EURASIP Journal on Image and Video Processing
a SpringerOpen Journal

RESEARCH Open Access

Low-complexity background subtraction based on spatial similarity

Sangwook Lee and Chulhee Lee*

Abstract

Robust detection of moving objects from video sequences is an important task in machine vision systems and applications. To detect moving objects, accurate background subtraction is essential. In real environments, due to complex and various background types, background subtraction is a challenging task. In this paper, we propose a pixel-based background subtraction method based on spatial similarity. The main difficulties of background subtraction include various background changes, shadows, and objects similar in color to background areas. In order to address these problems, we first computed the spatial similarity using the structural similarity method (SSIM). Spatial similarity is an effective way of eliminating shadows and detecting objects similar to the background areas. With spatial similarity, we roughly eliminated most background pixels such as shadows and moving background areas, while preserving objects that are similar to the background regions. Finally, the remaining pixels were classified as background pixels and foreground pixels using density estimation. Previous methods based on density estimation required high computational complexity. However, by selecting the minimum number of features and deleting most background pixels, we were able to significantly reduce the level of computational complexity. We compared our method with some existing background modeling methods. The experimental results show that the proposed method produced more accurate and stable results.

Keywords: Background subtraction; Background modeling; Structural similarity; Kernel density estimation

Introduction

As security monitoring emerges as an important issue, there has been an increasing demand for intelligent surveillance systems. Key operations in intelligent surveillance include object tracking, abnormal behavior detection, and behavior understanding. Accurate background subtraction plays an important role. The goal of background subtraction is to eliminate background components and detect meaningful moving objects. In real environments, due to various and complex background types such as moving escalators, waving tree branches, water fountains, and flickering monitors, background subtraction is a difficult task. Researchers have overcome these problems by using background modeling. Simple background models assume static background images. Background components can generally be eliminated by computing the difference between an input image and the background image that was modeled using average, low-pass filtering, and median filtering [1-4]. For instance, in [1], the median background image was used to subtract the background components. Since temporal median filtering is time-consuming, a fast algorithm utilizing the characteristics of adjacent frames was proposed [2]. Cheng et al. applied a recursive mean procedure to compute background images [3]. In [4], low-pass filtering was utilized to estimate a static background image. However, these approaches cannot handle dynamic backgrounds and are sensitive to threshold values.

In order to handle various background types, statistical approaches were introduced. Among these approaches, Gaussian modeling methods have been widely used. Initially, uni-modal distribution was used to model pixel values [5]. In [6], a background subtraction method using the HSV color space was presented based on single Gaussian modeling. A fast and stable linear discriminant approach based on uni-modal distribution and Markov random field was proposed [7]. Rambabu and Woo proposed a background subtraction method which is robust against noisy and changing illumination based on single Gaussian modeling [8]. Although these models have low complexity levels and produce satisfactory performances in controlled backgrounds, it is difficult to use them for dynamic scenes. The Gaussian mixture model (GMM) is usually used to

* Correspondence: chulhee@yonsei.ac.kr
Yonsei University, 134 Sinchon-dong, Seodaemun-gu, Seoul 120-749, Korea

model various background types. Stuffer and Grimson used the GMM for background subtraction in [9], and it is still a popular method for background subtraction [10-20]. A spatio-temporal GMM (STGMM) was proposed to handle complex background [10]. Using a GMM, a statistical framework was investigated to localize a foreground object [11] and a dynamic background was modeled for highly dynamic conditions such as active cameras and high motion activities in background regions [12]. Also, the subtraction of two Gaussian kernels (difference of Gaussians) was used to eliminate background regions in embedded platforms [13]. A general framework of regularized online classification EM for GMM was proposed [14]. Wang et al. proposed an adaptive local-patch GMM to detect moving objects in dynamic background regions [15]. In [16], a new update algorithm was proposed for learning adaptive mixture models, and Bin et al. proposed a self-adaptive moving object detection algorithm. The method improved the original GMM in order to adapt to sudden or gradual illumination changes [17]. In [18], in order to improve GMM performance, a new rate control method based on high-level feedback was developed. An improved adaptive-K GMM method was presented for updating background regions [19], and GMM was used for modeling background regions in a Bayer-pattern domain [20]. A disadvantage of these multimodal Gaussian modeling methods is that they require pre-defined parameters such as the number of the Gaussian distributions and the standard deviations of those distributions. Also, dynamic backgrounds cannot be accurately modeled by a few Gaussian distributions. In order to overcome parameter background modeling methods, nonparametric background modeling techniques have been developed for estimating background probabilities. Nonparametric background modeling methods have been used to estimate background distribution based on pixel values observed in the past. In [21], the Gaussian kernel was used for pixel-based background modeling. This nonparametric method is usually used to handle multiple modes of dynamic backgrounds without pre-defined parameters. However, these nonparametric methods use kernel density estimation (KDE), which requires heavy computational complexity and a large amount of memory. Various efforts have been made to address these problems. Using Parzen density estimation and foreground object detection, a fast estimation method was presented [22] and an automatic background modeling based on multivariate non-parametric KDE was proposed [23]. In [24], a non-parametric method was proposed for foreground and background modeling, which did not require any initialization. Han et al. proposed an efficient algorithm for recursive density approximation based on density mode propagation [25]. Also, depth information, on-line auto-regressive modeling, and Gaussian family distribution were used to eliminate background regions [26-28]. In [29], new object segmentation was proposed based on a recursive KDE. It used the mean-shift method to approximate the local maximum value of the density function. The background was modeled using real-time KDE based on online histogram learning [30].

Also, alternative approaches were proposed based on neural network techniques or the support vector machine (SVM) method [31-35]. A method was proposed based on self-organization through artificial neural networks [31]. Furthermore, a self-organization method was combined with fuzzy approach to update background [32]. In [33-35], an automatic algorithm was proposed to perform background modeling using SVM.

To develop a robust model with low complexity, we used a pixel-based background subtraction method based on spatial similarity computed using the structural similarity method (SSIM) [36]. Using spatial similarity, we measured the pixel similarity and eliminated background pixels. The remaining pixels were classified as either background or foreground pixels using KDE. Since we eliminated most background pixels and used only two features for KDE, the complexity of the proposed method was significantly reduced. The proposed method was evaluated using two datasets (Wallflower's and Li's datasets) and showed favorable performance over some existing methods.

The overall algorithm for efficient background subtraction

Preparation

The structure similarity for eliminating background components

To eliminate background components while preserving potential foreground components, we first computed the spatial similarity using the SSIM method that was developed for image quality assessment [36]. The SSIM was computed as follows:

$$\begin{aligned}
&\text{Luminance}: l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \\
&\text{Contrast}: c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \\
&\text{Structure}: s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \qquad (1) \\
&\text{SSIM}(x,y) = [l(x,y)]^{\alpha}[c(x,y)]^{\beta}[s(x,y)]^{\gamma} \\
&\qquad = \frac{\left(2\mu_x\mu_y + C_1\right)\left(2\sigma_{xy} + C_2\right)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)}
\end{aligned}$$

where α, β, and γ are parameters which determine the relative importance of $l(x, y)$, $c(x, y)$, and $s(x, y)$ and we

set α, β, and γ to 1. μ_x and μ_y are the local means, σ_x and σ_y are the local standard deviations, σ_{xy} is the local covariance coefficient between regions x and y, and we set C_3 to $C_2/2$ and C_1 and C_2 are constants that were set to 6.5025 and 58.5225 as proposed in [36]. In Equation 1, l, c, and s represent the luminance, contrast, and structure of two images. In this paper, we computed the SSIM for local regions (e.g., a 3×3 block) to eliminate background components. Figure 1a and b show the input and reference background images, respectively. Figure 1c and g show the intensity and hue difference images between Figure 1a and b, respectively. The SSIM difference image between Figure 1a and b is shown in Figure 1k. Thresholding (if a pixel value of the difference image was larger than the given threshold value, the pixel was eliminated) was applied to the difference images with various threshold values (low, medium, large), and the resulting images are shown in Figure 1d,e,f,h,i,j,l,m,n. For the intensity component (Figure 1c,d,e,f), the differences between the shadow regions and the corresponding background regions were high. The thresholding operation still left shadows when using a low threshold value (e.g., 80). When we used a larger threshold value (e.g., 120) to eliminate the shadows, potential foreground objects were also eliminated (Figure 1f).

For the hue component (Figure 1g,h,i,j), shadows were not retained, but many of the background regions contained high difference values. To eliminate these background regions, we tried using a larger threshold value (e.g., 120). However, the top portion of the person with the blue jacket and the red portion of the person on the right were also eliminated. Furthermore, the small object in the lower-left corner was almost deleted when the intensity component or the hue component was used. However, the method based on the SSIM correctly retained the object (Figure 1k,l,m,n). In the SSIM, global intensity and contrast changes were not determined as forms of distortion [36]. Therefore, the proposed method proved to be robust against shadows with lower intensity values while retaining internal structures. Furthermore, since the proposed method used the variances and covariance of two local regions, it could detect objects with similar colors. In

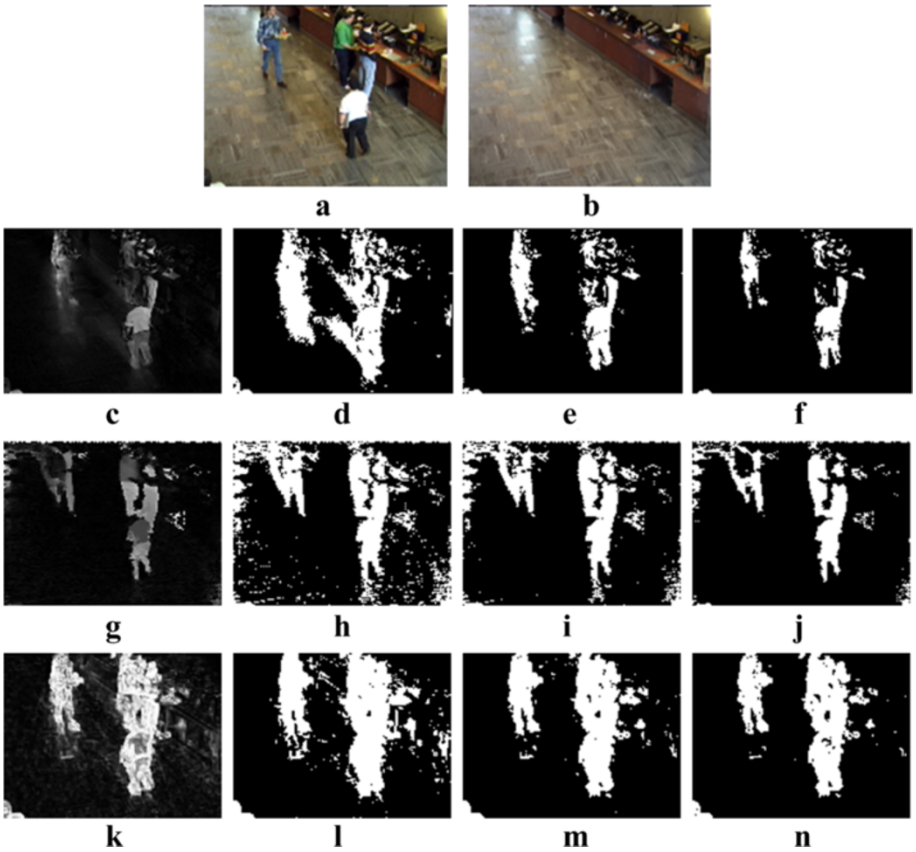

Figure 1 The characteristic of features. (a) An input image, **(b)** the reference background image, **(c)** intensity difference image between **(a)** and **(b)**, **(d)** thresholding of **(c)** with a low value (80), **(e)** thresholding of **(c)** with a middle value (100), **(f)** thresholding of **(c)** with a large value (120), **(g)** hue difference image between **(a)** and **(b)**, **(h)** thresholding of **(g)** with a low value (80), **(i)** thresholding of **(g)** with a middle value (100), **(j)** thresholding of **(g)** with a large value (120), **(k)** SSIM difference image between **(a)** and **(b)**, **(l)** thresholding of **(k)** with a low value (0.4), **(m)** thresholding of **(k)** with a middle value (0.5), and **(n)** thresholding of **(k)** with a large value (0.6).

Figure 2a, a person's head color was similar to the background regions. The proposed method showed improved performance compared to the other method [31] (http://www.na.icar.cnr.it/~maddalena.l/MODLab/SoftwareSOBS.html). Similarly, in Figure 2b, the woman's jacket color was similar to the background regions. The proposed method correctly classified the woman as a foreground object while the other method missed the jacket.

To apply the SSIM to local regions, we used a sliding window approach. For each pixel, we computed the SSIM of a 3×3 window centered at the pixel. Let $\mathbf{A}(i, j) = \lfloor A^R(i, j), A^G(i, j), A^B(i, j) \rfloor$ be a pixel in the RGB color space. Then, the similarity image (SI) between intensity images $A^I(i, j)$ and $B^I(i, j)$ was calculated as follows:

$$\mathrm{SI}_{A^I,B^I}(i,j) = \mathrm{SSIM}\left(A^I(i,j), B^I(i,j)\right) \tag{2}$$

where

$$\begin{aligned} A^I(i,j) &= \frac{1}{3}\left(A^R(i,j) + A^G(i,j) + A^B(i,j)\right), \mu_{A^I(i,j)} \\ &= \frac{1}{9}\left(\sum_{v=-1}^{1}\sum_{u=-1}^{1} A^I(i+u, j+v)\right) \end{aligned} \tag{3}$$

$$\sigma^2_{A^I(i,j)} = \frac{1}{9}\left(\sum_{v=-1}^{1}\sum_{u=-1}^{1}\left(A^I(i+u,j+v)^2 - \mu_{A^I(i,j)}\right)\right),$$

$$\sigma_{A^I(i,j)B^I(i,j)} = \frac{1}{9}\left(\sum_{v=-1}^{1}\sum_{u=-1}^{1}\left(A^I(i+u,j+v) - \mu_{A^I(i,j)}\right) \times \left(B^I(i+u,j+v) - \mu_{B^I(i,j)}\right)\right)$$

$A^I(i, j)$ represents an intensity value, $\mu_{A^I(i,j)}$ and $\mu_{B^I(i,j)}$ are intensity means, $\sigma_{A^I(i,j)}$ and $\sigma_{B^I(i,j)}$ are intensity standard deviations, and $\sigma_{A^I B^I(i,j)}$ is the intensity covariance. $SI_{A^I,B^I}(i, j)$ is close to 1 when two window regions were similar. C_1 and C_1 were set to 6.5025 and 58.5225, respectively [36]. By assuming that one image was a reference background image, we obtained a binary background image (BBI) by applying a thresholding operation:

$$\mathrm{BBI}_{A^I,B^I}(i,j) = \begin{cases} 0(\text{background}) \text{ if } \left(\mathrm{SI}_{A^I,B^I}(i,j) > T_1\right) \\ 1(\text{foreground candidate}) \quad \text{otherwise} \end{cases} \tag{4}$$

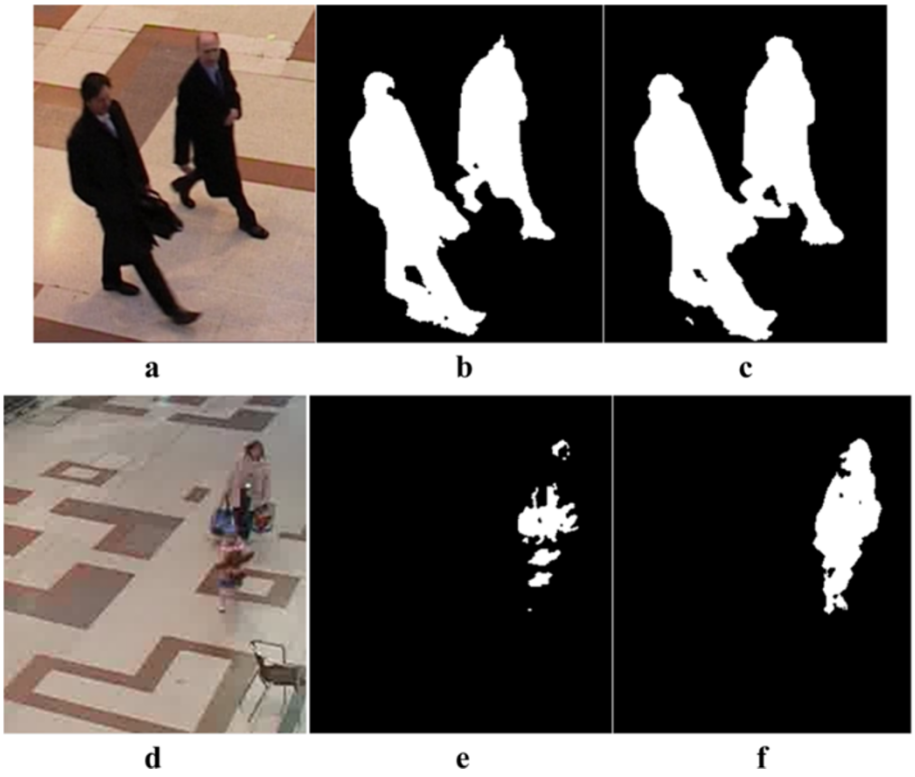

Figure 2 The effect for detecting the objects which are similar with backgrounds. (a) An input image 1, **(b)** the results of other method [31], **(c)** the results of the proposed method, **(d)** an input image 2, **(e)** the results of other method [31], and **(f)** the results of the proposed method.

T_1 is a threshold value which was empirically determined and set to 0.55. Figure 3 shows the effect of the threshold value. When we used a small value for T_1, most pixels were classified as background regions (Figure 3c). When we used a large value for T_1, most pixels were classified as foreground regions (Figure 3o). Based on this observation, we set T_1 to 0.55, though any value between 0.1 and 0.9 provided good performance.

Since we calculated the means and the variances, the computational complexity was low. However, some background pixels were still retained. In order to eliminate the background pixels, we used nonparametric kernel density estimation.

Determining foreground and background areas using KDE

Generally, KDE can model multi-modal probability distributions without requiring any prior information. It is effective for modeling the arbitrary densities of real environments. KDE was applied to each pixel of the training images. In other words, we extracted training samples at each pixel location of the training images. Let s_1, s_2, …, s_N be training samples and we used the Gaussian kernel function. Then, the probability of x_t was calculated as follows [21]:

$$p(x_t) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{1}{2\sigma}(s_i - x_t)^2} \tag{5}$$

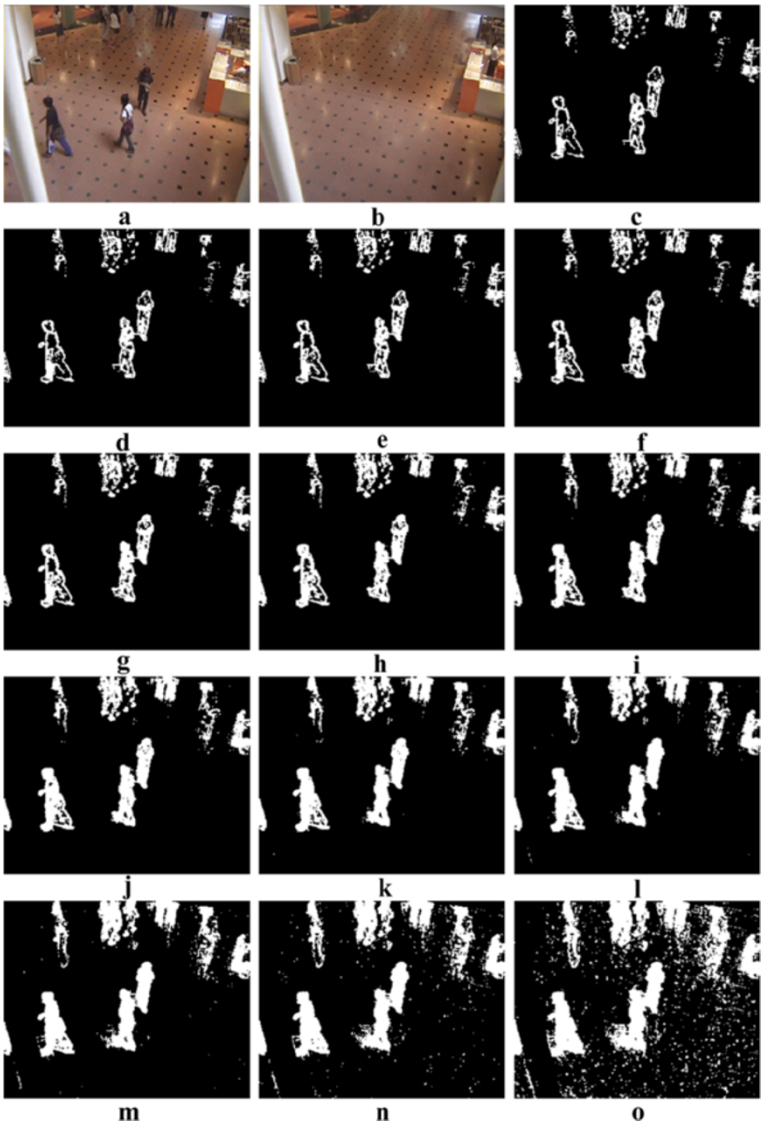

Figure 3 The effect of threshold T_1. (a) An input image, **(b)** the reference background image, **(c)** $T_1 = 0.1$, **(d)** $T_1 = 0.2$, **(e)** $T_1 = 0.3$, **(f)** $T_1 = 0.35$, **(g)** $T_1 = 0.4$, **(h)** $T_1 = 0.45$, **(i)** $T_1 = 0.50$, **(j)** $T_1 = 0.55$, **(k)** $T_1 = 0.60$, **(l)** $T_1 = 0.65$, **(m)** $T_1 = 0.70$, **(n)** $T_1 = 0.80$, and **(o)** $T_1 = 0.90$.

where σ represents the kernel function bandwidth and N is the number of training samples. A pixel was classified as a background pixel if the estimated probability was larger than the given threshold. It was observed that a large value of N produced more robust results. Consequently, a typical KDE method requires a large number of operations. On the other hand, we first eliminated most background pixels using the spatial similarity (SS) method and used only two features (one of the RGB components and one of the normalized RGB components). Also, we used a small number of samples (one hundred samples). Therefore, we were able to significantly reduce the computational complexity of the KDE without sacrificing performance. Figure 4 shows an example of the proposed method. We eliminated most background pixels using the SS method (Figure 4c). However, some background pixels were still retained and we eliminated these pixels using KDE. In this case, the candidate pixels made up 5% to 6% of the entire image. The processing time was also reduced accordingly.

Based on this observation, we propose a computationally efficient background subtraction method by eliminating background regions using spatial similarity in the spatial domain and the KDE method in the temporal domain. By combining spatial and temporal features, the proposed method produced better performance than the conventional KDE method. Figure 5 shows the comparison results. These sequences contain dynamic background regions. Tree branches were swaying and the curtain was moving in the wind. In dynamic background regions, it is difficult to accurately model the background in the conventional KDE method. Therefore, many background components are often classified as foreground components. However, since most of the background components in the proposed method were eliminated with spatial similarity, most of the background components misclassified as foreground components were correctly classified as background components.

The proposed method

Determine the background type

The reference background image (RBI) was computed as the average of the training intensity images:

$$\mathrm{RBI}^{I}(i,j) = \frac{1}{N}\sum_{t=0}^{N-1} A_t^I(i,j) \tag{6}$$

where $A_t^I(i,j)$ represents a pixel of the t-th intensity image of a video sequence and N is the number of training images, which was set to 100. In other words, the first 100 images of a given video sequence generally were used as training images. We also computed the averages of the RGB channels of the training images:

$$\mathrm{RBI}^{\Omega}(i,j) = \frac{1}{N}\sum_{t=0}^{N-1} A_t^{\Omega}(i,j) \text{ where } \Omega \in \{\mathrm{R},\ \mathrm{G},\ \mathrm{B}\} \tag{7}$$

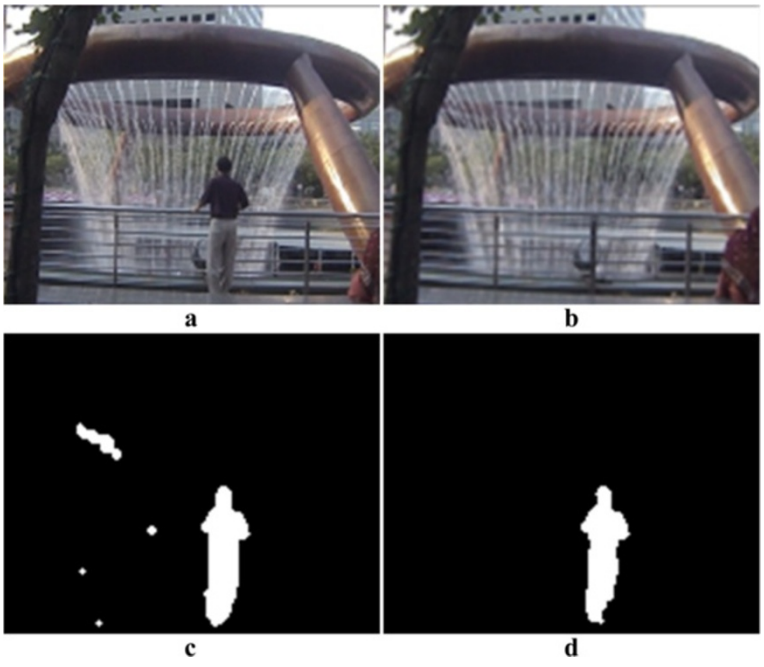

Figure 4 An example of the proposed method. (a) An input image, **(b)** the reference background image, **(c)** the similarity image, and **(d)** the final result.

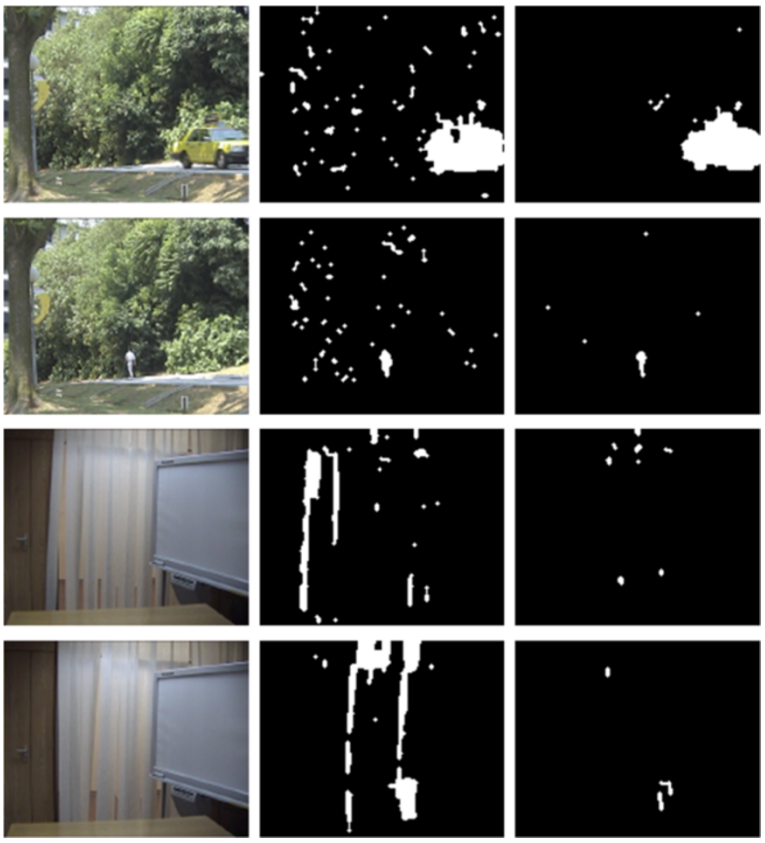

Figure 5 The results of the proposed method and the single KDE method. First column: input image; second column: results of single KDE method; and third column: the proposed method.

Then, a similarity image between the reference background and training intensity images was computed using Equation 2 and the reference binary background image (RBBI) was computed:

$$\begin{aligned}&\text{For each pixel } (i,j)\\ &r(i,j)=\frac{1}{N}\sum_{t=0}^{N-1}\mathrm{SI}_{\mathrm{RBI}^I,A_t^I}(i,j)\\ &\mathrm{RBBI}(i,j)=\begin{cases}0 \ \ (\text{static background}) & \text{if } (r(i,j)>0.8)\\ 1 \ \ (\text{moving background}) & \text{otherwise}\end{cases}\end{aligned} \tag{8}$$

The RBBI successfully detected moving background components such as moving escalators, waving tree branches, and water fountains.

Determine the foreground candidate pixels

When a new image was entered, a BBI was computed between the RBI and the input intensity image using Equations 3 to 4. If BBI(i, j) = 1, the pixel could have been either a foreground pixel or a moving background pixel. If RBBI(i, j) = 1 (moving background), we computed the difference between the intensity input image and the RBI^I. If the difference between the input intensity image and the RBI^I was small, the pixel could have been a background pixel. Also, the pixel was classified as a foreground candidate when the difference was larger than the given threshold, and the pixel was classified as a foreground candidate if BBI(i, j) = 1 and RBBI(i, j) = 0. The following procedure was used to classify a pixel:

$$\begin{aligned}&\text{For each pixel } (\mathrm{i},\mathrm{j})\\ &\quad \text{If } \left(\mathrm{BBI}_{\mathrm{RBI}^I,I_k^I}(i,j)\ =\ 1\right),\ \text{then}\\ &\qquad \text{If } (\ \mathrm{RBBI}(i,j)=\ 1),\ \text{then}\\ &\qquad\qquad \mathrm{FCI}_k(i,j)=\begin{cases}if\left(\left|\mathrm{RBI}^I(i,j)-{I_k}^I(i,j)\right|>T_2\right)\\ \ \ 1\ \ (\text{foreground candidate})\\ \text{otherwise}\\ \ \ 0\ \ (\text{background})\end{cases}\\ &\quad \text{Otherwise}\\ &\qquad \mathrm{FC}I_k(i,j)=1\ \ (\text{foreground candidate})\end{aligned} \tag{9}$$

where $\mathrm{FCI}_k(i,j)$ represents a candidate image, $I_k^I(i,j)$ represents the k-th input intensity image (see Equation 3),

and T_2 was empirically set to 30. If T_2 was too large, most pixels were classified as background pixels. In other words, many foreground pixels were misclassified as background pixels when T_2 was too large. Figure 6 shows the results for various values of T_2. Figure 6a,b shows an input image and the BBI. Figure 6c,d,e,f shows the FCI for various values of T_2. Most foreground pixels were eliminated when T_2 was set to 80 (Figure 6f), while most moving background pixels were retained when T_2 was set to 10 (Figure 6c). In order to choose an optimal threshold value, we tested the proposed method with various values of T_2 using some video sequences with dynamic background regions and chose the threshold value ($T_2 = 30$). At this point, most background regions were removed.

Subtract the background pixels using KDE

We classified only the foreground candidate pixels (i.e., $\mathrm{FCI}_k(i, j) = 1$) using KDE. Since there were high correlations among the R, G, and B components, and using all three channels produced only slight improvements, we used only the color with the largest difference. To improve performance, we also used one of the normalized RGB components that were robust against illumination changes and that represented the chrominance information well. We selected one of the RGB component channels as follows:

$$\begin{aligned} &\text{For each pixel } (i,j) \text{ of the } k\text{-th frame} \\ &\text{If } (\mathrm{FCI}_k(i,j) = 1) \\ &\quad d_{\max} = \max(\mathrm{Diff_R}, \mathrm{Diff_G}, \mathrm{Diff_B}) \end{aligned} \tag{10}$$

where

$$\begin{aligned} \mathrm{Diff_R} &= \left|\mathrm{RBI^R}(i,j) - I_k^{\mathrm{R}}(i,j)\right| \\ \mathrm{Diff_G} &= \left|\mathrm{RBI^G}(i,j) - I_k^{\mathrm{G}}(i,j)\right| \\ \mathrm{Diff_B} &= \left|\mathrm{RBI^B}(i,j) - I_k^{\mathrm{B}}(i,j)\right| \end{aligned}$$

where $d_{\max}$ represents the maximum difference. Let $\Omega_{\max}$ be the channel with the maximum difference.

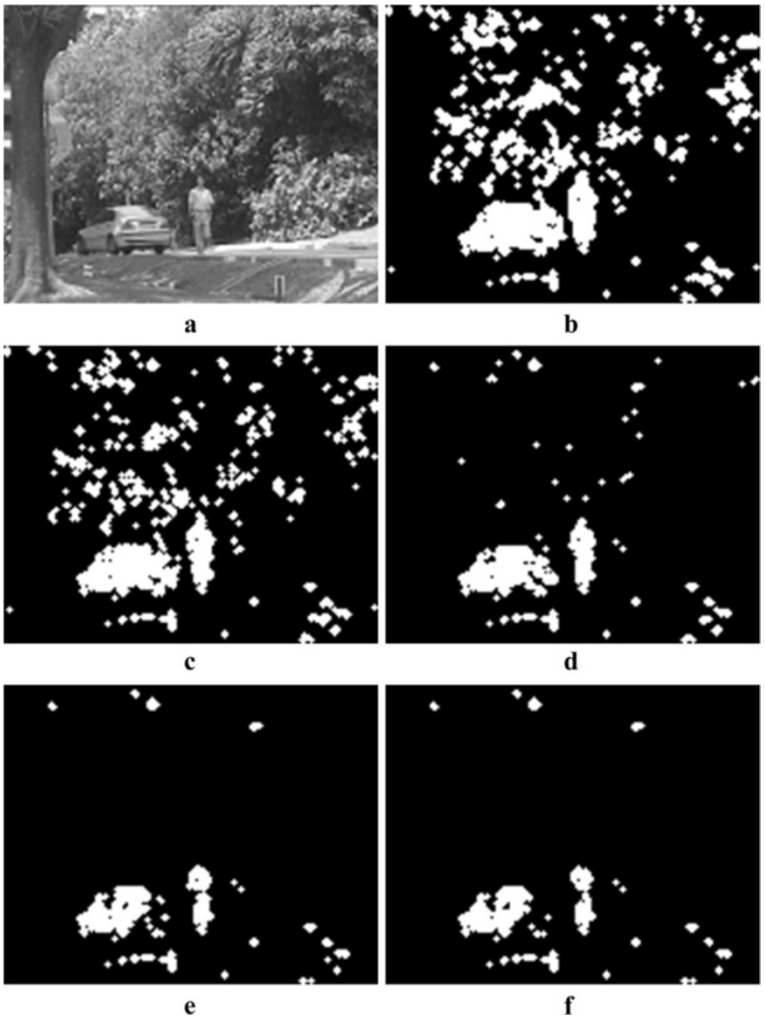

Figure 6 The results with various threshold (T_2) values. (a) An input image, **(b)** BBI, **(c)** $T_2 = 10$, **(d)** $T_2 = 30$, **(e)** $T_2 = 60$, and **(f)** $T_2 = 80$.

A foreground candidate pixel was classified as a background pixel when the estimated probability density function of the pixel value was larger than the given threshold as follows:

$$\begin{aligned}&\text{if } \frac{1}{N}\sum_{m=0}^{N-1}\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{1}{2\sigma}\left(I_k^{\Omega_{\max}}(i,j)-A_m^{\Omega_{\max}}(i,j)\right)^2} > T_3,\\ &\quad \text{decide the pixel as background}\\ &\text{otherwise,}\\ &\quad \text{decide the pixel as foreground}\end{aligned} \tag{11}$$

where σ represents the kernel width. Since the probability density function of the background pixel was unknown, we assumed that the probability densities for all intensity values were identical. Therefore, we set T_3 to 1/256. We used the standard deviation of the training images as the kernel width. This procedure was repeated using the normalized RGB color components, which were computed as follows:

$$I^{\Omega}_{\text{normalized}}(i,j) = \frac{255\cdot I^{\Omega}(i,j)}{I^{\mathrm{R}}(i,j)+I^{\mathrm{G}}(i,j)+I^{\mathrm{B}}(i,j)} \qquad \text{with } \Omega \in \{\mathrm{R},\ \mathrm{G},\ \mathrm{B}\} \tag{12}$$

where $I(i, j)$ represents the input image. If either the estimated probability density function of the pixel using the original RGB channels or the estimated probability density function of the pixel value of the normalized RGB channels was classified as a foreground component, the pixel was determined to be a foreground component. After this procedure, there were several small holes inside the foreground regions and some noise elements in the background regions. Most pixel-based methods suffer from this kind of problem. In order to address this, we applied a morphological operation to remove the small holes and noise elements. In particular, we used erosion followed by dilation and then a region filling technique was applied to the results [37].

Updating

After the decision procedure, the RBI and the pixels of the training images had to be updated to adapt to the changing background areas. We used a simple IIR filter to update the RBI as follows [38]:

$$\begin{aligned}&\text{If pixel}(i,j) \text{ is classified as background,}\\ &\quad \mathrm{RBI}^{\Omega}(i,j) = (1-\alpha)\mathrm{RBI}^{\Omega}(i,j) + \alpha I_k^{\Omega}(i,j)\\ &\quad \text{where} \quad \Omega \in \{\mathrm{R},\ \mathrm{G},\ \mathrm{B}\}\end{aligned} \tag{13}$$

where α represents the learning rate and was set to 0.01. The training images were updated by replacing the oldest pixel with the new background pixel. There is a trade-off in the choice of α. If a value for α was large, the RBI quickly reflected background changes. Figures 7 and 8 show the RBI changes for various learning rate values. As can be seen in Figure 7, the RBI was affected by shadows when we used a large value for α. Figure 7a, b shows the 372nd input and the initial RBI images. Figure 7c shows the RBI image when α was 0.6. Because of a large value for α, the RBI was quickly affected by the shadows. If we used a small value for α, the RBI did not quickly reflect background changes.

In some test sequences, the background gradually became brighter over a period (Figure 8). The RBI did not reflect this gradual background change with a small value of α (Figure 8c). Thus, we set $\alpha = 0.01$, and the learning rate was able to handle background changes adequately (Figure 8d).

If sudden background changes occurred, the results may have been erroneous. In order to handle such sudden background changes, we calculated the image intensity difference between the input image and the RBI and determined that sudden background changes occurred if the difference was larger than the given threshold:

$$\begin{aligned}&\text{if}\left(\frac{1}{N_x\cdot N_y}\sum_{i=0,j=0}^{i=N_x-1,j=N_y-1}\left|I_k^I(i,j)-\mathrm{RBI}^I(i,j)\right| > 30\right),\\ &\quad \text{a sudden background change occurs}\\ &\quad \text{at the } k\text{-th sequence.}\end{aligned} \tag{14}$$

When a sudden background change was detected at the k-th image, we calculated the image differences between the previous 100 images (from the (k-99)-th image to the k-th image) and the RBI. We selected the previous images that had larger frame differences than the threshold. The selected images were temporarily used as the training images. If the number of selected images was smaller than 15, all the pixels of the k-th image were classified as background components. However, the RBI was not updated when sudden changes were detected.

Figure 9 shows an example of the proposed background subtraction procedure. Figure 9a is an input image, and Figure 9b shows the reference background image. Figure 9c is the reference binary background image where the white areas represent moving backgrounds (the waving trees). Figure 9d shows the binary background image between Figure 9a and b. Figure 9e shows the foreground candidate image. Figure 9f shows the result obtained using the original RGB components, and Figure 9g shows the final result using the normalized RGB components and the morphological operation.

Experimental results

Experiments were performed using two datasets (Li's dataset and the Wallflower's dataset). Li's dataset contained several dynamic background video sequences (water surface (WS), campus (CAM), fountain (FT), and meeting

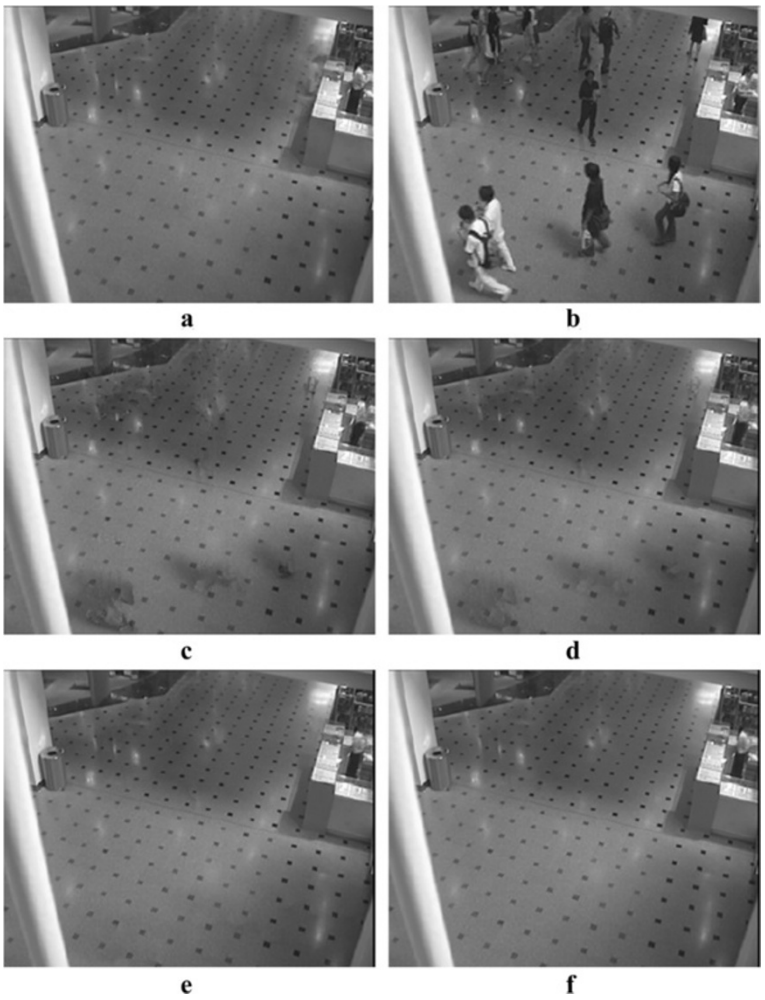

Figure 7 The RBI images at different values of the learning rate 1. (a) Initial RBI$^{\prime}$, **(b)** 372th input image, **(c)** RBI$^{\prime}$ with the $\alpha = 0.6$, **(d)** RBI$^{\prime}$ with the $\alpha = 0.3$, **(e)** RBI$^{\prime}$ with the $\alpha = 0.05$, and **(f)** RBI$^{\prime}$ with the $\alpha = 0.01$.

room (MR)) and static background video sequences (shopping center (SC), subway station (SS), airport (AP), lobby (LB), bootstrap (B)). The Wallflower's dataset contained various background types (bootstrap (B), camouflage (C), foreground aperture (FA), light switch (LS), moved object (MO), time of day (TD), and waving tree (WT)).

First, we measured the processing time of the proposed method. The proposed method took about 0.015 s per 10,000 pixels, while the processing time of a conventional method [38] was about 1.475 s per 10,000 pixels (using a 2.8-GHz Pentium IV with 1 GB of RAM) when the number of sample images was 100. For instance, the proposed method processed 66.7 frames of video per second when working with 160×128 video sequences. The complexity of KDE is $O_{\mathrm{KDE}}(MN)$ evaluations (the kernel function, multiplications and additions), assuming N image pixels and M sample points (N pixels per image and M training images). In the proposed method, we applied 'spatial similarity' to eliminate potential background pixels using a window processing operation (size of window = w). The computational complexity for calculating spatial similarity is $O_{\mathrm{similarity}}(w^2N)$ operations (multiplications and additions). Then, the remaining pixels (the number of remaining pixels: $K = \tau N$) are further processed using KDE ($O_{\mathrm{KDE}}(KM)$). Therefore, the computational complexity of the proposed method is calculated as follows:

$$\begin{aligned}\text{Number of operation} &= O_{\mathrm{similarity}}(w^2N) + O_{\mathrm{KDE}}(KM) \\ &= O_{\mathrm{similarity}}(w^2N) + O_{\mathrm{KDE}}(\tau NM)\end{aligned} \tag{15}$$

In the proposed method, the window size is 3 ($w = 3$), and the average remaining pixels were about 5% ~ 6% of the entire image pixels ($\tau \cong 0.05$). In other words, the KDE operation was reduced by approximately 95%. Although we needed to compute additional spatial similarity, it had a minor effect on the overall complexity. With 100 training images, the computational complexity for KDE and

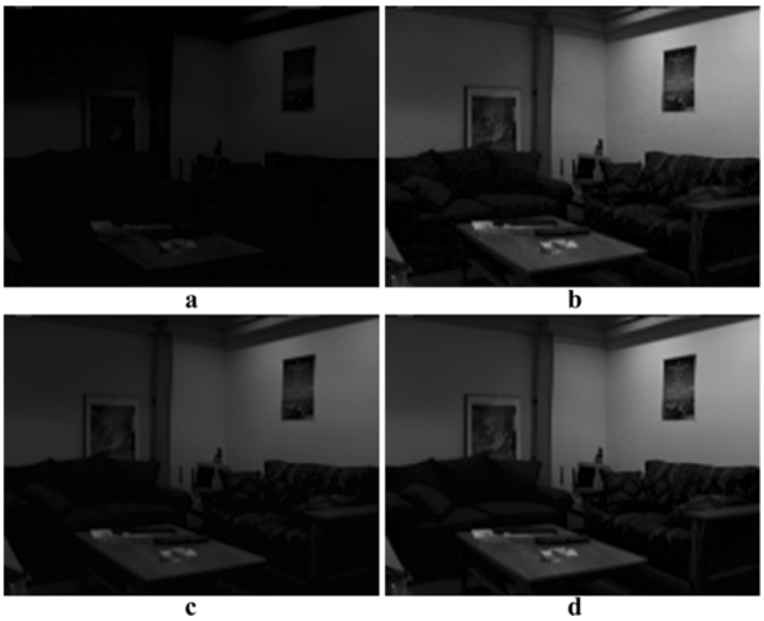

Figure 8 The RBI images at different values of the learning rate 2. (a) The first input image, **(b)** the 1,386th input image, **(c)** the 1,386th RBI^{t} with the $\alpha = 0.001$, and **(d)** the 1,386th RBI^{t} with the $\alpha = 0.01$.

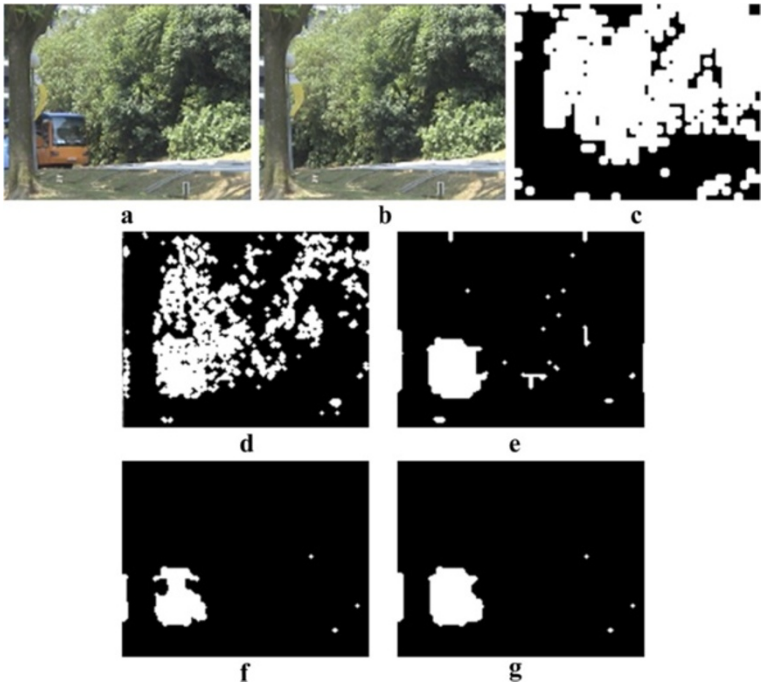

Figure 9 An example of the overall procedure of the proposed method. (a) An input image, **(b)** RBI, **(c)** RBBI, **(d)** the BBI between **(a)** and **(b)**, **(e)** the foreground candidate image, **(f)** the result obtained using the original RGB components, and **(g)** the final result.

the proposed method was $O(100\,N)$ and $O((9 + 0.05 \times 100)\,N) = O(14\,N)$, respectively. In this case, the complexity of the proposed method was about 14% of KDE.

Next, the proposed method was compared with some existing algorithms [31,38-40]. The Jaccard similarity was used as a performance measure [41]:

$$\mathrm{JS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{16}$$

where TP represents the number of true positive pixels, FP represents the number of false positive pixels, and FN represents the number of false negative pixels. Generally, a higher Jaccard similarity index indicates better performance.

Results using Li's dataset

Table 1 shows a performance comparison with Li's dataset based on Jaccard similarity.

Figure 10 shows the background subtraction results of the proposed method and Li's method using Li's dataset. The first column shows a test image, the second column shows the ground truth data of the test image, the third column shows the results of Li's method, and the fourth column shows the results of the proposed method. Using spatial similarity, the proposed method was robust against shadows. Noticeable improvements were observed in the SC, LB, B, and AP sequences which contained significant shadows. For these sequences, the proposed method showed about 8.4% ~ 14.9%, 7.8% ~ 20.6%, and 2.91% ~ 12.7% improvement compared to SOBS, Li's method, and Park's method, respectively, in terms of the Jaccard similarity. Since the proposed method used covariance, the variances of two local regions, and the normalized RGB color components, it was able to detect some objects that were similar to the background intensity. Therefore, in the WS and the FT sequences that contained objects whose intensity values were similar to the background regions, the proposed method showed improved performance compared to the other methods. For instance, a main difficulty of the WS sequence was detecting a person's leg when the intensity value of the leg was similar to the background intensity value. The other methods missed parts of the leg while the proposed method accurately detected the leg. For this WS sequence, the proposed method showed about 10.4%, 7.8%, and 2.91% improvements compared to SOBS, Li's method, and Park's method. A main difficulty of the FT sequence was that a person's pants color was similar to the background region when the person stood against the fountain. For the FT sequence, the Jaccard similarity of the proposed method was 0.820, and the proposed method showed about 16.5%, 14.6%, and 10.3% improvements compared to SOBS, Li's method, and Park's method, respectively. However, some sequences (e.g., CAM, SS, and MR) contained complex dynamic background sequences. For instance, in the CAM sequence, the background included tree branches that were constantly swayed by a strong wind. The SS sequence contained moving escalators and the MR sequence contained moving curtains). In these kinds of dynamic background sequences, Park's method (in CAM, SS, and MR) and Li's method (in MR) performed slightly better than the proposed method.

Table 1 Performance comparison with Jaccard similarity (Li's dataset)

Jaccard similarity	Proposed method	SOBS [31]	Li [39]	Park [38]
WS	0.929	0.825	0.851	0.8999
FT	0.820	0.655	0.674	0.7917
SC	0.752	0.668	0.645	0.6485
CAM	0.791	0.696	0.683	0.7935
LB	0.798	0.649	0.706	0.6706
SS	0.645	0.577	0.534	0.6826
B	0.723	0.602	0.564	0.6483
AP	0.714	0.594	0.508	0.6774
MR	0.852	0.817	0.911	0.8994
Average	0.780	0.676	0.675	0.746

Results using Wallflower's dataset

Table 2 shows a performance comparison with Wallflower's dataset based on FP + FN. Figure 11 shows the results of the proposed method and Wallflower method using the Wallflower's dataset. The first column shows a test image, the second column shows the ground truth data of the test image, the third column shows the results of Wallflower method, and the fourth column shows the results of the proposed method. The proposed method showed noticeable improvements for the C and B sequences. In the B sequence, the proposed method successfully detected objects that were similar to the background areas. On the other hand, since some moving trees of the WT sequence were classified as foreground components, the proposed method was not as good as Park's method. The LS sequence contained a sudden background change and the proposed method showed better performance. In the MO sequence, the proposed method classified the relocated objects (the chair and the phone) as foreground components. To handle this kind of problem, higher level processing such as that used in the Wallflower method might be required. The proposed method missed an object whose color was similar to that of the background area in the TD sequence.

The effects of thresholds

Next, we investigated the effects of thresholds (T_1 and T_2 in Equations 8 to 9). Figure 12 shows the Jaccard similarity of the proposed method as the T_1 and T_2 values increased with Li's dataset and wallflower's dataset. In order to

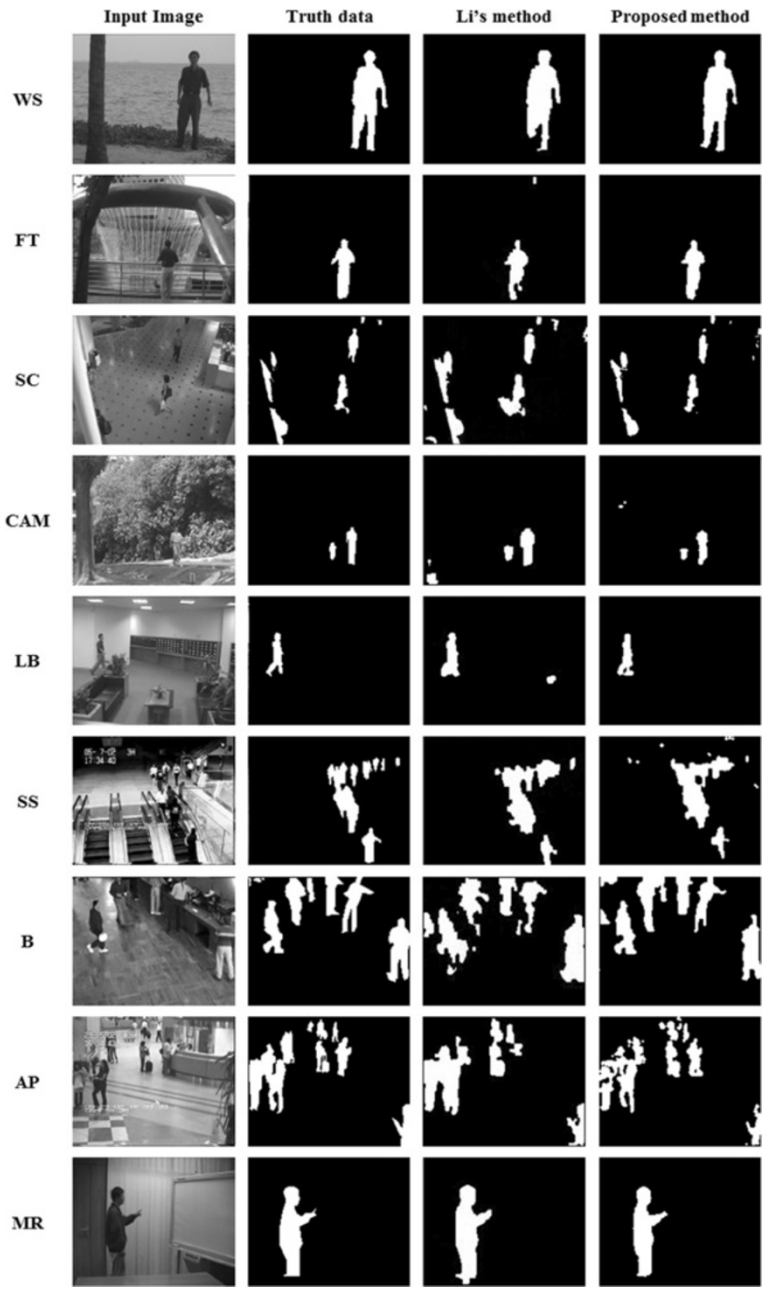

Figure 10 Background subtraction results of the proposed method and Li's dataset.

analyze the effect of T_1, we computed the false positive ratio (FPR) and false negative ratio (FNR) metrics as follows:

$$\begin{aligned} \mathrm{FPR} &= \frac{\mathrm{FP}}{\mathrm{TP}+\mathrm{FP}+\mathrm{FN}} \\ \mathrm{FNR} &= \frac{\mathrm{FN}}{\mathrm{TP}+\mathrm{FP}+\mathrm{FN}} \end{aligned} \tag{17}$$

When we used a large value for T_1, most foreground pixels were correctly classified as foreground pixels. However, many background pixels also were classified as foreground pixels. Therefore, FPR increased and FNR decreased. When we used a small value for T_1, most background pixels were classified as background pixels. However, many foreground pixels were classified as background pixels. Therefore, FPR decreased and FNR increased when we used a small value for T_1. Figure 13 shows the Jaccard similarity, and the FPR and FNR metrics with various values for T_1 (T_2 was fixed and set at 30).

We selected the optimal value for T_1 and T_2. When we set T_1 and T_2 to 0.55 and 30 respectively, the foreground candidate pixels were about 5% of the entire

Table 2 Performance comparison with the number of false positive and false negative pixels (Wallflower's dataset)

FP + FN	Proposed method	Wallflower [40]	Park [38]
C	325	2,395	1,492
WT	487	2,876	249
LS	1,140	1,322	2,260
MO	1,263	0	1,423
TD	685	986	306
FA	2,105	969	2,743
B	883	2,390	1,643
Sum	6,888	11,478	10,116

number of pixels, and the Jaccard similarity of the proposed method was about 0.78 with Li's dataset and the FP and FN numbers were about 6,888 with Wallflower's dataset. Experiments with various values of T_1 and T_2 show that the proposed method produced stable performance when the value of T_1 was from 0.5 to 0.65 and the value of T_2 was from 25 to 35.

Conclusions

In this paper, we proposed a background subtraction method that utilized structural similarity, which was robust against various background areas. The proposed method also significantly reduced the level of computational complexity since most pixels were eliminated using the similarity image. We tested the proposed method with two datasets and then compared the proposed method with some existing methods. The experimental results demonstrated that the proposed method

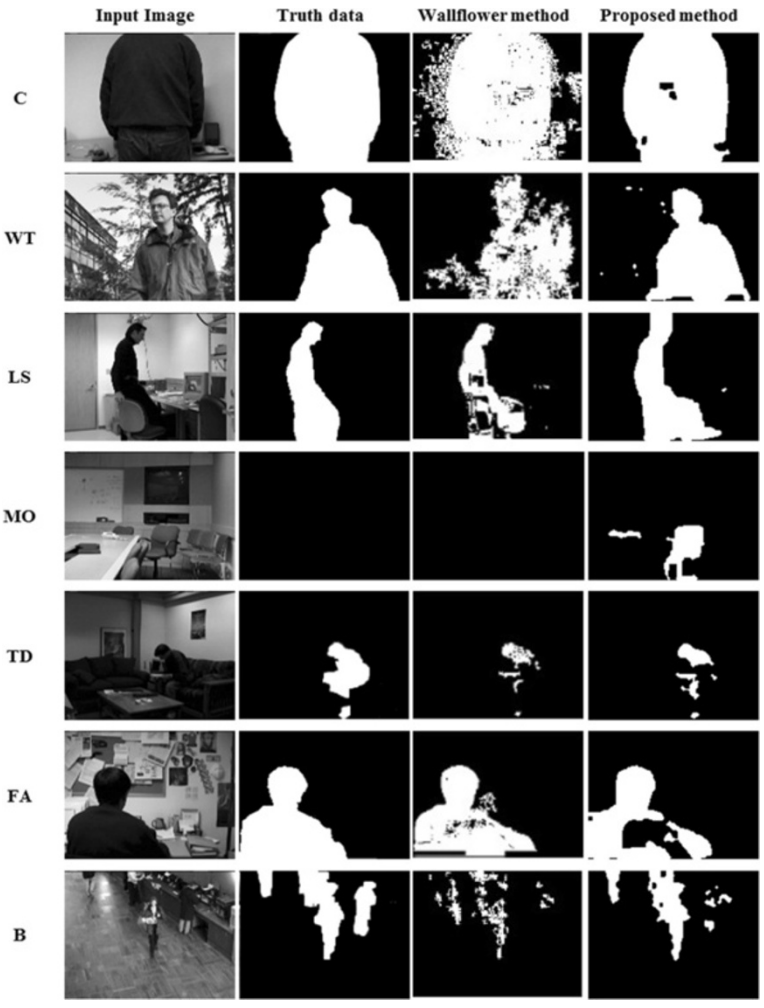

Figure 11 Background subtraction results of the proposed method and the Wallflower method using the Wallflower's dataset.

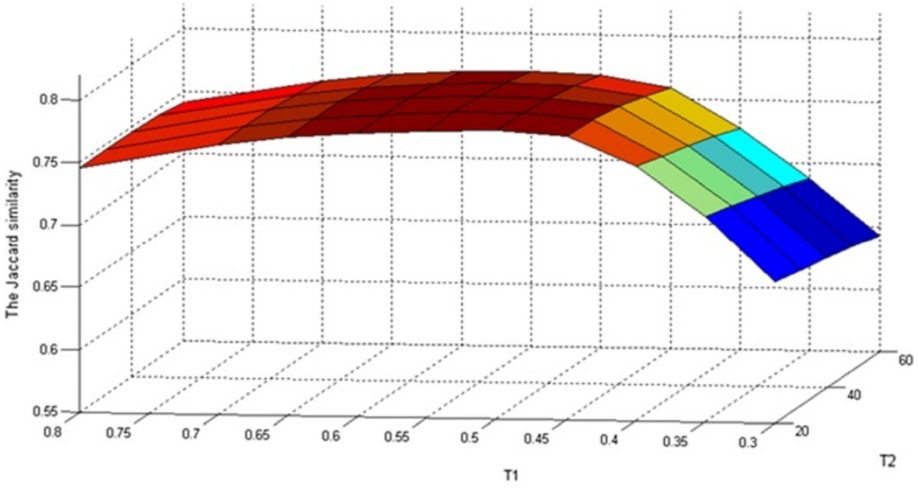

Figure 12 The effects of thresholds (T_1 and T_2).

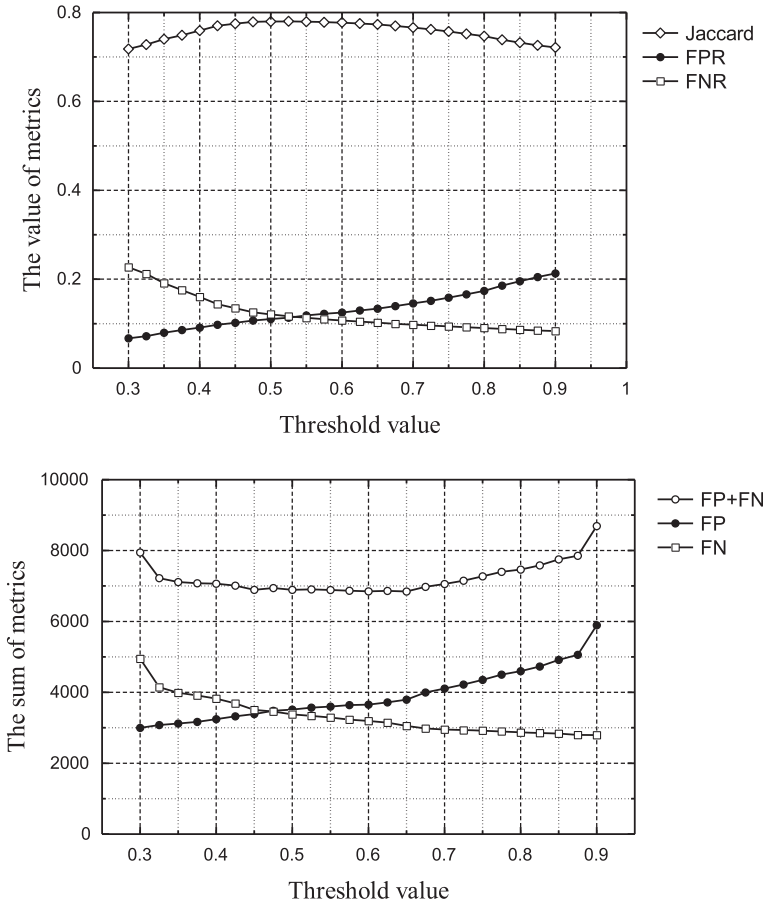

Figure 13 The evaluation metrics with various T_1 values.

was effective for various background scenes and compared favorably with some existing algorithms.

Competing interests
The authors declare that they have no competing interests.

Authors' information
Sangwook Lee received the BS and MS degrees in electrical and electronic engineering from Yonsei University, Seoul, Repiblic of Korea in 2004 and 2006, respectively. He is currently working toward the PhD degree from Yonsei University and a senior engineer at Samsung Electronics Co. Ltd., Republic of Korea. His research interests include machine vision, image/signal processing, and video quality measurement.
Chulhee Lee received the BS and MS degrees in electronic engineering from Seoul National University in 1984 and 1986, respectively, and a PhD degree in electrical engineering from Purdue University, West Lafayette, Indiana, in 1992. In 1996, he joined the faculty of the Department of Electrical and Computer Engineering, Yonsei University, Seoul, Republic of Korea. His research interests include image/signal processing, pattern cognition, and neural networks.

Acknowledgements
This work was supported by grant no. R01-2006-000-11223-0 from the Basic Research Program of the Korea Science & Engineering Foundation.

Received: 3 July 2013 Accepted: 2 June 2014
Published: 19 June 2014

References
1. NJB McFarlane, CP Schofield, Segmentation and tracking of piglets in images. Mach. Vision App. **8**(1), 187–193 (1995)
2. MH Hung, CH Hsieh, Speed up temporal median filter for background subtraction, in *Proceedings of the PCSPA, vol. 1* (Harbin, 2004), pp. 297–300
3. F Cheng, S Huang, S Ruan, Advanced motion detection for intelligent video surveillance systems, in *Proceedings of the ACM SAC, vol. 1* (984, Sierra, 2010), pp. 983–984
4. S Cohen, Background estimation as a labeling problem, in *Proceedings of ICCV, vol. 2* (Beijing, 2005), pp. 1034–1041
5. C Wren, A Azarbayejani, T Darrell, A Pentland, Pfinder: Real-time tracking of the human body. IEEE Trans. Pattern Anal. Mach. **19**(7), 780–785 (1997)
6. M Zhao, J Bu, C Chen, Robust background subtraction in HSV color space, in *Proceedings of SPIE MSAV, vol. 1* (Boston, 2002), pp. 325–332
7. X Pan, Y Wu, GSM-MRF based classification approach for real-time moving object detection. J. Zhejiang Univ. Sci. A **9**(2), 250–255 (2008)
8. C Rambabu, W Woo, Robust and accurate segmentation of moving objects in real-time video, in *Proceedings of International Symposium on Ubiquitous VR, vol. 191* (Yanji City, 2006), pp. 65–69
9. C Stauffer, E Grimson, Adaptive background mixture models for real-time tracking, in *Proceedings of IEEE Conf. Computer Vision Patt. Recog, vol. 2* (Fort Collins, 1999), pp. 246–252
10. W Zhang, X Fang, X Yang, Q Wu, Spatiotemporal Gaussian mixture model to detect moving objects in dynamic scenes. J. Electron. Imaging **16**(2), 023013-1–023013-6 (2007)
11. T Su, J Hu, Background removal in vision servo system using Gaussian mixture model framework, in *Proceedings of ICNSC, vol. 1* (Singapore, 2004), pp. 70–75
12. A Doulamis, Dynamic background modeling for a safe road design, in *Proceedings of PETRA, vol. 1* (Samos, 2010), pp. 1–9
13. MH Khan, I Kypraios, U Khan, A robust background subtraction algorithm for motion based video scene segmentation in embedded platforms, in *Proceedings of FIT, vol. 1* (Abbottabad, 2009), pp. 1–8
14. H Wang, P Miller, Regularized online mixture of Gaussians for background with shadow removal, in *Proceedings of AVSS, vol. 1* (Klagenfurt, 2011), pp. 249–254
15. SC Wang, TF Su, SH Lai, Detection of moving objects from dynamic background with shadow remova, in *Proceedings of ICASSP, vol. 1* (Prague, 2011), p. 925
16. L Zhao, X He, daptive Gaussian mixture learning for moving object detection, in *Proceedings of IC-BNMT, vol. 1* (Beijing, 2010), pp. 1176–1180
17. Z Bin, Y Liu, Robust moving object detection and shadow removing based on improved Gaussian model and gradient information, in *Proceedings of ICMT2010, vol. 1* (Ningbo, 2010), pp. 1–5
18. HH Lim, JH Chuang, TL Liu, Regularized background adaptation: a novel learning rate control scheme for Gaussian mixture modeling. IEEE Trans. Image Process. **20**(3), 822–836 (2011)
19. H Zhou, X Zhang, Y Gao, P Yu, Video background subtraction using improved adaptive-K Gaussian mixture model, in *Proceedings of ICACTE, vol. 5* (Chengdu, 2010), pp. 363–366
20. J Suhr, H Jung, G Li, J Kim, Mixture of Gaussians-based background subtraction for Bayer-pattern image sequences. IEEE Trans. Circuits Syst. Video Technol. **21**(3), 365–370 (2011)
21. A Elgammal, D Harwood, L Davis, Non-parametric model for background subtraction, in *Proceedings of ECCV, vol. 1* (Dublin, 2000), pp. 751–767
22. T Tanaka, A Shimada, D Arita, R Taniguchi, A fast algorithm for adaptive background model construction using Parzen density estimation, in *Proceedings of IEEE Conf. AVSS, vol. 1* (London, 2007), pp. 528–553
23. A Tavakkoli, M Nicolescu, G Bebis, Automatic robust background modeling using multivariate non-parametric kernel density estimation for visual surveillance, in *Proceedings of the International Symposium of Advances in Visual Computing LNCS, vol. 1* (Nevada, 2005), pp. 363–370
24. N Martel-Brisson, A Zaccarin, Unsupervised approach for building non-parametric background and foreground models of scenes with significant foreground activity, in *Proceedings of VNBA, vol. 1* (Vancouver, 2008), pp. 93–100
25. B Han, DCY Zhu, L Davis, Sequential kernel density approximation through mode propagation: applications to background modeling, in *Proceedings of ACCV, vol. 1* (Jeju, 2004), pp. 1–6
26. G Gordon, T Darrell, M Harville, J Woodfill, Background estimation and removal based on range and color, in *Proceedings of CVPR, vol. 1* (Fort Collins, 1999), pp. 2459–2464
27. A Monnet, A Mittal, N Paragios, V Ramesh, Background modeling and subtraction of dynamic scenes, in *Proceedings of ICCV, vol. 2* (Beijing, 2003), pp. 1–8
28. H Kim, R Sakamoto, I Kitahara, T Toriyama, K Kogure, Robust foreground extraction technique using Gaussian family model and multiple thresholds, in *Proceedings of ACCV, vol. 1* (Tokyo, 2007), pp. 758–768
29. Q Zhu, G Liu, Z Wang, H Chen, Y Xie, A novel video object segmentation based on recursive kernel density estimation, in *Proceedings of ICINFA, vol. 1* (Shenzhen, 2011), pp. 843–846
30. A Kolawole, A Tavakkoli, Robust foreground detection in videos using adaptive color histogram thresholding and shadow removal, in *Proceedings of ISVC, vol. 2* (Las Vegas, 2011), pp. 496–505
31. L Maddalena, A Petrosino, A self-organizing approach to background subtraction for visual surveillance applications. IEEE Trans. Image Process. **13**(4), 1168–1177 (2008)
32. L Maddalena, A Petrosino, Self organizing and fuzzy modelling for parked vehicles detection, in *Proceeding of ACVIS, vol. 1* (Bordeaux, 2009), pp. 422–433
33. H Lin, T Liu, J Chuang, A probabilistic SVM approach for background scene initialization, in *Proceedings of ICIP, vol. 3* (Rochester, 2002), pp. 893–896
34. L Cheng, M Gong, D Schuurmans, T Caelli, Real-time discriminative background subtraction. IEEE Trans. Image Process. **20**(5), 1401–1414 (2011)
35. I Junejo, A Bhutta, H Foroosh, Dynamic scene modeling for object detection using single-class SVM, in *Proceeding of International Conference on Image Processing, vol. 1* (Hong Kong, 2010), pp. 1541–1544
36. Z Wang, AC Bovik, HR Sheikh, EP Simoncelli, Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 1–14 (2004)
37. R Gonzalez, R Woods, *Digital Image Processing*, 2nd edn. (Prentice Hall, Englewood Cliffs, 2002)
38. JG Park, C Lee, Bayesian rule-based complex background modeling and foreground detection. Opt. Eng. **49**(2), 027006-1–027006-11 (2010)
39. L Li, W Huang, IYH Gu, Q Tian, Statistical modeling of complex backgrounds for foreground object detection. IEEE Trans. Image Process. **13**(1), 1459–1472 (2004)
40. K Toyama, L Krumm, B Brumitt, B Meyers, Wallflower: principles and practice of background maintenance, in *Proceedings of IEEE ICCV, vol. 1* (Kerkyra, 1999), pp. 255–261
41. P Jaccard, The distribution of flora in the alpine zone. New Phytol. **11**(2), 37–50 (1912)

doi:10.1186/1687-5281-2014-30
Cite this article as: Lee and Lee: **Low-complexity background subtraction based on spatial similarity.** *EURASIP Journal on Image and Video Processing* 2014 **2014**:30.