Open access • Journal Article • DOI:10.1109/TCSVT.2011.2105591

Kernel Similarity Modeling of Texture Pattern Flow for Motion Detection in Complex Background — Source link

Baochang Zhang, Yongsheng Gao, Sanqiang Zhao, Bineng Zhong

Institutions: Beihang University, Griffith University, Harbin Institute of Technology

Published on: 01 Jan 2011 - IEEE Transactions on Circuits and Systems for Video Technology (IEEE)

Topics: Kernel method, Motion detection, Motion estimation, Background subtraction and Kernel (image processing)

Kernel Similarity Modeling of Texture Pattern Flow for Motion Detection in Complex Background

Baochang Zhang[1], Yongsheng Gao[2,3], Sanqiang Zhao[2,3], Bineng Zhong[4]

[1]*National Key Laboratory of Science and Technology on Integrated Control Technology, School of Automation Science and Electrical Engineering, Beijing 100191, China*
[2]*Griffith School of Engineering, Griffith University, Nathan Campus, Brisbane, QLD 4111, Australia*
[3]*Queensland Research Laboratory, National ICT Australia (NICTA), Brisbane, QLD 4072, Australia*
[4]*Department of Computer Science and Engineering, Harbin Institute of Technology, Harbin, Heilongjiang Province 150001, China*

Abstract: *This paper proposes a novel Kernel Similarity Modeling of Texture Pattern Flow (KSM-TPF) for background modeling and motion detection in complex and dynamic environments. The Texture Pattern Flow encodes the binary pattern changes in both spatial and temporal neighborhoods. The integral histogram of Texture Pattern Flow is employed to extract the discriminative features from the input videos. Different from existing uniform threshold based motion detection approaches which are only effective for simple background, the Kernel Similarity Modeling is proposed to produce an adaptive threshold for complex background. The adaptive threshold is computed from the mean and variance of an extended Gaussian Mixture Model. The proposed KSM-TPF approach incorporates machine learning method with feature extraction method in a homogenous way. Experimental results on the publicly available video sequences demonstrate that the proposed approach provides an effective and efficient way for background modeling and motion detection.*

I. Introduction

Moving object detection from video stream or image sequence is one of the main tasks in many computer vision applications such as people tracking, traffic monitoring, scene analysis, and video surveillance. To achieve high sensitivity in the detection of moving objects while maintaining low false alarm rates, the background image has to be either a stationary scene without moving objects or regularly updated in order to adapt to dynamic environments such as swaying tree, rippling water, flickering monitors and shadows of moving objects. The most widely adopted approach for moving object detection is based on *background subtraction* [3], [17], [18]. An underlying assumption of background subtraction is that the images of the scene without the foreground objects exhibit a certain behavior that can be well described by a statistical model (i.e. background modeling). After the statistical model is created, moving objects can be detected by spotting the image parts that do not fit the model (i.e. foreground detection).

In the literature, many motion detection methods with adaptive thresholds have been proposed for both un-compressed [2] and compressed image sequences [16]. Tian and Hampapur [15] proposed a real-time algorithm to detect salient motion in complex backgrounds by combining temporal difference imaging and a temporal filtered motion field. A large number of background subtraction methods have also been proposed to model the dynamic backgrounds. These methods can be classified into two categories: *parametric* methods and *non-parametric* methods. A popular parametric technique is to model each pixel in a video frame with a Gaussian distribution and update the Gaussian parameters recursively with an adaptive filter [18], which has become a de facto baseline method in background modeling. However, the Gaussian model does not work well in dynamic natural environments, where the scene background is not completely stationary. A Gaussian Mixture Model (GMM) method was proposed in [14] to model dynamic background and applied to complex problems such as traffic monitoring. The parameters of GMM were updated using an on-line *K*-means approximation. In stead of using a fixed number of Gaussians to model each pixel, Zivkovic and van der Heijden [20] proposed a recursive method to automatically decide the number of Gaussians. Different from the above parametric methods, Elgammal et al. [4] proposed a non-parametric approach for background modeling, which utilizes a non-parametric kernel density estimation technique to build a statistical representation of the scene background. Mittal and Paragios [7] proposed an improved kernel density estimation, where the kernel has a variable bandwidth. Han et al. [5] proposed a sequential approach using an iterative algorithm to predict the modes of kernel density estimation to overcome the space complexity problem. Sheikh and Shah [13] proposed a Bayesian framework and kernel density estimation method for efficient motion detection without considering the relationship between two neighborhood pixels. Heikkilä and Pietikäinen [6] proposed to model the dynamic background using the Local Binary Pattern (LBP) histograms. The LBP operator, encoding the relationship between the central point and its neighbors, was originally designed for the texture description [8]. It was later extended to other applications including texture classification and face analysis [9], [11], [19]. In background modeling, the LBP method achieved a much better performance than GMM in terms of false negative and positive rates [6]. However, the LBP method does not consider the temporal variations in its pattern, which contains inherent motion information derived from moving objects in the video.

In this paper, we propose a novel Texture Pattern Flow (TPF) to encode the local pattern information in both spatial and temporal domains. In the spatial domain, the TPF is defined as a directional texture descriptor to reflect the fact that the foreground object always moves towards a certain direction. In the temporal domain, the TPF captures motion information over time in the image sequence. To speed up the proposed approach, the computationally efficient integral histogram [12] of TPF is employed to statistically model the image sequence. The complexity of integral histogram is linear to the length of the input data and computationally superior to the conventional histogram method that is exhaustive. Inspired from the property of histogram intersection as a kernel function, we propose to incorporate the histogram similarity measure into our background modeling framework. In stead of using a uniform threshold that treats different image regions and video frames equally and thus not effective for complex background, we propose to use the Kernel Similarity Modeling (KSM) to provide an adaptive threshold computed from the mean and variance of an Extended Gaussian Mixture Model (E-GMM). Background classification is based on a decision rule that a bigger similarity value has more contribution to the final decision. The flowchart of the proposed KSM-TPF approach is displayed in Fig. 1, in which both spatial and temporal information is exploited to construct the TPF features. We compared the proposed approaches with the LBP and GMM background modeling methods on three video sequences from two publicly available databases. It is a very encouraging finding that the proposed background subtraction approach performed consistently superior to the benchmark methods in all the comparative experiments.

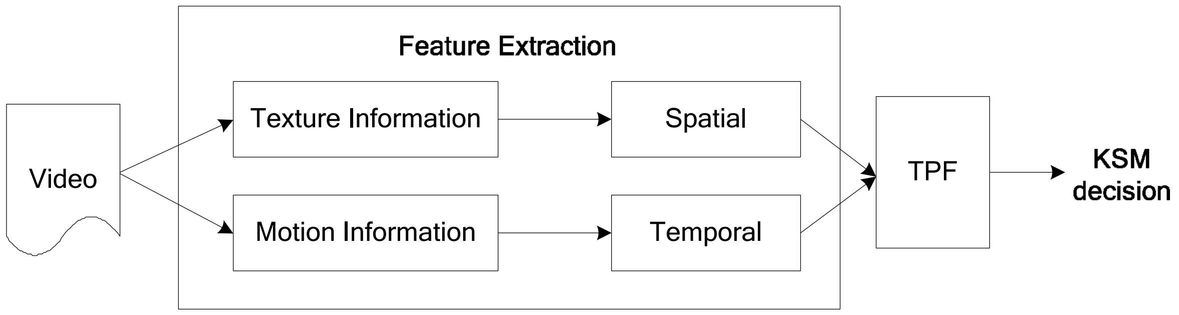


Fig. 1. The flowchart of the proposed KSM-TPF approach.

The remainder of this paper is organized as follows. Section II describes the concept of texture pattern flow for feature extraction. Section III presents the proposed kernel similarity modeling method for motion detection. Section IV gives comparative experimental results. Conclusions are drawn in Section V.

II. TEXTURE PATTERN FLOW

The goal of background modeling is to construct and maintain a statistical representation of the scene that the camera sees. In this section, we utilize the texture information for dynamic background modeling and motion detection. We propose a new operator, Texture Pattern Flow (TPF), for the measure of texture flow. TPF encodes texture and motion information in a local region from both spatial and temporal aspects. The integral histogram of TPF computed within a local region around the pixel is employed to extract the statistical discriminative features from the image sequence.

2.1. TPF Operator

Given a gray-scale image sequence $I_{x,y,t}$ where x, y and t represent space and time indexes respectively, the gradients in x and y directions can be calculated by

$$\frac{\partial I_{x,y,t}}{\partial x} = I_{x,y,t} - I_{x-1,y,t}, \tag{1}$$

$$\frac{\partial I_{x,y,t}}{\partial y} = I_{x,y,t} - I_{x,y-1,t}. \tag{2}$$

To capture the spatial variations in x and y directions, two thresholding functions, $f_x(\frac{\partial I_{x,y,t}}{\partial x})$ and $f_y(\frac{\partial I_{x,y,t}}{\partial y})$, are employed to encode the gradient information into binary representations:

$$f_x(\frac{\partial I_{x,y,t}}{\partial x}) = \begin{cases} 1, & if \ \frac{\partial I_{x,y,t}}{\partial x} \geq 0 \\ 0, & if \ \frac{\partial I_{x,y,t}}{\partial x} < 0 \end{cases}, \tag{3}$$

$$f_y(\frac{\partial I_{x,y,t}}{\partial y}) = \begin{cases} 1, & if \ \frac{\partial I_{x,y,t}}{\partial y} \geq 0 \\ 0, & if \ \frac{\partial I_{x,y,t}}{\partial y} < 0 \end{cases}. \tag{4}$$

The temporal derivative is defined as

$$\frac{\partial I_{x,y,t,i}}{\partial t} = I_{x,y,t,i} - \mu_{x,y,t-1,i}, \tag{5}$$

where $\mu_{x,y,t-1,i}$ is the mean of the i^{th} model in the Gaussian Mixture Model (GMM) at time $t-1$. GMM is exploited to model the distribution of the recent history of each pixel value over time as

$$P(I_{x,y,t}) = \sum_{i=1}^{K} \omega_{x,y,t,i} \varphi_{\mu_{x,y,t,i},\sigma^2_{x,y,t,i}}(I_{x,y,t}), \tag{6}$$

where K is the number of Gaussian models; $\omega_{x,y,t,i}$ is an estimate of the weight of the i^{th} model at time t; $\sigma^2_{x,y,t,i}$ is the variance of the i^{th} Gaussian probability density function $\varphi(\bullet)$ at time t; and $\varphi(\bullet)$ is defined as

$$\varphi_{\mu_{x,y,t},\sigma^2_{x,y,t}}(I_{x,y,t}) = \frac{1}{\sqrt{2\pi}\sigma_{x,y,t}} \exp\left(-\frac{(I_{x,y,t}-\mu_{x,y,t})^2}{2\sigma^2_{x,y,t}}\right). \tag{7}$$

In this way, the distribution of recently observed values of each pixel in the image sequence is characterized by a mixture of K Gaussian distributions. A new pixel value will be represented by one of the major components of the mixture model and used to update the model. Each new pixel in the image sequence is checked against the existing K Gaussian distributions until a match is found. If a pixel matches two mixtures, either Gaussian distribution can be used. A pixel value lying within 2.5 standard deviations ($2.5\sigma_{x,y,t}$) of a distribution is defined as a match. The thresholding function $f_t(\frac{\partial I_{x,y,t}}{\partial t})$ is used to encode the temporal derivative information ($I_{x,y,t} - \mu_{x,y,t-1,i}$) into binary representations based on whether a match can be found:

$$f_t\left(\frac{\partial I_{x,y,t}}{\partial t}\right) = \begin{cases} 1, & if \ \frac{\partial I_{x,y,t}}{\partial t} \geq \pm 2.5\sigma_{x,y,t-1,i} \\ 0, & if \ \frac{\partial I_{x,y,t}}{\partial t} < \pm 2.5\sigma_{x,y,t-1,i} \end{cases}. \tag{8}$$

If a pixel is checked with a match being found, the thresholding function gets 0; otherwise, the thresholding function gets 1.

After the temporal information is encoded, the current pixel is used to update the parameters of GMM through an online K-means approximation [14]. If a match has been found for the pixel, the parameters for the unmatched Gaussian distributions remain the same, while the mean and variance of the matched Gaussian distribution at time *t* are updated as follows:

$$\mu_{x,y,t} = (1-\rho)\mu_{x,y,t-1} + \rho I_{x,y,t}\,, \tag{9}$$

$$\sigma^2_{x,y,t} = (1-\rho)\sigma^2_{x,y,t-1} + \rho(I_{x,y,t} - \mu_{x,y,t})^2, \tag{10}$$

where ρ is the learning rate. The bigger the learning rate is, the faster the propagation speed. The weights of GMM at time *t* are updated as follows

$$\omega_{x,y,t,i} = (1-\alpha)\omega_{x,y,t-1,i} + \alpha(1 - f_t(\frac{\partial I_{x,y,t}}{\partial t}))\,, \tag{11}$$

where α is the second learning rate in direct proportion to ρ. All the weights are then re-normalized. If none of the *K* Gaussian distributions match the current pixel value, the least probable distribution is replaced with a new distribution whose mean is set to the pixel value, with an initial high variance and low prior weight.

By integrating both spatial and temporal information, the Texture Pattern Flow (TPF) is defined as

$$TPF(I_{x,y,t}) = \{f_x(\frac{\partial I_{x,y,t}}{\partial x}), f_x(\frac{\partial I_{x+1,y,t}}{\partial x}), f_y(\frac{\partial I_{x,y,t}}{\partial y}), f_y(\frac{\partial I_{x,y+1,t}}{\partial y}), f_t(\frac{\partial I_{x,y,t}}{\partial t})\}\,. \tag{12}$$

TPF reveals the relationship between derivative directions in both spatial and temporal domains. Fig. 2 illustrates the neighborhood pixels used for the TPF encoding in both the spatial domain and the temporal domain. To make TPF more robust against the negligible changes in pixel values, we add a perturbation factor (β) to the first order derivative in the spatial domain, e.g., $\frac{\partial I_{x,y,t}}{\partial x} = I_{x,y,t} - I_{x-1,y,t} + \beta$.

In our experiments, the perturbation factor is set to $\beta = 3$, the same as in [6].

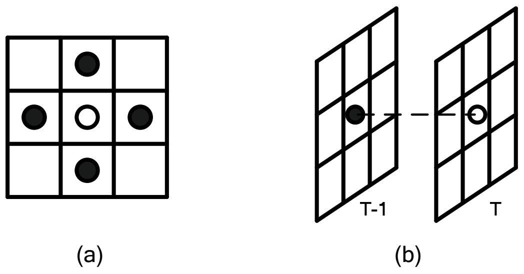


Fig. 2. The neighborhood pixels used for encoding the Texture Pattern Flow in the spatial domain (a) and the temporal domain (b).

2.2. Integral Histogram of TPF

After the TPF is calculated from each pixel of the input video, the discriminative statistics of TPF features are extracted as the video representation. To efficiently calculate the statistics, we extract the integral histogram [12] of TPF from a neighborhood region around each pixel $I(x, y)$. The integral histogram is more computationally efficient than the conventional histogram method in that its complexity is linear to the length of the data. Fig. 3 illustrates an example of the neighborhood region around a pixel for integral histogram extraction used in this paper. The typical values of width (w) and height (h) of a neighborhood region are $w = 8$ and $h = 8$. Using a neighborhood region instead of a pixel provides certain robustness against noise, as the neighborhood region has an effect similar to a smoothing window. However, when the local region is too large, more local details will be lost. Therefore, the selection of these two parameters is a compromise between noise reduction and detail preservation. We calculate TPF features based on the gray-level image in order to increase the robustness against complex and dynamic backgrounds. For simplicity, we model each pixel of the video frame in the same way, which allows for a high-speed parallel implementation if needed.

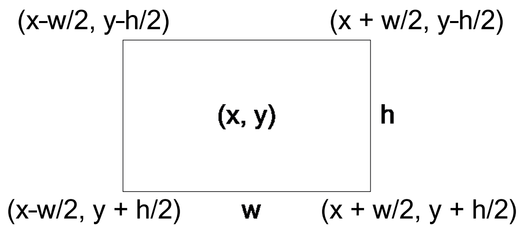


Fig. 3. An example of the local region around a pixel $I(x, y)$ for integral histogram extraction.

The similarity between two integral histograms $h1$ and $h2$ is calculated through the following similarity function:

$$HI(h1, h2) = \sum_{b=1}^{B} \min(h1_b, h2_b), \tag{13}$$

where $HI(h1, h2)$ is the histogram intersection operation and B is the number of histogram bins. This measure has an intuitive motivation in that it calculates the common parts of two histograms. Its advantage is that the computation is very fast as it only requires very simple arithmetic operations. Histogram intersection has been proved to be a Mercer's kernel [1], and thus there is highly effective tricks for computing scalar products in histogram spaces using powerful kernel functions.

III. Kernel Similarity Modeling

After the TPF integral histogram features are extracted to represent each pixel of the input video frame, a background model is established and dynamically adapted to complex situations. With new frames being processed, the background model is updated over time. The similarity measurement between the input video and the background model is conducted by a new Kernel Similarity Modeling (KSM) approach proposed in this section. The KSM approach integrates the TPF integral histogram features with a kernel similarity estimation method for dynamic similarity measurement. In the following subsections, we first introduce how to use TPF integral histogram features to model the background. The updating of background model is also presented in this process. We then describe how to use an adaptive threshold to distinguish a foreground pixel from background. The flowchart of the proposed KSM approach is shown in Fig. 4. In this flowchart, a Gaussian Mixture Model (GMM) is used to model the background into the *model integral histograms*, while an Extended Gaussian Mixture Model (E-GMM) is used to model the *histogram intersection kernel*. The similarity value between the input integral histogram and the model integral histograms is computed by the histogram intersection kernel, and implemented by the online *K*-means approximation [14]. Whether a pixel is background is decided by whether the computed similarity value is above an adaptive threshold. The procedures illustrated in Fig. 4 are repeated for each pixel of the input video frame. The pixel-wise approach usually offers more accurate results for shape extraction.

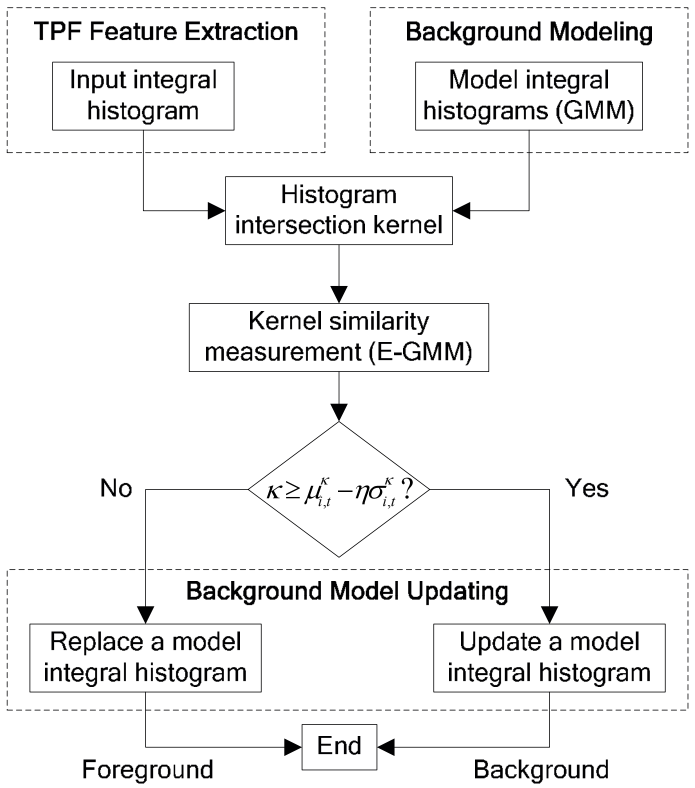

TPF Feature Extraction
Input integral histogram
Background Modeling
Model integral histograms (GMM)
Histogram intersection kernel
Kernel similarity measurement (E-GMM)
No
$\kappa \geq \mu_{i,t}^{\kappa} - \eta\sigma_{i,t}^{\kappa}$?
Yes
Background Model Updating
Replace a model integral histogram
Update a model integral histogram
End
Foreground
Background

Fig. 4. The flowchart of the Kernel Similarity Modeling for one pixel.

3.1. Background Modeling and Updating

In the KSM-TPF framework, each pixel of the video frame is modeled by a GMM of the TPF integral histograms, in the form of a set of model integral histograms with associated weights. The model integral histograms can be established from a training process. Once the TPF integral histogram is extracted from a pixel in an input video frame (called *input integral histogram*), it is compared with the model integral histograms through a similarity measurement (histogram intersection kernel). If the similarity value is above a threshold, the pixel will be considered as part of a background. Otherwise, it will be considered as part of foreground objects. The threshold that is computed adaptively from kernel similarity modeling will be described in the next subsection.

After the input pixel is classified either as part of background or foreground, the background model will be updated accordingly. If the pixel is labeled as foreground (i.e. the similarity value is less than the threshold for all the model integral histograms), a new model integral histogram is created with a low initial weight to replace the model integral histogram with the lowest weight. If the pixel is labeled as background (i.e. the similarity value is above the threshold for at least one of the model integral histograms), the model integral histogram with the highest similarity value is selected as the *best-matched model histogram*. The best-matched model histogram is updated with the new data through

$$\bar{H}_{i,t} = (1-\rho)\bar{H}_{i,t-1} + \rho H_t\,, \tag{14}$$

where $\bar{H}_{i,t}$ and H_t are the mean integral histogram of the i^{th} Gaussian distribution and the input integral histogram at time *t*, respectively [6]. The other model histograms remain the same. The weight $\omega_{i,t}$ of each Gaussian distribution at time *t* is updated through

$$\omega_{i,t} = (1-\alpha)\omega_{i,t-1} + \alpha(M_{i,t})\,, \tag{15}$$

where $M_{i,t}$ is set to 1 for the best-matched distribution, and set to 0 for the other distributions. All the weights are then re-normalized. Note Equation (15) gives a bigger weight to the best-matched distribution, and gives smaller weights to the other distributions. In practice, we sort the model integral histograms in descending order according to their weights. The first *N* model integral histograms are chosen to model the background:

$$\omega_{0,t} + \omega_{1,t} + \cdots + \omega_{N-1,t} > T_N , \tag{16}$$

where T_N is a user defined threshold. T_N is used to confine the number of Gaussian distributions *N*.

3.2. Adaptive Threshold Based Kernel Similarity Measurement

We use the variable κ to represent the *kernel similarity value* (which is measured by the *histogram intersection operation HI* in Equation (13)) between the input integral histogram *H* and a model integral histogram $\bar{H}$. The E-GMM model consists of three elements, $\mu_{i,t}^{\kappa}$, $\sigma^{2\kappa}_{i,t}$ and $\bar{H}$, where $\mu_{i,t}^{\kappa}$ and $\sigma^{2\kappa}_{i,t}$ represent the mean and the variance of κ corresponding to the i^{th} Gaussian distribution at time *t*, respectively. $\bar{H}$ is used to reserve both spatial and temporal information contained in TPF integral histograms. Both the mean $\mu_{i,t}^{\kappa}$ and the variance $\sigma^{2\kappa}_{i,t}$ in E-GMM are updated over time in a propagation way through the online *K*-means approximation [14]. The mean kernel similarity value is updated through

$$\mu_{i,t}^{\kappa} = (1-\rho)\mu_{i,t-1}^{\kappa} + \rho\kappa . \tag{17}$$

The variance of the similarity value in E-GMM is updated through

$$\sigma^{2\kappa}_{i,t} = (1-\rho)\sigma^{2\kappa}_{i,t-1} + \rho(\kappa - \mu_{i,t}^{\kappa})^2 . \tag{18}$$

The propagation speed is controlled by the learning rate ρ.

Traditionally, a uniform threshold is used in the update procedure (e.g. $\kappa \geq Threshold$). If the kernel similarity value κ is higher than the threshold for at least one model integral histogram, the input pixel is classified as background; otherwise, the input pixel is classified as foreground. However, the uniform threshold is not effective for complex and dynamic background, because different image regions have different ranges of background variation. In this paper, an adaptive threshold is used for kernel similarity measurement to segment the background from foreground objects. Specifically, the threshold is computed from the mean and the standard deviation of the histogram intersection kernel:

$$\kappa \geq \mu_{i,t}^{\kappa} - \eta \sigma_{i,t}^{\kappa}, \tag{19}$$

where $\sigma_{i,t}^{\kappa}$ is the standard deviation of the i^{th} Gaussian distribution, and η is a parameter to balance the mean value and the standard deviation. The adaptive threshold $\mu_{i,t}^{\kappa} - \eta \sigma_{i,t}^{\kappa}$ is used to reflect the statistics of dynamic similarity value through kernel similarity modeling.

IV. Experiments

Multiple video sequences from three publicly available databases are used to assess the proposed approach with both visual evaluation and numerical evaluation. The first video dataset, a synthesized dataset, is taken under a dynamic background with a computer graphics synthesized object. It is downloaded from the website at http://mmc36.informatik.uni-augsburg.de/VSSN06_OSAC/. The other two video datasets are the Wallflower video dataset [17] and the DynTex video dataset [10]. The reason of using a computer graphics synthesized video object for testing is that it can provide very accurate pixel-level ground truth for all the video frames (that manual labeling cannot achieve) for quantitative accuracy comparison with benchmark algorithms. For the realistic videos, the ground truth is provided by manually labeling one or several frames from the video sequence. The numerical evaluation is conducted in terms of *false negatives* (the number of foreground pixels that were misclassified as background) and *false positives* (the number of background pixels that were misclassified as foreground.

In all the following experiments, the test phases begin from the 50th frame of each video sequence. The following two parameters are used the same as in Reference [6]: the threshold $T_N = 0.7$ and the propagation rate $\rho = 0.01$. The remaining parameters in our approach can be heuristically determined by experiments. For example, the parameter η used to balance the mean value and the standard deviation has been demonstrated relatively stable in our experiment in the next subsection. For simplicity, three Gaussian distributions and three model integral histograms are used to describe all the Gaussian Mixture Models in both feature extraction and classification stages.

We compare the proposed approach with three benchmark methods, the Gaussian Mixture Model (GMM) method [14], the Carnegie Mellon University (CMU) method [2], and the Local Binary Pattern (LBP) method [6]. We also implement a simplified version of the proposed approach, named Texture Pattern Flow (TPF). The TPF method differs from the full version of the proposed approach only in that it uses a uniform threshold (i.e. $\kappa \geq Threshold$) to decide whether a pixel is background or foreground. It should be noted that all our experiments in this paper are conducted on gray-level values, although other implementation techniques (e.g. by concatenating RGB color information as used in [17]) may obtain a better performance.

4.1. Experiment 1

The first experiment is conducted on Video 3 (totally 898 frames) and Video 4 (totally 818 frames) from the synthesized dataset. These two video sequences focus on the outdoor scenes and contain sudden background changes and swaying trees. Fig. 5(a) shows an example frame in the Video 3. Two frames from each sequence are manually labeled as the ground truth.

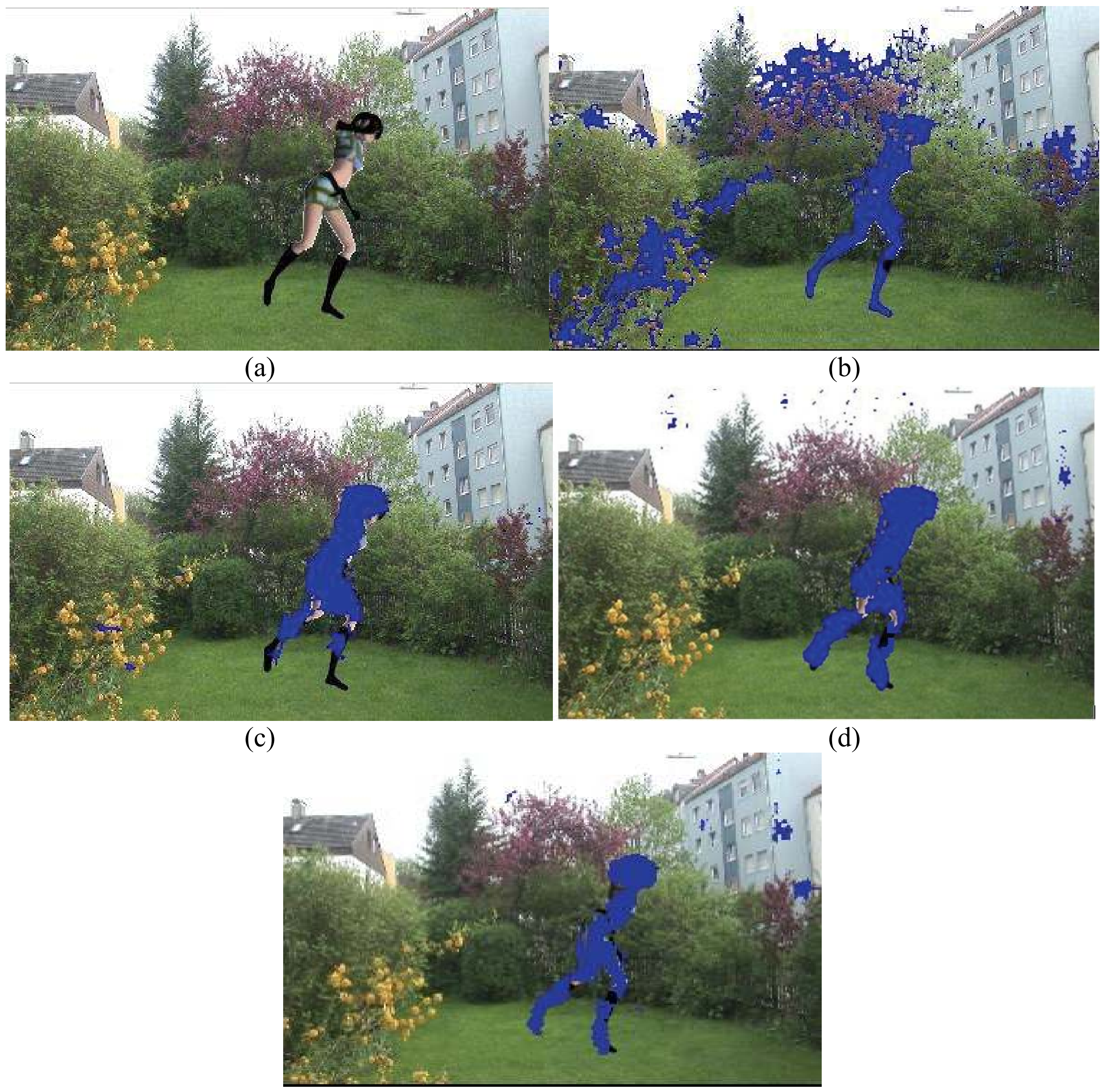
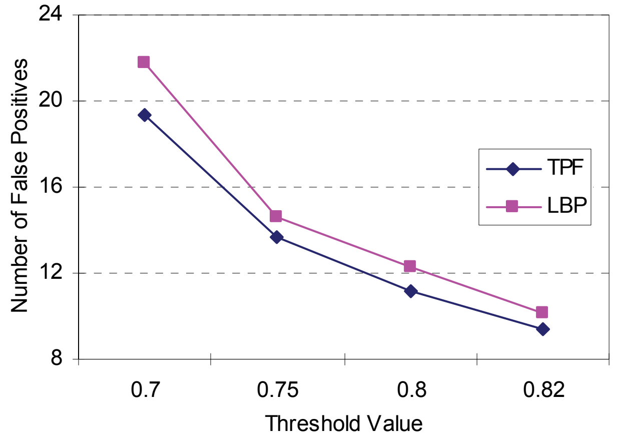
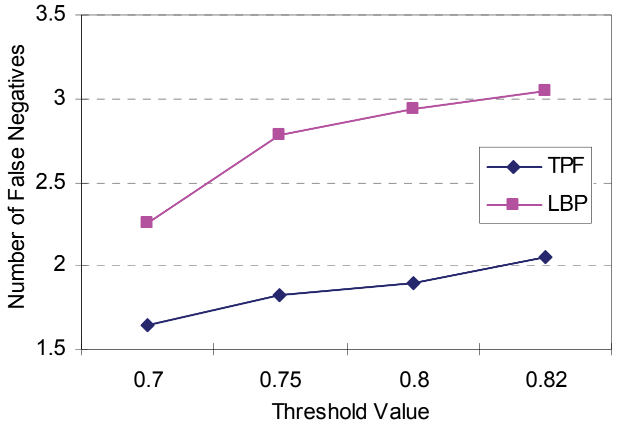


Fig. 5. The comparative performances on Video 3 from a synthesized dataset. (a) The original frame. (b) The GMM result. (c) The LBP result. (d) The TPF result. (e) The KSM-TPF result.

In this experiment, we first test the sensitivity of LBP and TPF with different uniform thresholds on Video 3. The experimental results in Fig. 6 show that both methods suffer from the threshold value changes, but TPF performs consistently much better than LBP, particularly in terms of false negatives. We then test the sensitivity of KSM-TPF with different η. The performances illustrated in Fig. 7 demonstrate that the proposed KSM-TPF approach performs relatively stable when η is varied from 0.4

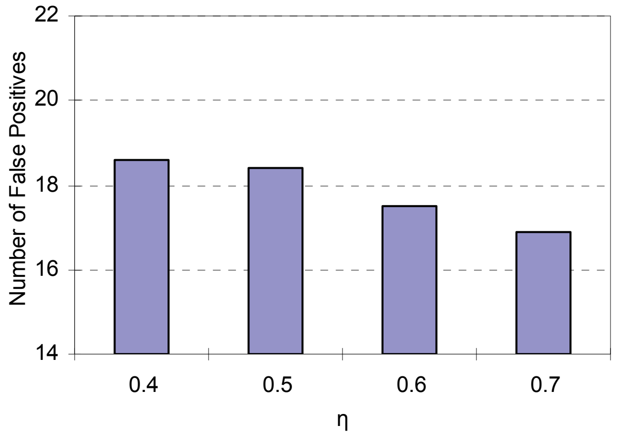
to 0.7. This can be explained by the fact that the standard deviation is much smaller than the mean value, and thus η does not significantly affect the final performance. Figs. 5(b)-(e) display some visual segmentation results of all the compared approaches, i.e. KSM-TPF, TPF, LBP and GMM. Tables 1 and 2 list their corresponding numerical results on Videos 3 and 4, respectively. The performance of the proposed KSM-TPF approach is clearly the best in terms of both false positives and false negatives, as well as the sum of two. The sum of false positives and false negatives is calculated by adding all the classification errors in the input frames compared with the ground truth frames. These experimental results show that the TPF based methods are better than LBP and GMM, indicating that TPF is effective in extracting the discriminate features from the input video sequences. Moreover, the proposed KSM-TPF approach achieves the best performance in terms of all three numerical evaluation factors (i.e. false positives, false negatives, and the sum). This proves that the adaptive threshold of the KSM-TPF approach can increase the performance of TPF to some extent. It should be noted that, for the proposed approach, most of the false positives occur on the contour areas of the moving objects (see Fig. 5), which is because the TPF features are extracted from the neighborhood of pixels.



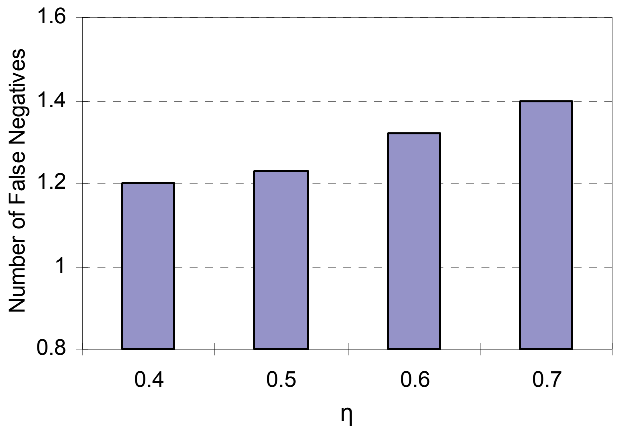


(a) (b)

Fig. 6. The comparative performances of TPF and LBP with different threshold values on Video 3. (a) The performances of false positives. (b) The performances of false negatives.



(a)



(b)

Fig. 7. The performances of KSM-TPF with different η on Video 3. (a) The performances of false positives. (b) The performances of false negatives.

Table 1. The comparative results of false positives and false negatives ($\times 10^5$) on Video 3

	False Positive	False Negative	Sum
KSM-TPF	18.4	1.2	19.6
TPF	19.3	1.64	20.94
LBP	21.76	2.26	24.02
GMM	66.84	2.06	68.91

Table 2. The comparative results of false positives and false negatives ($\times 10^5$) on Video 4

	False Positive	False Negative	Sum
KSM-TPF	26.4	1.42	27.82
TPF	29.7	1.95	31.65
LBP	40.6	2.98	43.58
GMM	68.80	3.01	71.81

In Tables 1 and 2, the threshold value ($\kappa \geq Threshold$) for both TPF and LBP is 0.7. For KSM-TPF, the parameter $\eta = 0.5$. For the integral histogram and the mean gray-level image, the size of the local subregions is 10×10 and 8×8, respectively. The number of histogram bins for LBP is equally quantized into 32. Therefore, the feature lengths for the LBP and TPF integral histograms are the same.

4.2. Experiment 2

The second experiment is conducted on the Wallflower video dataset [17]. Five frames from the sequence are manually labeled to generate the ground truth data. The numerical comparison results among KSM-TPF, TPF, LBP and GMM on the video sequence that has a dynamic background with heavily swaying trees (the first and second rows of Fig. 9) are shown in Fig. 8, in terms of false positives, false negatives and the sum. Some visual comparison results are also displayed in Fig. 9. The proposed KSM-TPF approach still achieves the best performance in terms of the sum of false positives and false negatives.

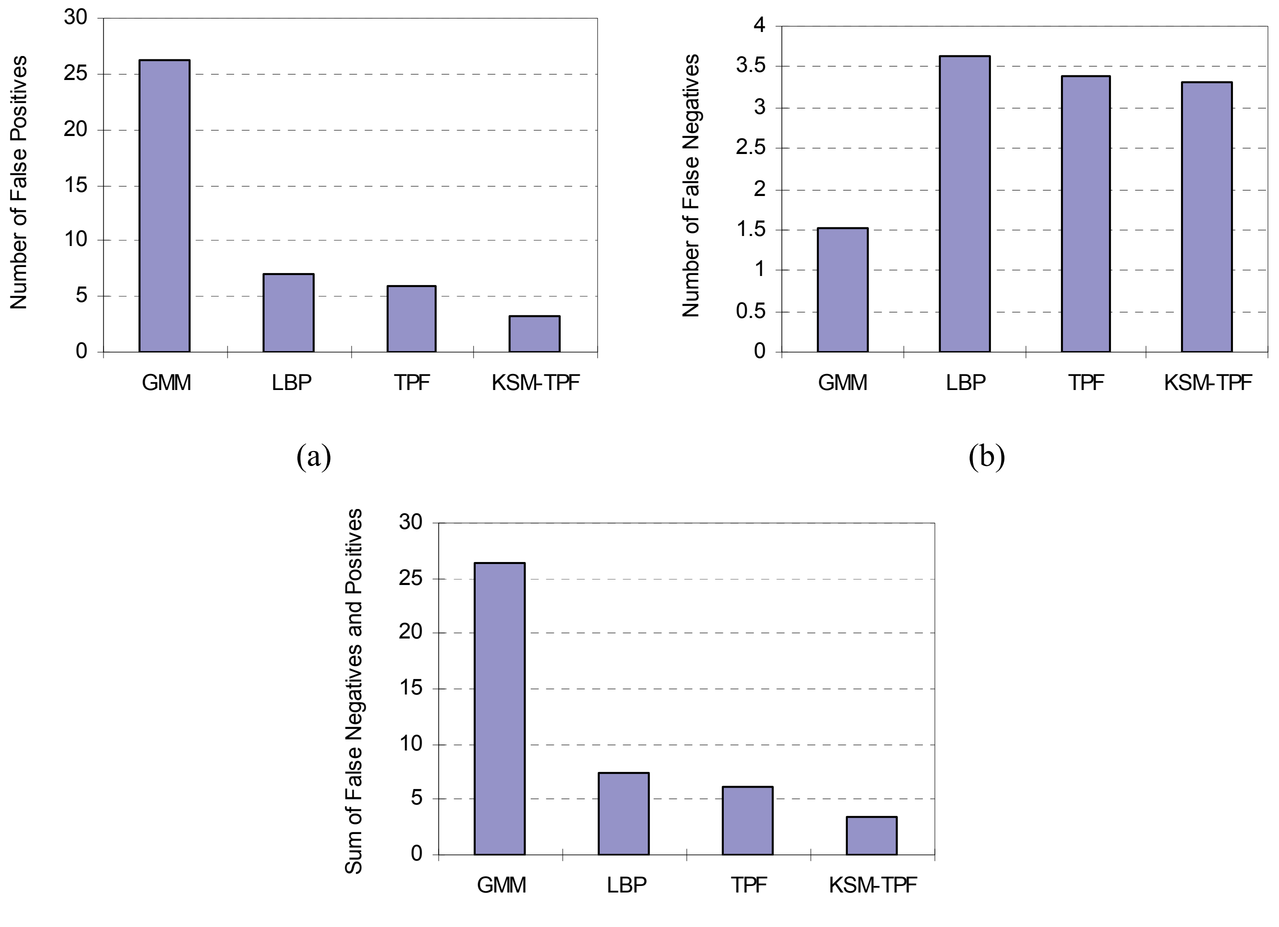


Fig. 8. The comparative performances on the video from the Wallflower video dataset. (a) The performances of false positives ($\times 10^5$). (b) The performances of false negatives ($\times 10^4$). (c) The performances of the sum ($\times 10^5$).

From the first two rows of Fig. 9, we can see that the proposed approach robustly handles the heavily swaying trees and correctly detects the walking person, because it combines both textural and motional information of background variations. Although we use the same set of parameters in this experiment as in the previous subsection, we would like to stress that it is straightforward to fine tune the parameters through experiments for a better performance.

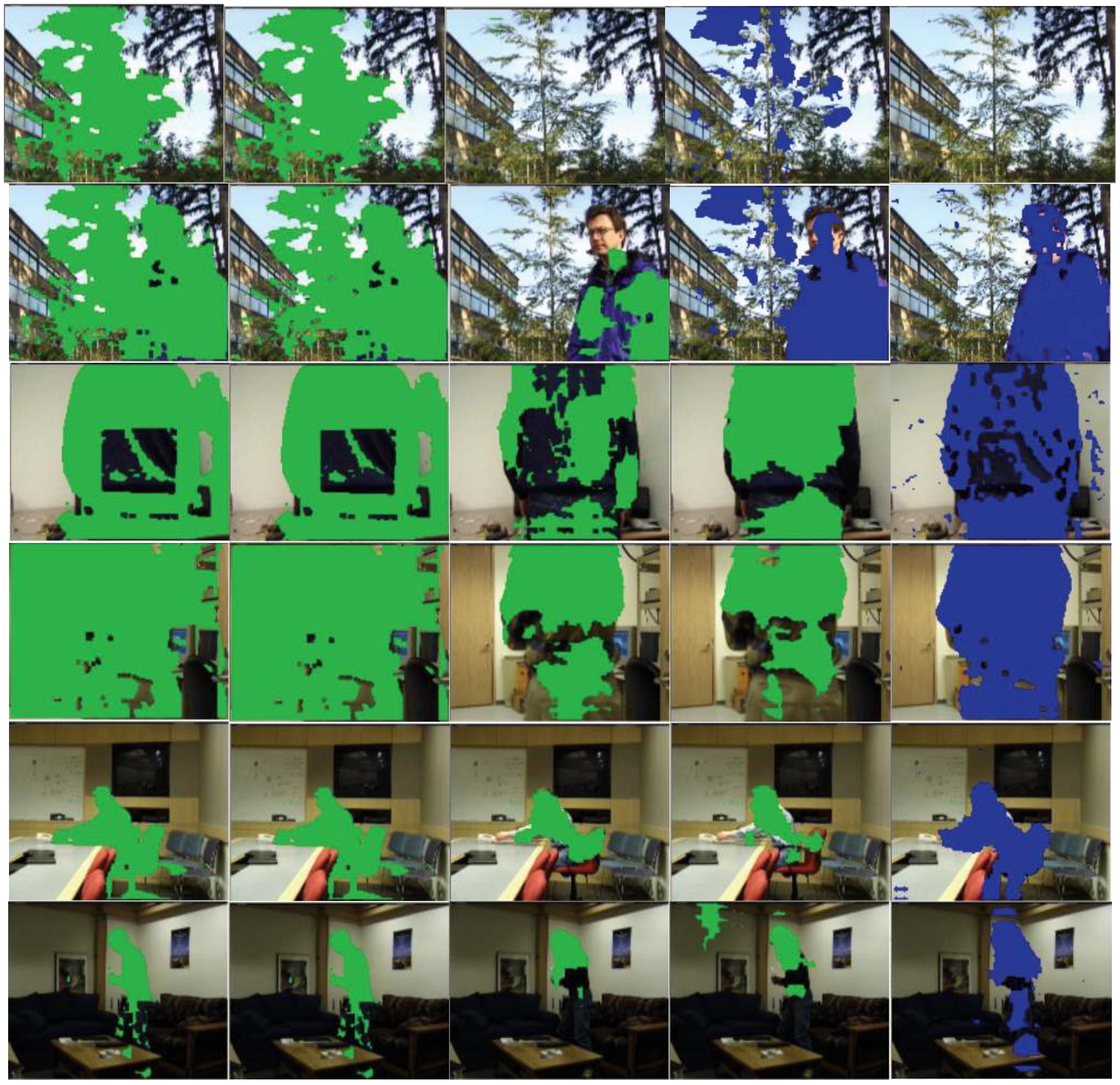
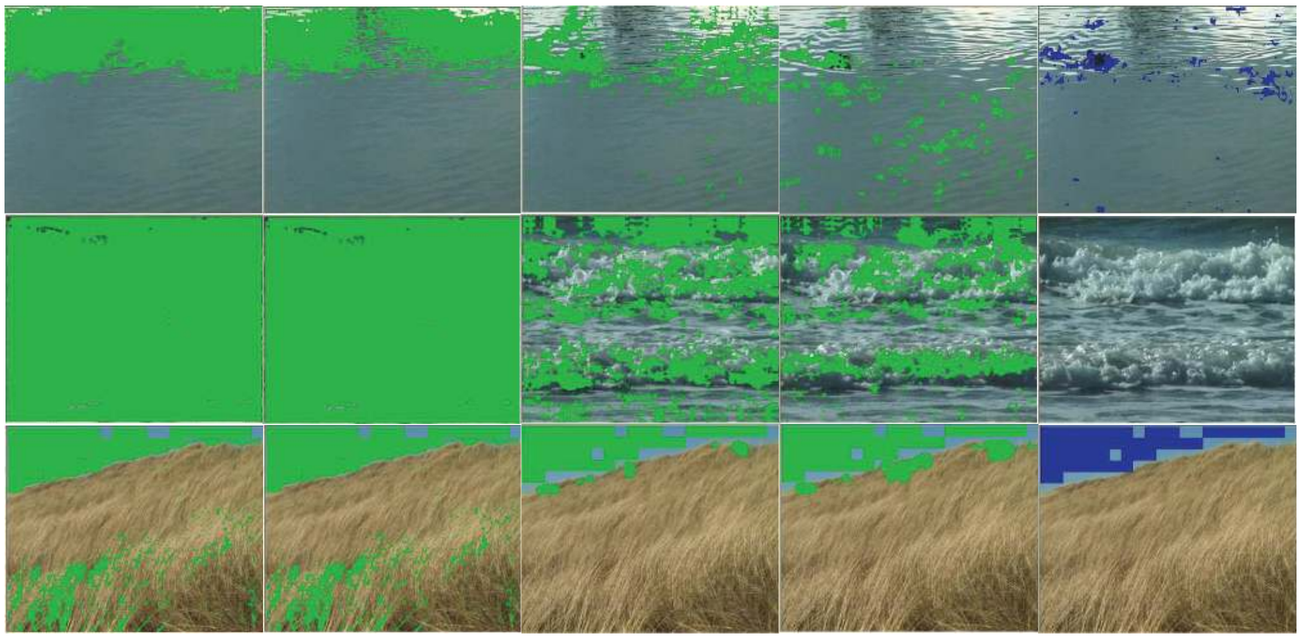

(a) (b) (c) (d) (e)

Fig. 9. The comparative performances on the Wallflower video dataset. (a) The GMM results. (b) The CMU results. (c) The LBP results. (d) The TPF results. (e) The KSM-TPF results.

4.3. Experiment 3

The third experiment is conducted on the DynTex video dataset [10], which includes ten videos for public downloading. The videos consist of ten different dynamic textures, such as swaying trees and moving escalators. Dynamic textures refer to spatially repetitive, time-varying visual patterns with a certain temporal stationarity. In dynamic texture, the notion of self-similarity central to conventional image textures is extended to the spatio-temporal domain.

A complete set of experimental results on all the ten videos are displayed in Fig. 10. Note that only one of the ten videos (the first row in Fig. 10) contains foreground objects, i.e., a duck moving into the scene. For the other nine videos, all the video frames are considered as dynamic backgrounds. The proposed KSM-TPF approach clearly outperforms the other four benchmark methods. For the video with vehicles running on the highway (the ninth row in Fig. 10), the traffic flow is considered by the proposed method as part of the dynamic background because the traffic flow consistently appears in the whole video clip from the beginning to the end and thus there is no clear highway road frames at the beginning of the video for our system training that can model an empty highway as static background.



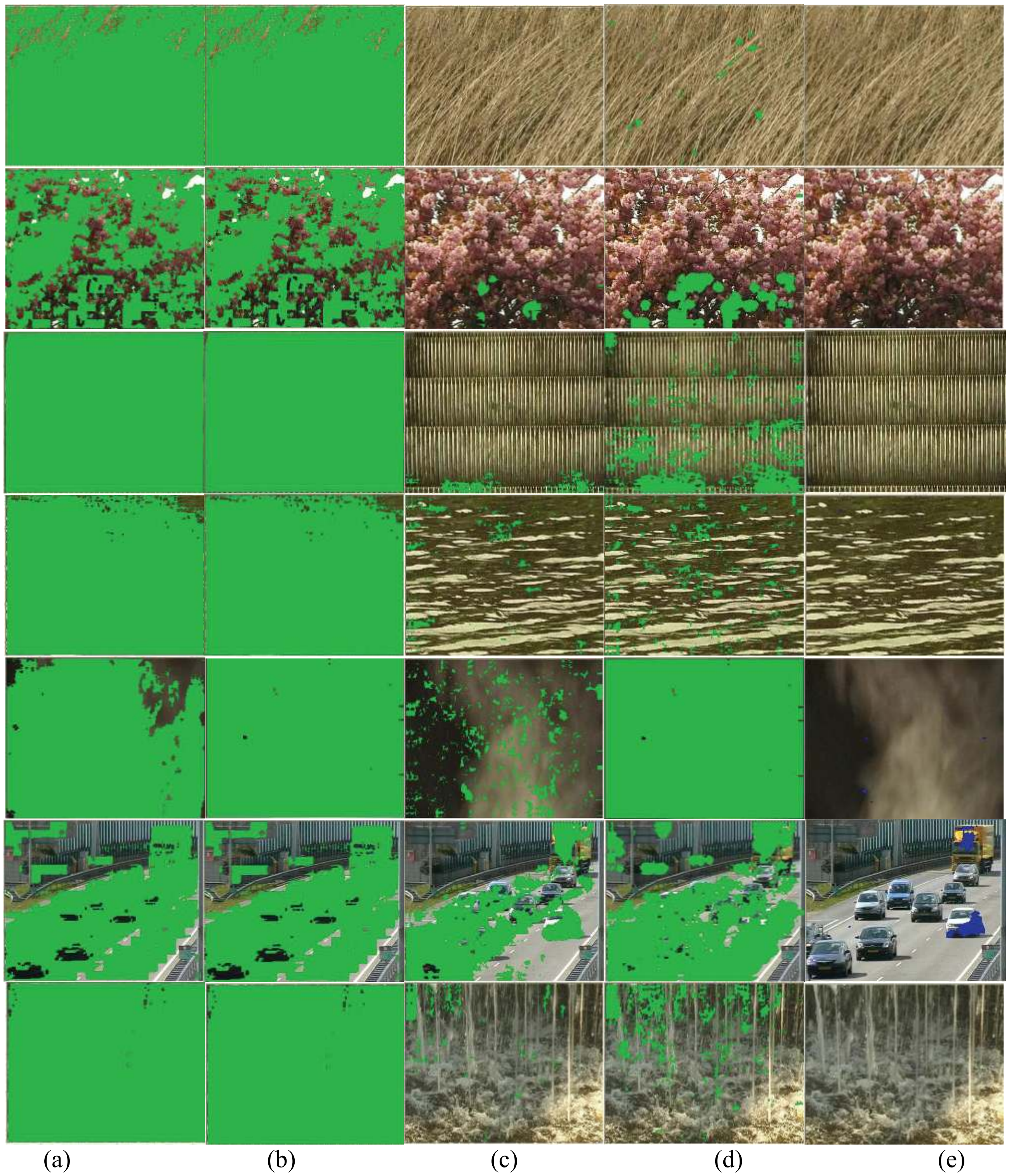

(a) (b) (c) (d) (e)

Fig. 10. The comparative performances on the DynTex video dataset. (a) The GMM results. (b) The CMU results. (c) The LBP results. (d) The TPF results. (e) The KSM-TPF results.

V. Conclusion and Future Work

We presented a new approach to background subtraction through Kernel Similarity Modeling of

Texture Pattern Flow (KSM-TPF). Our approach provides advantages over other methods in that it identifies more discriminative information in the spatio-temporal domain by using an adaptive threshold. The proposed KSM-TPF approach was tested on three video sequences from two publicly available databases and achieved a better performance than LBP and GMM. The experimental results showed that KSM-TPF is much more robust to significant background variations. The processing speed of the proposed approach is 5 frames per second on a Windows-platform PC with a 2.8GHz Intel Core Duo CPU and 2GB RAM, which reaches quasi real-time performance. The program is implemented using C/C++. This shows that the proposed method is less computationally efficient than the GMM method [14] or the LBP method [6]. In the future, we will focus on improving the computational efficiency of the proposed approach.

REFERENCES

[1] A. Barla, F. Odone, and A. Verri, "Histogram Intersection Kernel for Image Classification," *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, pp. 513-516, 2003.

[2] R.T. Collins, A.J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt, and L. Wixson, "A System for Video Surveillance and Monitoring," *Technical Report CMU-RI-TR-00-12, Carnegie Mellon University*, 2000.

[3] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting Moving Objects, Ghosts, and Shadows in Video Streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337-1342, 2003.

[4] A. Elgammal, R. Duraiswami, D. Harwood, and L.S. Davis, "Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1151-1163, 2002.

[5] B. Han, D. Comaniciu, Y. Zhu, and L. Davis, "Incremental Density Approximation and Kernel-

Based Bayesian Filtering for Object Tracking," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 638-644, 2004.

[6] M. Heikkilä and M. Pietikäinen, "A Texture-Based Method for Modeling the Background and Detecting Moving Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657-662, 2006.

[7] A. Mittal and N. Paragios, "Motion-Based Background Subtraction Using Adaptive Kernel Density Estimation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 302-309, 2004.

[8] T. Ojala, M. Pietikäinen, and D. Harwood, "A Comparative Study of Texture Measures with Classification Based on Feature Distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51-59, 1996.

[9] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971-987, 2002.

[10] R. Péteri, S. Fazekas, and M.J. Huiskes, "Dyntex: A Comprehensive Database of Dynamic Textures," *to appear in Pattern Recognition Letters*, 2010.

[11] M. Pietikäinen, T. Ojala, and Z. Xu, "Rotation-Invariant Texture Classification Using Feature Distributions," *Pattern Recognition*, vol. 33, no. 1, pp. 43-52, 2000.

[12] F. Porikli, "Integral Histogram: A Fast Way to Extract Histograms in Cartesian Spaces," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 829-836, 2005.

[13] Y. Sheikh and M. Shah, "Bayesian Modeling of Dynamic Scenes for Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1778-1792, 2005.

[14] C. Stauffer and W.E.L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 246-252, 1999.

[15] Y.-L. Tian and A. Hampapur, "Robust Salient Motion Detection with Complex Background for Real-Time Video Surveillance," *Proceedings of the IEEE Computer Society Workshop on Motion and Video Computing*, vol. 2, pp. 30-35, 2005.

[16] B.U. Toreyin, A.E. Cetin, A. Aksay, and M.B. Akhan, "Moving Object Detection in Wavelet Compressed Video," *Signal Processing: Image Communication*, vol. 20, no. 3, pp. 255-264, 2005.

[17] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and Practice of Background Maintenance," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 1, pp. 255-261, 1999.

[18] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland, "Pfinder: Real-Time Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780-785, 1997.

[19] G. Zhao and M. Pietikäinen, "Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915-928, 2007.

[20] Z. Zivkovic and F. van der Heijden, "Efficient Adaptive Density Estimation Per Image Pixel for the Task of Background Subtraction," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 773-780, 2006.