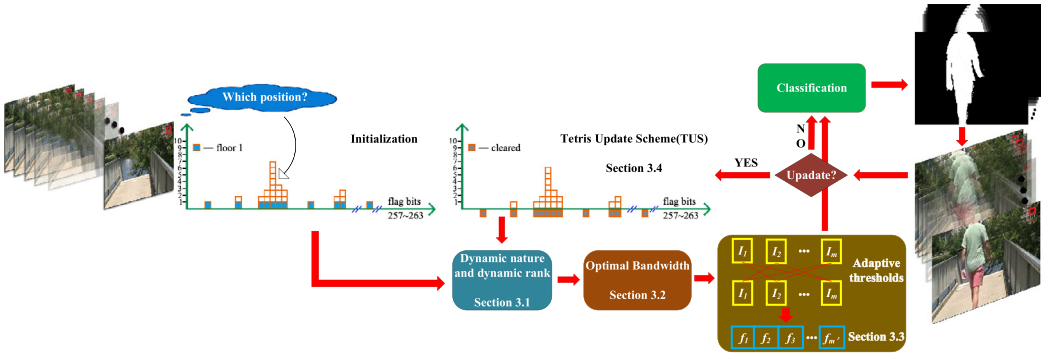

Fig. 1. Diagram of the whole framework for background subtraction modeling. Samples for initialization are firstly gathered into MVs for all pixels. Then the scene is described by DN and DR. With the structured data in MV, optimal bandwidth and adaptive thresholds are calculated. As classifying an incident pixel, if the update condition is met, TUS will be triggered, otherwise classification is performed.

weight. The varying kernel density estimation (VKDE) is a method associated with the objective rules which are embodied in the use of sample-point estimator. In this estimator, bandwidth is a function of each data point which is better than a function of estimated data point used in the balloon estimator. In this framework, we choose the sample-point estimator with a Gaussian kernel and resort to a normal distribution for the pilot estimate. When compared with other methods like median absolute deviation (MAD) used by Elgammal et al. [1], our approach significantly improves the accuracy in estimating the probability density so as to the detection result. Furthermore, by using MVs, we can avoid redundant repetitive computation for a higher efficiency.

In practice, choosing thresholds is a key step which determines final results in the background subtraction model. At present, there are two kinds of thresholds: one is in the global-level, and the other in the pixel-level, according to the scope of the action. Elgammal et al. [1] set a global threshold over all images and empirically adjust it to satisfy a certain false positive ratio. It may be a good choice for a stable scene, but evidently it is less flexible. Wang and Dudek [2] report a pixel-level scheme where threshold is bounded within a fixed range. The threshold fluctuates by the activity level of each pixel where the fluctuation is a measure of the noise level. Their work is similar to ours in terms of adaptation, but our method does not involve the use of extra parameters. We firstly calculate the densities of the observed data points which are used to get the optimal bandwidth. Then, these densities are sorted in an increasing order without repetition in every pixel to get an array of thresholds. The initial threshold of each pixel is determined by its DR (as a subscript in the array) which is already calculated before. Note that the length of *thresholds* is always larger than DR, so that the threshold can be tuned upon *thresholds* for matching an appropriate one over time.

Besides detection or classification by KDE, another necessary step for pursuing continuous outstanding performance for all background subtraction algorithms is the update process. We draw the inspiration from the popular game Tetris, in which, by rules of the game, the full bottom rows will be cleared and the remaining Minos will drop down to the ground by the gravity. The data structure, MV, filled with reference data is similar to the fallen Minos and we take another strategy to press them into the ground to eliminate the bottom rows. The effect of this operation also reduces the number of normal intensity values but does not reduce their capacities of accepting a new comer which belongs to them. Our method is robust to noises which always appear randomly within the intensity range (0 to 255). Benefiting by the idea of preserving reference data mentioned above, we can achieve a good speed in recalculating bandwidths and thresholds after the update procedure is performed.

In summary, this novel framework mainly includes four contributions: (1) we first propose the definition of dynamic nature (DN) and its sub-attribute dynamic rank (DR) for filling up the gap between dynamic and static scenes. As a result, we can describe and model all scenes in a unified way in terms of DN and DR. Then, we design the Mino Vector (MV) to serve the whole framework; (2) we adopt a varying KDE with a Gaussian kernel and choose normal distribution as the pilot estimate. This approach yields improvements in estimation accuracy; (3) we present a pixel-wise threshold which has a high adaptability, controlled by each pixel's own DR; (4) we put forward a novel update mechanism called Tetris update scheme (TUS), which results in performance improvements in suppressing the noise and enhancing the robustness. Our experimental results demonstrate that this novel framework achieves competitive results by comparing with the existing state-of-the-art approaches.

2. Related work

Since the late of last century, the rudiment of background subtraction is to simply calculate the difference between two frames [3,4] (one is purely the background, the other the current frame), but it is ill-suited when suffering even slow changes in the scenes. While statistics is good at analyzing the uncertainties underlying the data, a gamut of techniques in statistical area have been proposed.

Basic model. The seminal work by Wren et al. [5] uses a single Gaussian to model the background, assuming that the scene is a relatively static situation for their application Pfinder. Stauffer and Grimson [6] find that Pfinder does not work well in outdoor scenes. They model values of a pixel by using a mixture of Gaussians (MoG), and it becomes a milestone for most of the modified versions [7–12] in future. These methods have successively achieved progress in many situations. However, the characteristics of natural scenes cannot always be modeled by parametric ways. In this case, the non-parametric method is introduced into modeling the background scene. Elgammal et al. [1] choose the kernel function to be a Normal function, further assuming that the bandwidth is diagonal for reducing the complexity of computing probability density and is estimated by the median absolute deviation (MAD) over the sample for consecutive intensity values of the pixel. For a pixel at time t, I_t, they threshold the $P_r(I_t)$ by a global value th over all the images. If $P_r(I_t) < th$ holds, this pixel is considered as foreground, otherwise, background. Mittal and Paragios [13] define a hybrid density estimator with its bandwidth to be a function of both the estimated point and sample points which are called balloon estimator and sample-point estimator [14], respectively. In classification, they utilize a statistical approximation

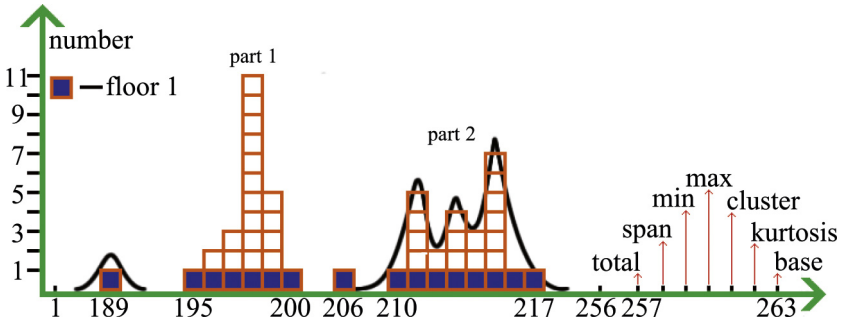

Fig. 2. Visualization of MV for a dynamic case with a large DN value and two main parts, part1 and part2.

method, i.e., sampling method, to obtain an incremental adaption of the threshold for achieving low false alarms and high detection rates. Sheikh and Shah [15] choose the Binned KDE [16,17] for the tradeoff between accuracy and computational burden, and then assert that the correlation of spatially proximal pixels is important. So they model the background and foreground both explicitly [18] and achieve high-quality detection results against dynamic elements like camera jitter, water ripples, etc.

Threshold. There are mainly two kinds of thresholds, the pixel-wise and the global-wise. At the earlier time, Elgammal et al. [1] take a global one over all the images in their KDE based background model and its value is determined heuristically. Barnich and Van Droogenbroeck [19] set global values for both the maximum sample distance threshold R and model update rate T. Global strategy means that all pixels will behave identically throughout the image sequence. However, this assumption cannot hold in reality since most of things in the background always change over time. Hofmann et al. [20] adopt the per-pixel threshold $R(x_i)$ (x_i is a pixel) in their PBAS model and tune $R(x_i)$ dynamically according to the average of minimal decision distances. St-Charles et al. [21] also set their threshold R in pixel-level and update this value by the minimal normalized color-LBSP distance and a 2D map of pixel-level accumulators.

Update scheme. An inappropriate update scheme will lead to low accuracy and weak adaptation. There are two types of update mechanisms, the selective update and the blind update mechanism, which are introduced and analyzed by Elgammal et al. [1]. Then they update their model by combing these two mechanisms together and computing the intersection of their detection results to eliminate the false positives which are ignored in a separate way. The blind update mechanism is adopted by Sheikh and Shah [15] and Sheikh et al. [22], which leads to somewhat more misdetections and thus depresses their model in performance. Cuevas and García [23] weight each sample in reference pixels (which are gathered in a blind way) for balancing the flexibility and robustness of the model. Xin et al. [24] propose an alternated updating mechanism in which there is an assumption of pure background frames given. Similar to ViBe [19,25], St-Charles et al. [21] update the pixel models using a conservative, stochastic, two-step approach. It uses the observation to replace a randomly picked sample of $B(x)$ with a probability of $1/T$ when a pixel is classified as background at x, and with the same probability to replace a neighbor of $B(x)$, where $B(x)$ is the background model at x. This conservative, stochastic approach is also adopted by Hofmann et al. [20] and Guo et al. [26]. Nevertheless, blind update mechanism, whatever applied totally, partially or in a modified version, will produce unavoidable false negatives due to the introduction of non-background pixels.

More detailed discussions about statistical methods can be found in recent surveys [27,28]. Apart from these statistical methods, researchers have innovatively introduced many other techniques recently, such as optical flow [29], superpixels [30,31], dynamic textures [32–34], binary patterns [21,26], Lasso [24], CNN [35], etc.

3. Construction of our background model

In this section, we will describe the procedure of constructing background subtraction model in detail. First of all, Mino Vector and a new manner of describing a scene will be introduced.

3.1. Mino vector, dynamic nature and dynamic rank

Mino Vector (MV) is exactly a vector with a fixed length of 263 and each pixel in a background scene will be equipped with a MV. This design derives from two concerns about the quality of reference data: efficiency and accuracy. In conventional ways, sample points are taken over a window in time of a fixed size, and then each point will take part in some operations with incident pixel sequentially. We find that an intrinsic property of reference data, concentricity, is neglected. That is, there are many sample points with the same intensity value which leads to repetitive operations and so inefficiency. Therefore, cells with subscripts from 1 to 256 in MV are designed to keep the number of every intensity value during the whole process for each pixel in background scene (see Fig. 2). In accuracy, MV also plays an important role, while this topic concerns the update scheme and it will be described in Section 3.4.

In Fig. 2, each block is called a Mino and indicates an intensity value. From the first frame of capturing a scene by a camera or of a recorded video, MV will be busy at finding places for the fallen intensity values and keep accumulating. This process is very similar to playing Tetris (the origin of the name of Mino Vector). The difference is that the fallen intensity value each time is only one block, not four, and we do not bother to find place strategically, but seek the subscript equal to intensity value plus 1 (just the intensity value in C array) and add its value by 1. Subscripts from 257 to 263 are flag bits for statistical information: “total” is the whole number of intensity values collected during a period ensuring its value non-zero after update (i.e., it is meaningless for a model without reference data); “min” and “max” are the minimal and maximal intensity value, separately, of all intensity values collected during a period, which are used to compute the “span”; “cluster” is the number of the group of Minos with successive intensity values and “kurtosis” is the number of peaks quantized from the probability distribution, which are used for monitoring and analyzing the dispersion degree and the shape of distribution of reference data separately; “base” is the number of Minos on floor 1 and used in update process focusing on a key problem, that is how many and which frames should be updated.

For a natural scene, those with dynamic objects like pendulums, escalators, or fountains, bushes, etc., are visually recognized as dy-

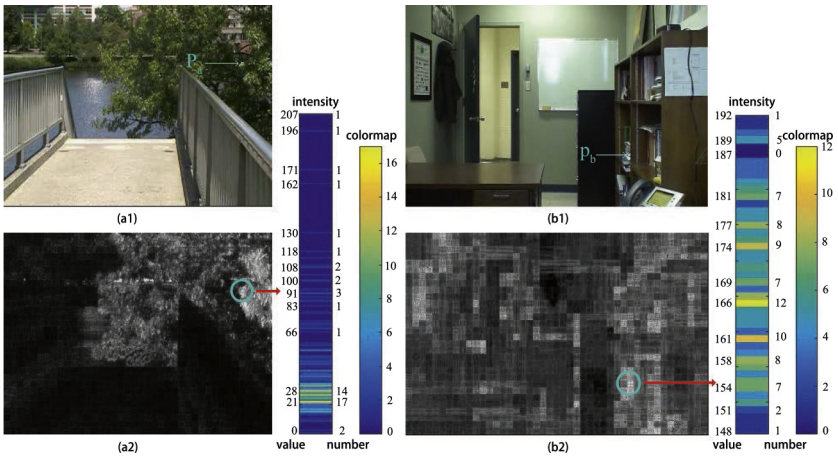

Fig. 3. (a1) and (b1) are two representative scenes for the dynamic and the static, which are from CDnet2012's two categories, *dynamicBackground* and *baseline*; (a2) and (b2) are the visualizations of DN values for (a1) and (b1) over 250 frames and 211 frames separately; coordinates of two points P_a and P_b are (284, 69) and (266, 178); two intensity bars indicate all the intensity values appeared and their number for background scenes (a1) and (a2) over time.

namic scenes (for example, in Fig. 3 (a1), it exists water ripples and swaying twigs) which are challenges in a background subtraction model. Soatto et al. [32] propose dynamic textures to model these dynamic things, but it cannot describe static scene and is somewhat complex (with six parameters). Our objective is to model all the scenes uniformly (without any discrimination between dynamic scenes and static scenes) and easily. To that end, we first propose a dynamic nature (DN) which is the span of the pixel's intensity value and so it can be easily acquired from MV's flag bit "span". Exemplified in Fig. 3, the second row displays the span of every pixel of two scenes, the dynamic and the static, during a period of time. Brighter the pixel is, longer the span. Those areas with brighter pixels are called dynamic areas. Inversely, the remains with dark pixels are static areas. It implies that the criterion of recognizing a dynamic scene is whether there is a dynamic area at least.

Reviewing the first row of Fig. 3, we can notice that the dynamic areas in Fig. 3 (a1) correspond to trees and ripples, while the scene in Fig. 3 (b1) seems stationary. That is, it cannot be explained where these dynamic areas in Fig. 3 (b2) come from. The reason is that, whether an area is dynamic largely depends on our subjective judgment, so the relationship between the dynamic and static area is relative; meanwhile, precisely because of this relationship, areas in static scene (e.g., Fig. 3 (b2)) with higher DN values relative to other static areas become dynamic. In our model, it is described by a second order attribute of DN, named dynamic rank (DR), representing which rank the DN belongs to. For a background scene, when all pixels' DNs are prepared, we can divide the DNs according to a predefined number (indicating the amount of DR and set to 10 in our model) or other strategies. For instance, in two scenes of Fig. 3, we separately choose a point with the largest DN (P_a and P_b). The DN value of P_b is 43 which is far lower than P_a's (208), but they have the same DR that is 10. This phenomenon coincides with a philosophical view that everything is relative. So far, we assert that there is no difference between dynamic scenes and static scenes, and so we can model all the scenes in a unified way, in terms of DN and DR. Apart from this contribution, DR further plays an important role in threshold setting (unveiled in Section 3.3). Then, the algorithm about how to get dynamic ranks of pixels is given in Algorithm 1.

Algorithm 1 Dynamic ranks acquisition.

Input: *Mino Vectors*, *dn_rank_num*
Output: array *dn_ranks*
1: Getting the *largest_DN* from 'span's
2: *largest_DN* ← *largest_DN* > *max_span*?*max_span* : *largest_DN*
3: Calculating the *dn_rank_intvl*: *round*(*largest_DN*/*dn_rank_num*)
4: **for** *each pixel* **do**
5: Getting the span from flag bit 'span'
6: Calculating the *dn_rank*: *dn_rank* ← *ceil*(*span*/*dn_rank_intvl*)
7: **if** *dn_rank* > *dn_rank_num* **then**
8: *dn_rank* ← *dn_rank_num*
9: **end if**
10: **end for**
11: **return** *dn_ranks*

3.2. Optimal bandwidth

In Fig. 2, we can know that the distribution of a point with a large DN value always belongs to multi-model from its kurtoses. However, structures exist in different places by their explicit or implicit change rule. In statistical literature, nonparametric methods are suitable for these arbitrarily structured data, and free from dilemma of selecting how many clusters of Kmeans or Gaussians in GMM. The most attractive one for modeling their distributions is the kernel-based density estimation (KDE) from observed data [13,14].

Assume that an image sequence as background has been gathered into MVs, and let $I = I_1, I_2, \ldots, I_N$ be a sample of intensity values for a pixel. At time t, a candidate point with intensity value I_t will be determined whether or not belonging to background by estimating its probability density function and compared with threshold:

$$\hat{f}(I_t) = \frac{1}{N}\sum_{i=1}^{N} K_H(I_t - I_i), \tag{1}$$

$$mask = \begin{cases} 1, & \hat{f}(I_t) < thresholds(dr_t) \\ 0, & \hat{f}(I_t) \geq thresholds(dr_t) \end{cases}, \tag{2}$$

where K in Eq. (1) is the kernel function and H the bandwidth matrix; in Eq. (2), 1 and 0 denote foreground and background, respectively, parameter dr_t the threshold rank of this point at time t (detailed in Section 3.3).

In [13,15], two types of bandwidth estimators have been introduced: sample-point estimator and balloon estimator, which are both better than fixed bandwidth undoubtedly. Jones [36] has helped to alleviate the confusion caused by these two variable methods, and further proved that the former performs rather bet-

ter than the latter, at least asymptotically. There is a tiny difference in formulae:

$$\begin{cases} H_b = H(I_t), & for\ balloon\ estimator \\ H_s = H(I_i), & for\ sample-point\ estimator \end{cases} \tag{3}$$

H_s is thought of as a function of individual I_is, while H_b as a function of the candidate point I_t where probability density is estimated. Sheikh and Shah [15] adopt none of them for considering the complexity of this technology and computation cost, but in fact, a pilot estimate proposed by Silverman [37] is feasible. For the first time, we choose a normal distribution $\tilde{f}_n$ as the pilot estimate with its variance σ^2 estimated from the observed data. Then an optimal but local bandwidth h_{loc} will be calculated:

$$\begin{cases} h_{loc} = 0.9\zeta N^{-0.2} \\ \zeta = min(\sigma,\ iqr/1.34) \end{cases}, \tag{4}$$

where *iqr* is the interquartile range of the sample I and can be calculated directly by using function $iqr(I)$ in Matlab. This equation can cope well with long-tail effects which exist in areas with large DN. Because these areas always have sparse distributions and the shape of distribution is flat, so leading to long-tail effect.

To get a variable bandwidth, we need to define a factor, η_i, which is controlled by each observed data:

$$\eta_i = [\tilde{f}_n(I_i)/geom]^{\alpha}, \tag{5}$$

where *geom* is the geometric mean of $\tilde{f}_n(I_i)$ and α is a sensitivity parameter (set to -0.5 in our model). Thus, the second equation in Eq. (3) can be instantiated as:

$$H_{s_i} = \eta_i h_{loc}, \tag{6}$$

and further we can rewrite the Eq. (1) by introducing H_s:

$$\hat{f}(I_t) = \frac{1}{NH_s^d}\sum_{i=1}^{N} K\left(\frac{I_t - I_i}{H_{s_i}}\right), \tag{7}$$

where d indicates the dimension and is three in this work (i.e., r, g and b three channels). Although Silverman [37] kindly advises to use Epanechnikov kernel for computational consideration. In our trials, however, E-kernel does not perform better than Gaussian in accuracy. For reducing the computational burden of a Gaussian kernel, we notice that what the structure, MV, saves are the number of intensity values, but not all the observed data. For instance, in Fig. 2, there are 49 data points filtered as reference data but only 16 kinds of intensity values (this number is kept in flag bit *base*). When computing optimal bandwidth, one procedure of each data point subtracted by all of them and then the difference participating further calculation in Gaussian is needed and the time complexity is $O(N^2)$ which is really very substantial in computation. By the merit of MV, the time complexity in our model will reduce to $O(MV[base]^2)$. Incidentally, the point selected for exemplification in Fig. 2 is the minority with large span of intensity values, while in most of scenes, there are many pixels with a short span about 10 and some extremely stable points present only one value for a long time, and so it saves computation substantially. Algorithm is listed below (see Algorithm 2).

Algorithm 2 Optimal bandwidth acquisition.

Input: *Mino Vectors*, *alpha*
Output: array *h*

```
for each pixel do
    Calculating standard deviation sigma and interquartile range iqr
    h_loc ← 0.9 * min(sigma, iqr/1.34) * N^0.2
    Calculating each reference data point's probability density f_i by introducing a normal distribution as a pilot estimate
    Calculating the geometric mean of f_i: geo_mean
    Calculating the optimal bandwidth for sample point estimator: h_i ← h_loc * (f_i/geo_mean)^alpha
end for
return h
```

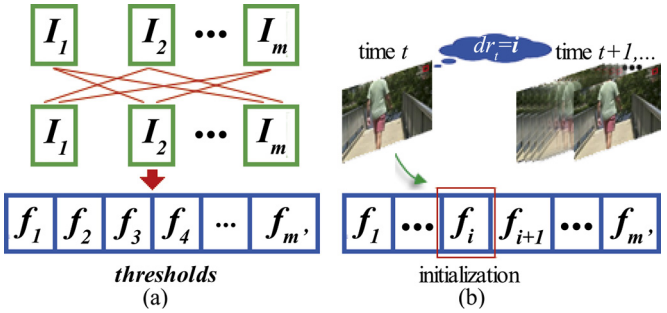

Fig. 4. (a) $I_1, I_2, \ldots, I_m$ are the intensity values of sample I of a pixel. Probability density of each reference data is estimated upon the others, then the results will be pushed into a queue without any repetition from small to big, i.e., $f_1 < f_2 < \cdots < f_{m'}$ where $m' \le m \times (m-1)$. The sorted queue will be used as the *thresholds*. (b) Threshold of each pixel will be initialized by its DR as an index in the *thresholds*. See Algorithm 3 below.

3.3. Thresholds

Whether a good result of classification can be obtained for background subtraction model at last will be dominated by thresholds. To pursue an adaptive and precise threshold selection, we calculate the probability densities within reference data and sort them in ascending order without repetition (see Fig. 4 (a)).

Recall that analyses of DN have helped simplify the model with a unified way when confronting the challenge of dynamic scenes. The other attribute, DR, is designed to cooperate with threshold establishing and to achieve the goal of high adaptability and precision. The DR of a pixel plays a role as a bridge for mapping to the *thresholds*. For instance, for a candidate point with intensity value I_t at time t assigned with the dynamic rank dr_t, its exclusive threshold will be selected as $thresholds[dr_t]$ initially. This idea is derived from the comparison between DR and *thresholds*, which implies that an appropriate division of DN can reflect the underlying law of scenes. Therefore, we try to establish the link between DR and *thresholds* by using part of training data for verification (see Fig. 5).

In Fig. 5, the blue dash line indicates the DR of each pixel point and the red solid line indicates, for each pixel point, the minimal position of 239 probability densities in the *thresholds*. We can notice that these two lines are very close (except for the pixel point 53 of Fig. 5 (a), because the remaining 239 data at this pixel point are very concentrative and lie in the scope of training data resulting in very large probability densities. Therefore we accept it as background directly) and most of them lower than 10 (that's why we set the amount of DR to 10), so the use of DR as index in *thresholds* can make sense and work well for most cases.

Additionally, an experiential statement is that the areas with lower DR are always initialized by smaller thresholds, and vice versa. The reason is that the shapes of distribution of low DR areas in MVs are tall and thin visually, and their bandwidths are also small. This results in a narrow acceptance range around the main part of the distribution of their intensity values. However, in practice a candidate who is not far from the main part should be classified into background, thus a lower threshold is reasonable to satisfy the second condition in formula (2).

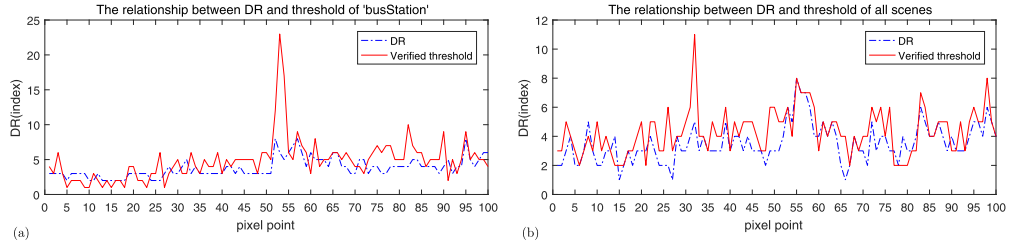

Fig. 5. (a) 100 pixel points are drawn randomly from scene *busStation* in category *Shadow* and then the first 60 frames of them are used for training and remaining 239 frames for verification. (b) for all scenes except *streetLight* in *intermittentObjectMotion*, *peopleInShade* in *shadow* and *park* in *thermal*, as these three scenes have less than 299 training data.

Algorithm 3 Thresholds calculation.

Input: *Mino Vectors*, *h*, *dn_rank_num*
Output: array *thresholds*
1: **for** *each pixel* **do**
2: Calculating the probability of each reference data on the others
3: Sorting ascendingly and getting a unique array: *probs_uarr*
4: Intercepting former *dn_rank_num* of *probs_uarr* as *threshold*
5: **end for**
6: **return** *thresholds*

Algorithm 4 TUS execution.

Input: *Mino Vectors*
Output: updated *Mino Vectors*
1: **for** *each pixel* **do**
2: for each intensity value with non-zero number in *Mino Vector*, reducing them by 1, and them updating the flag bits
3: **end for**
4: **return** *Mino Vectors*

3.4. Tetris update scheme

We propose an easy to understand and interesting update method, called Tetris update scheme (TUS), inspired by the video game Tetris. For obtaining a continuously more robust and effective background subtraction model, the variation of a scene (including illumination change) and random noises need to be considered. TUS partnering with MV provides a good solution about these concerns, and the process is visually illustrated in Fig. 6 and coded in Algorithm 4.

In TUS, the "normal data points" are depressed as well, but there is no side effects to influence the judgment that whether or not an estimated candidate belongs to them. Fig. 6 (a) presents a bimodal data point with 4 noises at t_i and 17 compatriots are recognized at time t_{i+17}. When preparing to verify closely next candidate (i.e., at time t_{i+18}), the update condition is met, and so the candidate has to temporarily stay in the air. The whole fallen Minos, including noises, will go down by 1 step into ground to bury the noises with sacrificing some compatriots settling in the same level. Multi-cluster and multi-modal are the main characters of the dynamic areas where long span of intensity value lowers the altitude of the entire distribution (see Fig. 6 (b) compared to 6 (a)). In Fig. 6 (b), for illustrative convenience, we choose two clusters and the span is shortened for space limit. The update process is analogous to those in Fig. 6 (a) and thus the main underlying structures are unchanged as well. In Fig. 6 (c), we simply and completely illustrate how the model adapts to environmental changes by six consecutive update procedures. By comparison of courses at time t_{i+14} and $t_{>i+75}$, we can notice that an obvious displacement of total fallen Minos appears from left to right at the horizontal axis. This enables our model to keep a close pace with varying scenes.

Besides robustness and effectiveness, another feature, adaptability, is also considered in TUS. However, this feature is always cursed by two fixed-problems in conventional update methods: one is the fixed update cycle, and the other is the fixed update window. Elgammal et al. [1] have set the cycle to 2 and the update window to 50 in their implementation; Sheikh and Shah [15] take a blind way to update and so they are all equal to 1. Evidently, they all ignore the adaptability simply, while in our model these two quantities are at TUS's own discretion. In fact, the first is the problem of when to update. In TUS, the relationship between the number of frames passed over from last update and the largest span of intensity value after last update has been established (e.g., simply set equal) as a condition for update. Noting that our model for dynamic scenes are slowly updated compared with that of static scenes, which seems against our intuition that dynamic scenes should be updated faster than static scenes. The reasons are that, Firstly, in a period of time, a static scene is stable and its model (SM) is easily become redundant, while the model of a dynamic scene (DM) needs more time to obtain more data for modeling changes and so it has less chance becoming redundant relatively. In other words, by using the same update frequency which is suitable for SM will weak the ability of DM to represent the dynamic scene, inversely which is suitable for DM will cause SM redundant. Secondly, according to the MV's update mechanism, the DM will clear its minos faster than the SM if our conventional intuition is applied, which will lead to an unexpected situation of DM being empty. Therefore, in our work, DM is indeed updated slower than SM for ensuring that our model can precisely represent scenes (static or dynamic) and SM will not become redundant (because of faster update frequency). The second is the problem of how many frames to update. From Fig. 6 we have already clearly known that how many and which frames are updated.

Summarily, each part in our framework is detailed above and their algorithms described in pseudo code are also tailed in each subsection. Then, the algorithm of the whole framework can be assembled by those components (See Alg. 1 of Sec. I in supplemental file).

4. Experiments

We test our framework on a PC with a 3.3 GHz Intel Xeon E3-1230 V2 CPU and 24 GB memory, and the environment is Matlab 2016. The dataset is a real-world benchmark - CDnet 2012 [38] which includes 6 video categories and 4 to 6 video sequences in each category. Kindly enough that not only the groundtruth, but also evaluation tools are provided, and so it is convenient to do evaluation and comparison.

We mainly compare our method with three classical work which use the statistics: KDE [1], GMM [7] and PBAS [20], and also two other state-of-the-art methods, SuBSENSE [21] and IUTIS-

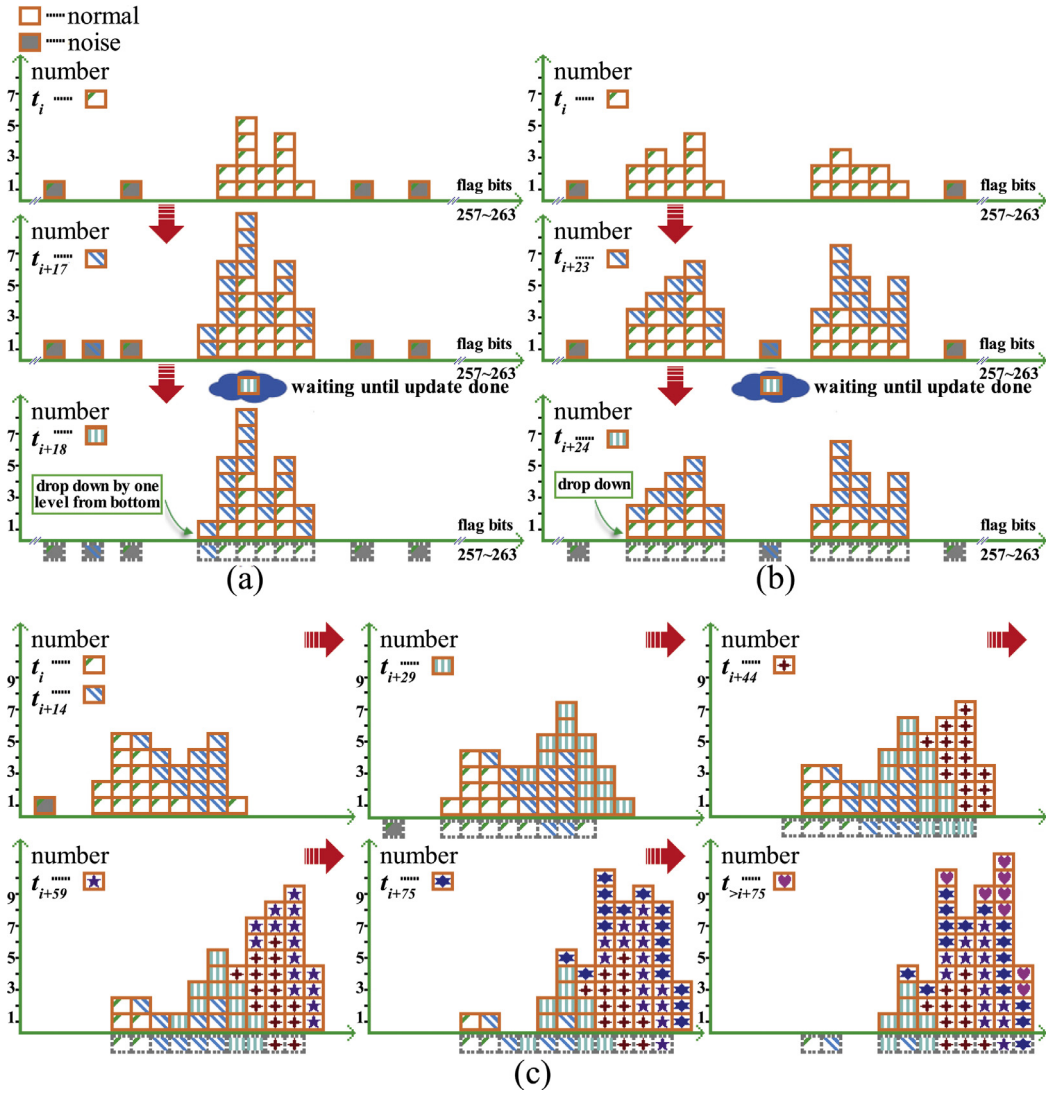

Fig. 6. (a) and (b) display the effects of depressing noises in areas with low DN and high DN respectively; (c) displays the effect of adapting to the variation of the environment. Note that this figure is purely illustrative; the numbers of the kurtosis and cluster are chosen for convenience. Should be viewed in color with dual-amplification.

Table 1
Parameters with the setting value and description.

Para. name	Value	Description
mv_len	263	The length of MV
dn_rank_num	10	The number of rank of dynamic nature
alpha	−0.5	A sensitive value in computing the optimal h
max_minos	300	The maximum of minos preserved in MV
max_span	200	The maximal difference between the smallest and largest intensity values for each pixel for limiting the maximal interval of DRs (see Algorithm 1)

5 [39], which exhibit the best performance in terms of "Recall", "Precision" and "F-Measure" metrics in CDnet 2014 [40].

4.1. Parameters

The parameters used in our framework are listed in Table 1.

Note that although the Minor Vector has a fixed memory cost (i.e., no matter how many the intensity values are), we limit the number of intensity values, *max_minos*, for taking into consideration the computation of geometric mean in calculating the optimal bandwidth, where the number of each intensity value will act as an exponent and too large of it will result in infinity.

4.2. Qualitative

We choose six representative scenes from each category of CDnet 2012, and they are *office* in *baseline, boats* in *dynamicBackground, sidewalk* in *cameraJitter, sofa* in *intermittentObjectMotion, peopleInShade* in *shadow* and *library* in *thermal*, separately. The results of five methods mentioned above are also from CDnet 2012 and are presented in Fig. 7 with ours. We can visually compare them with each other and against the ground truth.

For the scene *office*, although it is an indoor and static scene, changes caused by light exist and become evident when the young man comes in. Another challenge is the separation of young man's trouser legs from the skirting board, and it has not been coped with well to date. Our result keeps an intact contour of the young man (except the part of his legs truncated by skirting board), especially the gesture of the hand in turning the book page by page is separated out.

Fig. 7. Results of five classical methods compared with ours. From column (a) to column (f) are the scenes: *office* (*baseline*), *boats* (*dynamicBackground*), *sidewalk* (*camerJitter*), *sofa* (*intermittentObjectMotion*), *peopleInShade* (*shadow*) and *library* (*thermal*) (strings in parenthesis are category names); from second row to last row are the results of KDE [1], GMM [7], PBAS [20], SuBSENSE [21], IUTIS-5 [39], ours and GroundTruth.

The scene *boats* owns a large part of dynamic areas (i.e., water ripples) and an invader, motorboat, splashes the water with a long tail. Except the method SuBSENSE ignoring that tail, all others catch it. Interestingly, it is absent in GroundTruth.

The category *cameraJitter* is different from the other categories, because not only changes exist in scenes, but also the video acquisition device takes part in increasing the challenge by a global change. All methods suffer from this global change and so as to obtain a high rate of false positive. In this case, our method presents a recognizable result in walker's physical body and his shadow, while KDE and GMM lose the walker, and SuBSENSE and IUTIS-5 inversely lose the shadow of the walker, and PBAS loses both.

The difficulty in the scene *sofa*, apart from the intermittent movements of two men, is that the color of trousers of the sitting person is very close to the color of the sofa. Like the scene *office*, all methods miss a part of the man's legs again, but our result presents a high rate of true positive in sitting man's upper body. Additionally, the walking man is also identified precisely in our result except for some false positives caused by his refection on floor. IUTIS-5 and PBAS also yield good results.

The scene *peopleInShade* presents a synthesized challenge composed of the shade of the building and walking men's shadows on the ground, the change of environmental illumination, and two people standing for a while and then walking away. At first sight, all methods obtain good results, but carefully, we can notice that SuBSENSE almost over recognizing the foreground with two men and their shadows fused together. A tiny detail which will distinguish the rest methods to be good or bad is the right foot of the left person when that foot just leaves the ground. This detail is captured by KDE, GMM and our method, while IUTIS-5 and PBAS blur the boundary of that foot and its shadow.

Scenes in the category *thermal* are all grayscale images which are not conducive to the methods combing multiple features, like textures. In the scene *library*, GMM and SuBSENSE lose the most part of the student's body; KDE obtains the best result; PBAS, IUTIS-5 and our method also perform well, but by viewing the original image at first row of this scene, we can find that the student's thumb is apart from other fingers and this is clear in our result.

As to the computational saving in calculating the probability density by using MV against the conventional way (discussed in Section 3.2), we will show the results of the computational cost of these two manners in Fig. 8 (the quantitative results can be found in Table 3).

Evidently, our method by using MV is faster than the conventional way in runtime of per frame which is shown in Fig. 8 (a). Since the number of computation is reduced sharply which is exhibited in Fig. 8 (b). Noting the calculation number of 8th scene is largest than others but it has a lower runtime, the main reason is that in calculating the exponentiation of Gaussian, if the exponent

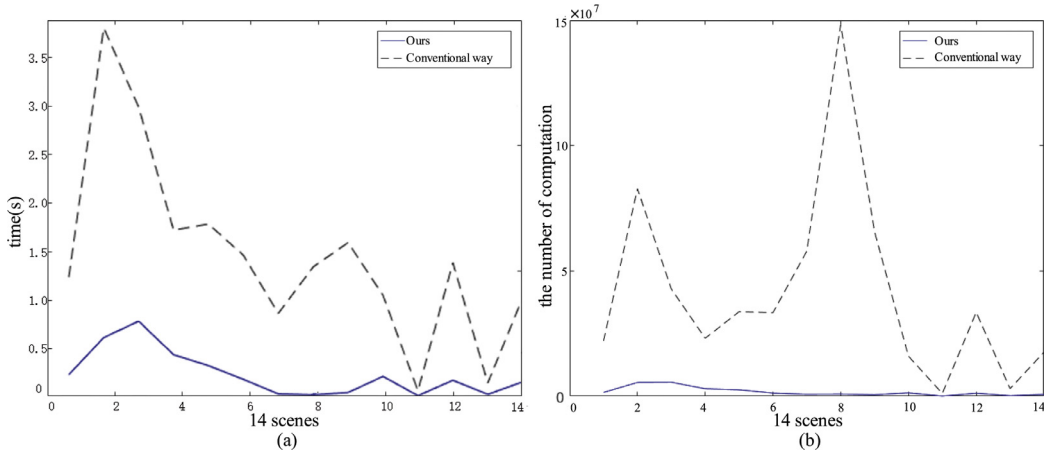

Fig. 8. (a) and (b) show the comparison of two methods in runtime and computational times of each frame for 14 scenes with QVGA resolution which are *highway* in *baseline*, *traffic* in *cameraJitter*, *canoe*, *boats*, *overpass* in *dynamicBackground*, *corridor*, *diningRoom*, *lakeSide*, *library* in *thermal*, *backdoor* in *shadow* and *parking*, *sofa*, *streetLight*, *winterDriveway* in *intermittentObjectMotion*.

Table 2
Results in seven metrics, Recall, Precision, F-Measure, Specficity, False Positive Rate, False Negative Rate and Percentage of Wrong Classifications, of all methods. Bold face denotes the best, and "*" at the top right corner of the number denotes the second-best.

Method	Re	Pr	FM	Sp	FPR	FNR	PWC
KDE [1]	0.7442	0.6843	0.6719	0.9757	0.0243	0.2558	3.4602
GMM [7]	0.6964	0.7079	0.6596	0.9845	0.0155	0.3036	3.1504
PBAS [20]	0.7840	0.8160	0.7532	0.9898	0.0102	0.2160	1.7693
SuBSENSE [21]	0.8281	0.8576*	0.8260	0.9955	0.0045	0.1443	1.0459
IUTIS-5 [39]	0.8471*	**0.8913**	**0.8542**	**0.9967**	**0.0033**	0.0993*	**0.7412**
Ours	**0.8522**	0.8421	0.8343*	0.9961*	0.0039*	**0.0898**	0.7456*

is zero, the computational burden will drop down rapidly. In fact, the runtime is roughly proportional to the number of computation.

4.3. Quantitative

There are seven metrics used in CDnet 2012, but three of them are more representative in describing the accuracy of methods, which are "Recall(Re)", "Precision(Pr)" and "F-Measure(FM)". The rest four metrics are Specficity(Sp), False Positive Rate(FPR), False Negative Rate(FNR) and Percentage of Wrong Classifications(PWC). The definitions of those metrics are as below:

$$Re = \frac{TP}{TP+FN}, Pr = \frac{TP}{TP+FP}, FM = 2 * \frac{Precision \times Recall}{Precision + Recall}, \quad (8)$$

$$Sp = \frac{TN}{TN+FP}, FPR = \frac{FP}{FP+TN}, FNR = \frac{FN}{TP+FN}, \quad (9)$$

$$PWC = 100 * \frac{FN+FP}{TP+FN+FP+TN}, \quad (10)$$

where *TP*, *FN*, *FP* and *TN* are the number of true positive, false negative, false positive and true negative, respectively. According to the definitions of these metrics, we calculate them on all the scenes of CDnet 2012 dataset for all methods, and the results are presented in Table 2 (the detailed results of each scene of our model can be referred to Sec. II of supplemental file).

We can notice that our method achieves a promising result compared with the existing state-of-the-art methods, especially obtains the best in terms of "Recall" and the second in "F-Measure". The main reason of "Precision" in our method smaller than the two best methods (SuBSENSE and IUTIS-5) at present is that our method has a high sensitivity in detecting the object of interest, and obtains more true positives. However, the side effect is the increase of the false positives, leading to the smaller "Precision" by formula (8).

Table 3
Computational time and the number of calculation of probability density per frame for fourteen videos with QVGA resolution ('Conv.' indicating 'conventional way').

Category	Scene	Runtime(s)		The number of calculation	
		Conv.	MV	Conv.	MV
baseline	highway	1.2361	0.2289	22,118,496	1,496,364
dynamicB...	canoe	2.9932	0.7811	42,762,100	5,572,161
	boats	1.7209	0.4351	23,122,643	2,958,446
	overpass	1.7841	0.3217	33,696,672	2,548,443
cameraJitter	traffic	3.8022	0.6118	82,793,120	5,482,059
intermitte...	parking	0.0619	0.0109	1,087,463	49,662
	sofa	1.3861	0.1706	33,462,772	1,170,891
	streetLight	0.1421	0.0248	2,994,693	192,350
	winterDriv...	1.0238	0.1543	17,563,113	713,851
shadow	backdoor	1.0468	0.2107	15,946,590	1,326,905
thermal	corridor	1.4619	0.1798	33,285,982	1,229,954
	diningRoom	0.8608	0.0297	57,915,192	752,017
	lakeSide	1.3421	0.0196	148,175,820	837,121
	library	1.5932	0.0441	65,600,927	568,284

In Fig. 8 we can perceptually know the advantage of MV in computing the probability density, then we will give the quantitative results in Table 3. We can clearly see that the number of calculating the probability density by MV is less than that of by using conventional way, resulting in a speedup in runtime. However, for scenes in *dynamicBackgournd* and *cameraJitter*, it is quasi-realtime because those scenes are so complex and large in DN that the rest number of calculation is still more than that of other relative simple scenes.

5. Conclusion

In this work, we propose a novel framework for modeling the background subtraction. There are several innovations in this

framework, like defining DN and DR, constructing MV, accelerating the Varying KDE (VKDE), adaptively selecting the thresholds and putting forward an update method, named Tetris update scheme (TUS). Experimental results show that our method achieves a promising result compared with contemporary state-of-the-art methods.

However, there are still several concerns worth studying deeply in our further work. Firstly, we need to establish a more clarified relationship between DN and thresholds mathematically. Secondly, in addition to being robust to noises and appearance changes, TUS will be further delved in for other challenges, like bad weather. Thirdly, we will optimize our algorithms to reduce the computational time, and then try to implement the framework by OpenCV with considering GPU for real-time applications.

6. Conflict of interest

None.

Acknowledgments

Firstly we would like to express our sincere appreciation to anonymous reviewers for their insightful comments, which have greatly aided us in improving the quality of the paper. This work was supported by Wuhan Municipal Science and Technology Bureau, China [grant number 2016010101010022]; and National Natural Science Foundation of China [grant number 61373107].

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2018.07.006.

References

[1] A. Elgammal, D. Harwood, L. Davis, Non-parametric model for background subtraction, in: Proceedings of the European Conference on Computer Vision, Springer, 2000, pp. 751–767.
[2] B. Wang, P. Dudek, A fast self-tuning background subtraction algorithm, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014, pp. 395–398.
[3] R. Jain, H.-H. Nagel, On the analysis of accumulative difference pictures from image sequences of real world scenes, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6 (1979) 206–214.
[4] N.M. Oliver, B. Rosario, A.P. Pentland, A Bayesian computer vision system for modeling human interactions, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 831–843.
[5] C.R. Wren, A. Azarbayejani, T. Darrell, A.P. Pentland, Pfinder: real-time tracking of the human body, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 780–785.
[6] C. Stauffer, W.E.L. Grimson, Adaptive background mixture models for real-time tracking, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2, IEEE, 1999, pp. 246–252.
[7] Z. Zivkovic, Improved adaptive gaussian mixture model for background subtraction, in: Proceedings of the Seventeenth International Conference on Pattern Recognition, ICPR 2004., 2, IEEE, 2004, pp. 28–31.
[8] M. Heikkila, M. Pietikainen, A texture-based method for modeling the background and detecting moving objects, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4) (2006) 657–662.
[9] H.-H. Lin, J.-H. Chuang, T.-L. Liu, Regularized background adaptation: a novel learning rate control scheme for Gaussian mixture modeling, IEEE Trans. Image Process. 20 (3) (2011) 822–836.
[10] X. Liu, C. Qi, Future-data driven modeling of complex backgrounds using mixture of Gaussians, Neurocomputing 119 (2013) 439–453.
[11] A. Shimada, H. Nagahara, R.-i. Taniguchi, Background modeling based on bidirectional analysis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1979–1986.
[12] S. Varadarajan, P. Miller, H. Zhou, Region-based mixture of Gaussians modelling for foreground detection in dynamic scenes, Pattern Recognit. 48 (11) (2015) 3488–3503.
[13] A. Mittal, N. Paragios, Motion-based background subtraction using adaptive kernel density estimation, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., 2, IEEE, 2004, pp. 295–302.
[14] M.P. Wand, M.C. Jones, Kernel Smoothing, Crc Press, 1994.
[15] Y. Sheikh, M. Shah, Bayesian modeling of dynamic scenes for object detection, IEEE Trans. Pattern Anal. Mach. Intell. 27 (11) (2005) 1778–1792.
[16] P. Hall, M.P. Wand, On the accuracy of binned kernel density estimators, J. Multivar. Anal. 56 (2) (1996) 165–184.
[17] S.R. Sain, Multivariate locally adaptive density estimation, Comput. Stat. Data Anal. 39 (2) (2002) 165–186.
[18] A. Criminisi, G. Cross, A. Blake, V. Kolmogorov, Bilayer segmentation of live video, in: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1, IEEE, 2006, pp. 53–60.
[19] O. Barnich, M. Van Droogenbroeck, Vibe: a universal background subtraction algorithm for video sequences, IEEE Trans. Image Process. 20 (6) (2011) 1709–1724.
[20] M. Hofmann, P. Tiefenbacher, G. Rigoll, Background segmentation with feedback: the pixel-based adaptive segmenter, in: Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2012, pp. 38–43.
[21] P.-L. St-Charles, G.-A. Bilodeau, R. Bergevin, Subsense: a universal change detection method with local adaptive sensitivity, IEEE Trans. Image Process. 24 (1) (2015) 359–373.
[22] Y. Sheikh, O. Javed, T. Kanade, Background subtraction for freely moving cameras, in: Proceedings of the 2009 IEEE Twelfth International Conference on Computer Vision, IEEE, 2009, pp. 1219–1225.
[23] C. Cuevas, N. García, Improved background modeling for real-time Spatio-temporal non-parametric moving object detection strategies, Image Vis. Comput. 31 (9) (2013) 616–630.
[24] B. Xin, Y. Tian, Y. Wang, W. Gao, Background subtraction via generalized fused Lasso foreground modeling, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4676–4684.
[25] M. Van Droogenbroeck, O. Paquot, Background subtraction: experiments and improvements for vibe, in: Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, 2012, pp. 32–37.
[26] L. Guo, D. Xu, Z. Qiang, Background subtraction using local SVD binary pattern, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016, pp. 86–94.
[27] T. Bouwmans, Recent advanced statistical background modeling for foreground detection-a systematic survey, Recent Patent. Comput. Sci. 4 (3) (2011) 147–176.
[28] T. Bouwmans, Traditional and recent approaches in background modeling for foreground detection: an overview, Comput. Sci. Rev. 11 (2014) 31–66.
[29] M. Chen, Q. Yang, Q. Li, G. Wang, M.-H. Yang, Spatiotemporal background subtraction using minimum spanning tree and optical flow, in: Proceedings of the European Conference on Computer Vision, Springer, 2014, pp. 521–534.
[30] G. Shu, A. Dehghan, M. Shah, Improving an object detector and extracting regions using superpixels, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3721–3727.
[31] S. Feldman-Haber, Y. Keller, A probabilistic graph-based framework for plug-and-play multi-cue visual tracking, IEEE Trans. Image Process. 23 (5) (2014) 2291–2301.
[32] S. Soatto, G. Doretto, Y.N. Wu, Dynamic textures, in: Proceedings of the Eighth IEEE International Conference on Computer Vision, 2, IEEE, 2001, pp. 439–446.
[33] A.B. Chan, N. Vasconcelos, Modeling, clustering, and segmenting video with mixtures of dynamic textures, IEEE Trans. Pattern Anal. Mach. Intell. 30 (5) (2008) 909–926.
[34] A. Mumtaz, W. Zhang, A.B. Chan, Joint motion segmentation and background estimation in dynamic scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 368–375.
[35] M. Babaee, D.T. Dinh, G. Rigoll, A deep convolutional neural network for video sequence background subtraction, Pattern Recognit. 76 (2017) 635–649.
[36] M. Jones, Variable kernel density estimates and variable kernel density estimates, Aust. J. Stat. 32 (3) (1990) 361–371.
[37] B.W. Silverman, Density Estimation for Statistics and Data Analysis, 26, CRC press, 1986.
[38] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, P. Ishwar, Change detection. net: a new change detection benchmark dataset, in: Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2012, pp. 1–8.
[39] S. Bianco, G. Ciocca, R. Schettini, How far can you get by combining change detection algorithms? in: Proceedings of the International Conference on Image Analysis and Processing, Springer, 2017, pp. 96–107.
[40] Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, P. Ishwar, Cdnet 2014: an expanded change detection benchmark dataset, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014, pp. 387–394.

Guian Zhang is a Ph.D. student at the School of Computer Science, Wuhan University, China. His research interests include computer vision and image processing.

Zhiyong Yuan is currently a professor at the School of Computer Science, Wuhan University, China. His research interests include virtual reality, human-computer interaction, embedded system, internet of things technology, machine learning and pattern recognition.

Qianqian Tong is a Ph.D. student at the School of Computer Science, Wuhan University, China. Her research interests include human-computer interaction, virtual reality and embedded system.

Mianlun Zheng is a master student at the School of Computer Science, Wuhan University, China. Her research interests include computer graphics and virtual reality.

Jianhui Zhao is currently an associate professor at the School of Computer Science, Wuhan University, China. His research interests include Computer graphics, digital image processing, human-computer interaction, virtual reality.