Machine Vision and Applications
DOI 10.1007/s00138-010-0262-3

SPECIAL ISSUE PAPER

Generalized Stauffer–Grimson background subtraction for dynamic scenes

Antoni B. Chan · Vijay Mahadevan · Nuno Vasconcelos

Received: 9 August 2008 / Revised: 30 November 2009 / Accepted: 10 March 2010

Abstract We propose an adaptive model for backgrounds containing significant stochastic motion (e.g. water). The new model is based on a generalization of the Stauffer–Grimson background model, where each mixture component is modeled as a dynamic texture. We derive an online K-means algorithm for updating the parameters using a set of sufficient statistics of the model. Finally, we report on experimental results, which show that the proposed background model both quantitatively and qualitatively outperforms state-of-the-art methods in scenes containing significant background motions.

Keywords Dynamic textures · Background models · Background subtraction · Mixture models · Adaptive models

1 Introduction

Background subtraction is an important first step for many vision problems. It separates objects from background clutter, usually by comparing motion patterns, and facilitates subsequent higher-level operations, such as tracking, object identification, etc. Because the environment can change substantially, both in the short term and throughout the lifetime of the vision system, background subtraction algorithms are expected to be robust. This is not always easy to guarantee, and many methods have been proposed in the literature (see [1] for a recent review). One approach, which was first introduced by Stauffer and Grimson (SG) in [2] and has since gained substantial popularity, is to model the distribution of colors (over time) of each pixel as a mixture of Gaussians. This accounts for the fact that, in scenes with multiple objects, pixel colors change as objects traverse the scene. For example, if an object stops, it should at some point be considered part of the background. As it departs, the un-occluded area should be quickly reassigned to the background. Some objects may even exhibit cyclic motion, e.g. a flickering light display, making a number of background pixels undergo cyclic variations of color over time. The mixture model captures these state transitions very naturally, while providing a compact summary of the color distribution. This simplifies the management of the background model, namely the problems of updating the distribution over time, or deciding which components of the mixture model should be dropped as the background changes (due to variable scene lighting, atmospheric conditions, etc.).

Nevertheless, the algorithm proposed by SG is not without problems. One of its main drawbacks is the assumption that the background is static over short time scales. This is a strong limitation for scenes with spatiotemporal dynamics, such as those of Fig. 1. Although the model allows each pixel to switch state, and tolerates some variability within the state, the Gaussian mixture assumes that the variability derives from noise, not the structured motion patterns that characterize moving water, burning fire, swaying trees, etc. One approach that has shown promise for modeling these spatiotemporal dynamic processes is the *dynamic texture* representation of [3]. Dynamic textures model a spatiotemporal volume as a sample from a *linear dynamical system* (LDS), and have shown surprising robustness for video synthesis [3], classification [4,5], segmentation [6,7], and image registration [8]. Not surprisingly, background models based on dynamic textures, or close relatives, have appeared in the literature. These utilize dynamic textures to model the whole

A. B. Chan (✉) · V. Mahadevan · N. Vasconcelos
Department of Electrical and Computer Engineering,
University of California, San Diego, 9500 Gilman Drive,
Mail code 0409, La Jolla, CA 92093-0409, USA
e-mail: abchan@cityu.edu.hk

Published online: 08 April 2010

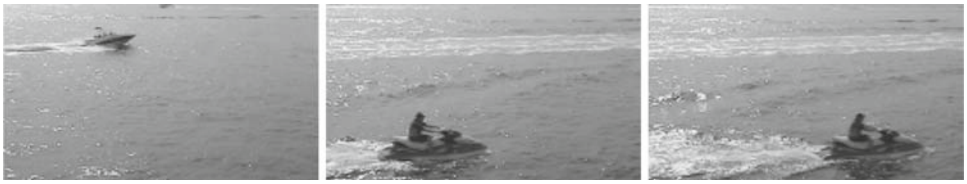


Fig. 1 A scene with dynamic background. The background consists of water waves, which are changed by the turbulent wake of a boat

video [9], or video patches extracted from the neighborhood of each location [10].

While able to capture background dynamics, these approaches lack the two most compelling (and dynamic) aspects of the SG method: (1) the ability to account for transitory events, due to motion of foreground objects; and (2) simple model management. Consider, for example, the aquatic scene of Fig. 1. As the jet-skier traverses a video patch, the video goes through the following state sequence: normal waves, occluded by jet-ski, turbulent waves that trail the jet-ski, return to normal waves. In the absence of a hidden discrete state variable, the dynamic texture will slowly interpolate through all these states. Both the transition from occluded to turbulent, and turbulent to normal waves, will generate outliers which are incorrectly marked as foreground. If the jet-ski cyclically passes through the same location, these errors will repeat with the same periodicity.

In summary, background subtraction requires both a *state-based* representation (as in SG) and the ability to *capture scene dynamics* within the state (as in the dynamic texture methods). This suggests a very natural extension of the two lines of work: to represent spatiotemporal video cubes as samples from a *mixture of dynamic textures* [11]. However, as in the static case, exact learning of the mixture parameters is computationally infeasible in an online setting. To address this problem, we combine two observations. The first is the main insight of SG: that parameter updates of a Gaussian mixture model only require a small set of *sufficient statistics*. The second is that, because the dynamic texture is a member of the exponential family [12], it exhibits the same property. We then *derive the sufficient statistics* required for *online* learning of the dynamic texture, and use them to *generalize the SG algorithm* to dynamic scenes. The generalized SG (GSG) algorithm inherits the advantages of [2]: (1) it adapts to long-term variations via online estimation; (2) it can quickly embrace new background motions through the addition of mixture components; and (3) it easily discards outdated information by dropping mixture components with small priors. The online, recursive, procedure for the estimation of dynamic texture parameters is of potential interest for any application of dynamic textures that requires online video processing. Experimental results show that background modeling with the mixture of dynamic textures substantially outperforms both static background models, and those based on a single dynamic texture.

The remainder of the paper is organized as follows: Sect. 2 briefly reviews related work in background subtraction. In Sect. 3, we review the SG method and introduce the proposed generalization. A novel online algorithm for the least squares estimation of dynamic texture parameters, based on recursive sufficient statistics, is derived in Sect. 4. In Sect. 5, we introduce the adaptive background subtraction algorithm based on mixtures of dynamic textures. Finally, an experimental evaluation is described in Sect. 6, and conclusions are drawn in Sect. 7.

2 Previous work

A number of extensions to the background subtraction method of SG have appeared in the literature. One of the main drawbacks of the original approach was the difficulty of detecting foreground objects of color similar to that of the background. This motivated a number of extensions based on properties of local image neighborhoods, e.g. texture information [13], optical flow [14,15], or spatiotemporal blocks [16,17]. Another drawback of [2] is the lack of consistency between the state of adjacent pixels, which sometimes leads to noisy foreground-background segmentations. This motivated various extensions that encourage global consistency, either through global image modeling (e.g. eigen-background methods [18,19]), by introducing priors that enforce global consistency (e.g. the combination of Markov random fields and maximum a posteriori probability decision rules [1]), or by allowing pixels to influence neighboring GMMs [20]. While most of these models can adapt to slow background changes (e.g. due to illumination) they share, with the mixture model of SG, the assumption that the background is *static* over short time scales.

Other methods of background subtraction are based on separating "salient" (foreground) motion from the background motion. Wixson [21] defines saliency as the distance that a pixel travels in a consistent direction, measured by accumulating the optical flow over several frames. When a pixel reverses direction, the saliency is reset to zero, and hence short oscillating motion (e.g. swaying tree branches) will have low saliency compared to foreground objects that move in a consistent direction. Tian and Hampapur [22] defines saliency in a similar way, but implements it with a combination of difference images, optical flow, and temporal filters. The disadvantage with these methods is that salient motion is defined as motion with a consistent direction. Thus, foreground objects that change direction will be marked as background, and backgrounds that move in a consistent direction (e.g. water waves) will be marked as foreground. Furthermore, it may not be possible to reliably compute optical flow on moving non-rigid backgrounds that do not obey the smoothness or brightness constancy constraints (e.g. moving water).

In the realm of *dynamic models*, two main methods have been proposed. In [9], the video is modeled with a robust Kalman filter. A dynamic texture models the entire video frame, and pixels that are not well explained by the LDS (i.e. outliers), conditioned on the previous frame, are marked as foreground. This method has limited effectiveness, because a global fit usually demands excessive representational power from the dynamic texture: background dynamics are seldom homogeneous throughout the scene, due to a combination of 3D effects (perspective and motion parallax) and background variability (e.g. trees swaying with different dynamics, or waves with normal and turbulent motions). A natural improvement is to rely on a localized representation. This was first proposed in [10], where the dynamic texture is applied to video patches, reducing the dimensionality of the model and producing a simpler learning problem. This work also exploits the principal component analysis (PCA) performed within the LDS to achieve computational efficiency. In particular, all model updates are performed with an incremental PCA algorithm. A spatial patch in the current frame is marked as foreground if either: (1) it is not well modeled by the PCA basis (i.e. large residual pixel error); or (2) the PCA coefficients extracted from the observed frame are different than those predicted from the previous frame (i.e. poor single-step prediction of the state variable).

The procedure now proposed makes three main contributions with respect to the state-of-the-art in dynamic background modeling [9,10]. The first is the derivation of a novel *online algorithm* for estimating the parameters of a dynamic texture, using sufficient statistics. This is an *exact* algorithm, which produces least-squares parameter estimates identical to those of [3]. Although incremental PCA-type of approximations are possible (and derived in the Appendix), they are not optimal or advisable when this exact solution is computationally feasible. The sufficient statistics are computed in *recursive* form and the procedure is suitable for *any* application of dynamic textures that requires online video processing, not just background subtraction. The second is a background subtraction algorithm with the ability to classify the background according to the statistics of the whole spatiotemporal cube, instead of single-step predictions. This leads to foreground-background assignments that account for motion over multiple frames, and is more robust. Finally, the generalized Stauffer–Grimson framework now proposed fully exploits the probabilistic modeling subjacent to the dynamic texture. This is unlike [9], which classifies pixels individually, or [10] which ignores the stochastic nature of the state variable. In result, the new algorithm is shown to substantially outperform the previous approaches.

3 Adaptive background modeling with online mixtures

In this section, we briefly review the static background subtraction method of SG [2]. The review emphasizes the insight that the background model can be updated using sufficient statistics. This suggests a natural generalization to dynamic scenes, which is introduced at the end of the section.

3.1 Probabilistic model

The method of SG models each background pixel with an adaptive mixture of Gaussians. The probability of observing pixel y is:

$$p(y) = \sum_{j=1}^{K} \omega_j G(y, \mu_j, \Sigma_j) \tag{1}$$

where K is the number of mixture components, ω_j the weight of each component, and $G(y, \mu, \Sigma)$ a multivariate Gaussian distribution of mean μ and covariance Σ (typically $\Sigma = \sigma I$). Given a video frame, this GMM is updated through an online approximation to the K-means algorithm.

3.2 Online parameter estimation

A newly observed pixel value y' is classified in a two-step procedure. First, the Mahalanobis distance to each Gaussian component

$$d_j = \|y' - \mu_j\|_{\Sigma_j} = \sqrt{(y' - \mu_j)^T \Sigma_j^{-1} (y' - \mu_j)} \tag{2}$$

is computed. The closest component

$$k = \operatorname*{argmin}_j d_j \tag{3}$$

is then identified. The pixel is considered to match this component if the distance is within a threshold θ, i.e. $d_k \leq \theta$ ([2] sets $\theta = 2.5$).

An online update of the model is performed after each pixel classification. If no match can be found, the model component of lowest weight ω_j is replaced with a new Gaussian of mean y', an initially high variance, and a low initial weight. If y' matches the kth component, then its parameters are updated according to

$$\mu_k \leftarrow (1 - \alpha)\mu_k + \alpha y', \tag{4}$$

$$\Sigma_k \leftarrow (1 - \alpha)\Sigma_k + \alpha(y' - \mu_k)(y' - \mu_k)^T, \tag{5}$$

and the component weights adjusted with

$$w_j \leftarrow (1-\beta)w_j + \beta\mathbb{I}(j=k),\ \forall j \tag{6}$$

where $\mathbb{I}(\cdot)$ is the indicator function, and re-normalized to sum to one.

It can be shown that the successive application of these online updates is equivalent to batch K-means estimation with exponentially decaying weights on the observations. This allows the mixture model to adjust to gradual non-stationary background changes (e.g. due to illumination). The adaptivity of the model is determined by the learning rates α and β: α controls how quickly each component adjusts to background changes, and β controls the speed at which components become relevant.[1] One appealing property of this online procedure is that the existing components of the background model do not have to be destroyed as new pixel processes are added. This is useful when pixels revert to a previously-observed process. Because the associated mixture component still exists (although with a small weight), it can be quickly re-incorporated into the background model.

3.3 Background detection

Because the online learning algorithm incorporates all pixel observations, the mixture model does not distinguish components that correspond to background from those associated with foreground objects, or simply due to noise. Due to this, background detection requires a list of "active" background components. In the method of SG [2], these are heuristically chosen as the Gaussians of most supporting evidence (high prior weight) and low variance. This is done by sorting components by decreasing ratio w_j/σ_j, and selecting the first B that explain T percent of the probability $\left(\sum_{j=1}^{B} w_j > T\right)$. Finally, the observed pixel is marked as background if its likelihood under these "active" background components is above a threshold, and foreground otherwise.

3.4 Extending Stauffer–Grimson to other distributions

The choice of Gaussian mixture components is suitable when background pixels are static or change slowly over time. While this assumption sometimes holds (e.g. a fixed camera overlooking a roadway), there are numerous scenes where it does not (e.g. outdoors scenes involving water or vegetation). Such scenes require extensions of the SG model, using mixtures of component distributions that are appropriate for the particular background process. The main insight of the procedure proposed by SG is that, for Gaussian components, all parameter estimates can be obtained efficiently, through online updates of a small set of *sufficient statistics*.

The extension of the procedure to more complex distributions $p(y|\Theta)$ requires the satisfaction of two conditions: (1) that the parameters Θ of these distributions can be estimated through a set of sufficient statistics ζ, and (2) that these statistics can be computed through efficient online updates, such as those of (4). The first condition is well known to be satisfied by any model in the exponential family [12]. Since the dynamic texture is a Gaussian model, and therefore in this family, it immediately satisfies this condition. The determination of the statistics is, however, not trivial. It is also not trivial that they can be computed in an efficient recursive form, so as to satisfy the second condition. We address these issues in the remainder of the paper.

4 Sufficient statistics for estimation of dynamic texture parameters

We start by deriving the set of sufficient statistics for the estimation of parameters of the dynamic texture model. For this, we briefly review the dynamic texture and the least-squares parameter estimation algorithm of [3]. We then derive a set of recursive sufficient statistics for the efficient implementation of this algorithm.

4.1 Dynamic textures

The dynamic texture [3] is a generative model for video, which it models as a sample from a linear dynamical system. The model separates appearance and dynamics (motion) into two stochastic processes. The dynamics are represented as a time-evolving state process $x_t \in \mathbb{R}^n$, while the appearance of frame $y_t \in \mathbb{R}^m$ is modeled as a linear function of the current state vector and observation noise. Formally, the system equations are:

$$\begin{cases} x_t = Ax_{t-1} + v_t \\ y_t = Cx_t + w_t + \bar{y} \end{cases} \tag{7}$$

where $A \in \mathbb{R}^{n\times n}$ is the state transition matrix, $C \in \mathbb{R}^{m\times n}$ the observation matrix, and $\bar{y} \in \mathbb{R}^m$ the video mean. The state and observation noise are modeled as Gaussian processes $v_t \sim \mathcal{N}(0, Q,)$ and $w_t \sim \mathcal{N}(0, R)$, respectively. Finally, the initial condition is distributed as $x_1 \sim \mathcal{N}(\mu, S)$.

The dynamic texture parameters are frequently learned with a least-squares algorithm proposed in [3]. Given an observed video sequence $Y_{1:\tau} = [y_1 \dots y_\tau]$, the mean is first estimated by the sample mean

$$\bar{y} = \frac{1}{\tau}\sum_{t=1}^{\tau} y_t \tag{8}$$

and the mean-subtracted video sequence

$$\tilde{Y}_{1:\tau} = [\tilde{y}_1 \dots \tilde{y}_\tau], \tag{9}$$

[1] Stauffer and Grimson [2] adaptively sets $\alpha = \beta G(y', \mu_k, \Sigma_k)$.

where $\tilde{y}_t = y_t - \bar{y},\ \forall t$, is used in all subsequent computations.

To estimate the model parameters, the video sequence is subject to a PCA, performed with recourse to the singular value decomposition (SVD)

$$\tilde{Y}_{1:\tau} = USV^T. \tag{10}$$

The observation matrix is estimated from the n principal components of the largest eigenvalues:

$$\hat{C} = [u_1 \dots u_n] \tag{11}$$

where u_i is the ith column of U, and it was assumed that the diagonal entries of S are ordered by decreasing value. The state-space variables are then estimated with:

$$\hat{X}_{1:\tau} = \left[\hat{x}_1 \dots \hat{x}_\tau\right] = \hat{C}^T \tilde{Y}_{1:\tau} \tag{12}$$

leading to the least square estimate of the state-transition matrix:

$$\hat{A} = \hat{X}_{2:\tau}(\hat{X}_{1:\tau-1})^\dagger, \tag{13}$$

where $X^\dagger = X^T(XX^T)^{-1}$ is the Moore–Penrose pseudo-inverse of X. The state noise is estimated from the state-prediction residual error:

$$\hat{V}_{1:\tau-1} = \hat{X}_{2:\tau} - \hat{A}\hat{X}_{1:\tau-1}, \tag{14}$$

$$\hat{Q} = \frac{1}{\tau-1}\hat{V}_{1:\tau-1}\left(\hat{V}_{1:\tau-1}\right)^T, \tag{15}$$

and the initial state is assumed constant

$$\hat{\mu} = \hat{x}_1, \tag{16}$$

$$\hat{S} = 0. \tag{17}$$

Finally, the observation noise is estimated from the reconstruction error

$$\hat{W}_{1:\tau} = \tilde{Y}_{1:\tau} - \hat{C}\hat{X}_{1:\tau}, \tag{18}$$

$$\hat{R} = \frac{1}{\tau}\hat{W}_{1:\tau}\hat{W}_{1:\tau}^T. \tag{19}$$

The covariance of the observation noise is usually assumed diagonal, $R = rI_m$. In this case, the variance estimate $\hat{r}$ is the mean of the diagonal elements of the full covariance estimate $\hat{R}$, i.e. $\hat{r} = \frac{1}{m}\sum_{i=1}^m [\hat{R}]_{i,i}$.

Although suboptimal in the maximum-likelihood sense, this procedure has been shown to lead to good estimates of dynamic texture parameters in various applications, including video synthesis [3] and recognition [4,5].

4.2 Estimation from sufficient statistics

The estimates of the state-space variables $\hat{X}_{1:\tau}$ play a central role in the least-squares algorithm of the previous section. However, because (1) they depend on the principal component basis (through the matrix $\hat{C}$), and (2) this basis changes at each time step, they are not suitable sufficient statistics for online parameter estimation. An online version of the least-squares procedure requires an alternative set of sufficient statistics, which can be updated at each time-step and do not depend on $\hat{C}$. We first look at computing the PCA basis. It suffices to note that the SVD computation of (10) is equivalent to computing the n principal components of $\hat{\Phi} = \tilde{Y}_{1:\tau}\tilde{Y}_{1:\tau}^T$. Hence $\hat{C}$ can be estimated as:

$$\hat{C} = \mathrm{PCA}\big(\hat{\Phi}, n\big), \tag{20}$$

where $\mathrm{PCA}(\Sigma, n)$ returns the top n principal components of Σ. Hence, $\hat{\Phi}$ is a sufficient statistic for computing the PCA basis, given data $Y_{1:\tau}$. Note that $\hat{\Phi}$ can be updated recursively, according to

$$\hat{\Phi}^{(\tau+1)} = \tilde{Y}_{1:\tau+1}\tilde{Y}_{1:\tau+1}^T \tag{21}$$

$$= \sum_{t=1}^{\tau+1}\tilde{y}_t\tilde{y}_t^T = \sum_{t=1}^{\tau}\tilde{y}_t\tilde{y}_t^T + \tilde{y}_{\tau+1}\tilde{y}_{\tau+1}^T \tag{22}$$

$$= \hat{\Phi}^{(\tau)} + \tilde{y}_{\tau+1}\tilde{y}_{\tau+1}^T, \tag{23}$$

where the superscript indicates the time-step to which the statistic refers. We next substitute (12) into (13) to rewrite the estimate of the state-transition matrix as:

$$\hat{A} = \left(\hat{C}^T\tilde{Y}_{2:\tau}\right)\left(\hat{C}^T\tilde{Y}_{1:\tau-1}\right)^\dagger \tag{24}$$

$$= \left(\hat{C}^T\tilde{Y}_{2:\tau}\tilde{Y}_{1:\tau-1}^T\hat{C}\right)\left(\hat{C}^T\tilde{Y}_{1:\tau-1}\tilde{Y}_{1:\tau-1}^T\hat{C}\right)^{-1} \tag{25}$$

$$= \left(\hat{C}^T\hat{\psi}\hat{C}\right)\left(\hat{C}^T\hat{\phi}\hat{C}\right)^{-1}, \tag{26}$$

where we define the sufficient statistics $\hat{\phi} = \tilde{Y}_{1:\tau-1}\tilde{Y}_{1:\tau-1}^T$ and $\hat{\psi} = \tilde{Y}_{2:\tau}\tilde{Y}_{1:\tau-1}^T$. Note that this is an estimate of A at time τ, which only depends on the estimate $\hat{C}$ of the PCA basis at this time step, and the sufficient statistics $\hat{\phi}$ and $\hat{\psi}$. The latter, again, can be updated recursively, e.g.

$$\hat{\psi}^{(\tau+1)} = \tilde{Y}_{2:\tau+1}\tilde{Y}_{1:\tau}^T = \tilde{Y}_{2:\tau}\tilde{Y}_{1:\tau-1}^T + \tilde{y}_{\tau+1}\tilde{y}_\tau^T \tag{27}$$

$$= \hat{\psi}^{(\tau)} + \tilde{y}_{\tau+1}\tilde{y}_\tau^T. \tag{28}$$

To derive an online estimate of the covariance Q, we note that:

$$\hat{Q} = \frac{1}{\tau-1}\left(\hat{X}_{2:\tau} - \hat{A}\hat{X}_{1:\tau-1}\right)\left(\hat{X}_{2:\tau} - \hat{A}\hat{X}_{1:\tau-1}\right)^T \tag{29}$$

$$= \frac{1}{\tau-1}\Big(\hat{X}_{2:\tau}\hat{X}_{2:\tau}^T - \hat{A}\hat{X}_{1:\tau-1}\hat{X}_{2:\tau}^T - \hat{X}_{2:\tau}\hat{X}_{1:\tau-1}^T\hat{A}^T + \hat{A}\hat{X}_{1:\tau-1}\hat{X}_{1:\tau-1}^T\hat{A}^T\Big) \tag{30}$$

$$= \frac{1}{\tau-1}\left(\hat{X}_{2:\tau}\hat{X}_{2:\tau}^T - \hat{A}\hat{X}_{1:\tau-1}\hat{X}_{2:\tau}^T\right) \tag{31}$$

$$= \frac{1}{\tau-1}\left(\hat{C}^T\tilde{Y}_{2:\tau}\tilde{Y}_{2:\tau}^T\hat{C} - \hat{A}\hat{C}^T\tilde{Y}_{1:\tau-1}\tilde{Y}_{2:\tau}^T\hat{C}\right) \tag{32}$$

$$= \frac{1}{\tau-1}\left(\hat{C}^T\hat{\varphi}\hat{C} - \hat{A}\hat{C}^T\hat{\psi}^T\hat{C}\right), \tag{33}$$

where we have used, in (30),

$$\hat{A}\hat{X}_{1:\tau-1}\hat{X}_{1:\tau-1}^T\hat{A}^T = \left(\hat{X}_{2:\tau}\hat{X}_{1:\tau-1}^{\dagger}\right)\hat{X}_{1:\tau-1}\hat{X}_{1:\tau-1}^T\hat{A}^T$$
$$= \hat{X}_{2:\tau}\hat{X}_{1:\tau-1}^T\hat{A}^T,$$

and we define the statistic $\hat{\varphi} = \tilde{Y}_{2:\tau}\tilde{Y}_{2:\tau}^T$. The mean and variance of the initial state are computed via

$$\hat{\mu} = \hat{x}_1 = \hat{C}^T\tilde{y}_1 = \hat{C}^T\hat{\xi}, \tag{34}$$
$$\hat{S} = \left(\hat{x}_1 - \hat{\mu}\right)\left(\hat{x}_1 - \hat{\mu}\right)^T = \hat{C}^T\tilde{y}_1\tilde{y}_1^T\hat{C} - \hat{\mu}\hat{\mu}^T \tag{35}$$
$$= \left(\hat{C}^T\hat{\eta}\hat{C}\right) - \hat{\mu}\hat{\mu}^T, \tag{36}$$

where we define $\hat{\eta} = \tilde{y}_1\tilde{y}_1^T$ and $\hat{\xi} = \tilde{y}_1$. Finally, the observation noise is estimated from the reconstruction error $\tilde{Y}_{1:\tau} - \hat{C}\hat{X}_{1:\tau}$,

$$\hat{R} = \frac{1}{\tau}\left(\tilde{Y}_{1:\tau} - \hat{C}\hat{X}_{1:\tau}\right)\left(\tilde{Y}_{1:\tau} - \hat{C}\hat{X}_{1:\tau}\right)^T \tag{37}$$
$$= \frac{1}{\tau}\left(\tilde{Y}_{1:\tau} - \hat{C}\hat{C}^T\tilde{Y}_{1:\tau}\right)\left(\tilde{Y}_{1:\tau} - \hat{C}\hat{C}^T\tilde{Y}_{1:\tau}\right)^T \tag{38}$$
$$= \frac{1}{\tau}\left(I - \hat{C}\hat{C}^T\right)\tilde{Y}_{1:\tau}\tilde{Y}_{1:\tau}^T\left(I - \hat{C}\hat{C}^T\right) \tag{39}$$
$$= \left(I - \hat{C}\hat{C}^T\right)\hat{\Phi}\left(I - \hat{C}\hat{C}^T\right). \tag{40}$$

In summary, the parameters of the dynamic texture can be computed as follows. The sufficient statistics are defined as:

$$\begin{aligned} &\hat{\Phi} = \tilde{Y}_{1:\tau}\tilde{Y}_{1:\tau}^T, \hat{\varphi} = \tilde{Y}_{2:\tau}\tilde{Y}_{2:\tau}^T, \quad \hat{\phi} = \tilde{Y}_{1:\tau-1}\tilde{Y}_{1:\tau-1}^T, \\ &\hat{\psi} = \tilde{Y}_{2:\tau}\tilde{Y}_{1:\tau-1}^T, \quad \hat{\eta} = \tilde{y}_1\tilde{y}_1^T, \hat{\xi} = \tilde{y}_1. \end{aligned} \tag{41}$$

These statistics can also be computed recursively, where at each time-step, we have the update equations:

$$\begin{aligned} \hat{\Phi}^{(\tau+1)} &= \hat{\Phi}^{(\tau)} + \tilde{y}_{\tau+1}\tilde{y}_{\tau+1}^T \\ \hat{\varphi}^{(\tau+1)} &= \hat{\Phi}^{(\tau+1)} - \hat{\eta} \\ \hat{\phi}^{(\tau+1)} &= \hat{\Phi}^{(\tau)} \\ \hat{\Psi}^{(\tau+1)} &= \hat{\Psi}^{(\tau)} + \tilde{y}_{\tau+1}\tilde{y}_{\tau}^T, \end{aligned} \tag{42}$$

where $\hat{\eta}$ is defined in (41), and $\hat{\Phi}^{(0)} = \hat{\Psi}^{(0)} = 0$. The PCA basis $\hat{C}$ is then computed via PCA on $\hat{\Phi}$, and the remaining parameters estimated as:

$$\begin{aligned} &\hat{A} = \left(\hat{C}^T\hat{\psi}\hat{C}\right)\left(\hat{C}^T\hat{\phi}\hat{C}\right)^{-1}, \quad \hat{\mu} = \hat{C}^T\hat{\xi}, \\ &\hat{Q} = \frac{1}{\tau-1}\left(\hat{C}^T\hat{\varphi}\hat{C} - \hat{A}\hat{C}^T\hat{\psi}^T\hat{C}\right), \quad \hat{S} = \hat{C}^T\hat{\eta}\hat{C} - \hat{\mu}\hat{\mu}^T, \\ &\hat{R} = \left(I - \hat{C}\hat{C}^T\right)\hat{\Phi}\left(I - \hat{C}\hat{C}^T\right), \end{aligned} \tag{43}$$

where $\hat{\xi}$ is as defined in (41). Note that the online estimates obtained with these recursive sufficient statistics are identical to the solution produced by the least-squares algorithm of Sect. 4.1.

Rewriting the least-squares solution in terms of sufficient statistics also exposes an interesting connection between the algorithm of [3] and the EM algorithm for dynamic textures [23,24]. In particular, the estimates of (43) utilize matrices of the form $\hat{C}^T\hat{\phi}\hat{C}$, which is equivalent to the projection of the image statistic ϕ into the state-space of the LDS. These projections can be viewed as approximations to the conditional expectations computed in the E-step of the EM algorithm:

$$\begin{aligned} \hat{C}^T\hat{\phi}\hat{C} &\approx \sum_{t=2}^{\tau}\mathbb{E}\left(x_{t-1}x_{t-1}^T|y\right), \\ \hat{C}^T\hat{\varphi}\hat{C} &\approx \sum_{t=2}^{\tau}\mathbb{E}\left(x_t x_t^T|y\right), \\ \hat{C}^T\hat{\psi}\hat{C} &\approx \sum_{t=2}^{\tau}\mathbb{E}\left(x_t x_{t-1}^T|y\right). \end{aligned} \tag{44}$$

Under this interpretation, the estimates $\hat{A}$, $\hat{Q}$, $\hat{S}$, and $\hat{\mu}$ in (43) are equivalent to those in the M-step of the EM algorithm. Hence, the least-squares solution of [3] can be viewed as a single iteration of the EM algorithm, where $\hat{C}$ is approximated as the PCA basis of the observations, and the E-step is approximated by projecting the image statistics into the state-space. In practice, the least-squares estimate serves as a good initialization for the EM algorithm, which typically converges after a few iterations.

At each time step, updating the sufficient statistics and the PCA basis has complexity $O(m^2)$. Assuming $n \ll m$, estimating the DT parameters from the sufficient statistics has complexity $O(nm^2)$. Hence, the complexity of one iteration of the online update is $O(nm^2)$ for a single frame. When the sufficient statistics are updated with T frames in each step, the complexity is $O((T+n)m^2)$.

Finally, the proposed procedure can also be used to estimate the parameters of a dynamic texture from multiple video samples. In particular, given a set of N mean-subtracted videos $\left\{\tilde{Y}_{1:\tau}^{(1)}, \ldots, \tilde{Y}_{1:\tau}^{(N)}\right\}$, the sufficient statistics are computed by averaging over all samples, e.g.

$$\hat{\Phi} = \frac{1}{N}\sum_{i=1}^{N}\tilde{Y}_{1:\tau}^{(i)}\left(\tilde{Y}_{1:\tau}^{(i)}\right)^T \tag{45}$$

and similarly for the other statistics. The parameters are then estimated with (43).

5 The generalized Stauffer–Grimson algorithm for dynamic textures

In this section, we introduce an adaptive background model based on the mixture of dynamic textures [11]. This model is used to extend the background subtraction algorithm of [2] to dynamic scenes. An overview of the proposed algorithm is shown in Fig. 2. Each video location is represented by a spatiotemporal neighborhood, centered at that location

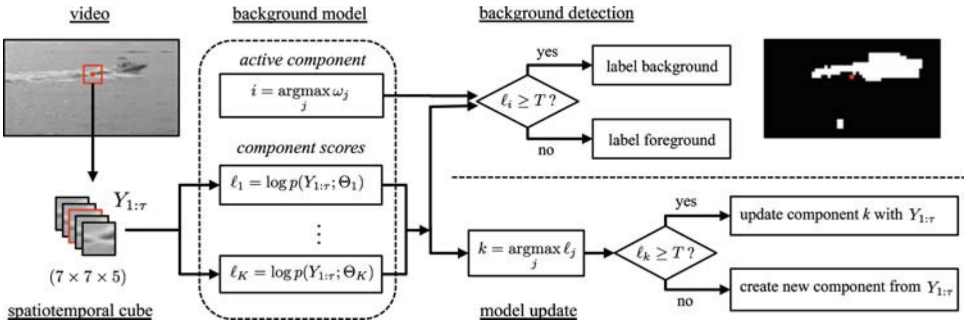

video
background model
background detection
active component
$i = \operatorname{argmax}_j \omega_j$
component scores
$\ell_1 = \log p(Y_{1:\tau}; \Theta_1)$
$\ell_K = \log p(Y_{1:\tau}; \Theta_K)$
$Y_{1:\tau}$
$(7 \times 7 \times 5)$
spatiotemporal cube
$\ell_i \geq T$?
yes
no
label background
label foreground
$k = \operatorname{argmax}_j \ell_j$
$\ell_k \geq T$?
yes
no
update component k with $Y_{1:\tau}$
create new component from $Y_{1:\tau}$
model update

Fig. 2 Overview of generalized Stauffer–Grimson background modeling for dynamic textures. A video location is represented by a neighboring spatiotemporal volume $Y_{1:\tau}$. The location is marked as background if the log-likelihood of $Y_{1:\tau}$ under the active background component is above a threshold. Next, the mixture component with the largest log-likelihood of generating $Y_{1:\tau}$ is updated, if the log-likelihood is above threshold. Otherwise, a new component is learned, replacing the component of lowest prior probability

(in this work we use a $7 \times 7 \times 5$ volume). The background scene is modeled as a mixture of K dynamic textures, from which spatiotemporal volumes are drawn. The jth dynamic texture is denoted (both its parameters and image statistics) by Θ_j, and a prior weight ω_j, s.t. $\sum_{j=1}^{K} \omega_j = 1$, is associated with each dynamic texture.

Given a spatiotemporal observation $Y_{1:\tau} \in \mathbb{R}^{m\times\tau}$ ($m = 49$ and $\tau = 5$ in all experiments reported), the location is marked as background if the log-likelihood of the observation under an "active" background component is above a threshold. The background model is then updated, using an online approximation to EM. As in SG, this consists of updating the mixture component with the largest log-likelihood of generating the observation $Y_{1:\tau}$, if the log-likelihood is above a second threshold. If not, a new dynamic texture component learned from $Y_{1:\tau}$ replaces the mixture component with lowest prior weight. In the remainder of the section, we discuss the two major components of the algorithm, background detection and online model updates.

5.1 Background detection

The determination of whether a location belongs to the background requires the assignment of mixture components to background and foreground. We have already seen that the procedure proposed by SG [2] accomplishes this by heuristically ranking the mixture components. Support for multiple background components is crucial for SG because background colors can switch quickly (e.g. a flashing light). Under the dynamic texture mixture model, *rapid changes in color and texture are modeled by the dynamic texture components themselves*. It follows that multiple active background components are not necessary, and we simply select the component of largest prior

$$i = \operatorname*{argmax}_j w_j \tag{46}$$

as the "active" background component. A video location is marked as belonging to the background if the log-likelihood of the corresponding spatiotemporal volume $Y_{1:\tau}$ under this mixture component is greater than a threshold T,

$$\log p(Y_{1:\tau}|\Theta_i) \geq T. \tag{47}$$

Note that the log-likelihood can be rewritten in "innovation" form

$$\log p(Y_{1:\tau}; \Theta_i) = \log p(y_1; \Theta_i) + \sum_{t=2}^{\tau} \log p(y_t|Y_{1:t-1}; \Theta_i),$$

which can be efficiently computed with recourse to the Kalman filter [11,23].

5.2 On-line updating of the background model

The background mixture model is learned with an online K-means algorithm. During training, an initial dynamic texture Θ_1 is learned with the procedure of Sect. 4.2, and mixture weights are set to $\omega = [1, 0, \ldots, 0]$. Given a new spatiotemporal observation $Y_{1:\tau}$, the mixture parameters are updated as follows. First, the mixture component with the largest log-likelihood of having generated the observation

$$k = \operatorname*{argmax}_j \log p(Y_{1:\tau}; \Theta_j) \tag{48}$$

is selected. If this log-likelihood is above the threshold T

$$\log p(Y_{1:\tau}; \Theta_k) \geq T, \tag{49}$$

the sufficient statistics of the kth component are combined with the sufficient statistics $\{\Phi', \phi', \varphi', \psi', \eta', \xi'\}$ derived from $Y_{1:\tau}$, in a manner similar to (4),

$$\begin{aligned}
\Phi &\leftarrow (1-\alpha)\Phi + \alpha\Phi', & \eta &\leftarrow (1-\alpha)\eta + \alpha\eta',\\
\phi &\leftarrow (1-\alpha)\phi + \alpha\phi', & \xi &\leftarrow (1-\alpha)\xi + \alpha\xi',\\
\psi &\leftarrow (1-\alpha)\psi + \alpha\psi', & \bar{y} &\leftarrow (1-\alpha)\bar{y} + \alpha\bar{y}',\\
\varphi &\leftarrow (1-\alpha)\varphi + \alpha\varphi'. & &
\end{aligned} \tag{50}$$

As before, α is a learning rate which weighs the contribution of the new observation. Finally, the parameters of the mixture component are re-estimated with (43), and prior weights are adjusted according to

$$w_j \leftarrow (1-\beta)w_j + \beta\mathbb{I}(j=k),\ \forall j, \tag{51}$$

(and normalized to sum to one). It can be shown that the successive application of the online estimation algorithm is equivalent to batch least-squares estimation with exponentially decaying weights on the observations. This allows the dynamic texture to adapt to slow background changes (e.g. lighting and shadows).

If (49) does not hold, the component with smallest prior ($i = \operatorname{argmin}_j w_j$) is replaced by a new dynamic texture learned from $Y_{1:\tau}$. A regularization term σI is added to the sufficient statistics $\{\Phi, \phi, \varphi, \eta\}$, to guarantee a large initial variance (in $\hat{Q}$, $\hat{S}$, and $\hat{R}$). As the component is updated with more observations, the influence of this regularization term vanishes. Finally, prior weights are adjusted according to:

$$\omega_j \leftarrow (1-\beta)\omega_j\mathbb{I}(j \neq i) + \beta\mathbb{I}(j=i),\ \forall j \tag{52}$$

and normalized to sum to one. The learning rate β adjusts the speed at which prior weights change, controlling how quickly a mixture component can become the "active" background component. The online background update algorithm is summarized in Algorithm 1.

For a single DT component, the complexity of computing the log-likelihood $p(Y_{1:\tau}; \Theta_j)$ with the Kalman filter is $O(\tau(n^3+nm))$. If the DT component is unchanged, then the Kalman filter can be cached and the subsequent complexity is $O(\tau nm)$. Re-estimating the parameters of a DT component with new video $Y_{1:\tau}$ has complexity $O((\tau+n)m^2)$, as shown in Sect. 4.2. Hence, the complexity of the background model is dominated by the update step of a single DT component.

When the dimension of the spatiotemporal volume, or the number of dynamic texture components, is large it may be impractical to compute and store the required sufficient statistics. In this case, the latter can be approximated by using a common PCA basis defined by $\hat{\Phi}$ (see Appendix for details), which is updated with an incremental PCA algorithm. One disadvantage of approximating the covariance Φ by its principal components is that some important directions of variation may not be captured (especially if the number of principal components is small). For example, a new PCA basis might not have strong enough support from a single video sample, but this support may increase considerably when aggregating over several iterations of the algorithm. In this case, the approximate algorithm will fail to include the new PCA basis, whereas the exact online algorithm will not.

Algorithm 1 Online updating of the background model

```
Input: Spatiotemporal observation $Y_{1:\tau}$, dynamic texture components $\{\Theta_j\}_{j=1}^K$, prior weights $\omega$, learning rates $\alpha$ and $\beta$, threshold $T$.
Find closest component: $k = \operatorname{argmax}_j \log p(Y_{1:\tau}; \Theta_j)$
if $\log p(Y_{1:\tau}; \Theta_k) \geq T$ then
    {Update component $k$}
    Update the sufficient statistics of $\Theta_k$ with $Y_{1:\tau}$ using (50).
    Estimate the dynamic texture parameters $\Theta_k$ with (43).
    Adjust priors (51) and normalize.
else
    {Create new component}
    Find smallest prior: $k = \operatorname{argmin}_j \omega_j$
    Compute new sufficient statistics $\Theta_k$ from $Y_{1:\tau}$.
    Estimate the parameters $\Theta_k$ with (43).
    Adjust priors (52) and normalize.
end if
```

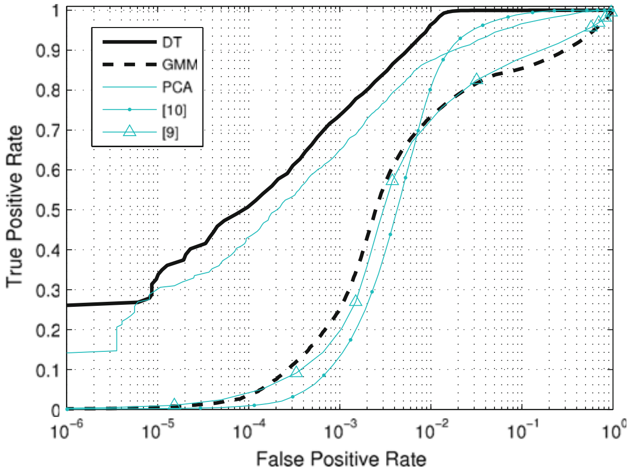


Fig. 3 ROC curves for foreground detection on Bottle video

6 Experiments

In this section, we present experiments on background subtraction using the adaptive background model based on dynamic textures. We present both quantitative and qualitative results for three aquatic video sequences, comparing results with other state-of-the-art models. A qualitative evaluation is also presented for several other challenging sequences.

6.1 Videos with water backgrounds

The first set of experiments is based on three videos of objects moving in water. Bottle, from [9], displays a bottle floating in water (see Fig. 4). The waves in the water move rapidly, pushing the bottle up the image. The sequence has dimensions 160 × 120, and the first 174 frames were used to train the background model, with the remaining 105 frames (which contain the actual bottle) being used for testing.

Boats1 and Boats2, depict boats moving in a harbor (see Figs. 6, 7). The boats create a turbulent wake that dramatically

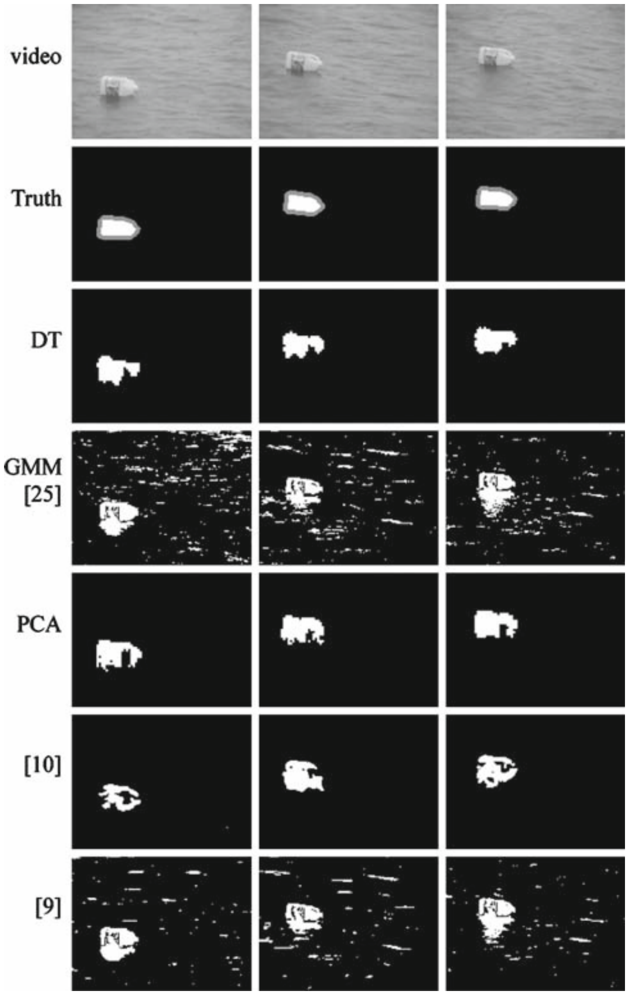


Fig. 4 Foreground detection results on Bottle sequence at 80% TPR

changes the motion and appearance of the water. The sequences also contains a large perspective effect, which makes the water motion in the foreground significantly different from that in the background. The strong motion parallax of this scene makes these sequences very challenging for background subtraction. They have dimensions 180 × 120 × 300. A separate sequence of 200 frames, containing just the baseline water motion, was used to train the background models.

6.2 Experimental setup

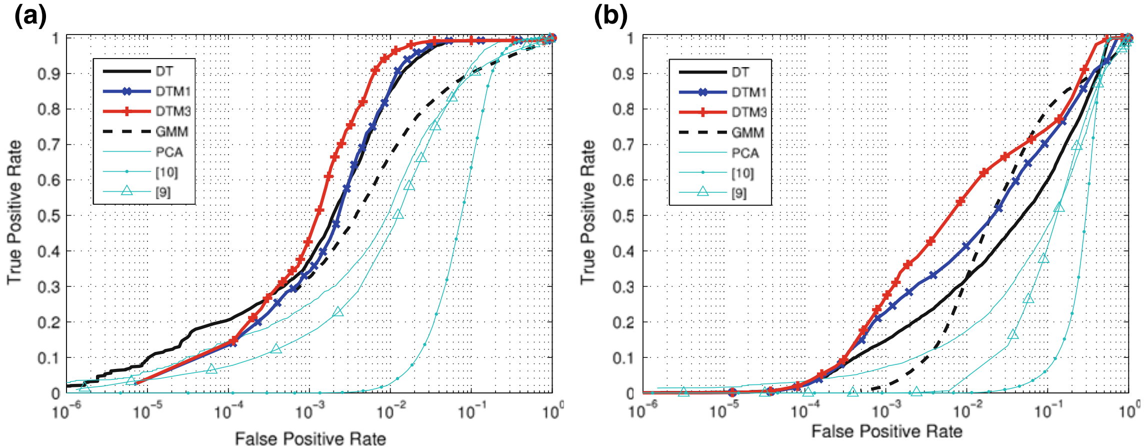
Several background models, based on dynamic textures, were compared. The first represents each spatiotemporal volume as a simple non-adaptive dynamic texture, and is denoted by DT. The second and third models use the adaptive dynamic texture mixture with $K = 1$ and $K = 3$, and are denoted by DTM1 and DTM3, respectively. These models are updated with Algorithm 1 (when $K = 1$ no new components are added in the update phase). Finally, the last two models are based on the adaptive dynamic texture mixture (with $K = 1$ and $K = 3$) using approximate sufficient statistics (with $q = n$), and are denoted by DTM1x and DTM3x. In all cases, a spatiotemporal cube size of 7 × 7 × 5 was adopted, and the state-space dimension of the dynamic textures was $n = 10$. The covariance matrix R was also assumed diagonal. For the adaptive dynamic texture mixture, the learning rate parameters were set to $\alpha = 0.16$ and $\beta = 0.08$. Finally, when adding a new mixture component, the covariance regularization parameter was $\sigma = 10$.

We compared performance against several state-of-the-art background models. The first is the GMM of [2] with $\alpha = 0.01$. Specifically, we used the extension of [25], that automatically selects the number of mixture components, and is available from [26]. We have also implemented the models from [10] (7 × 7 patches and $n = 10$) and [9] ($n = 10$). Finally, performance was compared against a simple background model, which represents 7 × 7 patches with $n = 10$ PCA components, and marks a patch as foreground if its reconstruction error is above a threshold.

All background models were evaluated by comparing foreground detections to ground-truth masks for foreground objects. Several examples of ground-truth are shown in Figs. 4, 6, and 7. The foreground is displayed as white and the background as black. The gray region is the boundary between object and background. Due to the length of the sequences, and the fact that pixels along the object boundary contain a mixture of foreground and background, it is both difficult and tedious to obtain pixel-accurate ground-truth masks. The gray boundary regions are ignored in the evaluation of foreground detection. Each background model was run for a large range of thresholds, and the true positive rate (TPR) and false positive rate (FPR), with respect to the ground-truth, were computed for each. The overall performance was measured by the area under the ROC curve (AUC). Videos of the results are available from [27].

6.3 Results on Bottle

We start by presenting results on Bottle. Since the water motion does not change significantly (i.e. its statistics are stationary), we only tested the non-adaptive dynamic texture (DT). The ROC curves of the different background models are shown in Fig. 3, and the area under the ROC (AUC) is listed in Table 1. We also report the false positive rate (FPR) at a true positive rate (TPR) of 0.80. The DT model performs significantly better than the GMM, with an AUC of 0.9985 for the former and 0.9229 for the latter. At 80% TPR, DT has a false-positive rate of 0.21%, while that of GMM is 2.36%. Note that although the DT model is non-adaptive, it models the background motion accurately. On the other hand, despite its adaptive nature, GMM cannot cope with the stochasticity of the water motion. The PCA model also performs



(b)

Fig. 5 ROC curves for foreground detection on (**a**) Boats1, and (**b**) Boats2

well, with an AUC of 0.9861, but is slightly inferior to DT. This suggests that modeling the water texture is sufficient for accurate performance, but the ability to further capture the texture dynamics gives the DT an additional performance boost.

Figure 4 shows the detected foreground regions at 80% TPR, for all background models. Note that the pixel-based methods (GMM and [9]) mark most of the reflection and shadow of the bottle as foreground, whereas the patch-based methods (DT, PCA, and [10]) classify it as background. Again, this illustrates the importance of modeling both texture and dynamics.

6.4 Results on Boats1 and Boats2

The ROC curves for Boats1 and Boats2 are shown in Fig. 5a and b, and the results summarized in Table 1. We start by noting that the adaptive dynamic texture mixture with $K = 3$ (DTM3) outperforms all other methods at almost all levels of TPR. For example on Boats1 at 90% TPR, DTM3 has an FPR of 0.61%, whereas DTM1, DT and GMM have FPRs of 1.2, 1.4 and 10.3%, respectively. On Boats2 at 55% TPR, DTM3 has an FPR of 0.88%, while DTM1, DT and GMM have FPRs of 3.18, 7.11 and 2.61%, respectively. This performance difference is illustrated in Figs. 6 and 7, which show the detected foregrounds. As the boat traverses the scene, DTM3 models its wake with a new mixture component, quickly including it into the background. The adaptive model with a single dynamic texture (DTM1) takes much longer to adapt to the wake, because it contains a single mode.

While the methods that fully exploit the probabilistic representation of the dynamic texture tend to do well on these sequences, the remaining methods perform fairly poorly. GMM is able to adapt to the wake of the boat, but the overall foreground detection is very noisy, due to the constant wave motion. The method of [10] fails to detect the wake as part of the background, and has trouble with the stochastic nature of the shimmering waves at the bottom of the video frame. On Boats1, the simple PCA model outperforms GMM, [9], and [10]. Finally, the ROC plots comparing DTM with the approximate DTM are shown in Fig. 8. The performance of DTM using the approximate sufficient statistics is similar to that of the standard DTM, albeit with a drop in performance on the difficult scene of Boats2. On this scene, the AUC drops from 0.9266 to 0.8955, suggesting that there is some loss in representative power when this approximation is used.

6.5 Results on other video sequences

In addition to the above quantitative evaluation, we present qualitative results on five additional challenging sequences. The first is "zod2" from the PETS2005 coastal surveillance data set [28], and is displayed in Fig. 9. It was captured with a thermal camera, and contains a small rubber Zodiac boat, at a variety of scales, traveling in the ocean. The boat starts close to the camera, and travels directly away from it. In the distance, it turns towards the right and travels across the scene. At this point the boat is very small relative to its initial size (only occupies a few pixels). The boat wake is clearly visible when the boat is close but not visible when it is far away.

Foreground detection with the adaptive dynamic texture mixture (DTM3) is shown in Fig. 9 (center). DTM3 marks the boat region as foreground, while ignoring the wake. The boat is also consistently marked as foreground, even when it is small relative to waves in other parts of the scene. This is in contrast to the GMM, which produces a significant amount of false detections. In particular, when the boat is small, the detected foreground region is very similar in shape and size

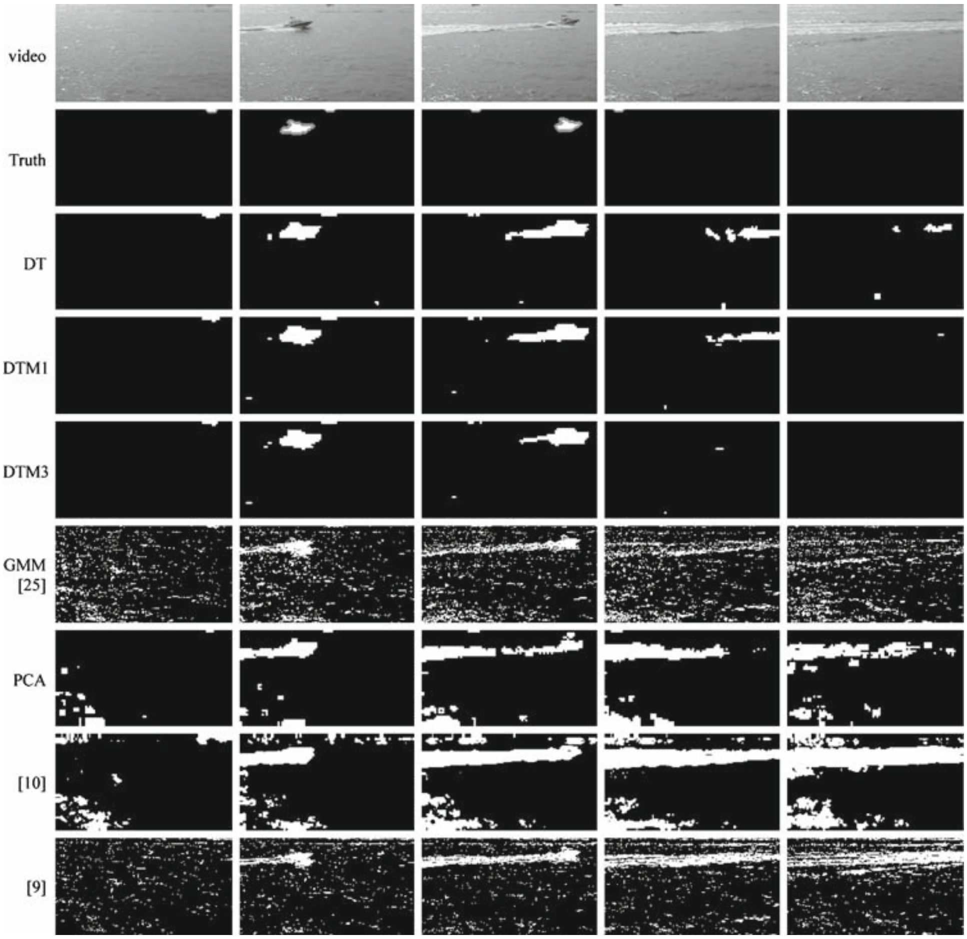


Fig. 6 Foreground detection results on Boats1 at 90% true-positive rate

to the false detections (e.g. see the last three rows of Fig. 9). These types of errors make subsequent operations, such as object tracking, very difficult. On the other hand, DTM3 performs well regardless of the boat scale, maintaining a very low FPR throughout the sequence.

The second sequence is the beach scene from [10], and appears in Fig. 10. The scene consists of two people walking on a beach, with part of the background composed of crashing waves. The waves have variable shape and intensity, making the background difficult to model. The foreground detections by DTM3 and GMM are shown in Fig. 10. Again, DTM3 is able to adapt to the wave motion, while detecting the foreground objects. On the other hand, the crashing waves cause false alarms by the GMM.

The final three videos illustrate the performance of the adaptive mixture on scenes containing dynamic backgrounds other than water. Figure 11 shows a sequence of a helicopter traveling in dense smoke, along with the detected foreground using the simple dynamic texture (DT). The helicopter is marked as foreground by the model, while the smoke is marked as background. The next sequence, displayed in Fig. 12, contains several people skiing down a mountain. The background includes falling snow flakes, and the scene is subject to an increasing amount of translation due to a camera pan. Figure 12 also shows the foreground detections by DTM3 and GMM. DTM3 successfully marks the skiers as foreground, but exhibits some errors when there is a significant amount of panning (e.g. the last two frames). On the other hand, the foreground detected by GMM contains a significant amount of noise, due to both the falling snow and the camera pan, and fails to find the skiers when they are small (e.g. in the first two frames).

The final sequence, shown in Fig. 13, shows a biker jumping in front of an explosion. In this scene the background is difficult to model because it changes significantly in a short period of time. Foreground detections by DTM3 are shown in Fig. 13 (middle), illustrating the ability of the model to quickly adapt to an unseen background motion. The first frame shows the background before the explosion occurs. In the next frame, the beginning of the explosion is detected as foreground, because it is an unseen motion process. As the explosion propagates outward (third and fourth frames), the areas within the explosion are marked as background, because the model adapts to the new motion. Meanwhile, the biker is still marked as foreground. Finally, after sufficient exposure (the fifth and sixth frames), the explosion no longer

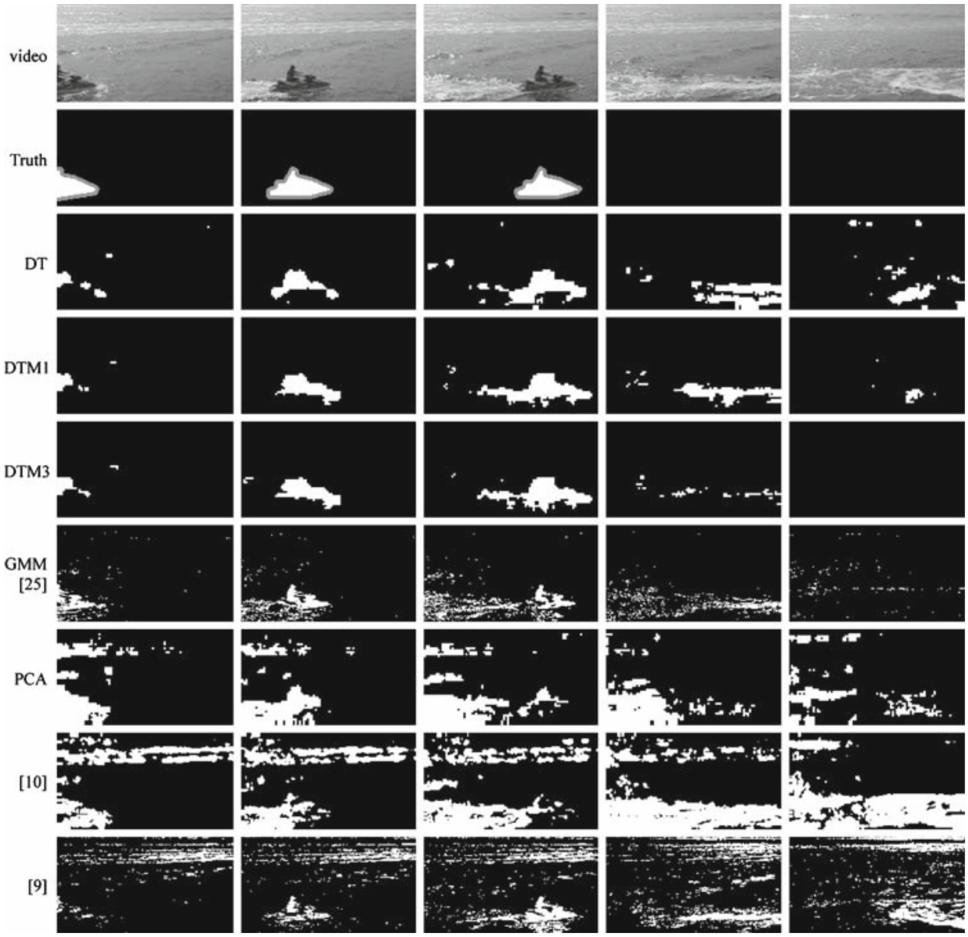


Fig. 7 Foreground detection results on Boats2 at 55% true-positive rate

Table 1 Quantitative results on three water scenes for different background models

Dataset		GMM	PCA	[10]	[9]	DT	DTM1	DTM3	DTM1x	DTM3x
Bottle	AUC	0.9229	0.9861	0.9912	0.9354	**0.9985**	–	–	–	–
Bottle	FPR (TPR = 0.80)	0.0236	0.0041	0.0098	0.0226	**0.0021**	–	–	–	–
Boats1	AUC	0.9516	0.9698	0.9061	0.9493	0.9884	0.9886	**0.9910**	0.9881	0.9902
Boats1	FPR (TPR = 0.90)	0.1030	0.0865	0.1762	0.1067	0.0140	0.0120	**0.0061**	0.0154	**0.0061**
Boats2	AUC	0.8952	0.8136	0.7073	0.7966	0.8685	0.8916	**0.9266**	0.8954	0.8955
Boats2	FPR (TPR = 0.55)	0.0261	0.1527	0.3105	0.1495	0.0711	0.0318	**0.0088**	0.0304	0.0097

Bold values indicate the best performance on each video

registers as foreground, leaving only the biker in this category. Under the GMM model (Fig. 13 bottom), the detected foreground again contains significant noise, indicating that the stochastic nature of the explosion is poorly modeled by the GMM pixel process.

7 Conclusions

In this work, we have introduced a generalization of the Stauffer–Grimson background subtraction algorithm. While the original algorithm restricts the background model to a Gaussian mixture, which is only suitable for static scenes, the generalization supports models with arbitrary component densities. The only restriction is that these densities can be summarized by sufficient statistics. We have applied the generalized SG model to the case where the component densities are dynamic textures, producing an adaptive background subtraction algorithm based on the mixture of dynamic textures, which is suitable for dynamic scenes. The new background subtraction algorithm includes a new online algorithm for dynamic texture parameter estimation using sufficient statis-

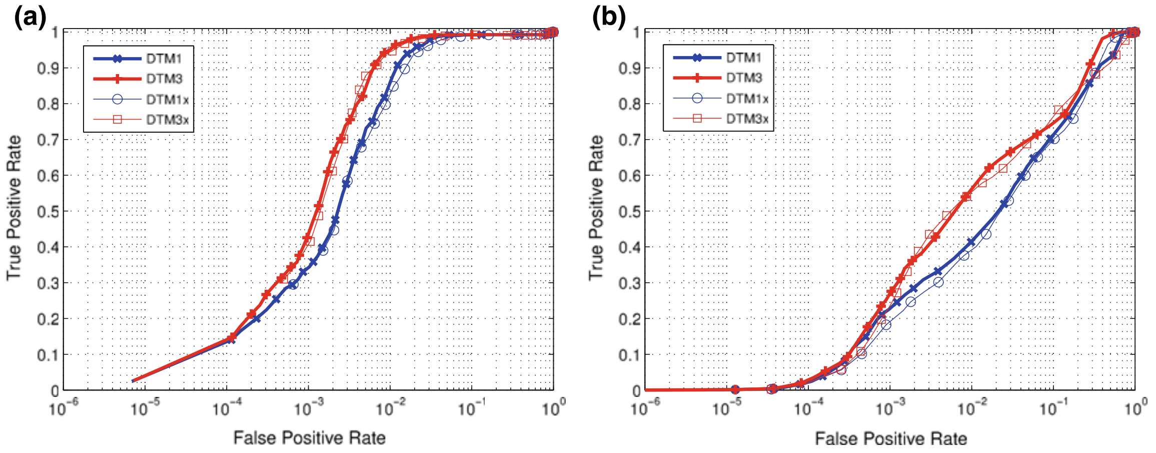


Fig. 8 ROC curves for foreground detection with DTM and approximate DTM on (**a**) Boats1 and (**b**) Boats2

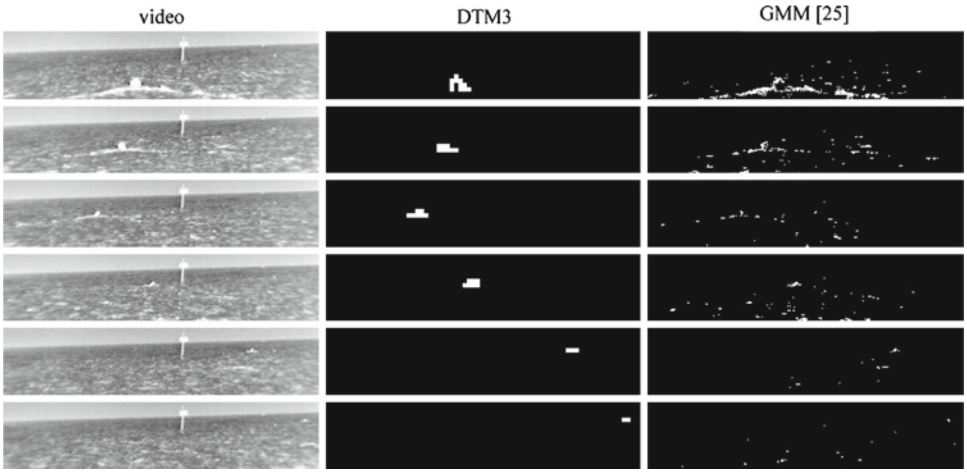


Fig. 9 Foreground detection results on "zod2" from the PETS2005 coastal surveillance dataset [28]

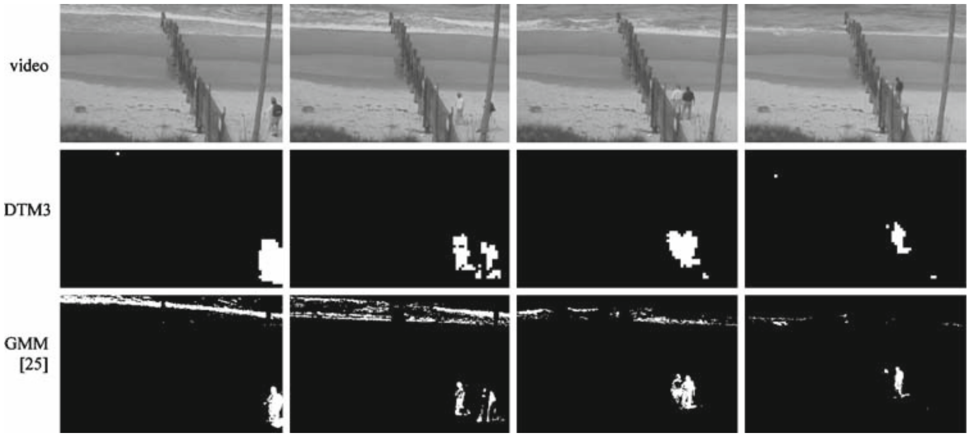


Fig. 10 Foreground detection results on the beach scene from [10]

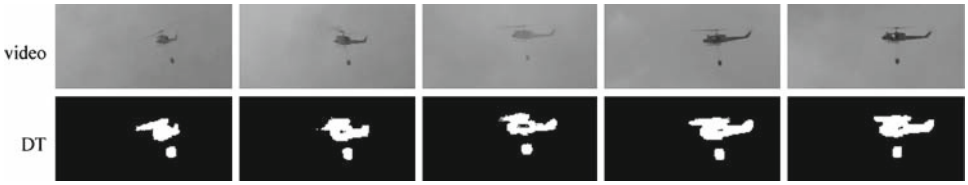


Fig. 11 Foreground detection results on Chopper video

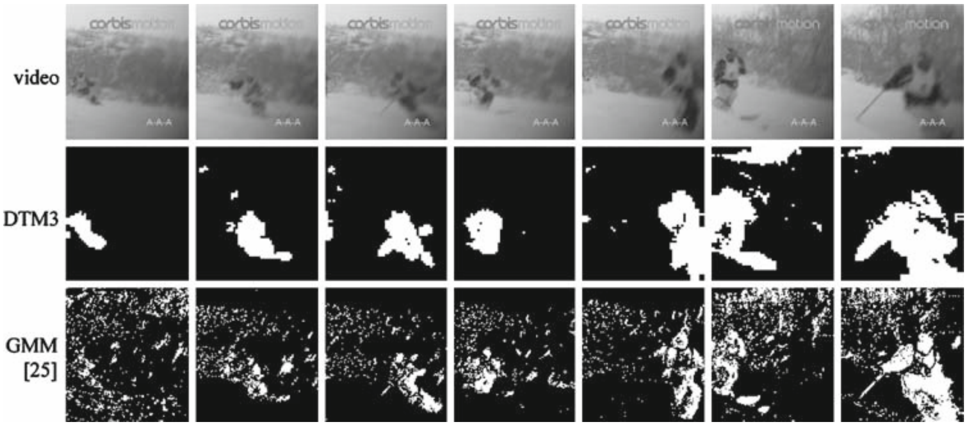


Fig. 12 Foreground detection results on Skiing video

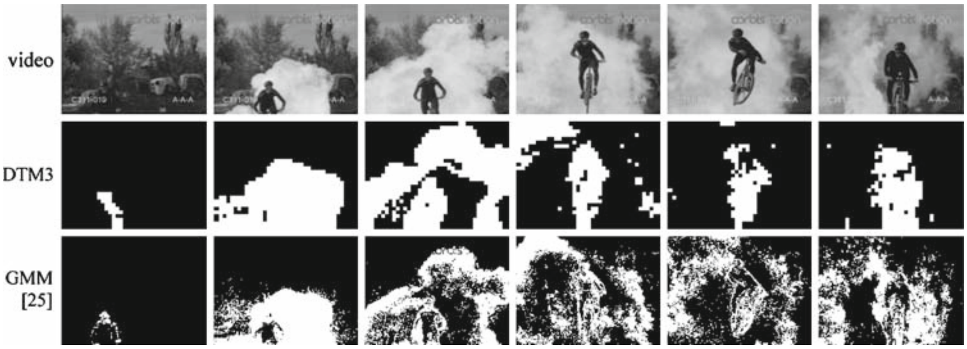


Fig. 13 Foreground detection results on CycleFire video

tics, which is equivalent to the least-squares algorithm of [3]. This algorithm can be used in any application of dynamic textures that requires online video processing, not just background subtraction. The performance of the new background subtraction algorithm was evaluated through extensive experiments, involving scenes with dynamic backgrounds. Substantial quantitative improvements over the state-of-the-art were demonstrated on aquatic scenes, where background dynamics are particularly challenging. For example, the proposed algorithm was shown to quickly incorporate boat wakes into the background, a task that proved too difficult for other state-of-the-art procedures. The efficacy of the proposed algorithm was also (qualitatively) demonstrated on other classes of dynamic backgrounds, including smoke, snow-fall, and fire. Finally, we note that the new background subtraction algorithm achieved high detection and low false-positive rates regardless of object scale, suggesting that it may be suitable for surveillance on dynamic scenes (e.g. harbors).

Acknowledgments The authors thank Stan Sclaroff for the Bottle video from [9], Terry Boult for the zodiac video from PETS2005 [28], Gerald Dalley for the Beach video from [10], and Zoran Zivkovic for the GMM code [25]. The authors also thank Mulloy Morrow for capturing the Boats1 and Boats2 video. This work was funded by NSF award IIS-0534985, NSF Career award IIS-0448609 and NSF Integrative Graduate Education and Research Traineeship (IGERT) award DGE-0333451.

Appendix: Online estimation with approximate sufficient statistics

In this appendix, we derive the online estimation algorithm for dynamic textures (Sects. 4.2 and 5.2) using approximate sufficient statistics. If the dimension of the video is large, then computing and storing the sufficient statistics may be impractical. In this case, the statistics can be approximated with the top q principal components of the image covariance Φ, i.e.

$$\begin{aligned}\Phi &\approx D\Phi_q D^T, \quad \phi \approx D\phi_q D^T, \quad \eta \approx D\eta_q D^T,\\ \psi &\approx D\psi_q D^T, \quad \varphi \approx D\varphi_q D^T, \quad \xi \approx D\xi_q,\end{aligned} \tag{53}$$

where $D \in \mathbb{R}^{m\times q}$ is the matrix containing the q top PCA basis vectors of Φ. The approximate sufficient statistics, which are the projections of the sufficient statistics into the PCA basis of D, are: $\Phi_q \in \mathbb{R}^{q\times q}$ which is diagonal, $\{\Phi_q, \phi_q, \eta_q, \psi_q, \varphi_q\} \in \mathbb{R}^{q\times q}$, and $\xi_q \in \mathbb{R}^q$. The online estimation algorithm proceeds in three phases: (1) update the PCA basis D, (2) update the approximate statistics, and (3) re-estimate the parameters of the dynamic texture. We will denote the previous PCA basis as D_{old}, the current approximate statistics $\{\Phi_q, \phi_q, \eta_q, \psi_q, \varphi_q, \xi_q\}$, and the new observation $Y_{1:\tau}$.

In the first phase of the approximate algorithm, the PCA basis D_{old} must be updated with the new data $Y_{1:\tau}$. An incremental PCA procedure is adopted, similar to [19]. Substituting the approximate image covariance into the update equation (50) yields:

$$\Phi \approx (1-\alpha) D_{\text{old}} \Phi_q D_{\text{old}}^T + \alpha \left(\frac{1}{\tau} \tilde{Y}_{1:\tau} \tilde{Y}_{1:\tau}^T \right) = Z Z^T \tag{54}$$

where

$$Z = \left[\sqrt{(1-\alpha)} D_{\text{old}} (\Phi_q)^{\frac{1}{2}}, \sqrt{\tfrac{\alpha}{\tau}} \tilde{Y}_{1:\tau} \right]. \tag{55}$$

Hence, the new PCA basis D can be computed from the SVD of $Z = USV^T$,

$$D = [u_1, \ldots, u_q], \tag{56}$$

$$\Phi_q = \text{diag}\left(\left[s_1^2, s_2^2, \ldots, s_q^2 \right] \right), \tag{57}$$

where u_i are the columns of U corresponding to the q largest singular values $\{s_1, \ldots, s_q\}$.

The second phase of the algorithm updates the approximate sufficient statistics with the statistics of the new video. Define the projection of the video $\tilde{Y}_{1:\tau}$ onto the basis of D as:

$$\hat{V}_{1:\tau} = [\hat{v}_1 \cdots \hat{v}_\tau] = D^T \tilde{Y}_{1:\tau}. \tag{58}$$

Pre-multiplying (53) by D^T and post-multiplying by D, the estimates of the approximate statistics $\left\{ \hat{\phi}'_q, \hat{\varphi}'_q, \hat{\psi}'_q, \hat{\eta}'_q, \hat{\xi}'_q \right\}$ of the new video $Y_{1:\tau}$ are:

$$\begin{aligned}
\hat{\phi}'_q &= D^T \hat{\phi} D = D^T \tilde{Y}_{1:\tau-1} \tilde{Y}_{1:\tau-1}^T D = \hat{V}_{1:\tau-1} \hat{V}_{1:\tau-1}^T, \\
\hat{\varphi}'_q &= D^T \hat{\varphi} D = D^T \tilde{Y}_{2:\tau} \tilde{Y}_{2:\tau}^T D = \hat{V}_{2:\tau} \hat{V}_{2:\tau}^T, \\
\hat{\psi}'_q &= D^T \hat{\psi} D = D^T \tilde{Y}_{2:\tau} \tilde{Y}_{1:\tau-1}^T D = \hat{V}_{2:\tau} \hat{V}_{1:\tau-1}^T, \\
\hat{\eta}'_q &= D^T \hat{\eta} D = D^T \tilde{y}_1 \tilde{y}_1^T D = \hat{v}_1 \hat{v}_1^T, \\
\hat{\xi}'_q &= D^T \hat{\xi} = D^T \tilde{y}_1 = \hat{v}_1.
\end{aligned} \tag{59}$$

The online update equations for the sufficient statistics (50) have the form:

$$\phi \leftarrow (1-\alpha)\phi + \alpha \phi'. \tag{60}$$

Substituting the approximate statistics into (60), we have:

$$D \hat{\phi}_q D^T \leftarrow (1-\alpha) D_{\text{old}} \hat{\phi}_q D_{\text{old}}^T + \alpha D \hat{\phi}'_q D^T, \tag{61}$$

$$\hat{\phi}_q \leftarrow (1-\alpha) \left(D^T D_{\text{old}} \right) \hat{\phi}_q \left(D_{\text{old}}^T D \right) + \alpha \hat{\phi}'_q. \tag{62}$$

Defining $F = D^T D_{\text{old}}$, which transforms the approximate statistics from the old PCA basis into the new basis, the approximate statistics are updated according to:

$$\begin{aligned}
\hat{\phi}_q &\leftarrow (1-\alpha) F \hat{\phi}_q F^T + \alpha \hat{\phi}'_q, & \hat{\eta}_q &\leftarrow (1-\alpha) F \hat{\eta}_q F^T + \alpha \hat{\eta}'_q, \\
\hat{\varphi}_q &\leftarrow (1-\alpha) F \hat{\varphi}_q F^T + \alpha \hat{\varphi}'_q, & \hat{\xi}_q &\leftarrow (1-\alpha) F \hat{\xi}_q + \alpha \hat{\xi}', \\
\hat{\psi}_q &\leftarrow (1-\alpha) F \hat{\psi}_q F^T + \alpha \hat{\psi}'_q, & \bar{y} &\leftarrow (1-\alpha) \bar{y} + \alpha \bar{y}'.
\end{aligned} \tag{63}$$

In the final phase, the dynamic texture parameters are estimated from the approximate sufficient statistics. From (20), the estimate of the observation matrix is:

$$\hat{C} = \text{PCA}\left(\hat{\Phi}, n \right) = \text{PCA}\left(D \hat{\Phi}_q D^T, n \right) = DJ, \tag{64}$$

where $J = I_{q,n}$ is the $q \times n$ identity matrix, which effectively selects the first n columns of D. Next, substituting with (53), we have:

$$\begin{aligned}
\hat{C}^T \hat{\phi} \hat{C} &\approx \hat{C}^T \left(D \hat{\phi}_q D^T \right) \hat{C} = J^T \hat{\phi}_q J, \\
\hat{C}^T \hat{\varphi} \hat{C} &\approx \hat{C}^T \left(D \hat{\varphi}_q D^T \right) \hat{C} = J^T \hat{\varphi}_q J, \\
\hat{C}^T \hat{\psi} \hat{C} &\approx \hat{C}^T \left(D \hat{\psi}_q D^T \right) \hat{C} = J^T \hat{\psi}_q J, \\
\hat{C}^T \hat{\eta} \hat{C} &\approx \hat{C}^T \left(D \hat{\eta}_q D^T \right) \hat{C} = J^T \hat{\eta}_q J, \\
\hat{C}^T \hat{\xi} &\approx \hat{C}^T (D \hat{\xi}_q) = J^T \hat{\xi}_q,
\end{aligned} \tag{65}$$

where $J^T \phi J$ selects the top-left $n \times n$ sub-matrix of ϕ. Hence substituting (65) into (43), the approximate parameter estimates are:

$$\begin{aligned}
&\hat{A} \approx \left(J^T \hat{\psi}_q J \right) \left(J^T \hat{\phi}_q J \right)^{-1}, \quad \hat{\mu} \approx J^T \hat{\xi}_q, \\
&\hat{Q} \approx \frac{1}{\tau - 1} \left(J^T \hat{\varphi}_q J - \hat{A} J^T \hat{\psi}_q^T J \right), \quad \hat{S} \approx \left(J^T \hat{\eta}_q J \right) - \hat{\mu} \hat{\mu}^T.
\end{aligned} \tag{66}$$

Finally, the covariance of the observation noise in (43) can be written as:

$$\begin{aligned}
\hat{R} &= \left(I - \hat{C} \hat{C}^T \right) \hat{\Phi} \left(I - \hat{C} \hat{C}^T \right) \\
&\approx \left(I - \hat{C} \hat{C}^T \right) Z Z^T \left(I - \hat{C} \hat{C}^T \right).
\end{aligned} \tag{67}$$

Assuming that $q \ll m$, the complexity of online estimation using the approximate statistics is dominated by the calculation of the PCA basis (i.e. taking the SVD of Z), which has complexity $O(m(q+\tau)^2)$. Note that using the approximate statistics reduces the complexity to be linear in m, rather than quadratic in m for the full statistics.

References

1. Sheikh, Y., Shah, M.: Bayesian modeling of dynamic scenes for object detection. IEEE Trans. PAMI **27**(11), 1778–1792 (2005)
2. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: CVPR, pp. 246–52 (1999)
3. Doretto, G., Chiuso, A., Wu, Y.N., Soatto, S.: Dynamic textures. Int. J. Comp. Vis. **51**(2), 91–109 (2003)
4. Saisan, P., Doretto, G., Wu, Y., Soatto, S.: Dynamic texture recognition. In: IEEE Conference Computer Vision and Pattern Recognition, vol. 2, pp. 58–63 (2001)
5. Chan, A.B., Vasconcelos, N.: Probabilistic kernels for the classification of auto-regressive visual processes. In: IEEE Conference Computer Vision and Pattern Recognition, vol. 1, pp. 846–851 (2005)

6. Doretto, G., Cremers, D., Favaro, P., Soatto, S.: Dynamic texture segmentation. In: IEEE International Conference Computer Vision, vol. 2, pp. 1236–1242 (2003)
7. Chan, A.B., Vasconcelos, N.: Mixtures of dynamic textures. In: IEEE International Conference Computer Vision, vol. 1, pp. 641–647 (2005)
8. Fitzgibbon, A.W.: Stochastic rigidity: image registration for nowhere-static scenes. In: IEEE International Conference Computer Vision, vol. 1, pp. 662–670 (2001)
9. Zhong, J., Sclaroff, S.: Segmenting foreground objects from a dynamic textured background via a robust Kalman filter. In: ICCV (2003)
10. Monnet, A., Mittal, A., Paragios, N., Ramesh, V.: Background modeling and subtraction of dynamic scenes. In: CVPR (2003)
11. Chan, A.B., Vasconcelos, N.: Modeling, clustering, and segmenting video with mixtures of dynamic textures. IEEE Trans. Pattern Anal. Mach. Intell. **30**(5), 909–926 (2008)
12. Kay, S.M.: Fundamentals of Statistical Signal Processing: Estimation Theory. Prentice-Hall, Englewood Cliffs (1993)
13. Heikkila, M., Pietikainen, M.: A texture-based method for modeling the background and detecting moving objects. IEEE Trans. PAMI **28**(4), 657–662 (2006)
14. Mittal, A., Paragios, N.: Motion-based background subtraction using adaptive kernel density estimation. In: CVPR (2004)
15. Elgammal, A., Harwood, D., Davis, L.: Non-parametric model for background subtraction. In: ECCV, pp. 751–57 (2000)
16. Latecki, L.J., Miezianko, R., Pokrajac, D.: Motion detection based on local variation of spatiotemporal texture. In: CVPR Workshops (2004)
17. Kahl, F., Hartley, R., Hilsenstein, V.: Novelty detection in image sequences with dynamic backgrounds. In: ECCV Workshop on Statistical Methods in Video Processing (2004)
18. Oliver, N.M., Rosario, B., Pentland, A.P.: Bayesian computer vision system for modeling human interactions. IEEE Trans. PAMI **22**(8), 831–843 (2000)
19. Li, Y.: On incremental and robust subspace learning. Pattern Recogn. **37**(7), 1509–1519 (2004)
20. Dalley, G., Migdal, J., Grimson, W.: Background subtraction for temporally irregular dynamic textures. In: Workshop on Applications of Computer Vision, Jan (2008)
21. Wixson, L.: Detecting salient motion by accumulating directionally-consistent flow. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 774–780 (2000)
22. Tian, Y.-L., Hampapur, A.: Robust salient motion detection with complex background for real-time video surveillance. In: IEEE Workshop on Motion and Video Computing (WACV/MOTION'05), vol. 2, pp. 30–35 (2005)
23. Shumway, R.H., Stoffer, D.S.: An approach to time series smoothing and forecasting using the EM algorithm. J. Time Series Anal. **3**(4), 253–264 (1982)
24. Ghahramani, Z., Hinton, G.: Parameter estimation for linear dynamical systems. Department of Computer Science, University of Toronto, Tech Report CRG-TR-96-2 (1996)
25. Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. In: ICVR (2004)
26. Zivkovic, Z.: [Online]. Available: http://staff.science.uva.nl/~zivkovic/DOWNLOAD.html (2004)
27. Chan, A.B.: Generalized Stauffer–Grimson background subtraction for dynamic scenes [Online]. Available: http://www.svcl.ucsd.edu/projects/dytexbkgnd (2008)
28. Boult, T.: Coastal surveillance datasets. Vision and Security Lab, U. Colorado at Colorado Springs [Online]. Available: http://www.vast.uccs.edu/~tboult/PETS2005 (2005)