Multi-Layer Background Subtraction Based on Color and Texture*

Jian Yao Jean-Marc Odobez
IDIAP Research Institute
Rue du Simplon 4, 1920 Martigny, Switzerland
{Jian.Yao,Jean-Marc.Odobez}@idiap.ch

Abstract

In this paper, we propose a robust multi-layer background subtraction technique which takes advantages of local texture features represented by local binary patterns (LBP) and photometric invariant color measurements in RGB color space. LBP can work robustly with respective to light variation on rich texture regions but not so efficiently on uniform regions. In the latter case, color information should overcome LBP's limitation. Due to the illumination invariance of both the LBP feature and the selected color feature, the method is able to handle local illumination changes such as cast shadows from moving objects. Due to the use of a simple layer-based strategy, the approach can model moving background pixels with quasi-periodic flickering as well as background scenes which may vary over time due to the addition and removal of long-time stationary objects. Finally, the use of a cross-bilateral filter allows to implicitely smooth detection results over regions of similar intensity and preserve object boundaries. Numerical and qualitative experimental results on both simulated and real data demonstrate the robustness of the proposed method.

1. Introduction

Foreground objects detection and segmentation from a video stream captured from a stationary camera is one of the essential tasks in video processing, understanding and visual surveillance. A commonly used approach to extract foreground objects consists of performing background subtraction. Despite the large number of background subtraction methods [2, 6, 7, 8, 10] that have been proposed in the past decade and that are used in real-time video processing, the task remains challenging when the background contains moving objects (e.g. waving tree branches, moving escalators) as well as shadows cast by the moving objects we want to detect, and undergoes various changes due to illumination variations, or the addition or removal of stationary objects.

Much work has been done since the introduction of the Mixture of Gaussian (MoG) model by Stauffer and Grimson [10]. In their approach, the mixture of $K(= 3, 4, 5)$ Gaussians representing the statistics of one pixel over time can cope with multi-modal background distributions. However, a common problem for this approach is to find the right balance between the speed at which the model adapts to changing background, and the stablity, i.e. how to avoid forgetting background which is temporarily occluded. Lee et al. [7] proposed an effective scheme to improve the update speed without compromising the model stability. To robustly represent multi-modal scenes (e.g. wavering trees or moving escalators), Tuzel *et al.* [11] proposed to estimate the probability distribution of mean and covariance of each Gaussian using recursive Bayesian learning, which can preserve the multi-modality of the background and estimate the number of necessary layers for representing each pixel. Most of these methods use only pixel color or intensity information to detect foreground objects. They may fail when foreground objects have similar color to the background. Heikkila *et al.* [2] developed a novel and powerful approach based on discriminative texture features represented by LBP histograms to capture background statistics. The LBP is invariant to local illumination changes such as cast shadow because LBP is obtained by comparing local pixels values. However, at the same time, it can not detect changes in sufficiently large uniform regions if the foreground is also uniform. In general, most of the methods that tackle the removal of shadow and highlight [3, 4] proposed to do it in a post-processing step. Jacques *et al.* [4] proposed to use the zero-mean normalized cross-correlation (ZNCC) to first detect shadow pixel candidates and then refine the results using local statistics of pixel ratios. Hu *al.* [3] proposed a photometric invariant model in the RGB color space to explain the intensity changes of one pixel w.r.t. illumination changes. Kim *et al.* [6] present a similar approach, but directly embedded in the background modeling, not as a post-processing step. They also proposed a multi-layer

*This work was supported by the European Union 6th FWP Information Society Technologies CARETAKER project (Content Analysis and Retrieval Technologies Applied to Knowledge Extraction of Massive Recordings, FP6-027231).

background scheme which, however, needs more memories and computation costs. Javed *et al.* [5] proposed to integrate multiple cues (color and gradient information) to detect moving foreground objects in three distinct levels, i.e. pixel level, region level and frame level.

In this paper, we propose a layer-based method to detect moving foreground objects from a video sequence taken under a complex environment by integrating advantages of both texture and color features. Compared with the previous method proposed by Heikkila *et al.* [2], several modifications and new extensions are introduced. First, we integrate a newly developed photometric invariant color measurement in the same framework to overcome the limitations of LBP features in regions of poor or no texture and in shadow boundary regions. Second, a flexible weight updating strategy for background modes is proposed to more efficiently handle moving background objects such as wavering tree branches and moving escalators. Third, a simple layer-based background modeling/detection strategy was developed to handle the background scene changes due to addition or removal of stationary objects (e.g. a car enters a scene and stay there for a long time). It is very useful for removing the ghost produced by the changed background scene, detecting abandoned luggage, etc. Finally, the fast cross bilateral filter [9] was used to remove noise and enhance foreground objects as a post-processing step.

The rest of this paper is organized as follows. A brief introduction on texture and color features is given in Section 2. Our proposed method for background modeling and foreground detection is described in Section 3. Experimental results on simulated and real data are reported in Section 4. Finally conclusions are given in Section 5.

2. Texture and Color Features

In this section, we introduce the local binary pattern used to model texture and the photometric invariant color measurements, which are combined for foreground detection.

2.1. Local Binary Pattern

LBP is a robust gray-scale invariant texture feature. The LBP operator consists of labeling a pixel with a binary number obtained by thresholding the gray-scale difference between the gray-scale value of each neighbor of the pixel and the pixel's gray-scale value, and considering the multiple 0, 1 output as a binary number. More formally, the LBP of the pixel $\mathbf{x}$ in an image $\mathbf{I}$ can be represented as follows:

$$\mathbf{LBP}_{P,R}(\mathbf{x}) = \{LBP^{(p)}_{P,R}(\mathbf{x})\}_{p=1,\ldots,P}, \quad (1)$$

$$LBP^{(p)}_{P,R}(\mathbf{x}) = s(\mathbf{I}^g(\mathbf{v}_p) - \mathbf{I}^g(\mathbf{x}) - n), \;\; s(x) = \begin{cases} 1 & x \geq 0, \\ 0 & x < 0, \end{cases}$$

where $\mathbf{I}^g(\mathbf{x})$ denotes the gray value of the pixel $\mathbf{x}$ in the image $\mathbf{I}$ and $\{\mathbf{v}_p\}_{p=1,\ldots,P}$ as a set of P equally spaced pixels located on a circle of radius R and center $\mathbf{x}$. The parameter n is a noise parameter which should make the LBP signature more stable against noise (e.g. like compression) in uniform areas. It is the minumum amount of positive gray-scale variation that is considered as a significant change. Note that the LBP can be extended to color images with the LBP computed on each separated color channel. Also, multi-scale LBP can be defined with different radiuses at different levels.

LBP has several properties that are beneficial to its usage in background modeling. As a (binary) differential operator, LBP is robust to monotonic gray-scale changes, whether global or local illumination. In the latter case, cast shadow can be coped with when the shadow areas are not too small and the chosen circle radius for the LBP features is small. Finally, LBP features are very fast to compute, which is an important property from the practical implementation point of view.

Heikkila *et al.* [2] proposed to represent LBP texture feature using a 2^P-bin LBP histogram over a neighborhood region. The main limitation is that both memories and computation costs increase exponentially with the increasing of P. In this paper, we prefer to represent the LBP feature by a set of P binary numbers, with memory and computation cost linearly proportional to P.

2.2. Photometric Invariant Color

The LBP features work robustly for background modeling in most cases. However, it fails when both the background image and the foreground objects share the same texture information. This is especially frequent in region of low (or no) texture, like image areas such as wall or floor and flat foreground object such as color clothes. To handle these situations, we proposed to utilize a shadow invariant color distance in the RGB color space to compare an observed color value with a color mode in our algorithm. Speak about invariant color descriptors (e.g. hue, saturation), whose computation is unstable for dark or gray color values. Hence, we observed how pixel values change over time under lighting variation using a color panel and found that there is the same phenomenon as described in [6]. We observe that pixel values changed due to illumination changes are mostly distributed along in the axis going toward the RGB origin point $(0, 0, 0)$. Thus, we proposed to compare the color difference between an observed color pixel and a background color pixel using their relative angle in RGB color space with respect to the origin and the changing range of the background color pixel up to last time instant.

3. Background Subtraction Algorithm

In this section, we introduce our approach to perform background modeling subtraction. We describe in turn the

background model, the overall algorithm, the distance used to compare image features with modes, and the foreground detection step.

3.1. Background Modeling

Background modeling is the most important part of any background subtraction algorithms. The goal is to construct and maintain a statistical representation of the scene to be modeled. Here, we chose to utilize both texture information and color information when modeling the background. The approach exploits the LBP feature as a measure of texture because of its good properties, along with an illumination invariant photometric distance measure in the RGB space. The algorithm is described for color images, but it can also be used for gray-scale images with minor modifications.

Let $\mathcal{I} = \{\mathbf{I}^t\}_{t=1,\ldots,N}$ be an image sequence of a scene acquired with a static camera, where the superscript t denotes the time. Let $\mathcal{M}^t = \{\mathbf{M}^t(\mathbf{x})\}_{\mathbf{x}}$ represent the learned statistical background model at time t for all pixels $\mathbf{x}$ belonging to the image grid. The background model at pixel $\mathbf{x}$ and time t is denoted by $\mathbf{M}^t(\mathbf{x}) = \{K^t(\mathbf{x}), \{\mathbf{m}_k^t(\mathbf{x})\}_{k=1,\ldots,K^t(\mathbf{x})}, B^t(\mathbf{x})\}$, and consists of a list of $K^t(\mathbf{x})$ modes $\mathbf{m}_k^t(\mathbf{x})$ learned from the observed data up to the current time instant, of which the first $B^t(\mathbf{x})(\leq K^t(\mathbf{x}))$ have been identified as representing background observations. Each pixel has a different list size based on the observed data variation up to the current instant. To keep the complexity bounded, we set a maximal mode list size $K_{\max}$. In the following unless explicitly stated or needed, the time superscript t will be omitted to simplify the presentation. Similarly, when the same operations applies to each pixel position, we will drop the $(\mathbf{x})$ notation.

For each pixel $\mathbf{x}$, each mode consists of 7 components according to $\mathbf{m}_k = \{\mathbf{I}_k, \hat{\mathbf{I}}_k, \check{\mathbf{I}}_k, \mathbf{LBP}_k, w_k, \hat{w}_k, L_k\}, k = 1, \ldots, K$. $\mathbf{I}_k$ denotes the average RGB vector $\mathbf{I}_k = (\mathbf{I}_k^R, \mathbf{I}_k^G, \mathbf{I}_k^B)$ of the mode. $\hat{\mathbf{I}}_k$ and $\check{\mathbf{I}}_k$ denote the estimated maximal and minimal RGB vectors[1] that the pixels associated with this mode can take. $\mathbf{LBP}_k$ denotes the average local binary pattern learned from all the LBPs that were assigned to this mode. $w_k \in [0, 1]$ denotes the weight factor, i.e. the probability that this mode belongs to the background. $\hat{w}_k$ represents the maximal value that this weight achieved in the past. L_k is the background layer number to which the mode belongs, where $L_k = 0$ means that $\mathbf{m}_k$ is not a reliable background mode and $L_k = l > 0$ indicates that it is a reliable background mode in the l-th layer). The use of layers allows us to model/detect multi-layer backgrounds. The motivation of multi-layered background modeling and foreground detection is to be able to detect foreground objects against all backgrounds which were learned from past observations but which were subsequently covered by long-time stationary objects, and then suddenly uncovered. Without these background layers, interesting foreground objects (e.g., people) will be detected mixed with other stationary objects (e.g., car). In addition, it should be useful to detect abandoned luggage and background scene changes (such as graffiti or posters) in visual surveillance scenarios.

[1]The maximal (minimal) RGB vector values are defined componentwise.

3.2. Background Model Update Algorithm

The algorithm works as follows. Given the $\mathbf{LPB}^t$ and RGB value $\mathbf{I}^t$ measured at time t (and position $\mathbf{x}$), the algorithm first seeks to which mode of the background it belongs to by computing a distance between these measurements and the data of each mode $\mathbf{m}_k^{t-1}$. This distance, denoted $Dist(\mathbf{m}_k^{t-1})$, will be described later. The mode that is closest to the measurements is denoted by $\tilde{k}$ (i.e. $\tilde{k} = \arg\min_k Dist(\mathbf{m}_k^{t-1})$). If the distance to the closest mode is above a given threshold (i.e. $Dist(\mathbf{m}_{\tilde{k}}^{t-1}) > T_{bgu}$), a new mode is created with parameters $\{\mathbf{I}^t, \mathbf{I}^t, \mathbf{I}^t, \mathbf{LBP}^t, w_{init}, w_{init}, 0\}$ where w_{init} denotes a low initial weight. This new mode is either added to the list of modes (if $K^{t-1} < K_{\max}$) or replaces the existing mode which has the lowest weight (if $K^{t-1} = K_{\max}$). On the contrary, if the matched mode $\tilde{k}$ is close enough to the data (i.e. if $Dist(\mathbf{m}_{\tilde{k}}^{t-1}) < T_{bgu}$) its representation is updated as follows:

$$\begin{cases} \check{\mathbf{I}}_{\tilde{k}}^t &= \min(\mathbf{I}^t, (1+\beta)\check{\mathbf{I}}_{\tilde{k}}^{t-1}), \\ \hat{\mathbf{I}}_{\tilde{k}}^t &= \max(\mathbf{I}^t, (1-\beta)\hat{\mathbf{I}}_{\tilde{k}}^{t-1}), \\ \mathbf{I}_{\tilde{k}}^t &= (1-\alpha)\mathbf{I}_{\tilde{k}}^{t-1} + \alpha\mathbf{I}^t, \\ \mathbf{LBP}_{\tilde{k}}^t &= (1-\alpha)\mathbf{LBP}_{\tilde{k}}^{t-1} + \alpha\mathbf{LBP}^t, \\ w_{\tilde{k}}^t &= (1-\alpha_w^i)w_{\tilde{k}}^{t-1} + \alpha_w^i, \\ & \quad \text{with} \quad \alpha_w^i = \alpha_w(1+\tau\hat{w}_{\tilde{k}}^{t-1}) \\ \hat{w}_{\tilde{k}}^t &= \max(\hat{w}_{\tilde{k}}^{t-1}, w_{\tilde{k}}^t), \\ L_{\tilde{k}}^t &= 1 + \max\{L_k^{t-1}\}_{k=1,\ldots,K^{t-1}, k\neq\tilde{k}}, \\ & \quad \text{if } L_{\tilde{k}}^t = 0 \text{ and } \hat{w}_{\tilde{k}}^t > T_{bw} \end{cases} \quad (2)$$

while the other modes are updated by recopy from the previous time frame (i.e. $\mathbf{m}_k^t = \mathbf{m}_k^{t-1}$) with the exception of the weight, which decreases according to:

$$w_k^t = (1-\alpha_w^d)w_k^{t-1} \text{ with } \alpha_w^d = \frac{\alpha_w}{1+\tau\hat{w}_k^{t-1}} \quad (3)$$

In the above, $\beta \in [0, 1)$ is the learning rate involved in the update rule of the minimum and maximum of color values, whose goal is to avoid the maximum (resp. minimum) value keeping increasing (resp. decreasing) over time. This make the process robust to noise and outlier measurements. The parameter $\alpha \in (0, 1)$ is the learning rate that controls the update of the color and texture information. The threshold T_{bw} is used to check whether the updated mode has become a reliable background mode.

For the update of the weight, we have proposed a novel 'hysteris' scheme which works as follows. First, note that the weight decreasing factor α_w^d is proportional to a constant factor α_w, as usually found in other approaches, but

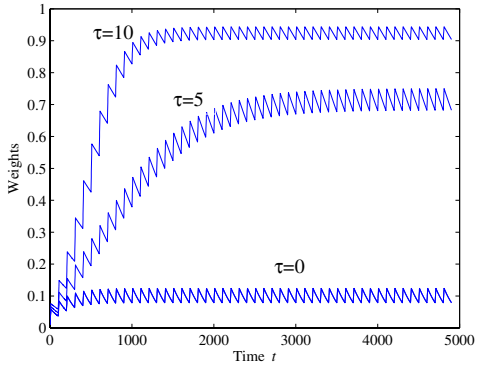

Figure 1. Evolution of a mode weight for a quasi-periodic pixel $\mathbf{x}$, where the data repeatedly match the mode for 10 frames, and don't match the mode for the subsequent 90 frames ($\alpha_w = 0.005, w_{init} = 0.01$), with different constants τ.

also depends on a constant τ and on the maximal weight $\hat{w}_k$. The larger the value of τ or the value of $\hat{w}_k$, the smaller the value of α_w^d, and thus the slower the weight decreases. Thus, if in the past, the mode has been observed for a sufficiently long amount of time, we will reduce the chances of forgetting it (e.g. this is the case when the background is covered by a stationary object, e.g. a parked car). Similarly, the increase weight factor α_w^i depends on α_w, the constant τ and the maximal weight $\hat{w}_k$. The larger the value of τ or the value of $\hat{w}_k$, the larger the value of α_w^i, i.e. the faster the weight increases. This proposed scheme allows to handle either background space repeatedly recovered by moving objects, or moving background pixels with quasi-periodic flickering, such as escalators. For instance, consider a pixel where a moving background matches a mode in 10 frames of the video and then disappears in the next 90 frames. The weight updating results with different constants τ are shown in Figure 1. With the classical setting ($\tau = 0$), the weight increases, but soon saturates at a small value (around 0.1 in the example). By using other reasonable values of τ (e.g. 2 or 3), the memory effect due to the introduction of the maximum weight can allow a faster increase of the weight, and a saturation at a larger value better reflecting that this mode may belong to the background. Note that at the same time, due to the use of both color and texture, the chances that moving foreground objects generate a consistent mode over time (and beneficiate from this effect) are quite small.

Finally, after the update step, all the modes $\{\mathbf{m}_k^t\}_{k=1,\ldots,K^t}$ are sorted in decreasing order according to their weights, and the number of modes deemed to belong to the background are the first B^t modes that satisfy

$$\sum_{k=1}^{B^t} w_k^t \Big/ \sum_{k=1}^{K^t} w_k^t \geq T_B, \tag{4}$$

where $T_B \in [0,1]$ is a threshold.

3.2.1 Texture- and Color-based Distance

The proposed measurement distance integrating texture information and color information is defined as follows:

$$\begin{aligned} Dist(\mathbf{m}_k^{t-1}) &= \lambda D_{text}(\mathbf{LBP}_k^{t-1}(\mathbf{x}), \mathbf{LBP}^t(\mathbf{x})) \\ &\quad + (1-\lambda) D_c(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x})), \end{aligned} \tag{5}$$

where the first term measures the texture distance, the second term measures the color distance and $\lambda \in [0,1]$ is a weight value indicating the contribution of the texture distance to the overall distance. The smaller the distance $Dist(\mathbf{m}_k^{t-1})$, the better the pixel $\mathbf{x}$ matches the mode $\mathbf{m}_k^{t-1}$.

The texture distance is defined as:

$$D_{text}(\mathbf{LBP}_a, \mathbf{LBP}_b) = \frac{1}{P} \sum_{p=1}^{P} D_{0|1}\Big(LBP_a^{(p)}, LBP_b^{(p)}\Big), \tag{6}$$

where $D_{0|1}(\cdot,\cdot)$ is a binary distance function defined as:

$$D_{0|1}(x,y) = \begin{cases} 0 & |x-y| \leq T_D, \\ 1 & \text{otherwise}, \end{cases} \tag{7}$$

where $T_D \in [0,1)$ is a threshold. Note that, from Eq. 5, the LBP values measured at time t, $\mathbf{LBP}^t(\mathbf{x})$, comprising either 0 or 1, will be compared to the LBP values of $\mathbf{LBP}_k^t(\mathbf{x})$, composed of averages of 0 or 1. Hence, the distance in Eq. 7 is quite selective: a measured LBP value (e.g. 0) will match its corresponding average only if this average is close enough (e.g. below $T_D = 0.2$). In other words, the distance of a measured data to a 'noisy' mode for which the previously observed data lead to average LBP values in the range $[T_D, 1-T_D]$ will systematically be 1. In this way, the selected distance will favor modes with clearly identified LBP patterns.

The color distance $D_c(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x}))$ is defined as:

$$\begin{aligned} D_c(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x})) &= \max\big(D_{angle}(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x})), \\ &\quad D_{range}(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x}))\big), \end{aligned} \tag{8}$$

where $D_{angle}(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x}))$ and $D_{range}(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x}))$ are two distances based on the relative angle formed by the two RGB vectors $\mathbf{I}_k^{t-1}(\mathbf{x})$ and $\mathbf{I}^t(\mathbf{x})$, and the range within which we allow the color changes to vary, respectively, as illustrated in Figure 2. The distance D_{angle} is defined as:

$$D_{angle}(\mathbf{I}_k^{t-1}(\mathbf{x}), \mathbf{I}^t(\mathbf{x})) = 1 - e^{-\kappa \max(0, \theta - \theta_n)}, \tag{9}$$

where θ is the angle formed by two RGB vectors $\mathbf{I}_k^{t-1}$ and $\mathbf{I}^t$ (w.r.t. the origin of the RGB color space) and θ_n is the largest angle formed by the RGB vector $\mathbf{I}^t$ and any of the virtual noisy RGB vectors $\{\tilde{\mathbf{I}}^t = \mathbf{I}^t + \mathbf{I}_n, \|\mathbf{I}_n\| \leq n_c\}$, where $\mathbf{I}_n$ denotes the noise (esp. compression noise) that can potentially corrupt the measurements, and where n_c parameterizes the maximum amount of noise that can be expected. As a result, we have $\theta_n = \arcsin(n_c/\|\mathbf{I}^t\|)$ [2]. Like

[2] Since in practice compression noise happens to be proportional to the intensity, we defined a minimum value $\tilde{\theta}_n$ for θ_n.

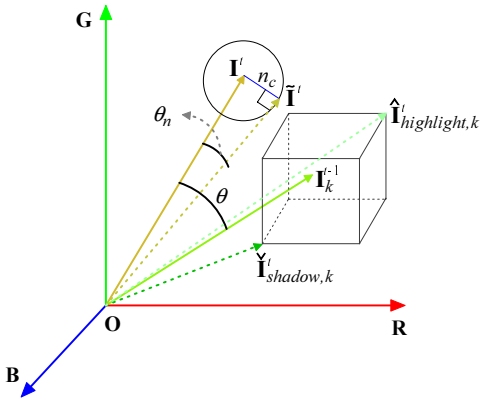

Figure 2. The proposed photometric invariant color model.

the noise parameter n presented in Eq. 1 for calculating the LBP, the parameter n_c (we used the same value for n and n_c) will allow to correctly account for noise in the color distance. This is particularly important for dark pixels where standard alternative color invariants (e.g. hue or saturation) are particularly sensitive to noise. The involved angles are illustrated in Figure 2.

The distance $D_{range}(\mathbf{I}^{t-1}_k(\mathbf{x}), \mathbf{I}^t(\mathbf{x}))$ is defined as:

$$D_{range}(\mathbf{I}^{t-1}_k, \mathbf{I}^t) = \begin{cases} 0 & \text{if } \mathbf{I}^t \in [\check{\mathbf{I}}^t_{shadow,k}, \hat{\mathbf{I}}^t_{highlight,k}], \\ 1 & \text{otherwise.} \end{cases} \quad (10)$$

where $\mathbf{I}^t(\mathbf{x}) \in [\check{\mathbf{I}}^t_{shadow,k}, \hat{\mathbf{I}}^t_{highlight,k}]$ means that the measurement belongs to the volume defined by the minimum and maximum color values of $\check{\mathbf{I}}^t_{shadow,k}$ and $\hat{\mathbf{I}}^t_{highlight,k}$, as illustrated in Figure 2. These extremes represent the potentially darkest "shadow" and brightest "highlight" color values that the pixel can take, and are defined by $\check{\mathbf{I}}^t_{shadow,k} = \min\left(\mu \mathbf{I}^t_k, \check{\mathbf{I}}^t_k\right)$ and $\hat{\mathbf{I}}^t_{highlight,k} = \max\left(\nu \mathbf{I}^t_k, \hat{\mathbf{I}}^t_k\right)$ where μ and ν are shadow and highlight factors, respectively, that define the range of measures that can correspond to a shadowed or highlighted pixel. Typically, $\mu \in [0.4, 0.7]$ and $\nu \in [1, 1.2]$.

3.3. Foreground Detection

Foreground detection is applied after the update of the background model. First, a background distance map $\mathbf{D^t} = \{\mathbf{D^t}(\mathbf{x})\}_\mathbf{x}$ is built, which can be seen as the equivalent of the foreground probabilities in the Mixture of Gaussian (MoG) approach. For a given pixel $\mathbf{x}$, the distance is defined as $\mathbf{D}^t(\mathbf{x}) = Dist(\mathbf{m}^{t-1}_{\tilde{k}}(\mathbf{x}))$, which is the distance to the closest mode as mentioned in Subsection 3.2, unless we have $\tilde{k} > B^t(\mathbf{x})$ and $L_{\tilde{k}}(\mathbf{x}) = 0$ (i.e. the mode was never identified as a reliable background mode in the past). In this latter case, the distance is set to $\max(Dist(\mathbf{m}^{t-1}_{\tilde{k}}(\mathbf{x})), 2T_{bg})$, where T_{bg} is a foreground/background threshold. To filter out noise, we propose to smooth the distance map using the cross bilateral filter introduced in [1]. It is defined as:

$$\tilde{\mathbf{D}}^t(\mathbf{x}) = \frac{1}{\tilde{\mathbf{W}}(\mathbf{x})} \sum_{\mathbf{v}} G_{\sigma_s}(\|\mathbf{v}-\mathbf{x}\|) G_{\sigma_r}(|\mathbf{I}^{g,t}(\mathbf{v})-\mathbf{I}^{g,t}(\mathbf{x})|) \mathbf{D}^t(\mathbf{v}),$$

Figure 3. Typical foreground objects (out of 50).

where $\tilde{\mathbf{W}}(\mathbf{x})$ is a normalizing constant, $\mathbf{I}^{g,t}$ denotes the gray-level image at time t, σ_s defines the size of the spatial neighborhood to take into account for smoothing, σ_r controls how much an adjacent pixel is downweighted because of its intensity difference, and G_σ denotes a Gaussian kernel. As can be seen, the filter smoothes values that belong to the same gray-level region, and thus prevents smoothing across edges. The filter is implemented using a fast approximation method [9]. Finally, the foreground pixels are those for which $\tilde{\mathbf{D}}^t(\mathbf{x})$ is larger than the T_{bg} threshold.

4. Experimental Results

In this section, we examined the performance of our proposed method on both simultated and real data.

4.1. Simulated Data

To evaluate the different components of our method, we performed experiments on simulated data, for which the ground truth is known:

Background Frames (BF): For each camera, 25 randomly selected background frames containing no foreground objects were extracted from the recorded video stream.

Background and Shadow Frames (BSF): In addition to the BF frames, we generated 25 background frames containing highlight and (mainly) shadow effects. The frames were composited as illustrated in Figure4, by removing foreground objects from a real image and replacing them with background content.

Foreground Frames, without (FF) or with Shadow (FSF): To evaluate the detection, we generated composite images obtained by clipping foreground objects (see Figure 3) at random locations into a background image[3]. This way, the foreground ground truth is known (see Figure 5). The number of inserted objects was randomly selected between 1 and 10. When a BF (resp. BSF) frame was used as background, we denote the result a FF (resp. FSF) image.

Evaluation protocol: The experiments were conducted as follows. First, a sequence of 100 BF frames was generated and used to build the background model. This model was then used to test the foreground detection algorithm on a simulated foreground image. This operation was repeated 500 times. Two series of experiments were conducted: in the first one, only FF images were considered as test images

[3]To generate photo-realistic images without sawtooth phenomenon, we blend the background image and foreground objects together using continuous alpha values (opacity) at the boundaries.

Figure 4. Generation of a Background and Shadow frame (BSF) (bottom right), by filling the holes in the bottom left image with the content of a background image (top right).

Figure 5. An example of simulated image, with its corresponding foreground ground truth mask.

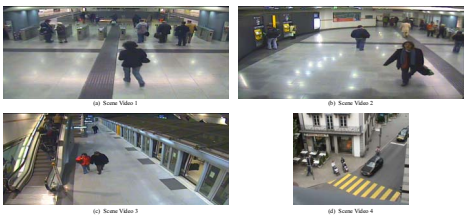

Figure 6. The four scenes considered.

(this corresponds to the 'Clean' condition). In the second case, only FSF images were used (this is the 'Shadow' condition). Finally, note that the experiments were conducted for 4 different scenes, as shown in Figure 6. The first three are real metro surveillance videos. Scene 1 and 2 contains strong shadows and reflections, while scene 3 contains a large number of moving background pixels due to the presence of the escalator. Scene video 4 is a typical outdoor surveillance video.

Parameters and performance measures: The method comprises a large number of parameters. However, most of them are not critical. Except stated otherwise, the same parameters were used for all experiments. The values were: the $\mathbf{LBP}_{6,2}$ feature was used, with $n=3$ as noise parameter; $T_{bgu}=0.2$, $w_{init}=\beta=\alpha=\alpha_w=0.01$, $\tau=5$ and $T_{bw}=0.5$ for the update parameters; $T_D=0.1$ for the texture distance; $n_c=3$, $\check{\theta}_n=3°$, $\mu=0.5$ and $\nu=1.2$ in the color distance computation. In these experiments, the bilateral filter was not used (only a gaussian smoothing with small bandwith $\sigma=1$). As performance measures, we used the $\texttt{recall}=N_c/N_{gt}$ and $\texttt{precision}=N_c/N_{det}$ measures, where N_c denotes the number of foreground pixels correctly detected, N_{gt} the number of foreground pixels in the ground-truth, and N_{det} the number of detected foreground pixels. Also, we used the `F-measure` defined as $F=2\cdot(\texttt{precision}\cdot\texttt{recall})/(\texttt{precision}+\texttt{recall})$.

Results: Figure 7 displays the different curves obtained by varying the threshold values T_{bg} (cf Subsection 3.3). We

Figure 8. Results on a metro video with a moving escalator (first row: original images; 2nd row: our method; 3rd row: MoG method).

also show the performance of a multi-scale LBP feature, $\mathbf{LBP}_{\{6,8\},\{2,4\}}$ where 14(=6+8) neighboring pixels located on two circles with radiuses 2 and 4 were compared to the central pixel. As can be seen, this does not produce obviously better results than the use of the LBP feature at the single scale. From these results, in the 'Clean' condition (Figure 7(a) and Figure 7(b)), we observe that the combination of both color and texture measures provide better results than those obtained with each of the feature taken individually. Overall, in this case, a value of λ=0.75 (with the distance threshold T_{bg}=0.2) gives the best performance. In the 'Shadow' condition (Figure 7(c) and Figure 7(d)), we can observe a performance decrease in all cases. However, the performance of the texture feature drops more, which indicates that the texture feature is not so robust when shadow or reflection exists in the scenes. Nevertheless, again, the combination of features is useful, and the best results are obtained with λ=0.25.

4.2. Real Data

For the real data, the experiments were as follows: for the first 100 frames, the parameters indicated in the simulated data were used to quickly obtain a background model. Then, the update parameters were modified according to: $w_{init}=\beta=\alpha=\alpha_w=0.001$. The parameters of the cross bilateral filter were set to σ_s=3 and σ_r=0.1 (with an intensity scale of 1), and we used λ=0.5 and T_{bg}=0.2, as a compromise between the clean and shadow simulated experiments. As a comparison, the MoG method [10] was used. We used the OpenCV implementation with default parameter as reference.

In the first experiment, a real metro surveillance video with a moving escalator was used. The foreground detection results on three typical frames are shown in Figure 8. We observe that our method provides better performance than the MoG method: not only the moving background pixels are well classified, but the foreground objects are also successfully detected. The results in the second example

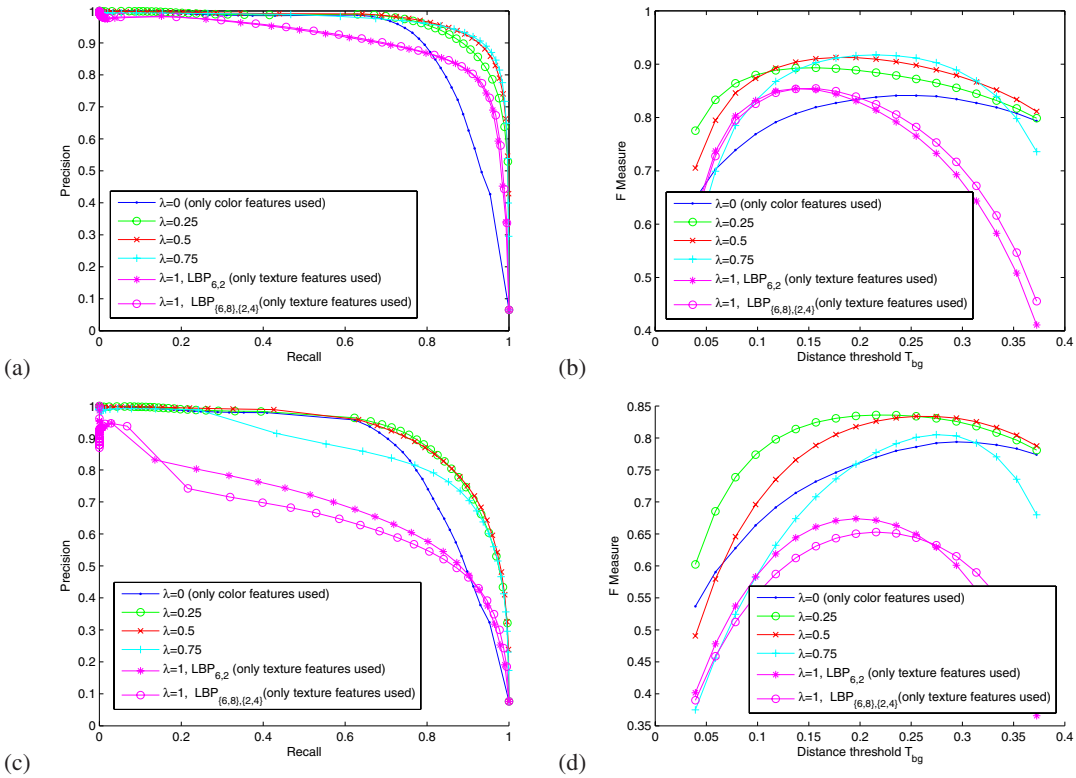

Figure 7. Precision-Recall and F-measure curves, for different values of λ, in the 'Clean' (a)-(b), and 'Shadow' (c)-(d) conditions.

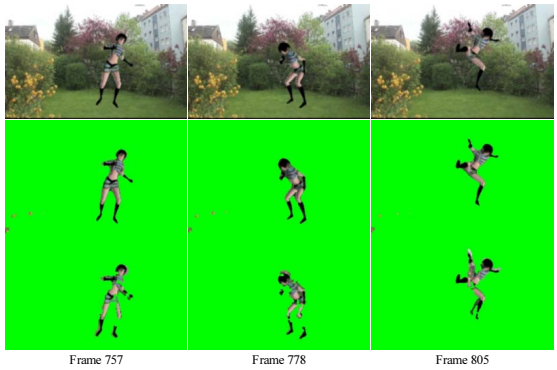

Figure 9. Results on a sythetic video with a real moving background scene and a synthetic moving people (first row:original images; 2nd row: our method; 3rd row: MoG method).

(Figure 9), where the background exhibits waving trees and flowers confirm the ability of the model to handle moving background.

In the third sequence, the video exhibits shadow and reflection components. The results (Figure 10) demonstrate that our method, though not perfect, handles this shadow better than the MoG method. The fourth sequence (Figure 11) is taken from the CAVIAR corpus. Results with both $\lambda = 0$ (only color is used) and $\lambda = 0.5$ are provided, and demonstrate the benefit of using both types of features.

In the last two experiments, we test our multi-layer scheme, which should be useful to avoid 'ghosts' produced by traditional approaches, and which should be useful for detecting left luggages for instance. The results on an outdoor camera monitoring traffic and pedestrians at a crossroad are shown in Figure 12, where a pedestrian (framed by red boxes) is waiting at a zebra crossing for a long time, and becomes part of the background before crossing the road. The MoG method produced a ghost after the pedestrian left. Thanks to the maintenance of previous background layers in our algorithm, such a ghost was not produced in our case. Another video from PETS'2006 was used for abandoned luggage detection. The results are shown in Figure 13 where a person left his luggage and went away.

5. Conclusions

A robust layer-based background subtraction method is proposed in this paper. It takes advantages of the complementarity of LBP and color features to improve the performance. While LBP features work robustly on rich texture regions, color features with an illumination invariant model produce more stable results in uniform regions. Combined with an 'hysteresis' update step and the bilateral filter (which implicitely smooths results over regions of the same intensity), our method can handle moving background pixels (e.g., waving trees and moving escalators) as well as multi-layer background scenes produced by the addition and removal of long-time stationary objects. Experiments

Figure 10. Results on a metro video with cast shadows and reflections (first row: original images; 2nd row: our results; 3rd row: MoG method).

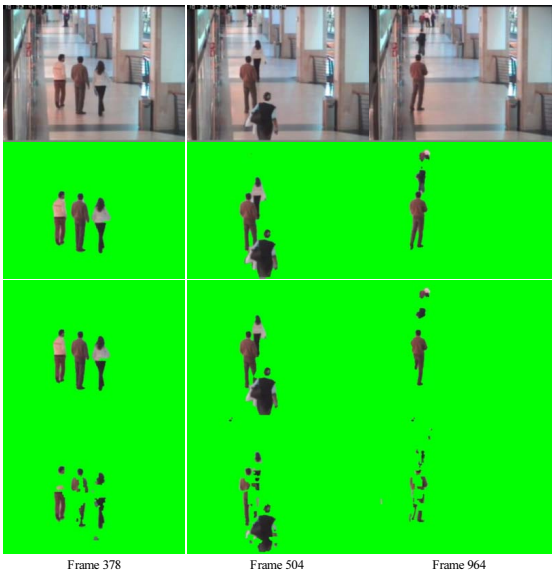

Figure 11. Results on a CAVIAR video (first row: original images; 2nd row: our method with $\lambda = 0.5$; 3rd row: our method with $\lambda = 0$; 4th row: MoG method).

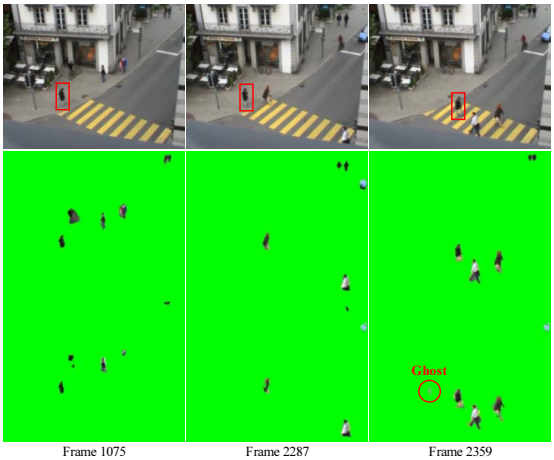

Figure 12. Results on an outdoor monitoring video (first row: original images; 2nd row: our results; 3rd row: MoG method).

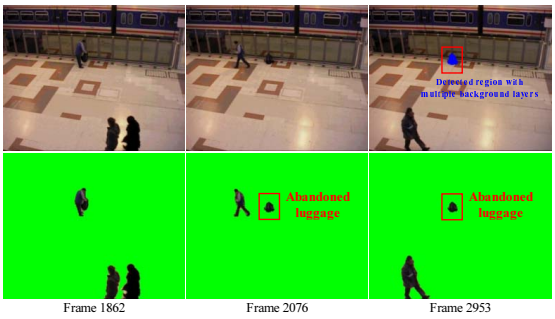

Figure 13. Left luggage detection on a PETS'2006 video (first row: original images with detected luggage covered by blue color; 2nd row: foreground detection results of our method).

on both simulated and real data with the same parameters show that our method can produce satisfactory results in a large variety of cases.

References

[1] E. Eisemann and F. Durand. Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics*, 23(2):673–678, July 2004.

[2] M. Heikkila and M. Pietikainen. A texture-based method for modeling the background and detecting moving objects. *IEEE Trans. Pattern Anal. Machine Intell.*, 28(4):657–662, April 2006.

[3] J.-S. Hu and T.-M. Su. Robust background subtraction with shadow and highlight removal for indoor surveillance. *EURASIP Journal on Advances in Signal Processing*, 14 pages, 2007.

[4] J. C. S. Jacques, C. R. Jung, and S. R. Musse. Background subtraction and shadow detection in grayscale video sequences. In *SIBGRAPI*, 2005.

[5] O. Javed, K. Shafique, and M. Shah. A hierarchical approach to robust background subtraction using color and gradient information. In *IEEE Workshop on Motion and Video Computing*, December 5-6 2002.

[6] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis. Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11(3):172–185, June 2005.

[7] D.-S. Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Trans. Pattern Anal. Machine Intell.*, 27(5):827–832, 2005.

[8] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Foreground object detection from videos containing complex background. In *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10, 2003.

[9] S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. In *ECCV*, volume 4, pages 568–580, 2006.

[10] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, volume 2, pages 246–252, 1999.

[11] O. Tuzel, F. Porikli, and P. Meer. A bayesian approach to background modeling. In *CVPR*, page 58, 2005.