Title	An efficient pattern-less background modeling based on scale invariant local states
Author(s)	Yuk, JSC; Wong, KYK
Citation	The 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS 2011), Klagenfurt, Austria, 30 Auguist-2 September 2011. In Proceedings of 8th AVSS, 2011, p. 285-290
Issued Date	2011
URL	http://hdl.handle.net/10722/137653
Rights	Creative Commons: Attribution 3.0 Hong Kong License

An Efficient Pattern-Less Background Modeling based on Scale Invariant Local States

Jacky S-C. Yuk
scyuk@cs.hku.hk

Kwan-Yee K. Wong
kykwong@cs.hku.hk

Computer Vision Group,
The University of Hong Kong
http://www.cs.hku.hk/visiongroup

Abstract

A robust and efficient background modeling algorithm is crucial to the success of most of the intelligent video surveillance systems. Compared with intensity-based approaches, texture-based background modeling approaches have shown to be more robust against dynamic backgrounds and illumination changes, which are common in real life videos. However, many of the existing texture-based methods are too computationally expensive, which renders them useless in real-time applications. In this paper, a novel efficient texture-based background modeling algorithm is presented. Scale invariant local states (SILS) are introduced as pixel features for modeling a background pixel, and a pattern-less probabilistic measurement (PLPM) is derived to estimate the probability of a pixel being background from its SILS. An adaptive background modeling framework is also introduced for learning and representing a multi-modal background model. Experimental results show that the proposed method can run nearly 3 times faster than existing state-of-the-art texture-based method, without sacrificing the output quality. This allows more time for a real-time surveillance system to carry out other computationally intensive analysis on the detected foreground objects.

1. Introduction

Background modeling has always been one of the key research topics in computer vision over the past few decades [1, 2, 6, 7, 9]. It is primarily used for foreground object extraction in video analysis [4, 8, 10].Since the output of background modeling is often subjected to analysis by other higher level processes (such as object tracking, behavior analysis, video profiling, face detection, etc.), a good background modeling algorithm should be as efficient as possible so as to reserve more computational time and power for the later possesses. A slow background modeling method is simply not useful, especially in a real-time system. However, background modeling is never an easy task in complicated real life scenarios. For instances, dynamic backgrounds (such as waving trees and rippling water surfaces) and illumination changes are two major challenges in background modeling. A common approach for tackling these problems is by adopting a mixture of distribution models to adaptively learn the underlying multi-modal background model.

Stauffer and Grimson [9] proposed using a mixture of Gaussians (MoG) in the color space to model a background pixel. Their method has been widely used for extraction of moving foreground regions in video surveillance. However, such a color-based method is extremely sensitive to illumination changes. Their method cannot handle highly dynamic backgrounds well with a limited number of Gaussian distributions, and performs even worse in handling soft-shadows. Texture-based methods [1, 3, 5], on the other hand, are more tolerable to dynamic backgrounds and illumination changes. In spite of that, texture-based methods often require more computational power as well as memory space than color-based methods. Heikkilä and Pietikäinen [2] modeled a background pixel using a histogram of local binary pattern (LBP) computed over a window centered at the pixel. Although LBP is a simple and effective feature for texture analysis, it is not stable in homogeneous regions where neighboring pixels have very similar intensities. Tan and Triggs [10] proposed to include an additional buffering state to solve the instability problem of LBP, and introduced the local ternary pattern (LTP) as a robust extension of LBP. Liao *et al*. [6] proposed the scale invariant local ternary pattern (SILTP) to make LTP more robust under different lighting conditions. They also introduced the pattern kernel density estimation (PKDE) technique to model the probability density function (pdf) of a background pixel in the discrete pattern space.

In this work, a texture-based pattern-less probabilistic background modeling framework is proposed. Compared with other texture-based methods, the proposed framework is more efficient in terms of memory space and computa-

978-1-4577-0845-9/11/$26.00

tional power required. Experiments show that the proposed framework not only produces results comparable to state-of-the-art PKDE [6], but also runs nearly 3 times faster than PKDE. In summary, this paper first introduces scale invariant local states (SILS) as a pixel feature. Based on this SILS feature, a pattern-less background model is developed which models a background pixel by the discrete probability density functions of the individual components of the SILS feature. A probabilistic measurement is then derived to estimate the probability of a pixel being a background pixel based on its SILS. An adaptive background modeling framework for a multi-modal background is also proposed to adaptively update the pattern-less background model. Extensive experiments have been carried out and the results show that the proposed method is very efficient and robust under extremely challenging video scenarios such as sudden changes of illumination and rippling water surfaces.

The rest of the paper is organized as follows. Section 2 introduces the scale invariant local states (SILS) pixel feature. Section 3 presents the pattern-less background model and introduces the probabilistic measurement for estimating the probability of a pixel being a background pixel. Section 4 gives the details of the adaptive background modeling framework for a multi-modal background. Experimental results are presented in Section 5, followed by the conclusions in Section 6.

2. Scale Invariant Local States

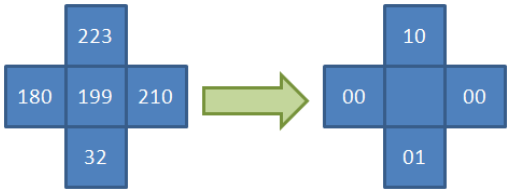


Figure 1. Examples of SILTP codes and the corresponding SILS with 4-connected neighbors and $\tau = 0.1$: SILTP code = 10000100 and SILS = $\{10, 00, 01, 00\}$.

In [6], Liao *et al*. proposed the robust SILTP feature for modeling a pixel. As shown in figure 1, this feature is constructed by comparing the intensity of a pixel with that of its neighbors and packing the 2-bit patterns representing the comparison results into one single long bit pattern. Given a pixel located at (x, y) with intensity I, its SILTP is defined as

$$SILTP_\tau(x, y) = \bigoplus_{n \in N_R} s_\tau(I, I_n) \qquad (1)$$

where N_R is the set of neighbors within a region of radius R centered at the pixel, I_n is the intensity of its n-th neighbor, and $\bigoplus$ is the bit-packing operations. $s_\tau(I, I_n)$ is defined as

$$s_\tau(I, I_n) = \begin{cases} 10 & if\ I_n > (1+\tau)I \\ 01 & if\ I_n < (1-\tau)I \\ 00 & otherwise \end{cases} \qquad (2)$$

where τ is a scale parameter to make the SILTP more robust under different lighting conditions.

Although SILTP feature is simple and has shown to be tolerable to illumination changes and dynamic backgrounds, its discrete nature makes the representation as well as updating of a background model much more complicated. In [6], Liao *et al*. proposed the pattern kernel density estimation (PKDE) to model and update the background using pdf in the pattern space. Although good results have been reported, their model has a huge demand on memory space and computational power. For instance, suppose N neighbors and K background models are used in modeling a background pixel. Their PKDE background modeling framework would require $O(3^N K)$ memory space to store and $O(3^N K)$ operations to update the background model for one single pixel. Such an exponential order in complexity prohibits their framework from using more neighbors in encoding a pixel using SILTP. The high memory and operations requirements also make their model not feasible in real-time embedded systems which have limited memory and computational power.

In order to overcome the weakness of using discrete patterns, this paper proposes to keep track of the discrete likelihood of the individual components of the scale invariant local states (SILS) instead of the code pattern itself. Formally, SILS is defined as

$$SILS_\tau(x, y) = \{s_\tau(I, I_n) : n \in N_R\}. \qquad (3)$$

From (3), it can be seen that SILS is in fact defined as a set of local states instead of a single bit pattern. The n-th state corresponds to the local state of the n-th neighbor (see figure 1).

3. Pattern-Less Background Modeling

In this section, a pattern-less background model is proposed based on the SILS feature introduced in the previous section. A probabilistic measurement is then developed to estimate the probability of a pixel being a background pixel based on its SILS and the current background model.

3.1. Pattern-Less Background Model

Traditionally, a background model of a pixel is built from the history of its pixel features $\{\mathbf{p}^0, \ldots, \mathbf{p}^{t-1}\}$. To determine whether a pixel observed at time t belongs to background or foreground, its feature $\mathbf{p}^t$ at time t will be compared against the current background $\mathbf{q}^t$. Besides, $\mathbf{p}^t$ will

also be used to update the background model from $\mathbf{q}^t$ to $\mathbf{q}^{t+1}$ for determining the background/foreground state of the pixel at time $t+1$. In [6], SILTP is employed as the pixel feature $\mathbf{p}^t$ and the background model $\mathbf{q}^t$ is represented as a discrete pdf of SILTP. The probability of a pixel being background can be directly obtained from the pdf and decision on whether the pixel is a background or foreground pixel can be made accordingly.

In this paper, the SILS introduced in (3) is employed as the pixel feature $\mathbf{p}^t$, and the background model $\mathbf{q}^t$ is represented by the discrete likelihood of the individual components of the SILS as

$$\mathbf{q}^t = \{\mathbf{S}_n^t : n \in N_R\} \tag{4}$$

where

$$\mathbf{S}_n^t = \{\rho_{n,c}^t : c \in \{00, 01, 10\}\}. \tag{5}$$

Here, $\rho_{n,c}^t$ is the likelihood of the n-th neighbor having a local state c given the pixel is a background pixel. This $\rho_{n,c}^t$ will be adaptively updated according to the previous pixel features $\{\mathbf{p}^0, \ldots, \mathbf{p}^{t-1}\}$. The details of the updating process will be described in the following subsection.

3.2. Probabilistic Measurement

In this subsection, a probabilistic measurement is introduced to determine the probability of a pixel observed at time t with feature $\mathbf{p}^t$ being a background pixel. Given the current background state $\mathbf{q}^t$ as defined in (4), the measurement $\phi(\mathbf{p}^t, \mathbf{q}^t)$ is defined as

$$\phi(\mathbf{p}^t, \mathbf{q}^t) = \Big(\prod_{n \in N_R} \sum_{c \in \{00,01,10\}} \big(\delta(s_n^t, c)\rho_{n,c}^t\big) \Big)^{\frac{1}{N}} \tag{6}$$

where $s_n^t \in \mathbf{p}^t$ is the state of the n-th neighbor as defined in (3), δ is a delta function such as $\delta(a,b) = 1$ when $a = b$ or 0 otherwise, $\rho_{n,c}^t$ is as defined in (5), and the power term $(.)^{\frac{1}{N}}$ is used for normalization. This measurement measures the likelihood of a pixel having a pixel feature $\mathbf{p}^t$ given it is a background pixel. Assuming the prior probabilities of a pixel being foreground and background are equal, then (6) gives a measure of the probability of the pixel being a background pixel given its pixel feature $\mathbf{p}^t$.

An adaptive update mechanism is applied to update the background model $\mathbf{q}^t$ such that it is able to tolerate changes in the background from time to time. This can be done by simply updating state likelihoods $\rho_{n,c}^t$ of $\mathbf{q}^t$ according to the following equation:

$$\rho_{n,c}^{t+1} = (1-\alpha)\rho_{n,c}^t + \alpha\delta(s_n^t, c) \tag{7}$$

where α is the learning rate ranging from 0 to 1.

4. Multi-Modal Background Modeling

In this section, an adaptive background modeling framework for a multi-modal background will be described. Details about how to classify a pixel as background or foreground, as well as how to update the multi-modal background model will be given.

4.1. Background Determination

Similar to other state-of-the-art background modeling approaches [6, 9], this paper also suggests to use a mixture of distribution models to model a multi-modal background. By keeping multiple hypotheses for a background pixel, the background model can better represent a multi-modal background under challenging scenarios such as dynamic backgrounds and illumination changes. In this paper, K models are used for background modeling, and a weight ω_k^t is associated with the k-th model at time t. Furthermore, at each time instance t, the K models are sorted by their associated ω_k^t in descending order.

Given a pixel with a pixel feature $\mathbf{p}^t$ and the current background state $\mathbf{Q^t} = \{\mathbf{q}_k^t : k \in \{1, 2, \ldots, K\}\}$, the probability of the pixel being a background pixel is given by

$$P(\mathbf{p}^t|\mathbf{Q}^t) = \frac{1}{\lambda} \sum_{k<M^t} \omega_k^t \phi(\mathbf{p}^t, \mathbf{q}_k^t) \tag{8}$$

where λ is a normalization factor, and M^t is the number of models that are currently considered as background. M^t is the maximum number less than K such that

$$\sum_{k<M} \omega_k^t < T_b \tag{9}$$

with T_b being a user-defined threshold. A pixel can then be considered as a background pixel if $P(\mathbf{p}^t|\mathbf{Q}^t)$ is larger than a user-defined threshold T_s.

4.2. Adaptive Background Update

After the classification of a pixel as either a foreground or a background pixel, its pixel feature $\mathbf{p}^t$ is used to update the current background model $\mathbf{Q}^t$. The model $\mathbf{q}_h^t$ with the highest probability $\phi(\mathbf{p}^t, \mathbf{q}_h^t) > T_h$ among all K models is selected for the update, where T_h is a user-defined threshold. The state likelihoods of $\mathbf{q}_h^t$ are updated according to (7). The weights of all the K models are also adaptively updated according to the following equation:

$$\omega_k^{t+1} = (1-\alpha)\omega_k^t + \alpha\delta(k, h) \tag{10}$$

On the other hand, if there does not exist a model $\mathbf{q}_h^t$ with $\phi(\mathbf{p}^t, \mathbf{q}_h^t) > T_h$, the model $\mathbf{q}_l^t$ with lowest weighting ω_l^t is then selected and replaced by a new model with state

likelihoods $\rho_{n,c}$ initialize according to the following equation:

$$\rho_{n,c}^{t+1} = \begin{cases} T_i & if\ s_n^t = c \\ 0 & otherwise \end{cases} \quad (11)$$

where T_i is a user defined value.

4.3. Memory Space and Operations Requirement

For each modeling pixel, there are 3 possible states for each of the N neighbors, and therefore, the proposed pattern-less probabilistic measurement (PLPM) requires totally $O(3NK)$ memory space for storing and $O(3NK)$ operations for updating the likelihoods of K models. Comparing to PKDE [6], which requires $O(3^N K)$ memory space for storing and $O(3^N K)$ operations for updating 3^N possible codes in each model, the proposed PLPM requires much less memory space and performs much faster.

5. Experimental Results

Similar to [6], the same block-based fusion technique is also applied to the proposed method for comparisons. The fusion is performed by downsampling the frame by scale r with each pixel value being the mean of the corresponding block with size r x r. Local states are then generated using (3), and the same background modeling is performed in the downsampled frame. The resulting probabilities are then upsampled bilinearly to the original space, and then compared with the threshold T_s for the final background decisions. Furthermore, [6] also suggested to combine the multi-scale fused background probabilities by $Final\_P = (\prod_i^n \hat{P}_{r_i})^{1/n}$ where $\hat{P}_r$ is the background probabilities defined in (8) with fusion scale r.

In the experiments, the proposed PLPM method together with state-of-the-art MoG [9] and PKDE [6] approaches were evaluated using nine challenging video sequences which are made publicly available by [5]. The proposed PLPM method was evaluated with fusion technique using scales 1, 2, and 3 respectively (labeled as PLPM$_{r=1}$, PLPM$_{r=2}$, and PLPM$_{r=3}$ respectively). Multi-scale fused PLPM with three scales 1, 2, and 3 (denoted as PLPM$_{r=1+2+3}$) was also evaluated. Similarly, the same fusion technique was also applied when evaluating PKDE denoted as (PKDE$_{r=1}$, PKDE$_{r=2}$, PKDE$_{r=3}$, and PKDE$_{r=1+2+3}$ respectively). The nine testing videos include six indoor scenes (AirportHall, Bootstrap, Curtain, Escalator, Lobby, and ShoppingMall), and 3 outdoor scenes (Fountain, Trees, and WaterSurface). AirportHall, Bootstrap, and ShoppingMall contain crowds with moving soft-shadow scenarios. Lobby is for testing the background models under sudden lighting changes. Curtain, Escalator, Fountain, Trees, and WaterSurface are the scenarios with highly dynamic backgrounds. Snapshots of the videos are shown in figure 2 (a). The corresponding ground truths and the background/foreground segmentation results are shown in figure 2(b)-(e). Similar to [6], no morphological operations were applied to the resulting foregrounds. Instead, a commonly used OpenCV post-processing technique was applied in order to eliminate small foreground regions and fill holes inside the regions. For both PKDE and PLPM, 4-connected neighbors were used for generating SILTP and SILS respectively. Since the number of neighbors N was fixed to 4 during the experiments, the normalization power term $(.)^{\frac{1}{N}}$ in (6) for PLPM was not necessary, so that the speed of the proposed method could be even faster. Furthermore, K = 5 models were used for all MoG, PKDE, and proposed PLPM. The parameters used for MoG and PKDE were set as suggested in [9] and [6] respectively. For the proposed PLPM, we set $T_b = 0.7$, $T_s = T_h = 0.015$, $T_i = 0.5$, and the learning rate $\alpha = 0.0005$ for both (7) and (10).

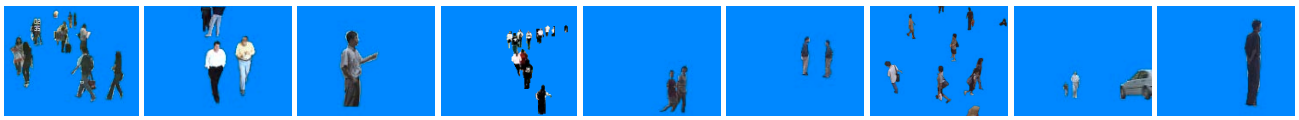
As shown in figure 2(c), the color-based MoG always suffered from errors caused by soft-shadows, and the method was sensitive to lighting changes, expecially in video sequences: AirportHall, Bootstrap, Escalator, and Trees. Due to the limited paper space, only the results of PKDE$_{r=1+2+3}$ and PLPM$_{r=1+2+3}$ are shown in figure 2(d) and (e) respectively. As expected, both texture-based PKDE and PLPM were insensitive to lighting changes, and effectively handled soft-shadows. They produced similar background/foreground segmentation results. However, these two methods sometimes generated holes in homogeneous foreground objects, especially for large objects. Fortunately, this "hole" problem can be relieved by further hole-filling mechanisms.

F-score quantitative evaluation method [6] is also used to evaluate the modeling techniques, and it is defined as

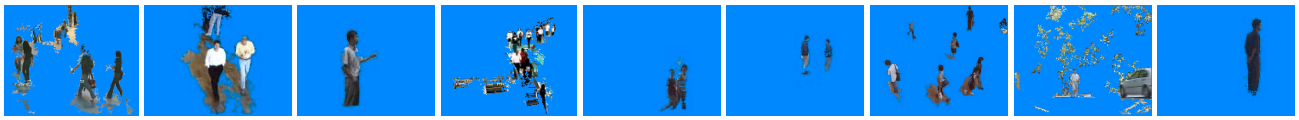
$$FScore = \frac{2TP}{2TP + FN + FP} \quad (12)$$

where TP, FN, and FP are true positive, false negative, and false positive respectively. 20 labeled frames from each testing video sequence with manually marked ground truths were selected for calculating the corresponding F-score as shown in table 1. The results show that the proposed PLPM generated comparable F-scores as PKDE, and outperformed MoG in most of the testing videos except the WaterSurface. The final average F-score of the proposed PLPM was even slightly better than PKDE. For the WaterSurface sequence, all three methods produced similar F-scores. The MoG had the highest F-score since the colors of the foregrounds were sharply differentiable from the background, and the texture-based PKDE and proposed PLPM had the "hole" problems for large homogeneous foreground objects. However, MoG was not able to handle soft-shadows and was sensitive to lighting changes, and therefore, performed the worst on average.

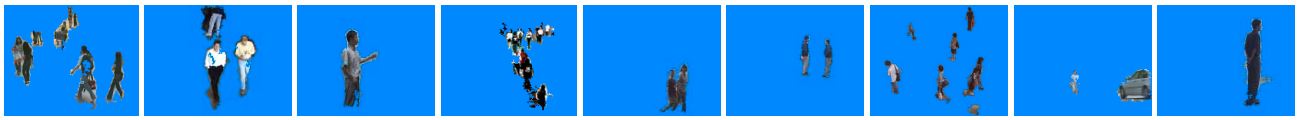
(a) Snapshots of testing videos: AirportHall, Bootstrap, Curtain, Escalator, Fountain, Lobby, ShoppingMall, Trees, and WaterSurface, respectively.



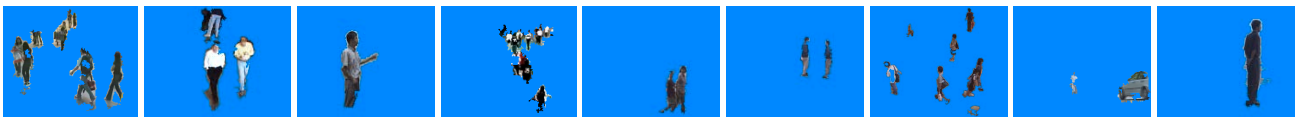
(b) Ground truths of the testing videos with the masked background regions.



(c) MoG [9] background/foreground segmentation results.



(d) PKDE [6] background/foreground segmentation results with fusion: PKDE$_{r=1+2+3}$.



(e) The proposed PLPM background/foreground segmentation results with fusions: PLPM$_{r=1+2+3}$.

Figure 2. (a) Testing video snapshots, (b) ground truths, and the background/foreground segmentation results of (c) MoG, (d) PKDE and (e) the proposed PLPM.

	MoG[9]	PKDE[6] r=1	PKDE[6] r=2	PKDE[6] r=3	PKDE[6] r=1+2+3	PLPM r=1	PLPM r=2	PLPM r=3	PLPM r=1+2+3
AirportHall	63.31	70.48	65.67	64.23	71.28	**74 12**	69.27	66.19	72.10
Bootstrap	59.32	**81 99**	78.03	69.95	80.92	81.39	77.55	69.16	80.22
Curtain	89.97	90.68	87.08	87.62	90.92	**91 46**	86.06	87.97	90.56
Escalator	31.83	64.46	70.83	66.84	**75 38**	67.81	69.62	63.81	73.67
Fountain	77.78	71.23	79.51	71.33	**85 38**	72.13	80.64	73.85	85.28
Lobby	76.47	**85 57**	71.83	69.46	84.47	84.91	76.60	71.31	85.05
ShoppingMall	71.38	80.28	77.50	76.39	**81 32**	80.51	78.90	76.64	80.29
Trees	23.39	37.74	68.97	67.75	70.83	47.75	63.05	66.96	**74 61**
WaterSurface	**87 93**	73.23	72.16	83.96	86.33	78.45	82.61	83.04	87.38
Average	64.60	72.85	74.62	73.06	80.76	75.39	76.03	73.21	**81 02**

Table 1. F-Scores of the background modeling (row 2-10). The last row shows the average F-scores of the testing videos.

The machine used for the experiments was a Dell notebook PC with Windows 7 OS, Intel Core 2 Duo CPU (2.40 Hz x 2) and 2 Gb memory. All the methods were implemented using C++ in VS .Net 2008 with no particular optimizations. The processing framerates of the testing videos are listed in table 2. Note that the processing overheads for generating the SILTP and SILS for PKDE and proposed PLPM, respectively, are also counted. Results show that the proposed PLPM was nearly 3 times faster than the PKDE approach when 4-connected neighbors (N=4) and 5 models (K=5) were used for the background modeling. In particular, PKDE$_{r=1}$ was only running at 8 fps for processing the ShoppingMall sequence (320 x 256 resolution). This is far from the real-time applcation requirement (25 fps). The

	Resolutions	MoG[9]	PKDE[6] r=1	PKDE[6] r=2	PKDE[6] r=3	PKDE[6] r=1+2+3	PLPM r=1	PLPM r=2	PLPM r=3	PLPM r=1+2+3
AirportHall	176x144	156.19	25.78	100.84	222.27	17.31	73.50	272.94	567.85	42.74
Bootstrap	160x120	200.25	34.48	135.84	301.07	23.01	95.28	363.19	752.00	55.46
Curtain	160x128	212.22	31.83	126.04	284.86	21.59	96.14	363.35	753.14	54.80
Escalator	160x130	187.33	32.71	126.85	281.93	21.60	99.46	398.07	810.48	54.92
Fountain	160x128	204.18	31.81	123.32	278.49	21.49	91.10	350.42	754.69	52.94
Lobby	160x128	209.05	31.76	125.47	282.68	21.48	95.83	366.22	749.39	55.11
ShoppingMall	320x256	48.20	8.21	32.12	71.70	5.44	25.02	95.79	196.18	14.23
Trees	160x128	187.61	32.49	123.97	277.53	21.35	85.12	327.90	695.51	50.47
WaterSurface	160x128	207.00	32.86	125.25	285.07	21.91	91.51	354.92	753.57	53.27

Table 2. The processing framerates of the testing videos measured in frame per second (fps).

proposed PLPM$_{r=1}$, on the other hand, fulfills the requrement. If a more accurate background modeling is needed, the proposed PLPM$_{r=1+2+3}$ (running at 14 fps) can be also used by skipping every odd frames which does not affect most of the real-time applications (e.g., tracking) too much. For the other videos with lower resolutions, the proposed PLPM at all fusions were running much faster than the real-time requirement, and thus, reserve more rooms for further complicated analysis of the foreground objects.

6. Conclusions

This paper proposed a robust an efficient and robust pattern-less probabilistic measurement (PLPM) based on the scale invariant local states (SILS) for background modeling. Unlike previous state-of-the-art texture-based background modeling, the proposed framework models the background as the probabilistic SILS instead of a single code pattern. Therefore, the background modeling framework can be constructed in a much simpler and efficient way. The memory space and operations required can also be dramatically reduced. Suppose N neighbors and K background models are used, the memory space and operations required for the proposed framework are only $O(3NK)$ comparing to Liao *et al.*'s framework [6] which requires $O(3^N K)$. This improvement allows the proposed method to reserve more rooms for further complicated analysis of the foreground objects, especially in the embedded real-time system in which the memory and the processing power are limited. Experimental results show that the proposed framework also produces comparable background results as [6] but being much faster in speed and uses less memory space.

References

[1] N. Armanfard, M. Komeili, M. Valizade, E. Kabir, and S. Jalili. A non-parametric pixel-based background modeling for dynamic scenes. In *Proc. IEEE Int. Conf. on Advances in Computational Tools for Engineering Applications (ACTEA)*, pages 369–373, Tehran, Iran, July 2009. 1

[2] M. Heikkilä and M. Pietikäinen. A texture-based method for modeling the background and detecting moving objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(4):657–662, April 2006. 1

[3] M. Heikkilä, M. Pietikäinen, and J. Heikkilä. A texture-based method for detecting moving objects. In *Proc. British Machine Vision Conference (BMVC)*, pages 187–196, London, September 2004. 1

[4] K. Kiratiratanapruk, P. Dubey, and S. Siddhichai. A gradient-based foreground detection technique for object tracking in a traffic monitoring system. In *Proc. IEEE Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, pages 377–381, September 2005. 1

[5] L. Li, W. Huang, I. Gu, and Q. Tian. Foreground object detection from videos containing complex background. In *Proc. of the Eleventh ACM International Conference on Multimedia*, pages 2–10, New York, USA, 2003. http://perception.i2r.a-star.edu.sg. 1, 4

[6] S. Liao, G. Zhao, V. Kellokumpu, M. Pietikäinen, and S. Z. Li. Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1301–1306, San Francisco, CA, June 2010. 1, 2, 3, 4, 5, 6

[7] T. Ojala, M. Pietikäinen, and M. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):971–987, July 2002. 1

[8] X. Song, J. Cui, H. Zha, and H. Zhao. Vision-based multiple interacting targets tracking via on-line supervised learning. In *Proc. of the 10th European Conference on Computer Vision (ECCV)*, volume 3, pages 642–655, Marseille, France, October 2008. 1

[9] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Fort Collins, CO , USA, June 1999. 1, 3, 4, 5, 6

[10] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Trans. on Image Processing*, 19(6):1635–1650, June 2010. 1