Radial Basis Function Based Neural Network for Motion Detection in Dynamic Scenes

Shih-Chia Huang and Ben-Hsiang Do

***Abstract*—Motion detection, the process which segments moving objects in video streams, is the first critical process and plays an important role in video surveillance systems. Dynamic scenes are commonly encountered in both indoor and outdoor situations and contain objects such as swaying trees, spouting fountains, rippling water, moving curtains, and so on. However, complete and accurate motion detection in dynamic scenes is often a challenging task. This paper presents a novel motion detection approach based on radial basis function artificial neural networks to accurately detect moving objects not only in dynamic scenes but also in static scenes. The proposed method involves two important modules: a multibackground generation module and a moving object detection module. The multibackground generation module effectively generates a flexible probabilistic model through an unsupervised learning process to fulfill the property of either dynamic background or static background. Next, the moving object detection module achieves complete and accurate detection of moving objects by only processing blocks that are highly likely to contain moving objects. This is accomplished by two procedures: the block alarm procedure and the object extraction procedure. The detection results of our method were evaluated by qualitative and quantitative comparisons with other state-of-the-art methods based on a wide range of natural video sequences. The overall results show that the proposed method substantially outperforms existing methods with Similarity and F_1 accuracy rates of 69.37% and 65.50%, respectively.**

***Index Terms*—Dynamic background, motion detection, neural network, video surveillance.**

I. Introduction

VIDEO surveillance systems are required to facilitate a wide range of applications in computer vision, including human activity understanding [1], [2], traffic monitoring and analysis [3], [4], endangered species conservation, and so on [5]–[12]. Many functions are involved in video surveillance systems. These include, but are not restricted to, motion detection, object classification, tracking, behavior recognition, and activity analysis. Motion detection, which is the segmentation of moving objects in video streams, is the first relevant step and plays an important role in video surveillance systems.

Numerous approaches have been proposed to achieve complete and accurate motion detection [13]–[39]. The three major categories of conventional motion detection approaches are optical flow, temporal difference, and background subtraction [13]. Optical flow [14], [15] can achieve robust detection by projecting motion on the image plane with proper approximation. However, it is very sensitive to noise and not computationally affordable for real-time applications. Temporal difference [16]–[18] detects moving objects by calculating the difference between consecutive frames and can effectively accommodate environmental changes. Nevertheless, the extracted shapes of moving objects are generally incomplete, especially when the moving objects in a scene are stationary or exhibit slow motion. Background subtraction [19]–[39] is a particularly popular method for motion detection. It detects moving objects by subtracting the current image from the reference background model of the previous image. This mode of detection has attracted the most attention due to the ability of this approach to extract moving objects while exhibiting only moderate time complexity. However, motion detection by background subtraction usually results in incomplete detection results of moving objects due to faulty background model generation. This can be especially aggravating when detecting moving objects in dynamic scenes [29]–[39].

Use of the Gaussian mixture model (GMM) has been proposed to model dynamic backgrounds for detecting moving objects [29]. It is the most widely used approach for motion detection applied to dynamic scenes. This approach models each pixel independently with a mixture of Gaussians and is updated by an online approximation. Several methods have been proposed to improve the classical GMM [30]–[35]. Most of these methods incorporate GMM into other well-known approaches such as optical flow and temporal difference [30], or improve the learning and parameter updating techniques [31]–[35]. Although the results show improvement, their computational complexity is still relatively high.

In [36], a new background subtraction method is proposed to solve dynamic background problems by employing a real-time dynamic background generation technique. This technique, based on a temporal median filter with exponentially weighted moving average filtering, can detect moving objects automatically with a low computational complexity.

The self-organizing background subtraction (SOBS) method uses the architecture of a self-organizing neural network map

Manuscript received January 16, 2012; revised July 20 2012, October 9, 2012; accepted February 4, 2013. This work was supported in part by the National Science Council under Grant NSC 100-2628-E-027-012-MY3. This paper was recommended by Associate Editor M. Cetin. *(Corresponding author: S.-C. Huang).*

The authors are with the Department of Electronic Engineering, National Taipei University of Technology, Taipei 106, Taiwan (e-mail: schuang@ntut.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2013.2248057

2168–2267/$31.00 © 2013 IEEE

to build a background model for detection of moving objects in complex environments [37]. A self-organizing neuronal map consisting of (3×3) weight vectors is used to construct the background model for each color pixel. However, this creates considerable memory and computational time requirements.

A highly compressed background model is obtained through application of the quantization/clustering technique by the use of the codebook background (CB) subtraction method from training sequences during long observation periods [38].

Unlike the methods mentioned above, the visual background extractor (ViBe) method constructs a background model by examining a set of pixel values taken either in the past or in the corresponding neighborhood. Moving objects are then detected by determining the differences between the background model and the current incoming pixel [39].

In contrast to the previously mentioned methods, this paper presents a novel motion detection approach based on the radial basis function (RBF) [40] artificial neural networks in order to segment moving objects in dynamic scenes. This method can effectively adapt to environmental changes and achieve accurate and complete detection in both dynamic and static scenes. Basically, the RBF neural network possesses the strong nonlinear mapping ability and the local synaptic plasticity of neurons with a minimal network structure. This allows it to be suitable for motion detection application in either dynamic or static scenes.

The remainder of this paper is organized as follows. Sections II presents a survey of several state-of-the-art related published works used in our comparison. Our motion detection method is described in Section III. The experimental results of our method then are compared with those of other state-of-the-art methods in section IV. Finally, Section V presents our conclusions.

II. Related Work

In the following section, we describe five state-of-the-art methods for motion detection in dynamic scenes: a real-time dynamic background generation method (RDBG) [36], a GMM [29], a SOBS [37], a CB subtraction method [38], and a visual background extractor method (ViBe) [39].

A. Real-Time Dynamic Background Generation

The RDBG [36], utilizes two bitmaps, B_t^L and B_t^S to generate the adaptive background model. The B_t^L bitmap holds the generated adaptive background model, while B_t^S holds the last frame from the camera. Each pixel in B_t^L has a long term timer $T^L(x, y)$ and a short term timer $T^S(x, y)$. The $T^L(x, y)$ timer is used to count the number of frames, in which the pixel of $B_t^L(x, y)$ features similar values. When the difference between $B_t^L(x, y)$ and the incoming pixel value $I_t(x, y)$ is within the predefined tolerance τ, the long term timer $T^L(x, y)$ is increased and $B_t^L(x, y)$ is replaced with the incoming pixel value $I_t(x, y)$.

The $T^S(x, y)$ timer is used to count the number of frames, in which the incoming pixel value $I_t(x, y)$ differs from $B_t^L(x, y)$. When the difference between $B_t^L(x, y)$ and the incoming pixel value $I_t(x, y)$ exceeds the predefined tolerance, the pixel value of $B_t^S(x, y)$ is replaced by incoming pixel value $I_t(x, y)$. Concurrently, if the incoming pixel value $I_t(x, y)$ differs from $B_{t-1}^S(x, y)$, $T^S(x, y)$ is reset to zero. Otherwise, $T^S(x, y)$ is increased.

Thus if a pixel is covered by a new object, $T^L(x, y)$ will stay the same and $T^S(x, y)$ will increase. When $T^S(x, y)$ is greater than $T^L(x, y)$, the new incoming pixel value $I_t(x, y)$ is assumed to be part of the background. In this case, the pixel values of $B_t^L(x, y)$ and $B_t^S(x, y)$ are replaced by the incoming pixel value $I_t(x, y)$, and $T^S(x, y)$ is reset to zero.

B. Gaussian Mixture Model

Use of the GMM method involves modeling [29] each pixel independently with a mixture of k Gaussians to maintain the probabilistic background model. The probability density function for the current pixel value is given by

$$P(X_t) = \sum_{i=1}^{k} \omega_{i,t} \eta\left(X_t,\ \mu_{i,t},\ \Sigma_{i,t}\right) \tag{1}$$

where X_t is each incoming pixel intensity value of the tth image frame, and $\omega_{i,t}$ is a estimation of the weight for the corresponding Gaussian distribution $\eta(X_t,\ \mu_{i,t},\ \Sigma_{i,t})$, which can be expressed as follows:

$$\eta\left(X_t,\ \mu_{i,t},\ \Sigma_{i,t}\right) = \frac{\exp[-((X_t-\mu_t)^T/2)\sum^{-1}(X_t-\mu_t)]}{2\pi^{n/2}|\Sigma|^{1/2}} \tag{2}$$

where $\mu_{i,t}$ and $\Sigma_{i,t}$ are the mean value and the covariance matrix of the ith Gaussian in the mixture model, respectively. The covariance matrix $\Sigma_{i,t}$ is assumed as follows:

$$\Sigma_{k,t} = \sigma_k^2 I. \tag{3}$$

Note that each pixel is checked against the k existing Gaussian distributions. If the pixel value is within 2.5 standard deviations of a distribution, the statement can be assumed as a match. The adaptive parameters of the first matched Gaussian model can be updated as follows:

$$\omega_{k,t} = (1-\alpha)\,\omega_{k,t-1} + \alpha \mathrm{M}_{k,t} \tag{4}$$

$$\mu_t = (1-\rho)\,\mu_{t-1} + \rho X_t \tag{5}$$

$$\delta_t^2 = (1-\rho)\,\delta_{t-1}^2 + \rho(X_t-\mu_t)^T\left(X_t-\mu_t\right). \tag{6}$$

The parameter $\mathrm{M}_{k,t}$ is 1 if the pixel matches the models; otherwise, it is set to 0. The adaptive background models are then obtained through the value ω/σ of each Gaussian, and the first B distributions are determined as follows:

$$\mathrm{B} = \operatorname{argmin}\left(\sum_{k=1}^{b} \omega_k > T_2\right) \tag{7}$$

where T_2 is the minimum portion of the data that should be classified as background.

C. Self-Organizing Background Subtraction

The SOBS method [37] consists of two basic steps. First, the initial background model is constructed by mapping each original image pixel to a (3 × 3) matrix of neuronal map structure. Second, the best match background candidate is found within the (3 × 3) matrix of each incoming pixel by a fixed threshold

$$d\left(c_m, p_t(x, y)\right) = \min_{i=1,\ldots,9} d(c_i, p_t(x, y)) \leq \epsilon \tag{8}$$

where $p_t(x, y)$ is the incoming pixel, c_i is the ith candidate in the (3 × 3) matrix, and c_m is the best match. If no best match is found for incoming pixel $p_t(x, y)$, then $p_t(x, y)$ is deemed part of a moving object; otherwise, $p_t(x, y)$ is regarded as a background pixel. If the best match $\mathrm{c_m}$ is located at position $(\overline{x}, \overline{y})$ in the background model, the background model is updated as follows:

$$\begin{aligned} A_t(i, j) = \left(1 - \alpha_{i,j}(t)\right) A_{t-1}(i, j) + \alpha_{i,j}(t) p_t(x, y) \quad s.t. \\ i = \overline{x} - 1 \ldots \overline{x} + 1,\ j = \overline{y} - 1 \ldots \overline{y} + 1 \end{aligned} \tag{9}$$

where A is the neuronal map background model, and α is the learning rate.

D. Codebook Background Subtraction

The CB method [38] models the background by employing a quantization/clustering technique based on observations over a long period of time. A codebook consisting of one or more codewords was generated for each pixel as follows:

$$C = \{\mathrm{c}_1, \mathrm{c}_2, \ldots, \mathrm{c}_L\} \tag{10}$$

where C is the codebook consisting of L codewords, c_i is the ith codeword consisting of a RGB vector v_i and a tuple u_i that can be presented as (R_i, G_i, B_i) and $\langle I_i^{\min}, I_i^{\max}, f_i, \lambda_i, p_i, q_i \rangle$, respectively.

Note that the $I^{\min}$ and $I^{\max}$ are the respective minimum and maximum brightness intensity values of the codewords. f represents the frequency at which the codeword has occurred. λ is expressed as the longest interval of the training period, in which the codeword has not recurred. p and q are used to record the first and last access times, respectively.

When there are an input pixel $x_t = (R, G, B)$ and a RGB vector v_i of a codeword c_i, they can be described as follows:

$$\|x_t\|^2 = R^2 + G^2 + B^2 \tag{11}$$

$$\|v_i\|^2 = \overline{R}_i^2 + \overline{G}_i^2 + \overline{B}_i^2 \tag{12}$$

$$\langle x_t, v_i \rangle^2 = (\overline{R}_i R + \overline{G}_i G + \overline{B}_i B)^2. \tag{13}$$

Thus, the color distortion δ can be obtained by

$$p^2 = \|x_t\|^2 \cos^2\theta = \frac{\langle x_t, v_i \rangle^2}{\|v_i\|^2} \tag{14}$$

$$\text{color dist}\,(x_t, v_i) = \delta = \sqrt{\|x_t\|^2 - p^2} \tag{15}$$

where v_i is the RGB vector of the codeword c_i and δ is the color distortion between x_t and v_i. The logical brightness function is defined as follows:

$$\text{brightness}\left(I, \langle \check{I}, \hat{I} \rangle\right) = \begin{cases} \text{true}, & if\, I_{\text{low}} \leq \|x_t\| \leq I_{\text{hi}} \\ \text{false}, & \text{otherwise} \end{cases}. \tag{16}$$

Moreover, the range $[I_{\text{low}}, I_{\text{hi}}]$ for each codeword is defined as follows:

$$I_{\text{low}} = \alpha \hat{I} \tag{17}$$

$$I_{\text{hi}} = \min\{\beta \hat{I}, \frac{\check{I}}{\alpha}\} \tag{18}$$

where α and β are the predefined parameters. Typically, α ranges between 0.4 and 0.7, and β ranges between 1.1 and 1.5. Finally, the detection result can be attained via two conditions as follows:

$$\text{color dist}\,(x, c_m) \leq \varepsilon_2, \tag{19}$$

$$\text{brightness}\left(I, \langle \check{I}_m, \hat{I}_m \rangle\right) = \text{true} \tag{20}$$

where ε_2 is the detection threshold, and c_m is the codeword of the background. If the incoming pixel fits within these two conditions, it is regarded as a background pixel; otherwise, it is part of a moving object. According to experiments conducted in previous studies [38], the average number of codewords per pixel for background acquisition is 6.5.

E. Visual Background Extractor

The ViBe approach [39] detects moving objects by calculating the difference between the background model $\mathcal{M}$ and the incoming pixel $p(x, y)$. Initially, the background model is initialized from the first frame. The tth background sample $\mathcal{M}^t(x, y)$ is randomly chosen by N neighboring pixels in the 8-connected neighborhood of location (x, y).

Second, a good match occurs when the Euclidean distance between $\mathcal{M}^t(x, y)$ and $p(x, y)$ is lower than a predefined threshold R. If the number of occurrences of good matches is larger than or equal to the given threshold $\#_{\text{min}}$, then the current pixel $p(x, y)$ is classified as background. Otherwise, $p(x, y)$ is regarded as a foreground pixel.

Finally, if $p(x, y)$ is determined to be a background pixel, then two randomly chosen background samples—one at location (x, y) and the other at a location in the 8-connected neighborhood—are replaced by $p(x, y)$.

III. Proposed RBF-based approach

In general, most existing methods can work well for static scenes. However, complete and accurate motion detection in dynamic scenes, such as those containing swaying trees, spouting fountains, rippling water, and so on, is still a very difficult task [18]. The main reason for this is the inherent difficulty in discrimination between moving objects and the dynamic background caused by the intensity fluctuations of both background and foreground pixels [28].

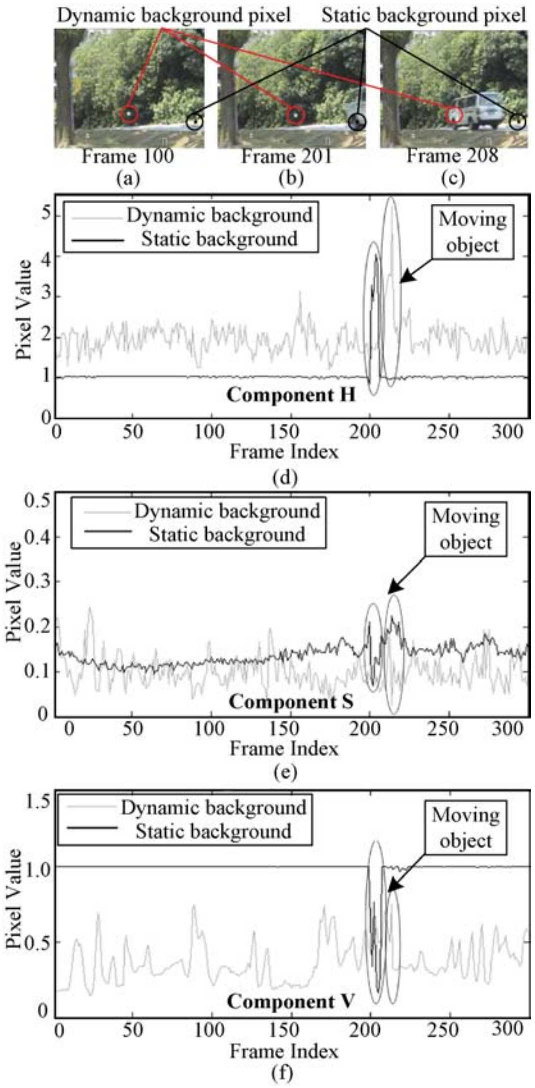


Fig. 1. Intensity variations of dynamic and static background pixels. (a)–(c) 100th, 201th, and 208th frames of sequence CAM with two pixels marked by white and black points. (d)–(f) Intensity variations in *H*, *S*, and *V* components, over time, of the sampled background pixels.

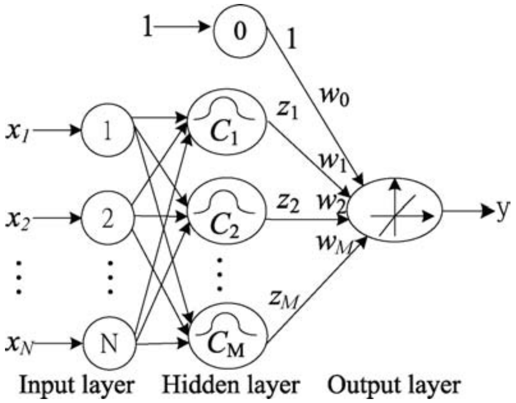


Fig. 2. Radial basis function neural network.

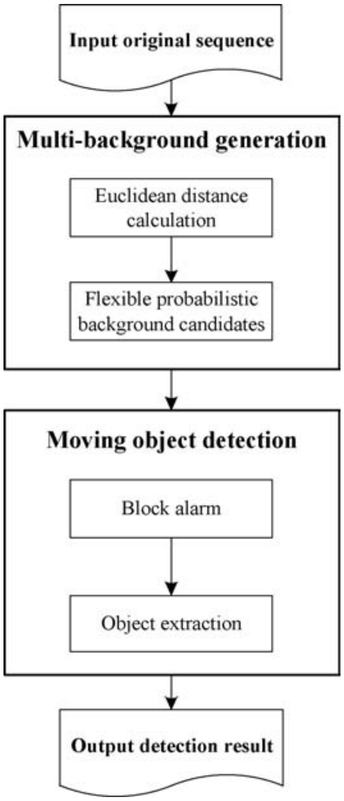
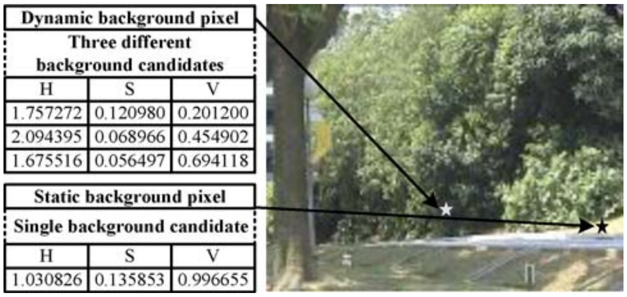

Input original sequence
Multi-background generation
Euclidean distance calculation
Flexible probabilistic background candidates
Moving object detection
Block alarm
Object extraction
Output detection result

Fig. 3. Overview of the modules involved in the proposed method.

For example, as shown in Fig. 1, two pixels marked by white and black points are sampled to plot their intensity variations in hue (H), saturation (S), and value (V) components for 300 frames of a sample sequence. The white point belonging to the waving tree is regarded as a dynamic background pixel in Fig. 1(a)–(c). Conversely, the black point is regarded as a static background pixel. In Fig. 1(b) and (c), a vehicle passes through the dynamic background pixel and the static background pixel, respectively. Fig. 1(d)–(f) shows the plots of intensity variations of two sample pixels marked by white and black points for H, S, and V components, respectively. It is obvious that the signals of the static background pixel are stable, allowing easy extraction of the moving object when the vehicle passes through. However, due to frequent signal oscillations, it is difficult to discriminate between the signal of the moving object and that of the dynamic background.

In this section, we propose a novel motion detection approach based on RBF artificial neural networks. This process extracts moving objects from dynamic scenes and static backgrounds in order to effectively avoid misjudging dynamic backgrounds as moving objects. The RBF neural network shown in Fig. 2 consists of an input layer, a hidden layer, and an output layer. It has certain advantages that include simple network configurations, fast learning speed by locally tuned neurons [40], and good approximation properties [41].

As shown in Fig. 3 our approach involves two important modules: a multibackground generation module and a moving object detection module. The proposed multibackground generation (MBG) module generates a flexible probabilistic background model automatically by calculating the Euclidean distance [42] from each incoming pixel to the corresponding reference background candidates; it then relays this information to the network as hidden layer neuron centers. The



Dynamic background pixel

Three different background candidates

H	S	V
1.757272	0.120980	0.201200
2.094395	0.068966	0.454902
1.675516	0.056497	0.694118

Static background pixel

Single background candidate

H	S	V
1.030826	0.135853	0.996655

Fig. 4. A scene with a dynamic background pixel and a static background pixel. The white point is the dynamic background pixel whose dynamic range can be expressed by three different candidates. The black point is a static background pixel which requires only a single candidate.

probabilistic background model can express the dynamic range of each pixel within the background and is used to construct a hidden layer in the RBF network structure.

After employing the MBG module, the proposed moving object detection (MOD) module is employed. The MOD module accomplishes complete and accurate detection of moving objects by using two procedures: a block alarm procedure and an object extraction procedure. The block alarm procedure eliminates unnecessary examination of the dynamic and static background region, after which the object extraction procedure processes those blocks that have a high probability of containing moving objects.

A. Multibackground Generation

In this section, three perceptual variables, hue (H), saturation (S), and value (V) of the input layer are built in HSV color space that is very similar to human visual capability [43]. Let (h, s, v) represent hue, saturation, and value component values of a pixel $p_t(x, y)$ in each incoming frame I_t.

A sufficient number of hidden neurons can improve the accuracy. Nevertheless, too many neurons may result in enlargement of the network structure and reduction in performance quality. Therefore, it is very important to construct a proper flexible probabilistic background model that can represent the hidden neurons.

In order to construct a proper flexible probabilistic background model, each incoming pixel intensity value $p_t(x, y)$ of the tth frame I_t is compared to the corresponding candidates of background intensity values $B(x, y)_1$ to $B(x, y)_n$. If the intensity of incoming pixel $p_t(x, y)$ is close to the related candidates of background intensity—e.g., if the incoming pixel belongs to the background candidates—we update the related background candidates; otherwise, $p_t(x, y)$ is declared as a new background candidate.

To determine whether or not the incoming pixel $p_t(x, y)$ is close to the related candidates of background intensity, we employ the Euclidean distance of vectors in the HSV color hexcone [42]. This is calculated using the distance from pixel $p_i = (h_i, s_i, v_i)$ to pixel $p_j = (h_j, s_j, v_j)$ by

$$\begin{aligned} d(p_i, p_j) = \|(v_i s_i \cos(h_i), v_i s_i \sin(h_i), v_i) - \\ (v_j s_j \cos(h_j), v_j s_j \sin(h_j), v_j)\|_2^2 \end{aligned} \quad . \tag{21}$$

Use of this metric can circumvent problems in the periodicity of hue and the unsteadiness of hue for small saturation values [42].

An empirical tolerance, ϵ, is used to determine whether or not incoming pixel $p_t(x, y)$ belongs to background candidates $B(x, y)_k$, where $k \in \{1, n\}$. This decision rule can be expressed as

$$p_t(x, y) \begin{cases} \in B(x, y)_k, \text{if } d(p_t(x, y), B(x, y)_k) \le \epsilon \\ \notin B(x, y)_k, \text{otherwise.} \end{cases} . \tag{22}$$

Background candidates close to incoming pixel $p_t(x, y)$ are updated by

$$B(x, y)'_k = (1 - \beta)B(x, y)_k + \beta p_t(x, y) \tag{23}$$

where $B(x, y)_k$, $B(x, y)'_k$ are the original and updated kth candidates at position (x, y), and β is a predefined parameter. Fig. 4 illustrates candidates of background intensity from dynamic and static areas. This background construction approach can be regarded as an unsupervised learning process of the centers' location in the RBF network.

B. Moving Object Detection

1) *Block Alarm Procedure:* The structure of the RBF network considered here consists of three input neurons, one output neuron, and a hidden layer of M neurons. The MBG module determines the number M and center points $C_1, \ldots, C_M$ of the hidden layer neurons in the RBF network, as shown in Fig. 2; it also determines the structure of the network. After structure determination, the HSV components (h, s, v) of the incoming pixel $p_t(x, y)$ are used as the input vector. The input neurons propagate the input vector to the hidden layer neurons. After Euclidean distances between the input vector and center points of the hidden neurons are calculated, the output of each hidden neuron is generated by the basis function as follows:

$$z_i(p) = \phi(\|p - C_i\|), \text{ where } i = 1, 2, \ldots, M, \tag{24}$$

where $\phi(\cdot)$ is the basis function, C_i is the center point of the ith neuron, p is the input vector, M is the number of hidden neurons, and $\|p - C_i\|$ is the Euclidean distance between p and C_i.

Several types of basis functions are commonly used such as the Gaussian function, linear function, cubic function, thin plate spline function, and so on [44]. For our approach, we use the most common basis function that is the Gaussian function [45]. The representative function is as follows:

$$\phi(\|p - C_i\|) = \exp\left(\frac{-\|p - C_i\|^2}{2\sigma^2}\right) \tag{25}$$

where σ is defined as ϵ, and ϵ is the empirical tolerance of Euclidean distance in (22). The reason for this is that a lower ϵ value correlates to the generation of more background candidates in the probabilistic model. Lower standard division σ values can make the Gaussian curve more gradual. This can later prevent the summation in the output layer from getting too high and misjudging dynamic background. Therefore, σ is in proportion to ϵ. According to our experiment, σ can be empirically defined as ϵ.

Because the Gaussian function is factorizable and localized [45], [46], it is suitable for our application. Moreover, the

Gaussian function can be used to provide a fine fit for checking the block state empirically. The larger the output value of basis function is, the more closely the input vector is located to the center points—e.g., the higher the probability of the incoming pixel being background. In order to eliminate unnecessary examination of the dynamic and static background region, the incoming frame is split into w×w blocks. The calculation of the sum of basis functions within each block is as follows:

$$\delta = \sum_{p\in\mu} \sum_{i=1}^{M} \phi(\| p - C_i \|) \tag{26}$$

where p is each independent pixel of the corresponding block μ, M is the number of hidden neurons, and the block size w can be set to 4.

When the calculated sum of block (i, j) exceeds a threshold S, the block $A(i, j)$ is labeled with 0, which indicates that it does not contain pixels belonging to moving objects. Otherwise, block $A(i, j)$ is labeled with 1, meaning that it is highly probable that it contains pixels of moving objects

$$A(i, j) = \begin{cases} 0, \text{if } \delta \ge S \\ 1, \text{otherwise} \end{cases}. \tag{27}$$

Table I, illustrates the sum of basis functions within blocks in a sampled video frame. By setting S equal to 12, blocks that may possibly contain moving objects can be detected.

Finally, the background candidates are updated in the hidden layer by

$$B(x, y)_k^t = \begin{cases} B(x, y)_k^{t-1}, \text{if } p_t(x, y) \notin B(x, y)_k^{t-1} \\ \alpha p_t(x, y) + (1 - \alpha) B(x, y)_k^{t-1}, \text{otherwise} \end{cases} \tag{28}$$

where $B(x, y)_k^{t-1}$, $B(x, y)_k^t$ are the kth candidates at position (x, y) of the previous and current flexible background models, and α is a predefined parameter. The decision rule of whether $p_t(x, y)$ belongs to $B(x, y)_k^{t-1}$ is determined according to (22).

2) *Object Extraction Procedure:* After the block alarm procedure, unnecessary examinations are eliminated and the object extraction procedure processes only blocks containing moving objects. As the last step of our approach, the output layer of the RBF network is used to compute the binary motion detection mask as the detection result. The output layer computes a function of the weighted linear combination of the values emerging from the hidden layer as follows:

$$F = \sum_{i=1}^{M} w_i(z_i(p)) + w_0 \tag{29}$$

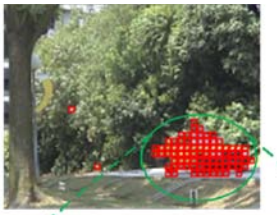
where w_i is the weight that connects the ith hidden neuron and the output layer, z_i is the output value of ith hidden neuron, and w_0 is a fixed threshold. Initially, w_i is experimentally set to 1. The binary motion detection mask is obtained as follows:

$$D(x, y) = \begin{cases} 1, \text{if } F(x, y) < 0 \\ 0, \text{otherwise} \end{cases}. \tag{30}$$

To label $D(x, y)$ with 1, means that pixel $p_t(x, y)$ is part of a moving object; otherwise, $p_t(x, y)$ is part of the background and is labeled with 0. After finishing operations for the current incoming frame, we adjust the weights for operations for the next incoming frame. In the beginning, all weights are initialized to 1, after which the weights are adjusted as follows:

TABLE I

SUM OF BASIS FUNCTIONS WITHIN BLOCKS IN A VIDEO FRAME



15.05	13.12	15.13	15.87	15.96	15.97	14.70	14.87	15.14	15.62	15.39	12.88	15.98	15.46	15.23	14.73	14.64
15.71	15.57	15.57	15.82	15.84	15.91	15.48	15.79	14.70	15.02	15.25	14.14	15.78	15.35	15.98	15.85	15.86
15.94	15.95	15.54	15.72	15.70	15.03	15.00	11.25	12.79	15.64	14.90	14.89	15.86	15.60	15.32	15.90	15.26
15.87	15.79	15.82	15.08	14.39	15.78	12.50	6.60	9.19	12.02	14.34	14.69	14.92	15.88	15.95	15.98	15.93
15.20	15.91	15.73	15.08	13.73	10.86	9.37	9.03	7.83	7.93	8.66	13.04	15.79	15.55	15.29	15.82	15.86
15.68	14.93	14.59	10.74	6.77	6.79	9.49	3.54	3.48	7.16	5.56	10.28	14.14	15.76	15.72	15.93	15.23
14.61	11.82	6.50	2.19	6.12	8.53	8.29	5.20	5.44	5.87	10.63	4.16	5.19	6.37	6.92	8.92	15.57
15.17	8.54	1.30	3.28	8.22	6.86	3.36	8.30	8.61	11.89	9.63	8.37	2.56	11.01	8.49	5.02	15.67
14.83	14.69	14.49	13.33	6.15	9.09	9.40	7.12	2.20	2.36	2.00	2.53	3.64	2.66	2.55	0.94	8.29
13.99	15.21	15.30	6.82	6.85	11.41	7.12	6.84	5.75	1.13	4.47	3.55	4.05	2.27	0.13	4.81	15.36
15.45	15.53	15.67	8.97	9.74	13.48	12.04	9.25	6.79	6.56	10.59	10.15	11.09	12.17	8.31	13.62	15.04
15.93	15.89	15.96	15.91	15.91	15.97	15.96	15.97	15.96	15.92	15.47	15.91	15.79	15.12	13.64	15.57	15.04
15.94	15.98	15.99	15.97	15.95	15.97	15.92	15.96	15.85	15.96	15.98	15.10	15.91	14.29	14.53	15.19	15.49

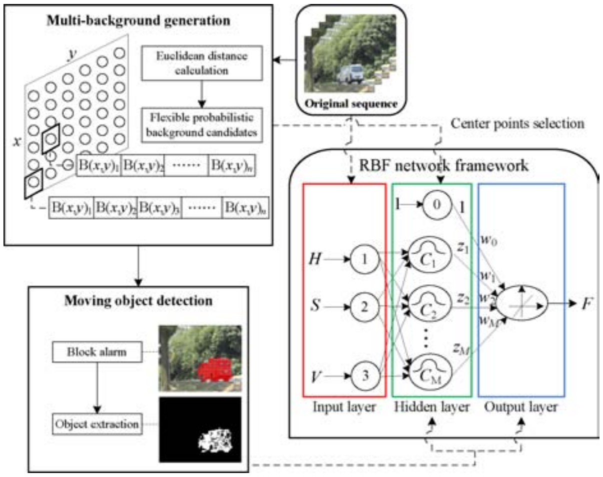


Fig. 5. Flowchart of proposed RBFMD approach.

$$w_i^{t+1} = (w_i^t + \eta \cdot z_i) \cdot \frac{M}{M + \eta \cdot \sum_{i=1}^{M} z_i} \tag{31}$$

where w_i^t is the weight among the output layer and ith hidden neuron at frame I_t, η is the learning rate, and M is the number of hidden neurons.

After weight adjusting, the weights among the output layer and the hidden neurons that are close to the input vector are reinforced, and the others are decreased. Fig. 5 shows the flowchart of the proposed RBF-based motion detection approach (RBFMD).

IV. EXPERIMENTAL RESULTS

The intention of this section is to present a comparison between our RBFMD method and several other state-of-the-art methods. Experimental results of object extraction performed by the RBFMD method were analyzed through qualitative and quantitative comparisons with other state-of-the-art methods

TABLE II
SPECIFIC PARAMETER VALUES OF RBFMD

β	ϵ	S	α	w_0	η
0.3	0.2	12	0.1	−0.7	0.01

TABLE III
NUMBER OF TRAINING FRAMES ADOPTED IN RBFMD

CAM	*FT*	*WS*	*MR*	*MSA*	*IR*
150	150	150	300	40	50

for several natural video sequences representative of dynamic and static scenes.

Sequences *CAM*, *WS*, *FT*, and *MR* were employed to test the results of object extraction from dynamic backgrounds. The first sequence, *CAM* is of moving vehicles and pedestrians in front of waving trees. The second sequence, *WS* is of a person walking at a waterfront where a rippling water surface can be seen in the background. The background dynamics in the third sequence (*FT*) are caused by an active water fountain. The fourth sequence, *MR* was captured in a meeting room where the background curtain was moving in the wind.

Two additional sequences were utilized in order to gauge detection results regarding static backgrounds. Sequence *MSA* is of a person walking in a corridor and features objects stopping temporarily. Sequence *IR* was taken in an initially vacant meeting room with one person entering and slowly walking around.

All the parameters in each method are set to the optimum values. According to [36], the predefined tolerance τ of RDBG method can be set to 5. For maintaining the probabilistic background model by the use of GMM, the number of Gaussians components k is fixed to 3 with the learning rate α equal to 0.001 [29]. The literature [37] of the SOBS method indicates that the learning rate α is obtained by predefined constants c_1 and c_2, which range over 0.01 to 1.0. Moreover, the threshold ϵ_1 and ϵ_2 ranges from 0.1 to 0.2 and 0.01 to 0.07, respectively. According to [38], the minimum and maximum brightness of all pixels in the CB method are derived from typical values α and β, which range from 0.4 to 0.7 and 1.1 to 1.5, respectively.

In regard to the ViBe method [39], the number of background samples per pixel N can be set to 20, the number of times matches occur, $\#_{\min}$, can be set to 2 for determination of a background pixel. The predefined threshold of the Euclidean distance R is set to 20. Specific values of all parameters and the numbers of training frames of the RBFMD method are shown in Tables II and III.

We subsequently evaluated the memory requirement of the probabilistic background model generated by the RBFMD method and compared it to the probabilistic background models of other methods. Finally, to verify the computational feasibility for real-time applications, we measured the processing speed of the RBFMD method.

A. Quantitative Evaluation

In order to objectively evaluate the accuracy of binary objects masks detected by RBFMD and other state-of-the-art methods, quantitative evaluations through *Recall*, *Precision*, F_1, and *Similarity* metrics [27], [47]–[52] were utilized on the test video sequences.

Recall provides the percentage of detected true positives by a comparison with the total count of items in the ground truth

$$Recall = tp/(tp + fn) \tag{32}$$

where tp is the total count of true positive pixels, fn is the total count of false negative pixels, and $(tp + fn)$ represents the total count of items in the ground truth.

Precision provides the percentage of detected true positives by a comparison with the total count of items in the binary objects mask detected by the method

$$Precision = tp/(tp + fp) \tag{33}$$

where fp is the total count of false positive pixels, and $(tp + fp)$ represents the total count of detected items in the binary objects mask.

Nevertheless, *Recall* selectively measures only the incorrect association of internal lost items to moving objects, and *Precision* selectively measures only the incorrect association of superfluous detected items. Accordingly, using the above mentioned metrics alone cannot offer a satisfactory comparison between the different methods.

In order to facilitate an effective measurement, accuracy was evaluated in terms of two additional metrics—F_1 and *Similarity*. Use of these two metrics was accomplished by weighting the harmonic means of *Recall* and *Precision*

$$F_1 = 2(Recall)(Precision)/(Recall + Precision) \tag{34}$$

$$Similarity = tp/(tp + fp + fn). \tag{35}$$

All attained values through the above considered metrics range from 0 to 1, with higher values meaning greater accuracy.

Average accuracy values for all test sequences were obtained through utilizing the above four metrics. These were generated by GMM [29], RDBG [36], SOBS [37], CB [38], ViBe [39], and RBFMD methods, and are reported in Table IV. We can readily observe that the RBFMD method achieves the best *Similarity* and F_1 values in comparison to other state-of-the-art methods for the *CAM*, *WS*, *FT*, *MR*, *MSA*, and *IR* sequences. In particular, the RBFMD method is the only method that attains accuracy rates of all metrics exceeding 80% for the *WS* and *MR* sequences, which contain dynamic background.

In comparison, the accuracy rates obtained through *Similarity* and F_1 for the GMM method were up to 38% and 34% lower than those achieved by the RBFMD method, respectively; the accuracy rates produced by *Similarity* and F_1 for the RDBG method were up to 45% and 44% lower than those achieved by the RBFMD method, respectively; the accuracy rates produced through *Similarity* and F_1 for the SOBS method were up to 25% and 21% lower than those achieved by the RBFMD method, respectively; the accuracy rates produced through *Similarity* and F_1 for the CB method were up to 39% and 31% lower than those achieved by the RBFMD method, respectively; the accuracy rates produced

TABLE IV

COMPARISON OF THE OBTAINED AVERAGE *Similarity*, F_1, *Precision*, AND *Recall* VALUES OF EACH METHOD

Sequence	Evaluation	RBFMD	GMM	RDBG	SOBS	CB	ViBe
CAM	*Similarity*	**0.6971**	0.5713	0.2479	0.6649	0.6471	0.3352
	F_1	**0.8204**	0.6546	0.3815	0.7919	0.7342	0.4865
	Recall	0.7444	0.5504	0.6632	0.6904	**0.7694**	0.4100
	Precision	0.9152	0.9266	0.3010	**0.9460**	0.7088	0.7054
WS	*Similarity*	**0.8119**	0.5885	0.6538	0.7204	0.7958	0.6910
	F_1	**0.8957**	0.7408	0.7900	0.8372	0.8851	0.8166
	Recall	**0.9129**	0.6091	0.8055	0.7356	0.8240	0.7650
	Precision	0.8811	0.9481	0.7778	**0.9722**	0.9035	0.8790
FT	*Similarity*	**0.5764**	0.5427	0.3513	0.4343	0.4464	0.4049
	F_1	**0.7289**	0.7001	0.5064	0.5986	0.6160	0.5741
	Recall	0.5931	**0.8063**	0.6991	0.4580	0.4817	0.5417
	Precision	**0.9526**	0.6221	0.4206	0.8795	0.8693	0.6168
MR	*Similarity*	**0.8029**	0.7578	0.6824	0.5485	0.7894	0.7731
	F_1	**0.8885**	0.8591	0.8089	0.6758	0.8819	0.8700
	Recall	0.8279	0.9153	0.8419	0.5638	0.8269	**0.9157**
	Precision	**0.9662**	0.8133	0.7830	0.9646	0.9455	0.8705
MSA	*Similarity*	**0.8541**	0.4725	0.8158	0.8387	0.5834	0.7866
	F_1	**0.9211**	0.5774	0.8980	0.9096	0.7338	0.8800
	Recall	**0.9383**	0.5316	0.9357	0.9190	0.6113	0.8821
	Precision	0.9010	0.8216	0.8655	0.9110	0.9260	**0.9492**
IR	*Similarity*	**0.8138**	0.5045	0.7005	0.8025	0.4286	0.5820
	F_1	**0.8962**	0.6404	0.8222	0.8887	0.5851	0.7318
	Recall	0.9016	0.5724	**0.9350**	0.9229	0.4298	0.6307
	Precision	0.9002	0.8176	0.7478	0.8642	**0.9955**	0.9052

TABLE V

AVERAGE NUMBERS OF BACKGROUND INTENSITIES PER PIXEL OF DIFFERENT PROBABILISTIC BACKGROUND MODELS

	CAM	*FT*	*WS*	*MR*	*MSA*	*IR*
RBFMD	1.53	1.10	1.34	1.04	1.01	1.01
GMM	3–5	3–5	3–5	3–5	3–5	3–5
SOBS	9	9	9	9	9	9
CB	2.61	1.31	1.40	1.31	1.28	1.26
ViBe	20	20	20	20	20	20

through *Similarity* and F_1 for the ViBe method were up to 36% and 33% lower than those achieved by the RBFMD method, respectively. It is important to note that each method is implemented by using optimum parameters according to previous studies [29], [36]–[39] and all accuracy values of each method are obtained by considering every frame of each test sequence.

B. Qualitative Evaluation

Here, qualitative evaluation of the object extraction results for different test sequence by each of the methods is performed via visual inspection. The qualitative evaluation results of each video sequence along with *Similarity* and F_1 accuracy values of the binary object masks detected by each method are shown in Figs. 6–9. Comparing the generated binary objects masks with the ground truths, we find that the RBFMD method achieves robust detection not only in dynamic scenes but also in static scenes, and that the detection results of RBFMD are more accurate than those obtained by the GMM, RDBG, SOBS, CB, and ViBe methods.

It is evident from the results of qualitative and quantitative evaluations that the RBFMD method was successful in detection of moving objects in both dynamic and static backgrounds, outperforming other state-of-the-art methods.

C. Multibackground Analysis

In an advanced video surveillance system, a proper background model is necessary for accurate detection. Most approaches utilize a single image to characterize the background while trying to achieve accurate detection. However, it is usually difficult to get robust estimates of dynamic backgrounds by using a single background model. In general, a probabilistic background model is more suitable for the handling of dynamic backgrounds. However, using probabilistic background models may increase the memory requirement. The flexible probabilistic background model generated by the RBFMD method stores a different number of background intensities at each pixel position according to the dynamic range of each pixel.

Table V shows the average number of background intensities per pixel of the different probabilistic background models generated by the RBFMD, GMM, SOBS, CB and ViBe methods for several test sequences.

We can see that the average number of stored background candidates per pixel by the RBFMD method for sequences containing dynamic background (*CAM*, *FT*, *WS*, and *MR*) were 1.53 or fewer, and the number for sequences with static background (*MSA* and *IR*) were close to 1. In comparison, GMM, SOBS, and ViBe store fixed numbers of background intensities at each pixel position. In previous papers, the stored numbers of GMM, SOBS and ViBe are recommended to be 3 to 5, 9, and 20, respectively.

The average memory requirements for images of different sizes for the probabilistic background models of the different methods are shown in Table VI. Test sequences *CAM*, *WS*, *FT*, and *MR* with image size 160 × 128 pixels and sequences

TABLE VI
AVERAGE MEMORY REQUIREMENTS (IN KB) OF PROBABILISTIC BACKGROUND MODELS

	RBFMD (kB)	GMM (kB)	SOBS (kB)	CB (kB)	ViBe (kB)
160 × 128 pixels	76.82	184.32–307.20	552.96	305.51	1228.80
320 × 240 pixels	232.69	691.20–1152.00	2073.60	887.82	4608.00

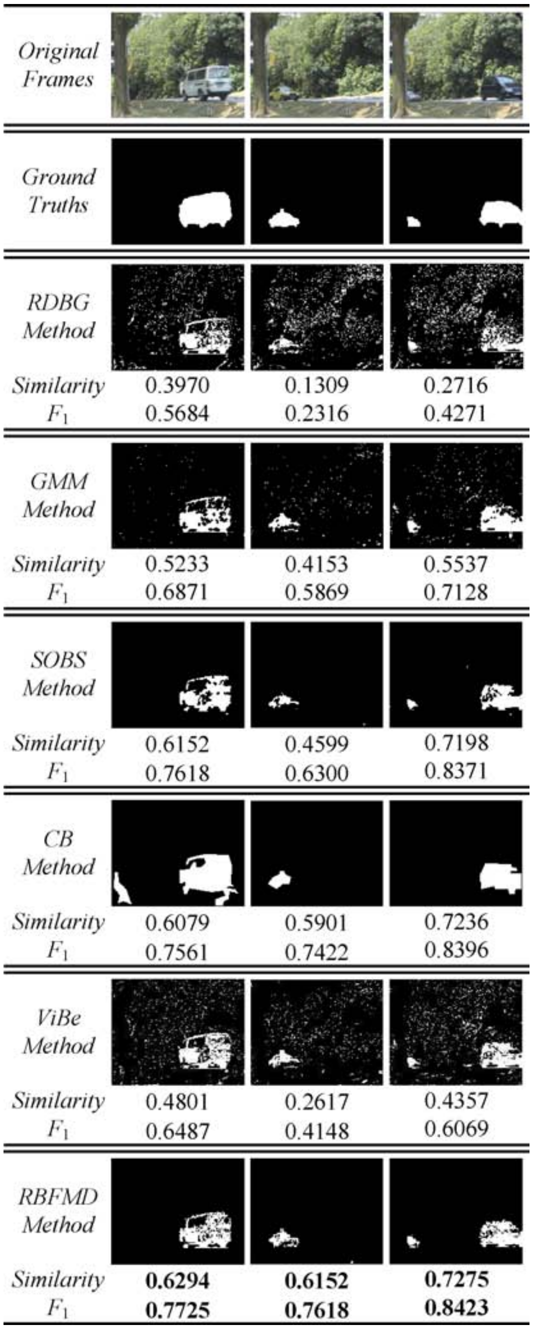

Original Frames
Ground Truths
RDBG Method
Similarity 0.3970 0.1309 0.2716
F_1 0.5684 0.2316 0.4271
GMM Method
Similarity 0.5233 0.4153 0.5537
F_1 0.6871 0.5869 0.7128
SOBS Method
Similarity 0.6152 0.4599 0.7198
F_1 0.7618 0.6300 0.8371
CB Method
Similarity 0.6079 0.5901 0.7236
F_1 0.7561 0.7422 0.8396
ViBe Method
Similarity 0.4801 0.2617 0.4357
F_1 0.6487 0.4148 0.6069
RBFMD Method
Similarity **0.6294** **0.6152** **0.7275**
F_1 **0.7725** **0.7618** **0.8423**

Fig. 6. Detection results of sequence *CAM*.

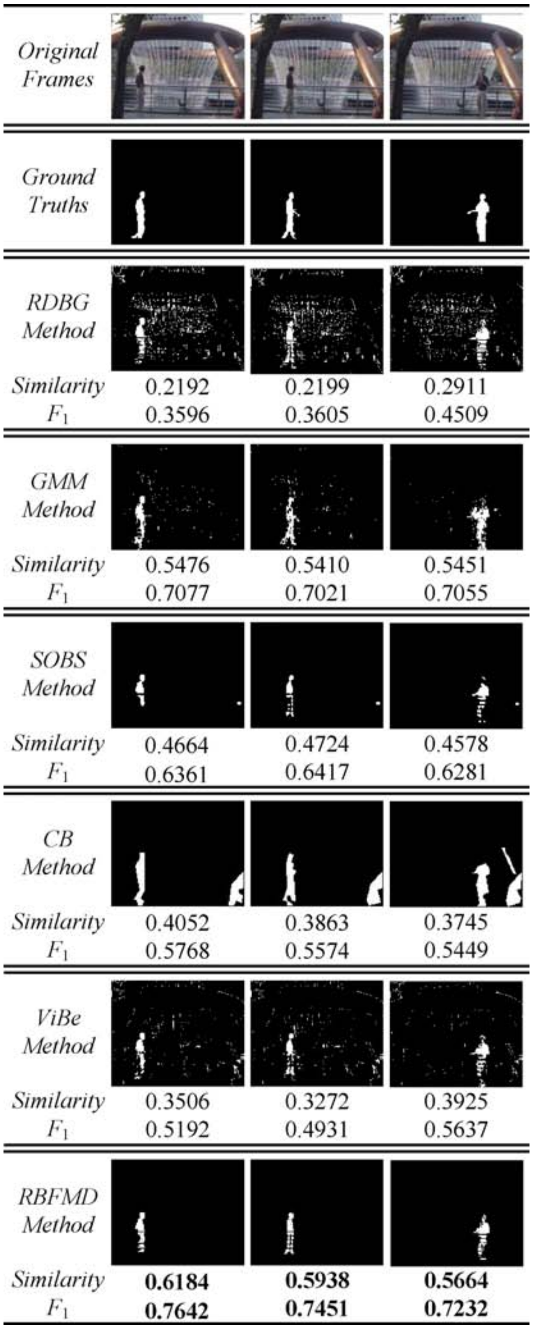

Original Frames
Ground Truths
RDBG Method
Similarity 0.2192 0.2199 0.2911
F_1 0.3596 0.3605 0.4509
GMM Method
Similarity 0.5476 0.5410 0.5451
F_1 0.7077 0.7021 0.7055
SOBS Method
Similarity 0.4664 0.4724 0.4578
F_1 0.6361 0.6417 0.6281
CB Method
Similarity 0.4052 0.3863 0.3745
F_1 0.5768 0.5574 0.5449
ViBe Method
Similarity 0.3506 0.3272 0.3925
F_1 0.5192 0.4931 0.5637
RBFMD Method
Similarity **0.6184** **0.5938** **0.5664**
F_1 **0.7642** **0.7451** **0.7232**

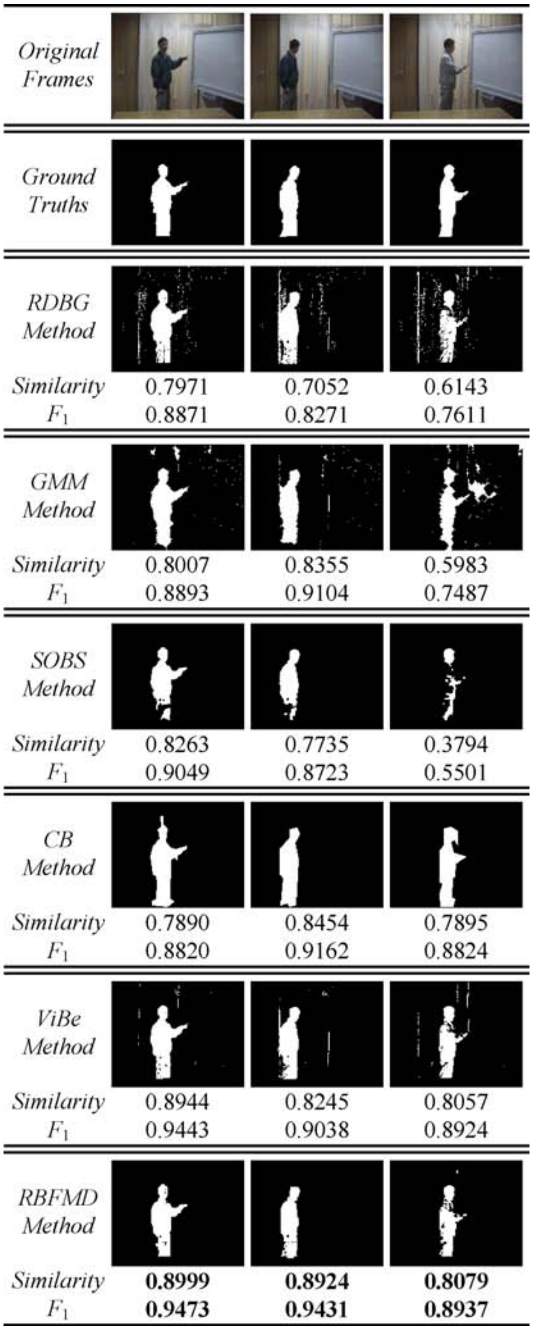
Fig. 7. Detection results of sequence *FT*.

MSA and *IR* with image size 320 × 240 pixels were employed to evaluate average memory requirements.

For sequences with image size 160 × 128 pixels, the average required memory of the RBFMD method is approximately 76.82 kB. While for image size 320 × 240 pixels, 232.69 kB memory is required. Compared with the memory requirements for probabilistic background models of other methods, use of the proposed method can result in a 58 to 95 percent reduction in memory requirement.

D. Performance Results

To verify the computational feasibility for real-time applications, we report the processing speeds of the RBFMD method



RDBG Method Similarity	0.7971	0.7052	0.6143
F_1	0.8871	0.8271	0.7611
GMM Method Similarity	0.8007	0.8355	0.5983
F_1	0.8893	0.9104	0.7487
SOBS Method Similarity	0.8263	0.7735	0.3794
F_1	0.9049	0.8723	0.5501
CB Method Similarity	0.7890	0.8454	0.7895
F_1	0.8820	0.9162	0.8824
ViBe Method Similarity	0.8944	0.8245	0.8057
F_1	0.9443	0.9038	0.8924
RBFMD Method Similarity	**0.8999**	**0.8924**	**0.8079**
F_1	**0.9473**	**0.9431**	**0.8937**

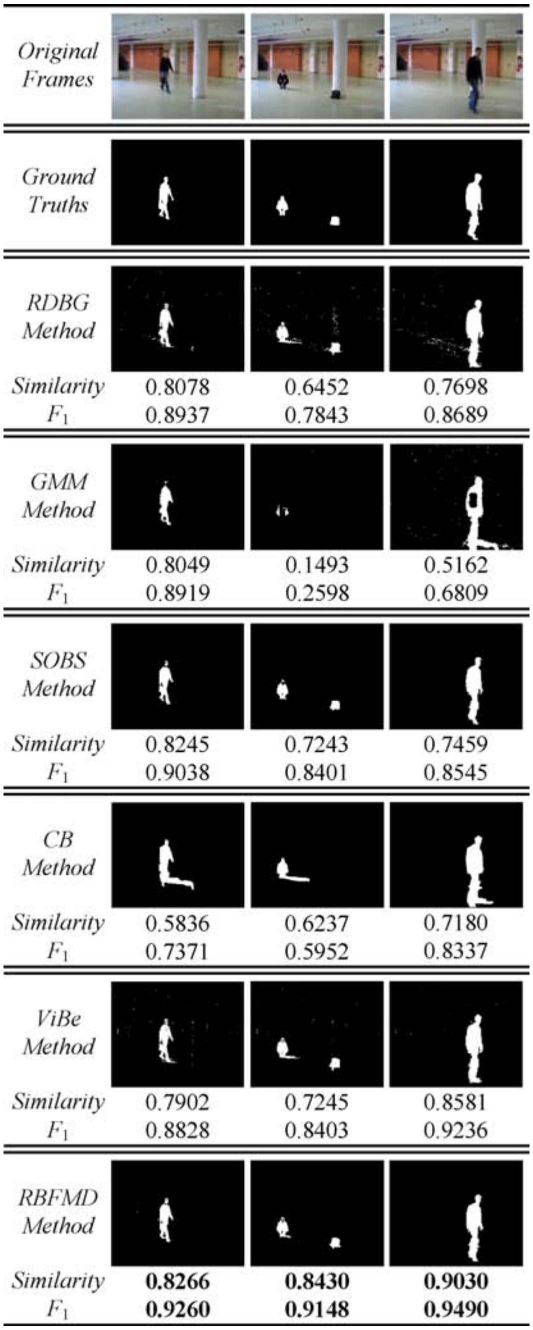
Fig. 8. Detection results of sequence *MR*.

TABLE VII
PROCESSING SPEED (IN F/S) OF RBFMD

CAM	*FT*	*WS*	*MR*	*MSA*	*IR*
40.35	53.59	58.29	54.57	34.10	35.03

for several test sequences in Table VII. The RBFMD method was implemented using C programming language on an Intel Core2Quad 2.33 GHz processor and 2 GB of RAM, running a Windows 7 operating system.

The performance results indicate that for all sequences with dynamic and static backgrounds, the RBFMD method can achieve speeds higher than 34 f/s, which is sufficient for real-time applications.



RDBG Method Similarity	0.8078	0.6452	0.7698
F_1	0.8937	0.7843	0.8689
GMM Method Similarity	0.8049	0.1493	0.5162
F_1	0.8919	0.2598	0.6809
SOBS Method Similarity	0.8245	0.7243	0.7459
F_1	0.9038	0.8401	0.8545
CB Method Similarity	0.5836	0.6237	0.7180
F_1	0.7371	0.5952	0.8337
ViBe Method Similarity	0.7902	0.7245	0.8581
F_1	0.8828	0.8403	0.9236
RBFMD Method Similarity	**0.8266**	**0.8430**	**0.9030**
F_1	**0.9260**	**0.9148**	**0.9490**

Fig. 9. Detection results of sequence *MSA*.

V. CONCLUSION

A novel motion detection approach for moving object segmentation in both static and dynamic scenes was proposed. The proposed method was based on the RBF neural network and featured a combination of a unique multibackground generation (MBG) module along with a novel two-procedure moving object detection (MOD) module to achieve accurate and complete detection in both static and dynamic scenes. The MBG module effectively generated a flexible probabilistic model that can express the dynamic range of each pixel within the background and can be used to construct the hidden layer in the RBF network structure. This minimized the network structure and increased the processing speed. After a high-quality probabilistic background model was generated, the MOD module utilized a block alarm procedure to eliminate unnecessary examination of the entire background region, after which an object extraction procedure can efficiently detect the pixels of moving objects. For the final step, the output weights were adjusted for operations on the next incoming frame. The experimental results were evaluated by qualitative and quantitative comparisons with other state-of-the-art methods based on a wide range of natural video sequences. The quantitative and qualitative evaluation results indicated that the proposed method was capable of achieving complete and accurate detection in both static and dynamic scenes, outperforming other methods. Moreover, we demonstrated empirically that the proposed method had the lowest memory requirement for probabilistic background model generation, compared to other probabilistic background modeling techniques. In addition, our method was demonstrated to be feasible for real-time applications. Analyses indicated that the proposed RBFMD method can detect moving objects in both dynamic and static scenes with inexpensive computation and low memory requirement.

References

[1] K. Huang, D. Tao, Y. Yuan, X. Li, and T. Tan, “View-independent human behavior analysis,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 4, pp. 1028–1035, Aug. 2009.

[2] Y. Ran, Q. Zheng, R. Chellappa, and T. M. Strat, “Applications of a simple characterization of human gait in surveillance,” *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 40, no. 4, pp. 1009–1020, Aug. 2010.

[3] L. Li, W. Huang, I. Y.-H. Gu, R. Luo, and Q. Tian, “An efficient sequential approach to tracking multiple objects through crowds for real-time intelligent CCTV systems,” *IEEE Trans. Syst. Man Cybern. B Cybern.*, vol. 38, no. 5, pp. 1254–1269, Oct. 2008.

[4] Z. Zhu, G. Xu, B. Yang, D. Shi, and X. Lin, “Visatram: A real-time vision system for automatic trafc monitoring,” *Image Vis. Comput.*, vol. 18, no. 10, pp. 781–794, Jul. 2000.

[5] K. Huang, D. Tao, Y. Yuan, X. Li, and T. Tan, “Biologically inspired features for scene classification in video surveillance,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 307–313, Feb. 2011.

[6] D. Makris and T. Ellis, “Learning semantic scene models from observing activity in visual surveillance,” *IEEE Trans. Syst. Man Cybern. B Cybern.*, vol. 35, no. 3, pp. 397–408, Jun. 2005.

[7] G. L. Foresti, “A real-time system for video surveillance of unattended outdoor environments,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 6, pp. 697–704, Oct. 1998.

[8] C. M. Huang and L. C. Fu, “Multitarget visual tracking based effective surveillance with cooperation of multiple active cameras,” *IEEE Trans. Syst. Man Cybern. B Cybern.*, vol. 41, no. 1, pp. 234–247, Feb. 2011.

[9] J. Xue, N. Zheng, J. Geng, and X. Zhong, “Tracking multiple visual targets via particle-based belief propagation,” *IEEE Trans. Syst. Man Cybern. B Cybern.*, vol. 38, no. 1, pp. 196–209, Feb. 2008.

[10] D. Labonte, P. Boissy, and F. Michaud, “Comparative analysis of 3-D robot teleoperation interfaces with novice users,” *IEEE Trans. Syst. Man Cybern. B Cybern.*, vol. 40, no. 5, pp. 1331–1342, Oct. 2010.

[11] I. Haritaoglu, D. Harwood, and L. S. Davis, “W^4: Real-time surveillance of people and their activities,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 809–830, Aug. 2000.

[12] N. M. Oliver, B. Rosario, and A. P. Pentland, “A bayesian computer vision system for modeling human interactions,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 831–843, Aug. 2000.

[13] W. Hu, T. Tan, L. Wang, and S. Maybank, “A survey on visual surveillance of object motion and behaviors,” *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 34, no. 3, pp. 334–352, Aug. 2004.

[14] L. Wixson, “Detecting salient motion by accumulating directionally consistent flow,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 774–780, Aug. 2000.

[15] B. K. P. Horn and B. G. Schunck, “Determining optical flow,” *Artificial Intelligence*, vol. 17, no. 1–3, pp. 185–204, Aug. 1981.

[16] A. J. Lipton, H. Fujiyoshi, and R. S. Patil, “Moving target classification and tracking from real-time video,” in *Proc. IEEE Workshop Appl. Comput. Vision*, Oct. 1998, pp. 8–14.

[17] S. Yalamanchili, W. N. Martin, and J. K. Aggarwal, “Extraction of moving object description via differencing,” *Computer Graphics Image Process.*, vol. 18, no. 2, pp. 188–201, Feb. 1982.

[18] J. E. Ha and W. H. Lee, “Foreground objects detection using multiple difference images,” *Optical Eng.*, vol. 49, no. 4, p. 047201, Apr. 2010.

[19] H. Wang and D. Suter, “A consensus-based method for tracking: Modelling background scenario and foreground appearance,” *Pattern Recognit.*, vol. 40, no. 3, pp. 1091–1105, 2007.

[20] A. Manzanera, “Local jet feature space framework for image processing and representation,” in *Proc. Int. Conf. SITIS*, 2011, pp. 261–268.

[21] A. Manzanera and J. C. Richefeu, “A robust and computationally efficient motion detection algorithm based on Σ-Δ background estimation,” in *Proc. ICVGIP*, 2004, pp. 46–51.

[22] G. Pajares, “A hopeld neural network for image change detection,” *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1250–1264, Sep. 2006.

[23] D. Culibrk, O. Marques, D. Socek, H. Kalva, and B. Furht, “Neural network approach to background modeling for video object segmentation,” *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1614–1627, Nov. 2007.

[24] A. Manzanera and J. C. Richefeu, “A new motion detection algorithm based on Σ-Δ background estimation,” *Pattern Recognit. Lett.*, vol. 28, pp. 320–328, Feb. 2007.

[25] M. Oral and U. Deniz, “Centre of mass model—a novel approach to background modelling for segmentation of moving objects,” *Image Vis. Comput.*, vol. 25, no. 8, pp. 1365–1376, Aug. 2007.

[26] W. Wang, J. Yang, and W. Gao, “Modeling background and segmenting moving objects from compressed video,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 5, pp. 670–681, May 2008.

[27] S. C. Huang, “An advanced motion detection algorithm with video quality analysis for video surveillance systems,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 1, pp. 1–14, Jan. 2011.

[28] B. H. Do and S. C. Huang, “Dynamic background modeling based on radial basis function neural networks for moving object detection,” in *Proc. IEEE ICME*, Jul. 2011, pp. 1–4.

[29] C. Stauffer and W. E. L. Grimson, “Learning patterns of activity using real-time tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 747–757, Aug. 2000.

[30] D. Zhou and H. Zhang, “Modified GMM background modeling and optical flow for detection of moving objects,” in *Proc. Int. Conf. Syst., Man, Cybernet.*, vol. 3. Oct. 2005, pp. 2224–2229.

[31] Z. Zivkovic, “Improved adaptive Gaussian mixture model for background subtraction,” in *Proc. Int. Conf. Syst., Man, Cybernet.*, vol. 2. 2004, pp. 28–31.

[32] K. T. P. Pakorn and B. Richard, “A real time adaptive visual surveillance system for tracking low-resolution color targets in dynamically changing scenes,” *Image Vis. Comput.*, vol. 21, no. 10, pp. 913–929, Sep. 2003.

[33] P. KaewTraKulPong and R. Bowden, “An improved adaptive background mixture model for real-time tracking with shadow detection,” in *Proc. 2nd Eur. Workshop Adv. Video Based Surv. Syst.*, Sep. 2001.AQ: Please provide page range in Ref. [].

[34] D. S. Lee, “Effective Gaussian mixture learning for video background subtraction,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 827–832, May 2005.

[35] H. C. Zeng and S. H. Lai, “Adaptive foreground object extraction for real-time video surveillance with lighting variations,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 1, Apr. 2007, pp. 1201–1204.

[36] B. Shoushtarian and N. Ghasem-aghaee, “A practical approach to real-time dynamic background generation based on a temporal median filter,” *J. Sci. Islamic Republic Iran*, vol. 14, no. 4, pp. 351–362, 2003.

[37] L. Maddalena and A. Petrosino, “A self-organizing approach to background subtraction for visual surveillance applications,” *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1168–1177, Jul. 2008.

[38] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Background modeling and subtraction by codebook construction,” in *Proc. ICIP*, vol. 5. 2004, pp. 3061–3064.

[39] O. Barnich and M. Van Droogenbroeck, “ViBe: A universal background subtraction algorithm for video sequences,” *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, Jun. 2011.

[40] J. Moody and C. J. Darken, “Fast learning in network of locally-tuned processing units,” *Neural Comput.*, vol. 1, no. 2, pp. 281–294, 1989.

[41] F. Girosi and T. Poggio, “Networks and the best approximation property,” *Biol. Cybern.*, vol. 63, no. 3, pp. 169–176, 1990.

[42] R. B. Fisher, *Change Detection in Color Images*, (1999) [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/PAPERS/iccv99.pdf

[43] A. R. Smith, "Color gamut transform pairs," *Computer Graphics*, vol. 12, no. 3, pp. 12–19, 1978.

[44] M. T. Musawi, W. Ahmed, K. H. Chan, K. B. Faris, and D. M. Hummels, "On the training of radial basis function classifiers," *Neural Netw.*, vol. 5, no. 4, pp. 595–603, Jul.–Aug. 1992.

[45] H. Simon, Neural networks, in *A Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice-Hall, 1994.

[46] M. J. Er, S. Wu, J. Lu, and H. L. Toh, "Face recognition with radial basis function (RBF) neural networks," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 697–710, May 2002.

[47] G. Gualdi, A. Prati, and R. Cucchiara, "Video streaming for mobile video surveillance," *IEEE Trans. Multimedia*, vol. 10, no. 6, pp. 1142–1154, Oct. 2008.

[48] C. Benedek and T. Sziranyi, "Bayesian foreground and shadow detection in uncertain frame rate surveillance videos," *IEEE Trans. Image Process.*, vol. 17, no. 4, pp. 608–621, Apr. 2008.

[49] C. Y. Chen, T. M. Lin, and W. H. Wolf, "A visible/infrared fusion algorithm for distributed smart cameras," *IEEE J. Selected Topics Signal Process.*, vol. 2, no. 4, pp. 514–525, Aug. 2008.

[50] M. Albanese, R. Chellappa, V. Moscato, A. Picariello, V. S. Subrahmanian, P. Turaga, and O. Udrea, "A constrained probabilistic Petri net framework for human activity detection in video," *IEEE Trans. Multimedia*, vol. 10, no. 6, pp. 982–996, Oct. 2008.

[51] F. Bartolini, A. Tefas, M. Barni, and I. Pitas, "Image authentication techniques for surveillance applications," *Proc. IEEE*, vol. 89, no. 10, pp. 1403–1418, Oct. 2001.

[52] D. Avitzour, "Novel scene calibration procedure for video surveillance systems," *IEEE Trans. Aerospace Electronic Syst.*, vol. 40, no. 3, pp. 1105–1110, Jul. 2004.

Shih-Chia Huang received the Doctorate degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2009.

He is currently an Associate Professor with the Department of Electronic Engineering, National Taipei University of Technology, Taipei. He has published more than 25 journal and conference papers and holds more than 30 patents in the U.S. and Taiwan. His current research interests include image and video coding, wireless video transmission, video surveillance, error resilience and concealment techniques, digital signal processing, cloud computing, mobile applications and systems, embedded processor design, and embedded software and hardware codesign.

Dr. Huang was presented the Kwoh-Ting Li Young Researcher Award in 2011 by the Taipei Chapter of the Association for Computing Machinery.

Ben-Hsiang Do received the B.S. degree in electrical engineering from National Chi Nan University, Nantou, Taiwan, in 2009, and the M.S. degree from the Graduate Institute of Computer and Communication Engineering, National Taipei University of Technology, Taipei, Taiwan, in 2011.

His current research interests include computer vision, video surveillance, neural networks, and embedded systems.