Moving Object Segmentation by Pursuing Local Spatio-Temporal Manifolds

Yuanlu Xu

Technical Report, Sun Yat-Sen University, 2012.

Abstract

Although it has been widely discussed in video surveillance, background subtraction is still an open problem in the context of complex scenarios, e.g., dynamic backgrounds, illumination variations, and indistinctive foreground objects. To address these challenges, we propose a simple yet effective background subtraction method that learns and maintains dynamic texture models within spatio-temporal video patches (i.e. video bricks). In our method, the scene background is decomposed into a number of regular cells, within which we extract a series of video bricks. The background modeling is solved by pursuing manifolds (i.e. learning subspaces) with video bricks at each background location (cell). By treating the series of video bricks as consecutive signals, we adopt the ARMA (Auto Regressive Moving Average) Model to characterize spatio-temporal statistics in the subspace. In the initial learning stage, each manifold can be analytically learned, given sequences of video bricks. In the real-time detection stage, we segment foreground objects by estimating the appearance and state residuals of the new video bricks within the corresponding manifolds. Afterwards, the structure of each manifold is automatically updated by the Incremental Robust PCA (IRPCA) algorithm and its state variation by estimating the state of the new brick and re-solving linear problems. In the applications, we apply the proposed method in ten complex scenes outperform other state-of-the-art approaches. Moreover, the empirical studies of parameter settings and algorithm analysis are reported as well.

Index Terms

Video shot, unsupervised categorization, category discovery, graph partition.

I. Introduction

The problem of background subtraction (also referred as foreground extraction) has been extensively studied in the last decades, yet still remains open in surveillance applications due to the following difficulties:

Fig. 1. three difficult background scenes and our algorithm result: floating bottle: large dynamic area (left column), waving curtain: dynamic background and indistinctive foreground (middle column), lab: sudden light changes (right column)

- Dynamic backgrounds: it is not always static but randomly dynamic for a scene background, e.g., rippling water, heavy rain and camera jitter.
- Lighting and illumination variations, particularly with sudden changes.
- Indistinctive foreground objects: similar appearances between objects and scenes, or dim objects in low contrast.

The objective of this paper is a novel solution of background subtraction addressing the above mentioned difficulties. Fig. 1 exhibits some representative results generated by our method.

In the literature, the background subtraction techniques can be roughly divided into three categories, according to their representations.

The **pixel-processing** approaches are most widely used that represent scene backgrounds as a set of pixel processes. The values of a pixel in the scene is described by multi-modal distributions to adapt temporal changing in the scenes, which can be estimated in either parametric [7], [31], [8] or nonparametric [2], [27] way. To tolerate the background variations, a number of robust image features are utilized to improve robustness instead of using raw pixel values, such as Local Binary Pattern (LBP) operator and its extensions [20], [23], and color texture features [14]. Some typical noises (i.e. illumination changes and shadows) can be explicitly alleviated by extra estimations [5], [12]. These methods can usually handle the gradual changes as well as regularly dynamic variations, but suffer from inconsistent and noisy predictions in highly and randomly dynamic backgrounds.

To deal with the indistinctive foreground objects, some **motion-based** approaches are proposed to segment objects by salient motion detection [17], [1], [25]. However, the limitations come from the assumptions on the motion characteristics of the foreground object.

The **dynamic-appearance** methods mainly address dynamic background modeling that capture spatial interactions between pixels as well as temporal changes. A batch of diverse approaches are proposed to represent spatial structure of scenes, such as the joint distribution of image pixels [27], block-wise classifiers [11], structured adjacency graphs [16], auto-regression models [4], [18], random fields [28], and multi-layer model [15] etc. These models are usually automatically learned and updated with observations. For example, Monnet et al. [4] operate on image blocks and maintain the model by Candid Covariance-free Incremental PCA (CCIPCA) [13], and Cheng et al. [16] employ the generalized 1-SVM algorithm for model learning. Methods in this category show promising results in complex scenes and achieve impressive performance, but often suffer from degeneration during the maintenance. Moreover, these methods separate the spatial and temporal information in the modeling process.

The proposed method, belonging to a fourth category, builds background models on spatio-temporal patches (i.e. video bricks) that are the atomic units of processing. In the literature, Mahadevan et al. [25] propose to detect salient appearance and motion of foreground objects within spatiotemporal patches. They separate foreground from background by judging the distinguished regions which have different motions with the majority of the whole scene.

The **key contribution** of this work is to pursue dynamic texture manifolds within spatio-temporal representation and achieve a simple yet effective background model for almost all the real challenging cases in surveillance. More specifically, the proposed method advances in the following aspects.

1. Representing spatio-temporal statistics. We decompose the scene background into a number of regular cells, within which a batch of video bricks (e.g. $4 \times 4 \times 5$ pixels) are extracted as observations. In order to better represent video bricks and enhance the robustness to illumination variations, we propose a brick-based descriptor, namely Center Symmetric Spatio-temporal Local Ternary Pattern (CS-STLTP), instead of using pixel intensity. CS-STLTP is inspired by the 2D local pattern descriptor [23] and adapted to characterize video bricks.

2. Pursuing dynamic manifolds. We treat the series of video bricks as consecutive signals, and adopt the linear dynamic system (i.e. Auto Regressive Moving Average, ARMA model) to capture the spatio-temporal statistics in the subspace. A data matrix is formed at each location of background, in which each video brick is stretched as a vector. Then the basis vectors (i.e. eigenvectors) of the matrix can be estimated by using singular value decomposition, being treated as appearance parameters of the ARMA

model. The parameters of state variations are further estimated by minimizing reconstruction error, with the fixed appearance parameters. In the previous dynamic appearance methods [24], [4], appearance and dynamic motion are modeled separately, i.e. the former is related to a spatial texture model while the latter a temporal prediction model. In contrast, we jointly utilize the spatial and temporal information in the subspace of video bricks.

3. Maintaining dynamic manifolds online. Given new video bricks in the real-time detection stage, foreground objects is segmented by estimating the appearance and state residuals within the corresponding manifolds. Afterwards, the appearance of each manifold is automatically updated using incremental subspace learning. We employ IRPCA [26] to update the appearance of each manifold, and compare it with CCIPCA [13], which is applied in [4], [30]. The state variation of the sequence of video bricks is updated by estimating the state of the new brick and re-solving linear problems. By constructing the selective video bricks for model update, we effectively compensate the missing background samples caused by foreground occlusion and thus avoid the model disturbed by foreground objects.

The remainder of this paper is arranged as follows. We first present the feature and model representation in Section II. The initial learning, foreground segmentation and online learning mechanism are discussed in Section III. We demonstrate the experiments and comparisons in Section IV and finally conclude this paper in Section V with a summary.

II. Representation

In this section, we first discuss the assumptions we make in this paper, and then propose our video-brick-based feature as well as the dynamic manifold model.

A. Motivation

To handle the problem of dynamic background, we find previous works on a similar field: dynamic texture. Dynamic textures are image sequences of certain textures moving and changing under certain properties. Many previous works [24], [21], [10], [6] focus on modeling, recognizing, segmenting and synthesizing different dynamic textures. In our method, we regard scenes with dynamic backgrounds as a composition of many small and independent dynamic textures. The goal of representing dynamic background is achieved by modeling each dynamic texture separately.

As for the problem of illumination changes, Y. Zhao et al. [29], [30] did empirical studies on the manifold that background spatio-temporal video bricks live in. According to their observations, back-

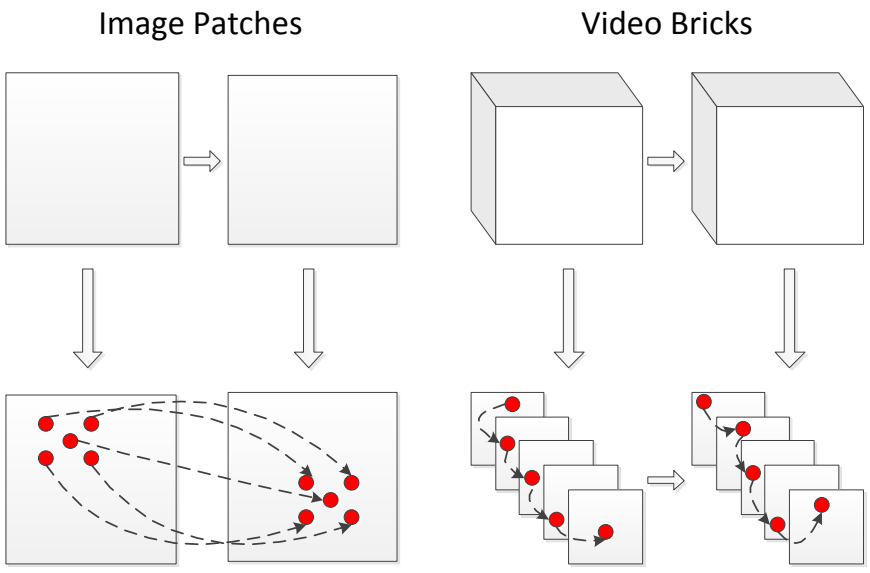

Fig. 2. Illustration of comparison between modeling with image patches and modeling with video bricks.

ground bricks, with only lighting changes, lie in a low-dimension manifold, while bricks with foreground occlusion lie in a much higher manifold.

Since indistinctive foreground objects shares similar appearance with the background, one way to solve this problem is that we model the variations of the background and predict the background appearance in the new frame. If the new appearance contradicts the predicted results, we can regard the background as indistinctive foreground occlusions.

In this paper, we propose a unified framework to settle all three problems above, based on two assumptions:

Assumption 1: Given the sequence of video bricks at a given background location, under dynamic variations or illumination changes, all bricks in this sequence lie in a low-dimension manifold, and the variations along the sequence obey local-linear.

Assumption 2: The bricks with indistinctive and distinctive foreground occlusions can be well separated from the background by distinguishing differences in both appearance and variations.

B. Spatio-temporal Video Brick

In this paper, we segment a video into cells and regard background in each cell as a spatio-temporal manifold consisting of small brick-like video patches. Video bricks of small size (e.g., $4\times4\times5$ pixels) are

less affected by compositionality and thus remains "pure". That is, they include relative simple content and can be thus generated by few bases. When new frames $I_n(n = 1, 2, \dots)$ arrive, we divide them into $h \times w$ 2-D patches $u_{i,n}$, where i indicates the cell index; at each cell i, each of the t patches are combined together to form a brick. We can then obtain a brick sequence $V_i = \{v_{i,1}, v_{i,2}, \dots, v_{i,n}, \dots\}$ at each cell for a video.

The video brick captures simple and non-linear motions of a group of pixels, while the image patch only describe the spatial statistics of pixels. As illustrated in Fig. 2, when stacking image patches together, the motion of the pixels we capture are linear uniform motions. As for stacking video bricks together, we actually loosen the constraints of linear uniform motions into linear uniform accelerated motions. That is, we capture the acceleration, rather than motion, of the pixels, which can describe more complicated motions. With the loosened constraints, pixels can move more "chaotic" and thus are more favorable in complicated dynamic background scenes.

Although we assume the low-dimension manifold can contain background video bricks under all possible lighting conditions, there is usually not enough training samples to pursue the right manifold, because the lighting changes is usually sudden (e.g., lights on, lights off) for most indoor scenes. As a result, we design a novel descriptor, CS-STLTP, for encoding video bricks instead of using RGB values. For a pixel in the brick, we first compute the local ternary patterns (LTP) of a few 2D spatio-temporal planes centering at the pixel, which extracted from 3D video volume (e.g., $3 \times 3 \times 3$ pixels), as Fig. 3 illustrates. The feature vectors from the planes are then concatenated together to form the descriptor of CS-STLTP.

CS-STLTP operator in each spatio-temporal plane compares the intensities in a center-symmetric direction with a contrast threshold. This operator has several advantages compared to original LTP: integrating of spatio-temporal statistics, lower feature dimension and better robustness to noises and shadows. Given a pixel located at (x_c, y_c, z_c) and its neighborhood pixels indexed by $\{0, \dots, M\}$, the feature vector on plane j is calculated as:

$$F^j(x_c, y_c, z_c) = \biguplus_{i=0}^{\frac{M}{2}-1} s_\tau(p_i, p_{i+\frac{M}{2}}), \tag{1}$$

where p_i and $p_{i+\frac{M}{2}}$ are the gray scales of center-symmetric pairs of neighborhood pixels i and $i + M/2$; The sign $\biguplus$ indicates stretching elements into a vector, and τ a contrast factor for the comparing range.

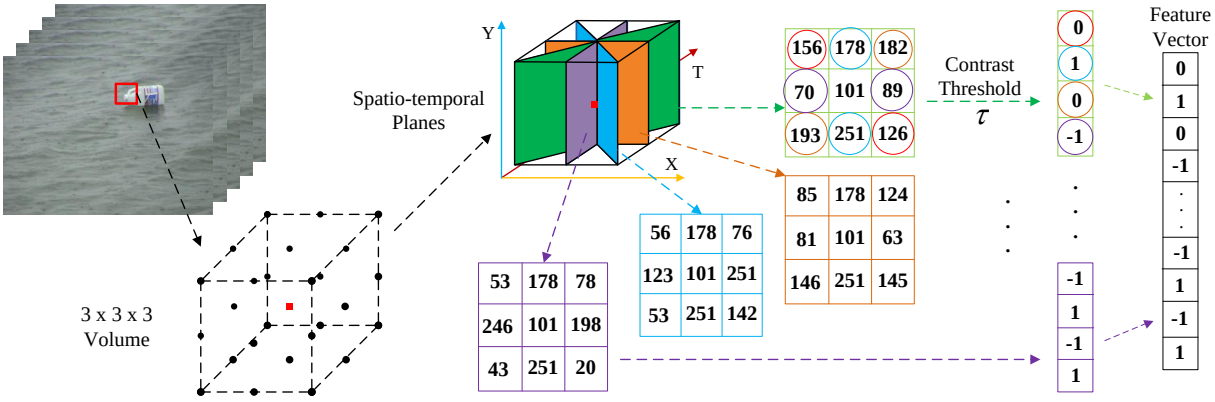

Fig. 3. Illustration of spatio-temporal representation using video bricks and CS-STLTP descriptor. For a pixel in the brick, the center-symmetric local ternary patterns are encoded with four spatio-temporal planes extracted from $3 \times 3 \times 3$ volume, and then stretched into the descriptor vector. We use the same color to represent the center-symmetric pairs of pixels.

The function s_τ is defined as follows:

$$s_\tau(p_i, p_{i+\frac{M}{2}}) = \begin{cases} 1, & \text{if } p_i > (1+\tau)p_{i+\frac{M}{2}}, \\ 0, & \text{if } p_i < (1-\tau)p_{i+\frac{M}{2}}, \\ \text{-1}, & \text{otherwise.} \end{cases} \tag{2}$$

In one spatio-temporal plane centering at a pixel, $M = 8$ neighborhood pixels are taken into account and the dimensions of feature vector is 4. For all 4 planes, the total dimension of CS-STLTP operator is $16d$ ($4d \times 4$ planes), which provides both the efficiency and the effectiveness of our method. In practice, we can apply the operator in each color channel.

C. Dynamic Manifold Model

Let $V = \{v_1, v_2, \ldots, v_n\}$ be a sequence of video bricks at a given background location. According to the assumption, we can use a set of bases $C = [C_1, C_2, \ldots, C_d]$ to represent the d-dimension manifold that V lies in. Each video brick v_i in V can be represented as

$$v_i = \sum_{j=1}^{d} z_{i,j} C_j + \omega_i, \tag{3}$$

where C_j is the j-th basis (j-th column of matrix C) of the manifold, $z_{i,j}$ the coefficient for C_j to represent v_i, ω_i the appearance residual (noises). We call C the appearance consistency term of the dynamic manifold model.

To model variations along V within the manifold, we treat the coefficients z_i as the state for each video brick v_i. and thus find the linear relationship in the sequence of coefficients $Z = \{z_1, z_2, \ldots, z_n\}$. That is,

$$z_{i+1} \;=\; Az_i \;+\; \epsilon_i, \tag{4}$$

where z_{i+1}, z_i are two consecutive states, ϵ_i the state residual. We regard A as the temporal coherence term of the dynamic manifold model.

Therefore, the problem of pursuing dynamic manifold (i.e. manifold which video bricks lies in and state variations which the sequence follows) is to solve the appearance consistency C and temporal coherence A. Due to the state sequence Z unknown, we jointly solve C, A, Z by minimizing an empirical energy function $f_n(C, A, Z)$:

$$\begin{aligned} &min. \quad f_n(C, A, Z) \;=\; \frac{1}{n}\sum_{i=1}^{n}\,(\,\frac{1}{2}\parallel v_i - Cz_i \parallel_2^2 \;+\frac{1}{2}\parallel z_i - Az_{i-1} \parallel_2^2\,) \\ &(\,V \in \mathbb{R}^{m\cdot n},\, Z \in \mathbb{R}^{d\cdot n},\, C \in \mathbb{R}^{m\cdot d},\, A \in \mathbb{R}^{d\cdot d}\,) \end{aligned} \tag{5}$$

Note that there exist two problems in the energy function $f_n(C, A, Z)$: 1). $f_n(C, A, Z)$ is not jointly convex, but convex with respect to each of the two groups (C, A) and Z when the other is fixed. 2). the variables C, A, and Z are time-invariant in $f_n(C, A, Z)$, while the problem of background subtraction requires online maintenance.

Therefore, we re-formulate Equ.(8) into a linear dynamic system plus a statistical robustness term for efficiency, and regard it as the dynamic manifold model:

$$\begin{aligned} &v_{i,n} \;=\; C_{i,n}\, z_{i,n} \;+\; \omega_{i,n}, \\ &z_{i,n+1} \;=\; A_{i,n}\, z_{i,n} \;+\; B_{i,n}\, \epsilon_{i,n}, \\ &\omega_{i,n} \overset{IID}{\sim} N(0, \Sigma_{\omega_i}),\;\; \epsilon_{i,n} \overset{IID}{\sim} N(0, I_{d_\epsilon}), \end{aligned} \tag{6}$$

where appearance consistency term $C_{i,n} \in \mathbb{R}^{h\,w\,t\times d}$ indicates the basis elements and $z_{i,n} \in \mathbb{R}^d$ the varying state. The temporal coherence $A_{i,n} \in \mathbb{R}^{d\times d}$ and statistical robustness term $B_{i,n} \in \mathbb{R}^{d\times d_\epsilon}$ maintain the evolution of z_i over time effectively and efficiently. d and d_ϵ are the subspace dimensions of spatio-temporal statistics and noise, respectively. $\omega_{i,n}$ and $\epsilon_{i,n}$ are defined for the measurement noise and the process noise, respectively.

Unlike most linear dynamic systems which are time-invariant, the formulation we propose is a time-variant linear dynamic system. The parameters C, A, B can be maintained incrementally, in order to process online.

To solve Equ. 6, though there are iterative solutions [22] which leads to a local minimum, we instead seek analytical solution due to the huge computational complexity. In this paper, we employ the sub-optimal solution proposed by S. Soatto et al. [24]. See Section III-A for details about the solution.

III. Learning Algorithm

In this section, we discuss the learning of the background dynamic manifold model. The model learning process is split into two stages: initial model learning and online model maintenance. For initial model learning, we provide "pure" background samples (i.e. background video bricks) to roughly identify the dynamic manifold. For online model maintenance, we adapt the dynamic manifold so that the model can remain effective.

A. Initial Model Learning

In the initial stage, we simplify Equ. 6 into a time-invariant linear dynamic system. Given a brick sequence $V_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,n}\}$ for initial model learning, we present an algorithm to identify the model parameters $C_{i,n}$, $A_{i,n}$, $B_{i,n}$, based on the sub-optimal solution proposed in [24].

In the following discussion, due to the independency among each cell, we ignore the notation of cell index i for simplicity. To guarantee the Equ.(6) has an unique and canonical solution, we postulate

$$n \gg d, \;\; \text{Rank}(C_n) \;=\; d, \;\; C_n^{\top} C_n \;=\; I_d, \tag{7}$$

where I_d is the identity matrix of dimension $d \times d$.

The best estimate of the appearance consistency term C_n is formulated as

$$\begin{aligned} C_n^* \;&=\; \arg\max_{C_n} \mid W_n - C_n [\; z_1 \; z_2 \; \cdots \; z_n \;] \mid \\ &=\; \arg\max_{C_n} \mid [\; v_1 \; v_2 \; \cdots \; v_n \;] - C_n [\; z_1 \; z_2 \; \cdots \; z_n \;] \mid, \end{aligned} \tag{8}$$

where W_n is the data matrix composed of observed video bricks. Following in [21], Equ.(8) satisfies the full rank approximation property and can be thus solved by SVD. The appearance matrix C_n is treated as the first d principal components of W_n, and the state matrix $[\; z_1 \; z_2 \; \cdots \; z_n \;]$ as the product of $d \times d$ sub-matrix of Σ and corresponding first d columns of $V^{\top}$. To solve with SVD, we calculate the mean vector μ_n that is concatenated from averages of each row of W_n.

The temporal coherence term A_n is calculated by solving the following linear problem:

$$A_n^* \;=\; \arg\max_{A_n} \mid [\; z_2 \; z_3 \; \cdots \; z_n \;] - A_n [\; z_1 \; z_2 \; \cdots \; z_{n-1} \;] \mid. \tag{9}$$

The statistical robustness term B_n is estimated by the reconstruction error E

$$\begin{aligned} E &= [\, z_2\ z_3\ \cdots\ z_n\,] - A_n [\, z_1\ z_2\ \cdots\ z_{n-1}\,] \\ &= B_n [\, \epsilon_1\ \epsilon_2\ \cdots\ \epsilon_{n-1}\,], \end{aligned} \tag{10}$$

where $B_n \cong \frac{1}{\sqrt{n-1}} E$. Since the rank of A_n is d and $d \ll n$, the rank of input-to-state noise d_ϵ is assumed to be smaller and equal to d. That is, the dimension of E can be further reduced by computing SVD: $E = U_\epsilon \Sigma_\epsilon V_\epsilon^\top$, and we have

$$B_n = \frac{1}{\sqrt{n-1}} \left[\, U_\epsilon^1 \quad \cdots \quad U_\epsilon^{d_\epsilon} \,\right] \begin{bmatrix} \Sigma_\epsilon^1 & & \\ & \ddots & \\ & & \Sigma_\epsilon^{d_\epsilon} \end{bmatrix}. \tag{11}$$

The values of d, d_ϵ represent the manifold dimensions, which are essentially lie in the spatio-temporal complexity of the dynamic textures. For example, video bricks containing static contents can be well described with a function of low dimensions while highly dynamic video bricks (e.g., from an active fountain) require more basis to generate. Therefore, in the proposed method, we adaptively determine d, d_ϵ by thresholding eigenvalues in Σ and Σ_ϵ.

$$\begin{aligned} d^* &= \arg\max_d \Sigma^d > T_d, \\ d_\epsilon^* &= \arg\max_{d_\epsilon} \Sigma_\epsilon^{d_\epsilon} > T_{d_\epsilon}, \end{aligned} \tag{12}$$

where Σ^d indicates the d-th eigenvalue in Σ, $\Sigma_\epsilon^{d_\epsilon}$ the d_ϵ-th eigenvalue in Σ_ϵ.

B. Online Model Maintenance

In the model maintenance stage, we have two goals: segmenting foreground objects and adaptively updating model parameters.

Given the new video brick v_{n+1}, according to the assumption, we can determine whether a pixel in v_{n+1} belongs to the foreground objects or background by thresholding its appearance residual and state residual. The state of v_{n+1} within the manifold is estimated as

$$z'_{n+1} = C_n^\top v_{n+1}. \tag{13}$$

According to the estimated state z'_{n+1}, we define the appearance residual of v_{n+1} as

$$\omega_{n+1} = v_{n+1} - C_n z_{n+1}. \tag{14}$$

Since we assume the noises in the state z'_{n+1} obey a Gaussian distribution, we regard the noise ϵ_n in the state z'_{n+1} as the state residual of v_{n+1}. That is,

$$\begin{aligned} B_n\epsilon_n &= z'_{n+1} - A_n z_n, \\ \epsilon_n &= \mathrm{pinv}(B_n)\,(z'_{n+1} - A_n z_n), \end{aligned} \tag{15}$$

where pinv denotes the pseudo-inverse.

Given the state residual ϵ_n of the new video brick v_{n+1}, if ϵ_n is greater than the threshold T_ϵ, v_{n+1} can be intuitively postulated "containing foreground objects". To achieve pixel-wised segmentation, we assume that the appearance residual of a pixel in v_{n+1} is the corresponding sub-vector of ω_{n+1}. If the appearance residual of a pixel in v_{n+1} is greater than the threshold T_ω, we label this pixel as foreground and reversely as background.

After the foreground segmentation, we employ the background pixels to update the model. Due to foreground occlusion, we define a noise-free video brick under the current model to compensate the missing background samples. The noise-free video brick $\hat{v}_{n+1}$ is defined as

$$\begin{aligned} \hat{z}_{n+1} &= A_n z_n, \\ \hat{v}_{n+1} &= C_n\,\hat{z}_{n+1}, \end{aligned} \tag{16}$$

If a pixel is labeled as foreground, we use the corresponding sub-vector of the noise-free video brick $\hat{v}_{n+1}$ as the update sample and otherwise we use the sample from the new video brick v_{n+1}. Therefore, we construct a selective video brick $\bar{v}_{n+1}$ for model update. As the empirical analysis in Fig. 4, the selective video brick $\bar{v}_{n+1}$ is helpful to avoid model drift, in the case of foreground objects appearing.

Recall in Equ.(8), W_n is spanned by the appearance consistency term C_n. To update C_n, we regard W_{n+1} as the extension by adding a new column $\bar{v}_{n+1}$ to W_n. Then this problem of updating C_n can be formulated as the task of incremental subspace learning.

To cope with incremental subspace learning, incremental PCA (IPCA) is an extensively studied subject [26], [9], [19], [3] focused on how to express the covariance matrix incrementally. In this paper, we adopt the state-of-the-art work [26], which employs an extra robust estimation procedure to guarantee further robustness.

With the new observation $\bar{v}_{n+1}$, the parameter of the robust function ρ is firstly estimated

$$\begin{aligned} \rho &= [\rho_1, \rho_2, \ldots, \rho_{|\bar{v}_{n+1}|}]^\top \\ \rho_j &= \beta\sigma_j,\ j = 1, 2, \ldots, |\bar{v}_{n+1}| \\ \sigma_j &= \max_{i=1}^{d} \sqrt{\lambda_{n,i}}\,|\,c^j_{n,i}\,|\,, \end{aligned} \tag{17}$$

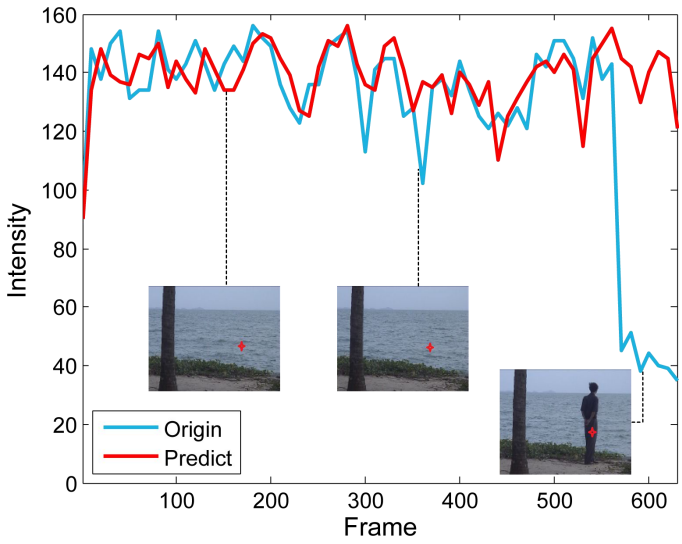

Fig. 4. Robustness of the model maintenance. We analyze the model maintenance using a video including dynamic water surface. When foreground object appears during frame 551 to 632, the values (i.e. intensities) predicted by the model remain stable.

where β is a fixed coefficient, σ_j denotes the standard deviation of the jth element of the observation vectors $\bar{v}_{n+1}$. σ_j is approximated by the maximal projection of the current eigenvectors on the jth dimension (weighted by their corresponding eigenvalues).

Next we compute the residual error of a new vector $\bar{v}_{n+1}$ by

$$r_{n+1} = C_n C_n^\top \bar{v}_{n+1} - \bar{v}_{n+1}. \tag{18}$$

By using the robust function $w(t) = \frac{1}{1+(t/c)^2}$, we define the weighted update sample

$$\mathbf{v}_{n+1} = \sqrt{w(r_{n+1})}\, \bar{v}_{n+1}. \tag{19}$$

The weighted update sample , instead of $\bar{v}_{n+1}$, is used in the following incremental subspace learning. The mean vector μ_{n+1} of W_{n+1} is updated by,

$$\mu_{n+1} = \alpha\mu_n + (1-\alpha)\mathbf{v}_{n+1}, \tag{20}$$

where α indicates the update weight.

For a d-dimension manifold, with eigenvectors C_n and corresponding eigenvalues Λ_n, the covariance matrix can be approximated as

$$\mathrm{Cov}_n \approx \sum_{i=1}^{d} \lambda_{n,i} c_{n,i} c_{n,i}^\top = C_n \Lambda_n C_n^\top, \tag{21}$$

where $c_{n,i}$ and $\lambda_{n,i}$ denote the i-th eigenvector and eigenvalue, respectively. With the new observation $\bar{v}_{n+1}$, the new covariance matrix Cov_{n+1} is formulated as

$$\begin{aligned}\mathrm{Cov}_{n+1} &= \alpha\,\mathrm{Cov}_n + (1-\alpha)\mathbf{v}_{n+1}\,\mathbf{v}_{n+1}^\top \\ &\approx \alpha C_n \Lambda_n C_n^\top + (1-\alpha)\mathbf{v}_{n+1}\,\mathbf{v}_{n+1}^\top \\ &= \sum_{i=1}^{d} \alpha\,\lambda_{n,i}\,c_{n,i}\,c_{n,i}^\top + (1-\alpha)\,\mathbf{v}_{n+1}\,\mathbf{v}_{n+1}^\top.\end{aligned} \tag{22}$$

To simplify computation, the new covariance matrix Cov_{n+1} is re-formulated as

$$\mathrm{Cov}_{n+1} = Y_{n+1}Y_{n+1}^\top, \tag{23}$$

where $Y_{n+1} = [y_{n+1,1}\, y_{n+1,2}\, \ldots\, y_{n+1,d+1}]$ and each column $y_{n+1,i}$ in Y_{n+1} is defined as

$$y_i = \begin{cases} \sqrt{\alpha\lambda_i}\,c_{n,i}, & \text{if } 1 < i < d, \\ \sqrt{1-\alpha}\,\mathbf{v}_{n+1}, & \text{if } i = d+1. \end{cases} \tag{24}$$

Instead of directly estimating principal components from Cov_{n+1}, Y. Li proposes to eigen-decompose a smaller matrix $Y_{n+1}^\top Y_{n+1}$,

$$(Y_{n+1}^\top\, Y_{n+1})\, e_{n+1,i} = \lambda_{n+1,i}\, e_{n+1,i} \qquad i = 1, 2, \ldots, d+1, \tag{25}$$

where $e_{n+1,i}$ and $\lambda_{n+1,i}$ is the i-th eigenvector and eigenvalue of matrix $Y_{n+1}^\top Y_{n+1}$, respectively. Defining $c_{n+1,i} = Y_{n+1}e_{n+1,i}$ and left-multiplying Y_{n+1} to each side of Equ.(25), we have

$$\begin{aligned} Y_{n+1}\,Y_{n+1}^\top\,Y_{n+1}\,e_{n+1,i} &= \lambda_{n+1,i}\,Y_{n+1}\,e_{n+1,i}, \\ \mathrm{Cov}_{n+1}\;c_{n+1,i} &= \lambda_{n+1,i}\,c_{n+1,i} \qquad i = 1, 2, \ldots, d+1. \end{aligned} \tag{26}$$

That is, we obtain the updated eigenvectors C_{n+1} and corresponding eigenvalues Λ_{n+1} of the new covariance matrix Cov_{n+1}. Noting each time when a new observation is added, the dimension of the subspace is automatically increased by one. To guarantee the appearance of the manifold remains stable, we keep the first d eigenvectors and corresponding eigenvectors and discard the least significant principal component.

After the appearance of each manifold has been updated, we focus on updating parameters of the state variation A_{n+1}, B_{n+1}, based on the latest state z_{n+1}. z_{n+1} under C_{n+1} is re-estimated as follows,

$$z_{n+1} = C_{n+1}^\top \mathbf{v}_{n+1}, \tag{27}$$

and A_{n+1} can be further calculated, by re-solving the linear problem of a fixed number of latest observed states,

$$A_{n+1}\,[\; z_{n-l+1}\;\cdots\; z_n\;] = [\; z_{n-l+2}\;\cdots\; z_{n+1}\;], \tag{28}$$

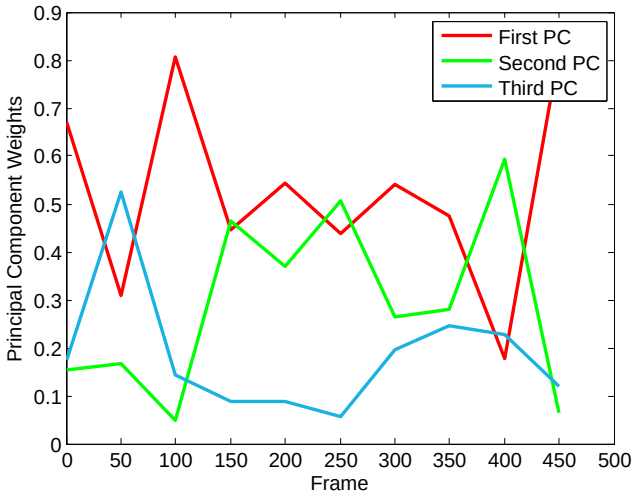

Fig. 5. Model maintenance against degeneration. The weights of 3 principal components (normalization $z(n)$ to 1) updated by IRPCA are visualized during maintaining the model with the dynamic backgrounds (i.e. the fountain). Each component keeps updating actively in a proper range rather than becoming trivial or monopolizing.

where l indicates the number of latest observed states, i.e. the span of observations. And similarly, we update B_{n+1} by computing the new reconstruction error $E = [z_{n-l+2} \ \cdots \ z_{n+1}] - A_{n+1} [z_{n-l+1} \ \cdots \ z_n]$ and evaluate the first d_ϵ components of E.

To verify the model maintenance resists the model degeneration, we observe the state $z(n)$ (the weights of all principal components) over the video. As the empirical study shown in Fig. 5, all three principal components (PCs) remain active range rather than becoming trivial or monopolizing.

Applying the above mechanism, we summarize the Spatio-temporal Dynamic Manifolds (STDM) algorithm in Algorithm 1. Our algorithm is separated into two steps: initial learning and online learning. Recall Section III-A, we mainly employ SVD and linear programming in the initial learning. Therefore the time complexity of initial learning is $O(n^3)+O(n^2) = O(n^3)$, where n denotes the number of video bricks in the sequence at each cell. As for online learning, incremental subspace learning and linear programming are utilized. Given a d-dimension manifold, the time complexity for IRPCA is $O(dn^2)$. And thus the total time complexity for online learning is $O(dn^2)+O(l^2)$, where l is the number of states used to solve the linear problem.

Algorithm 1: Spatio-temporal Dynamic Manifolds

Input: Brick sequence $V_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,n}\}$ for all the cells in a video

Output: Segmented foreground pixels in V_i

forall the *Cells in a video* **do**

  for *Initial Learning bricks* $v_j \in \{v_{i,1} \ldots v_{i,n}\}$ **do**

    Calculate brick descriptor by (1)-(2);

  end

  Identify model parameters $C_n, A_n, B_n, \mu_n, d, d_\epsilon$ by (7)-(12);

  for *Online Learning bricks* $v_j \in \{v_{i,n+1} \ldots\}$ **do**

    Calculate brick descriptor by (1)-(2);

    Segment foreground objects by (13)-(15);

    Construct the selective video brick $\bar{v}_j$ for update;

    Update parameters of the appearance of the manifold μ_j, C_j by IRPCA (17)-(26);

    Update parameters of the state variation of the manifold A_j, B_j by (27)-(28);

  end

end

IV. Experiments

We first specify the datasets used in the experiments as well as the parameter settings, and then present the experimental results and comparisons. The further discussions of system parameters are proposed at last.

A. Datasets and settings

We collect a number of challenging videos to validate our approach, which are publicly available or from real surveillance systems. Two of them (AirportHall and Trainstation) from the PETS database[1] include crowded pedestrians and moving cast shadows; five highly dynamic scenes [2] including waving curtain active fountain, swaying trees, water surface; the others contain extremely difficult cases such as heavy rain, sudden and gradual light changing. Most of the videos have several thousand frames, with 20 frames manually labeled for each as groundtruth.

[1]Downloaded from http://www.cvg.rdg.ac.uk/slides/pets.html.

[2]Downloaded from http://perception.i2r.a-star.edu.sg

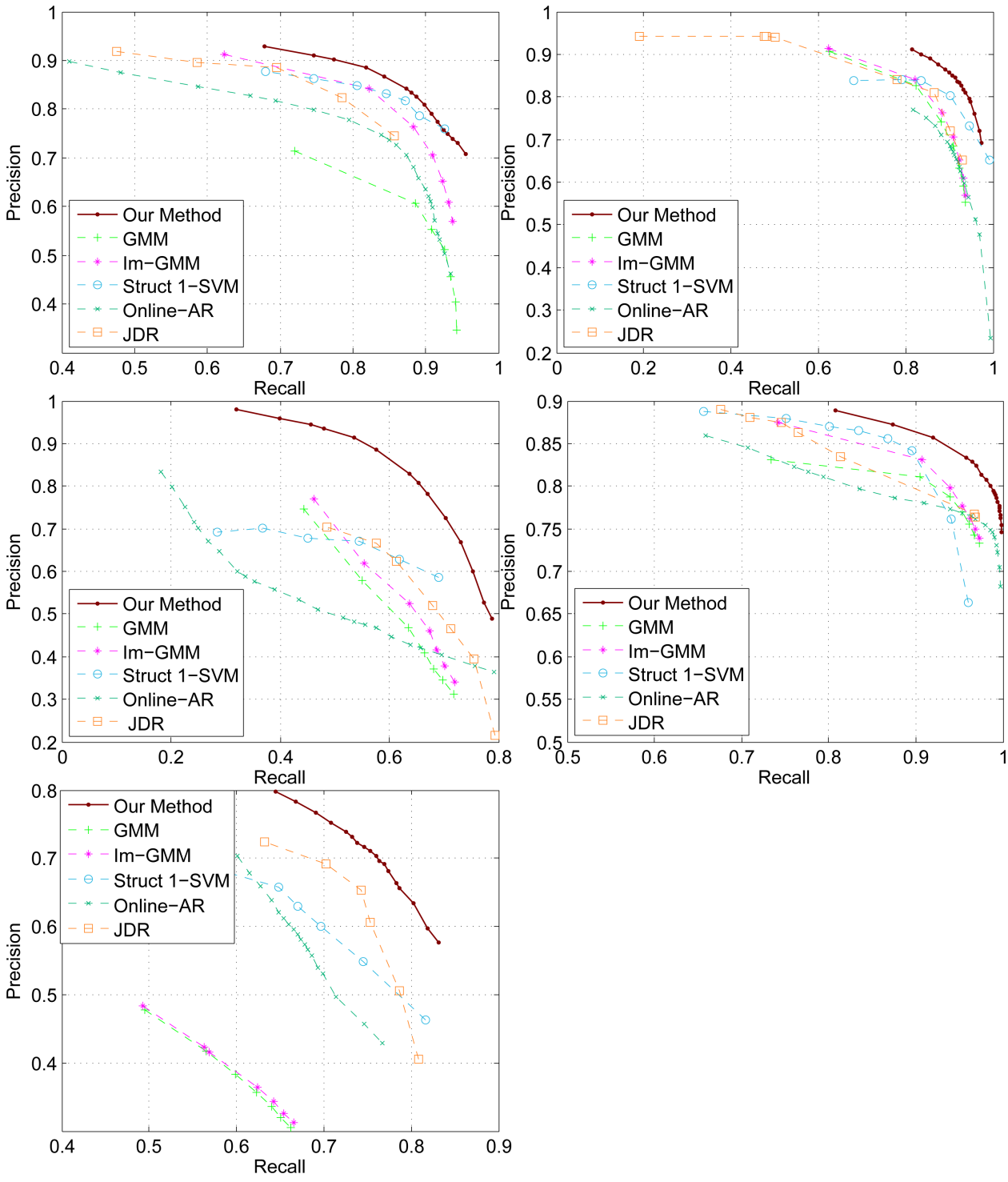

Fig. 6. Experimental results (using CS-STLTP as feature and CCIPCA as update strategy) and comparisons on 5 videos: first row left, the scene including a dynamic curtain and indistinctive foreground objects (i.e. having similar appearance with backgrounds); first row right, the scene with heavy rain; second row left, an indoor scene with the sudden lighting changes; second row right, the scene with dynamic water surface; third row, a busy airport. The precision-recall (PR) curve is introduced as the benchmark measurement for all the 6 algorithms.

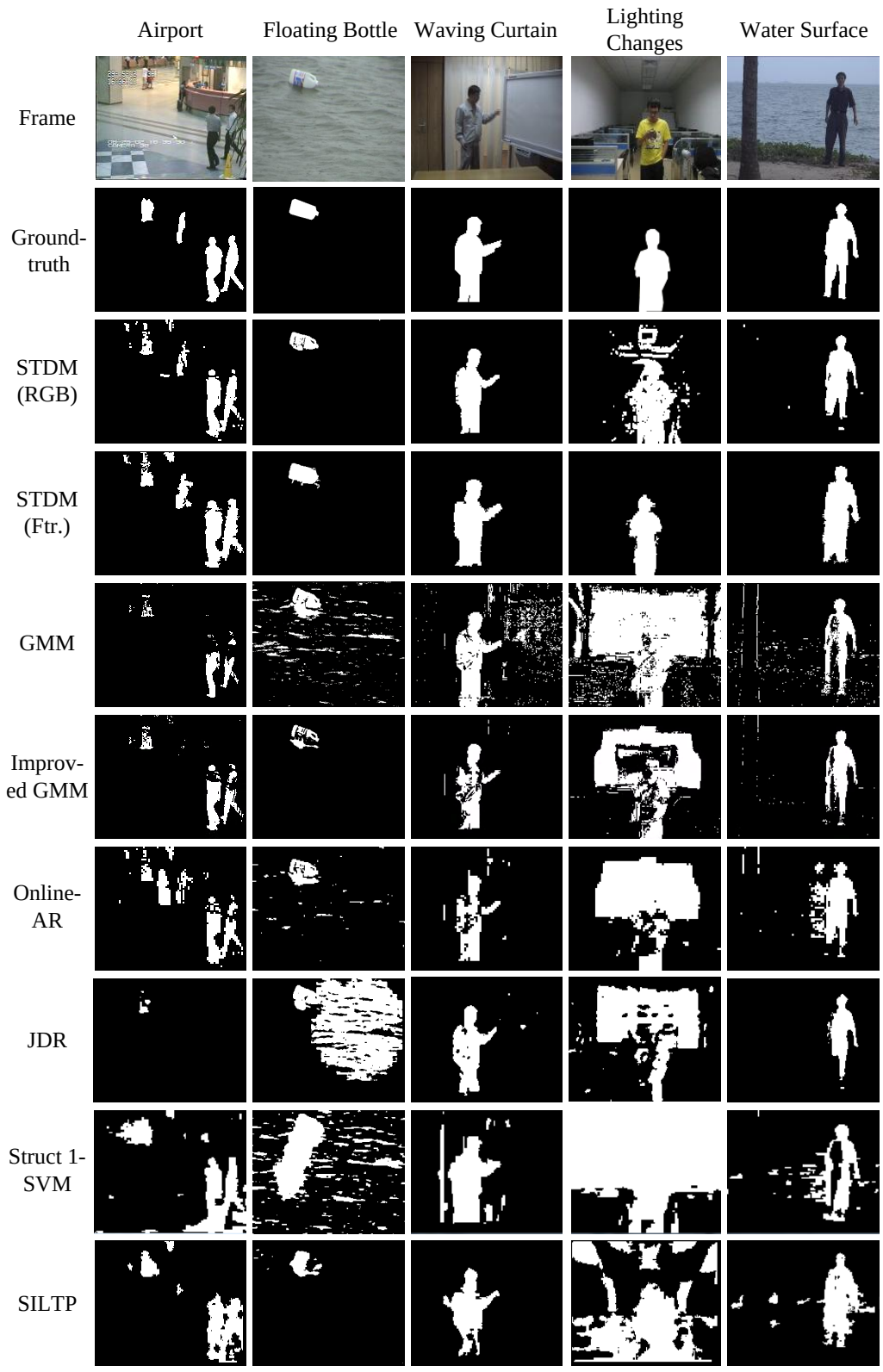

Fig. 7. Sampled results of background subtraction generated by our approach (using RGB or CS-STLTP as feature and CCIPCA as update strategy) and other competing methods. The complete results can be found in the supplemental videos.

TABLE I

QUANTITATIVE RESULTS AND COMPARISONS ON THE 10 COMPLEX VIDEOS USING THE F-SCORE (%) MEASUREMENT. THE LAST TWO COLUMNS REPORT THE RESULTS OF OUR METHOD WITH EITHER RGB OR CS-STLTP AS FEATURE AND CCIPCA AS UPDATE STRATEGY.

Scene	GMM [7]	Im-GMM [31]	Online-AR [4]	JDR [27]	Struct1-SVM [16]	SILTP [23]	STDM(RGB)	STDM(Ftr.)
1# Airport	46.99	47.36	62.72	60.23	65.35	68.14	**75.52**	66.40
2# Floating Bottle	57.91	57.77	43.79	45.64	47.87	59.57	69.04	**75.85**
3# Waving Curtain	62.75	74.58	77.86	72.72	77.34	78.01	**87.74**	79.57
4# Active Fountain	52.77	60.11	70.41	68.53	74.94	76.33	76.85	**79.68**
5# Heavy Rain	71.11	81.54	78.68	75.88	82.62	76.71	**86.86**	81.35
6# Sudden Light	47.11	51.37	37.30	52.26	47.61	52.63	51.56	**70.23**
7# Gradual Light	51.10	50.12	13.16	47.48	62.44	54.86	54.84	**72.52**
8# Train Station	65.12	68.80	36.01	57.68	61.79	67.05	**73.43**	66.46
9# Swaying Trees	19.51	23.25	**63.54**	45.61	24.38	42.54	43.7	48.49
10# Water Surface	79.54	86.01	77.31	84.27	83.13	74.30	**88.54**	87.88
Average	55.39	59.56	57.02	60.23	59.79	63.08	70.81	**72.84**

Our algorithm has been adopted in a real video surveillance system and achieves a satisfactory performance. The system is capable of processing $8 \sim 10$ frames per second in the resolution 160×128 pixels.

All the parameters are fixed in the experiments, including the contrast threshold of CS-STLTP descriptor $\tau = 0.2$, the dimension threshold of ARMA model $T_d = 0.5$, $T_{d_\epsilon} = 0.5$, the span of observations for model updating $l = 60$, and the size of bricks $4 \times 4 \times 5$. For foreground segmentation, the threshold of appearance residual $T_\omega = 3$, update threshold $T_\epsilon = 3$ for CS-STLTP feature and $T_\omega = 5$, $T_\epsilon = 4$ for RGB. In online model maintenance, the robust coefficient $\beta = 2.3849$, the update weight $\alpha = 0.94$ for IRPCA.

In our experiment, we find that segmentation with CS-STLTP detects is sensitive to the contours of foreground objects and yet insensitive to the flat regions inside foreground objects. Therefore, we utilize a standard OpenCV postprocessing to fill contours whose area is greater than 20 pixels and eliminate pieces less than 20 pixels.

We utilize the F-score as the benchmark metric, which measures the segmentation accuracy by con-

sidering both the recall and the precision. The F-score is defined as

$$F = \frac{2TP}{2TP + FP + FN}, \tag{29}$$

where TP is true positives (foreground objects), FN false negatives (false background pixels), FP false positive (false foreground pixels).

B. Experimental results

Experimental results. We compare the proposed method (STDM) with six start-of-the-art online background subtraction algorithms including Gaussian Mixture Model (GMM) [7] as baseline, improved GMM [31][3], online auto-regression model [4], non-parametric model with scale-invariant local patterns [23], discriminative model using generalized Struct 1-SVM [16][4], and the Bayesian joint domain-range (JDR) model [27][5]. We adopt the provided codes of the methods [7], [31], [16], [27] and implement the methods [23], [4] according to their descriptions. The F-scores ($\%$) over all 10 videos are reported in Table I. We also exhibit the results and comparisons using the precision-recall (PR) curves, as shown in Fig. 6. Due to space limitation, we only show results on 5 videos. From the results, we can observe that the proposed method outperforms the other methods. For the scenes with highly dynamic backgrounds (e.g., the scene # 2 #5 and #10), the improvements made by our method are more than 10%. And the system enable us to well handle the indistinctive foreground objects (i.e. small objects or background-like objects in the scene #1, #3). Moreover, we make significant improvements (i.e. $15\% \sim 25\%$) in the scene #6 and #7 including both sudden and gradual lighting changes, without extra illumination estimates. The benefit of using the proposed CS-STLTP feature is clearly validated as well. A number of sampled results of background subtraction are exhibited in Fig. 7.

C. Discussion

We discuss the selection of model parameters and incremental subspace learning method, based on the following empirical study.

Adaption efficiency. Like other online-learning background models, there is a trade-off between the model stability and adaption efficiency. The corresponding parameter in our method is the learning rate r. We tune r in the range of $0 \sim 12$ fixed the other model parameters and visualize the quantitative

[3]Available at http://dparks.wikidot.com/background-subtraction

[4]Available at http://www.cs.mun.ca/∼gong/Pages/Research.html

[5]Available at http://www.cs.cmu.edu/∼yaser/

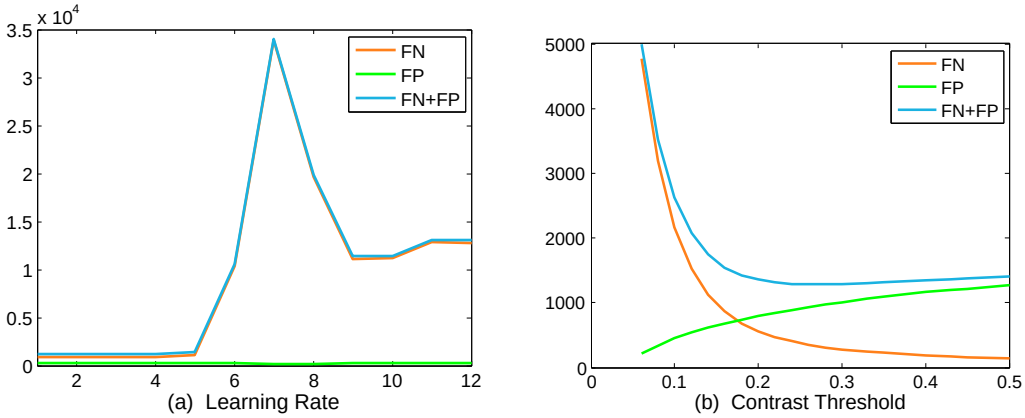

Fig. 8. Discussion of parameter selection: (i) learning rate r for model adaption (in (a)) and (ii) the contrast threshold of CS-STLTP feature τ (in (b)). In each figure, the horizontal axis represents the different parameter values; the three lines in different colors denote, respectively, the false alarm (FA), false negative (FN), and the sum of FA and FN.

results of background subtraction, as shown in Fig. 8(a). From the results, we can observe this parameter is insensitive in range $0 \sim 5$ in our model. In practice, once the scene is extremely busy and crowded, it could be set as a relative small value to keep model stable.

Feature effectiveness. The contrast threshold τ is the only parameter in CS-STLTP operator, which affects the characterizing power for dynamic textures (i.e. spatio-temporal information within video bricks). From the empirical results of parameter tuning, as shown in Fig. 8 (b), we can observe that the appropriate range for τ is $0.15 \sim 0.25$. In practice, a very small value (i.e. < 0.15) for τ could lead the model sensitive to noises while some foreground regions of homogeneous appearances could be missed under a great value setting (i.e. > 0.25).

Background decomposition. The size of video bricks is critical to the performance of the proposed algorithm. In the empirical study shown in Fig. 9, we visualize the prediction residuals (calculated by Equ.(16)) of the synthesized bricks against observations during the initial learning stage and online updating stage of the model, respectively. These curves provide a rough but intuitive measure: the consistency of synthesized data and observations. Once we degenerate the brick-based representation into string-wise (e.g., $1 \times 1 \times 3$ and $1 \times 1 \times 5$) or block-wise (e.g., $2 \times 2 \times 1$ and $4 \times 4 \times 1$), the consistencies of changing unavoidably decrease. This observation is accordant with the previous theoretical studies in dynamic texture modeling [24], [10]. Representing dynamic textures with the LDS (i.e. ARMA model) assumes the subspaces of video bricks to be linear and analytical, and thus the decomposition of subspaces

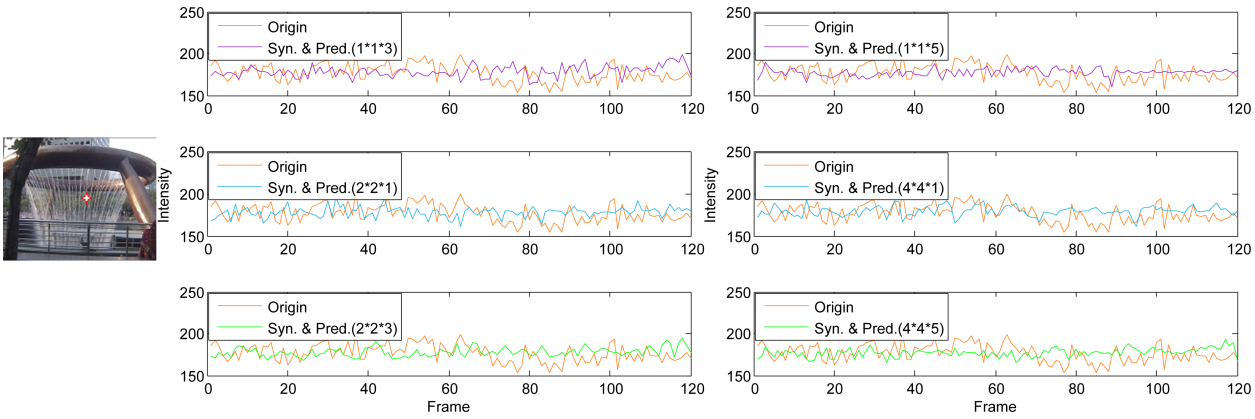

Fig. 9. Empirical study of the size of video bricks. We display the scene in the left image and supervise a fixed location (labeled by the red star) at the background. The other 6 figures show the appearance changes of the observations (yellow curves) and synthesized data (green curves), with respect to different sizes of video bricks. The vertical axis of each figure represent average pixel intensity at the location. The brick sizes for figures (from left to right) are fixed as, respectively, $1 \times 1 \times 3$, $1 \times 1 \times 5$, $2 \times 2 \times 1$, $4 \times 4 \times 1$, $2 \times 2 \times 3$ and $4 \times 4 \times 5$.

is potentially related to the dynamic variants and magnitudes in the scene. Although the size $2 \times 2 \times 3$ also looks good from the curves, we use the larger size $4 \times 4 \times 5$ in our system for better efficiency. In practice, we can flexibly reduce the size to adapt the extremely dynamic appearances.

Incremental subspace learning method. To verify IRPCA is better than CCIPCA, we substitute IRPCA with CCIPCA and compare the quantitative results, as shown in Table II. In our experiment, we set the learning rate $r = 3$, the iteration number $it = 2$ for CCIPCA. From the results, IRPCA performs much better in scenes with dynamic background and slightly better in scenes with illumination changes, while CCIPCA behaves much better in scenes with heavy foreground occlusions. The results make sense in that CCIPCA assumes the appearance of the manifold not greatly changes and IRPCA is a universal incremental subspace learning approach, which is more suitable for updating background with great dynamics. Moreover, IRPCA is also more efficient than CCIPCA.

V. Conclusion

This paper studies a simple yet effective method for background subtraction, in which we represent scene backgrounds with brick-like spatio-temporal video patches and pursue the dynamic manifolds with the linear dynamic system. Sufficient experiments and analysis are presented to validate that our method is solid and applicable for the real video surveillance systems.

TABLE II

QUANTITATIVE ACCURACY (F-SCORE) AND EFFICIENCY ON THE 10 COMPLEX VIDEOS.

Comparison		RGB + CCIPCA	RGB + IRPCA	CS-STLTP + CCIPCA	CS-STLTP + IRPCA
Accuracy (%)	#1	**75.52**	65.13	**66.4**	57.6
	#2	69.04	**70.02**	75.85	**78.17**
	#3	**87.74**	78.47	**79.57**	70.93
	#4	76.85	**81.38**	79.68	**85.46**
	#5	**86.86**	79.84	**81.35**	75.29
	#6	51.56	**53.63**	70.23	**74.57**
	#7	54.84	**59.79**	72.52	**77.41**
	#8	**73.43**	68.69	**66.46**	60.35
	#9	43.7	**70.17**	48.49	75.89
	#10	88.54	**89.43**	87.88	**88.68**
	Average	70.81	**71.66**	72.84	**74.44**
Efficiency (fps)		2.3	**4.1**	0.6	**1.0**

REFERENCES

[1] A. Bugeau and P. Perez. Detection and segmentation of moving objects in highly dynamic scenes. *Computer Vision and Image Understanding*, 113(4):459–476, 2009.

[2] 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*, 90(7):1151–1163, 2002.

[3] A. Levy and M. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *IEEE Transactions on Image Processing*, pages 1371–1374, 2000.

[4] A. Monnet, A. Mittal, N. Paragios, V. Ramesh. Background modeling and subtraction of dynamic scenes. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 2003.

[5] A. Prati et al. Detecting moving shadows: Algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):918–923, 2003.

[6] A. Ravichandran, and R. Vidal. Optical flow estimation and segmentation of multiple moving dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):158–171, 2010.

[7] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. *CVPR*, 1999.

[8] D. Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832, 2005.

[9] D. Skocaj and A. Leonardis. Weighted and robust incremental method for subspace learning. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1494–1500, 2003.

[10] G. Doretto, D. Cremers, P. Favaro, and S. Soatto. Dynamic texture segmentation. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 2:1236–1242, 2003.

[11] H. H. Lin et al. Learning a scene background model via classfication. *IEEE TSP*, 57(5):1641–1654, 2009.

[12] J. Pilet et al. Making background subtraction robust to sudden illumination changes. *ECCV*, 4:567–580, 2008.

[13] J. Weng, Y. Zhang, and W. Hwang. Candid covariance-free incremental principal component analysis. *IEEE TPAMI*, 25(8):1034–1040, 2003.

[14] J. Yao, and J. Odobez. Multi-layer background subtraction based on color and texture. *CVPR*, pages 1–8, 2007.

[15] K. A. Patwardhan et al. Robust foreground detection in video using pixel layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):746–751, 2008.

[16] L. Cheng et al. Realtime background subtraction from dynamic scenes. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2066–2073, 2009.

[17] L. Wixson. Detecting salient motion by accumulating directionally-consistent flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):774–780, 2000.

[18] L. Wixson. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 1:44–50, 2003.

[19] M. Artač, M. Jogan, and A. Leonardis. Incremental pca for on-line visual learning and recognition. *in Proc. ICPR*, pages 781–784, 2002.

[20] M. Heikkilä, and M. Pietikäinen. A texture-based method for modeling the background and detecting moving objects. *IEEE TPAMI*, 28(4):657–662, 2006.

[21] P. Saisan, G. Doretto, Y.N. Wu, and S. Soatto. Dynamic texture recognition. *CVPR*, 2:58–63, 2001.

[22] P.V. Overschee, and B.D. Moor. N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30:75–93, 1994.

[23] S. Liao et al. Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes. *CVPR*, pages 1301–1306, 2010.

[24] S. Soatto, G. Doretto, and Y. Wu. Dynamic textures. *IJCV*, 52(2):91–109, 2003.

[25] V. Mahadevan, and N. Vasconcelos. Spatiotemporal saliency in dynamic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):171–177, 2010.

[26] Y. Li. On incremental and robust subspace learning. *Pattern Recognition*, 37(7):1509–1518, 2004.

[27] Y. Sheikh and M. Shah. Bayesian modeling of dynamic scenes for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1778–1792, 2005.

[28] Y. Wang, K. F. Loe, and J. K. Wu. A dynamic conditional random field model for foreground and shadow segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(2):279–289, 2007.

[29] Y. Zhao, H. Gong, L. Lin, and Y. Jia. Spatio-temporal patches for night background modeling by subspace learning. *ICPR*, pages 1–4, 2008.

[30] Y. Zhao, H. Gong, Y. Jia, and S.C. Zhu. Background modeling by subspace learning on spatio-temporal patches. *Pattern Recognition Letters*, 33:1134–1147, 2012.

[31] Z. Zivkovic. Improved adaptive gaussian mixture models for background subtraction. *ICPR*, 2004.