International Journal of Pattern Recognition
and Artificial Intelligence
Vol. 25, No. 1 (2011) 1−35

DOI: 10.1142/S0218001411008464

World Scientific
www.worldscientific.com

BACKGROUND ESTIMATION IN KERNEL SPACE

HAMIDREZA BARADARAN KASHANI
Electrical Department, Ferdowsi University of Mashhad
Azadi Square, Mashhad, Iran
hamidreza.baradaran@gmail.com

HADI SADOGHI YAZDI
Computer Department, Ferdowsi University of Mashhad
Azadi Square, Mashhad, Iran
h-sadoghi@um.ac.ir

SEYED ALIREZA SEYEDIN
Electrical Department, Ferdowsi University of Mashhad
Azadi Square, Mashhad, Iran
seyedin@um.ac.ir

One problem in background estimation is the inherent change in the background such as waving tree branches, water surfaces, camera shakes, and the existence of moving objects in every image. In this paper, a new method for background estimation is proposed based on function approximation in kernel domain. For this purpose, Weighted Kernel-based Learning Algorithm (WKLA) is designed. WKLA includes a weighted type of kernel least mean square algorithm with ability to function approximation in the presence of noise. So, the proposed background estimation method includes two stages: firstly, a novel algorithm for outlier detection namely Fuzzy Outlier Detector (FOD) is applied. Then obtained results are fed to the WKLA. The proposed approach can handle scenes containing moving backgrounds, gradual illumination changes, camera vibrations, and non-empty backgrounds. The qualitative results and quantitative evaluations on various indoor and outdoor sequences relative to existing approaches show the high accuracy and effectiveness of the proposed method in background estimation and foreground detection.

Keywords: Background estimation; fuzzy outlier detector; weighted kernel-based learning algorithm.

1. Introduction

Background estimation in video streams is often the first crucial step of information extraction in many computer vision applications, including automatic video surveillance, traffic monitoring, people tracking, military application, and semantic description of videos. It includes the removal of moving regions as foreground objects

from the background layers and the maintenance of changes relevant to the background.

1.1. *Background estimation/modeling approaches*

Many algorithms have been proposed for this purpose. A common principle of these algorithms is to construct a model of the static or dynamic background in order to compare this model to each new frame for distinguishing the regions of unusual motion. Some of the commonly used algorithms have a good performance when the background is static. In a static background, each pixel of the background can be represented by one value or state, corrupted by noise. It means that a pixel is a visual display of one and only one object in the scene. It is valid in indoor scenes where there is no background motion. However, because of inherent changes in the real background, the background may not be completely stationary. Nonstationary backgrounds would be caused by waving leaves, fluttering flags, rippling water, fluorescent light, and so on. Even when the background is static, camera jittering and signal noise may still cause nonstationary problems. In these backgrounds, each pixel of the background has a multimodal representation, so that a pixel is a visual representation of one or more objects in the scene.

Several classifications of methods for performing background estimation/modeling have been proposed in the recent literature (see surveys in Refs. 4, 12 and 31). Here, we classify the existing methods into several categories based on a common basic technique or concept employed for each group. Then we review some widely used approaches of each group.

(i) *Filter-based approaches*

Many linear and nonlinear filter-based methods have been used to model background pixels. For example, median filters in Refs. 3 and 7, $\Sigma - \Delta$ filters in Refs. 1, 29 and 41, minimum-maximum filter,[15] Wiener filter,[42] single Gaussian[46] and Kalman filtering[19,22] for updating the model presented in Ref. 46 are some approaches of this class. Although they have fast computation, ghost effects in the estimated background, lack of robustness to the nonstationary backgrounds, and crowded scenes are main disadvantages of the group's approaches.

(ii) *Mixture of Gaussians-based approaches*

Since a single Gaussian assumption will not hold for the probability density function (pdf) of the pixel intensity values in multimodal scenes, a mixture of Gaussians (MoG) modeling technique was proposed in Refs. 13, 37 and 38 to solve this problem. In these models, a few Gaussians are used to represent the color distributions at each background pixel. The parameters (mean, variance, and weight) for each Gaussian are updated using an IIR filter to adapt to gradual background changes. However, there are several shortcomings for mixture learning methods. First, fast variations of background cannot be accurately modeled with only a few Gaussians. Second, even

using incremental expectation maximization (EM) algorithm, parameter estimation and its convergence are noticeably slow where the Gaussians adapt to a new cluster.[39] Finally, it has relatively high computational complexity and parameters should be tuned carefully. Many authors have proposed improvements and extensions to this algorithm. New update algorithms for learning mixture models were presented in Ref. 18. They also detected moving shadows using the proposed model. In Ref. 48, an online algorithm which estimates the MoG parameters and simultaneously adapts the number of components based on the multimodality of each pixel was used. Also, in order to speed up the convergence, a recursive filter formulation was proposed in Ref. 24.

(iii) *Kernel density estimation-based approaches*

To address the issues of parametric methods like MoG, a nonparametric kernel density estimation (KDE) method for pixel-wise background modeling was proposed in Ref. 11. In the method, the probability density function (pdf) for pixel intensity is estimated directly from the data without any assumptions about the underlying distributions. Memory and time consuming, choosing a proper kernel bandwidth for each pixel, and the model convergence in situations where the illumination suddenly changes are major issues to be addressed in these methods. A modified version of Ref. 11 as an adaptive kernel density estimation (AKDE) algorithm was also presented in Ref. 39. Against conventional KDE in Ref. 11, the new algorithm considered dependencies between the pixel features and applied an independent threshold for each pixel instead of a global threshold for all pixels. A good discussion of kernel estimation techniques can be found in Ref. 34.

(iv) *Hidden markov model-based approaches*

In Refs. 30 and 32, Hidden Markov Models (HMM) were proposed to handle the environments such as night/day and sunny/cloudy. The model was able to learn sudden illumination changes. However, slow model training speed and sensitivity to model selection and initialization are problems of these methods.[39]

(v) *Neural network-based approaches*

Background modeling based on neural network structures has been also proposed. In Ref. 8, a feed-forward neural network to achieve background subtraction as a combination of a probabilistic neural network and a winner-takes-all neural network was proposed. In addition, the rules for adaptation of network weights based on Bayesian formulation were employed. Another neural network with simple structure was presented in Ref. 28. The method generates the background model by learning in a self-organizing manner many background variations, i.e. background motion cycles, seen as trajectories of pixels in time. The algorithm detects motion based on the learnt background model through a map of motion and stationary patterns. Both approaches (i.e. Refs. 8 and 28) are robust to moving backgrounds and crowded

scenes. Details about self-organizing maps (SOMs) or Kohonen networks can be seen in Ref. 21.

(vi) *Fuzzy-based approaches*

Recently, background modeling based on fuzzy concepts has been presented. In Ref. 36, fuzzy background subtraction and fuzzy running average algorithms were proposed instead of classic versions of them. A novel fuzzy background subtraction based on cellular automata (CA) was also presented in Ref. 35. In that approach, each frame sequence was considered as a 2D cellular space, and so each pixel as a cell of cellular automata. Based on this assumption, each frame sequence was modeled by a cellular automata and specific cellular automata rules applied to pixels. A good survey on CA can be found in Ref. 14. However, both approaches in Refs. 35 and 36 applied only to traffic scene sequences, and their performance on scenes including moving backgrounds is problematic. Using fuzzy operators such as Sugeno integral[47] and Choquet integral[9] as aggregation operators for fusing color and texture features are other presented approaches based on fuzzy concepts and theories.

1.2. *Motivation*

After a study of the aforementioned works, the issue of background estimation in this paper is considered from the viewpoint of a function estimation problem. We also interpret the foreground samples as outliers relative to the background ones. So, we propose a new function estimator based on kernel machine joint with learning schemes which is named Weighted Kernel-based Learning Algorithm (WKLA). It is capable of good approximation of functions in the presence of noise. A Fuzzy Outlier Detector (FOD) is also proposed which adapts to our interpretation of outlier. Finally, the background estimation problem is addressed by applying the obtained results by FOD to WKLA approach. The presented method can remove the influences of moving objects in generated background perfectly. It can also retain the dynamic properties of the background as much as possible. Most of the ghost effects and noises are reduced in the proposed algorithm. No requirement to empty background (nonempty background problem), high accuracy, and fair computational complexity are other features of the proposed method. The experimental results over indoor and outdoor scenes demonstrate the accuracy and effectiveness of the proposed method. These results are displayed as both qualitative and quantitative evaluations.

The paper is organized as follows: Section 2 provides an introduction to the Kernel Least Mean Square (KLMS) algorithm. Section 3 explains the proposed WKLA approach and two examples on function estimation with noisy data and outliers. Proposed background estimation approach using WKLA is described in Sec. 4. Experimental results are shown in Sec. 5. Finally, conclusions and future work are given in Sec. 6.

2. Kernel Least Mean Square Algorithm

The fundamental concept of kernel algorithms is to transform the data $\boldsymbol{x}_i$ from the input space to a high dimensional Hilbert space of vectors $\boldsymbol{\Phi}(\boldsymbol{x}_i)$. This space is often called feature space where the inner products can be calculated using a positive definite kernel function satisfying Mercer's conditions:[43]

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{\Phi}(\boldsymbol{x}_i), \boldsymbol{\Phi}(\boldsymbol{x}_j) \rangle \tag{1}$$

where K is a kernel function, $\boldsymbol{\Phi}$ is a mapping, and $\langle \ldots \rangle$ represents the inner product in the kernel Hilbert space. This idea is commonly known as the "kernel trick." Without loss of generality, in this paper we will only consider the translation-invariant radial basis (Gaussian) kernel, which is the most greatly used Mercer kernel.

$$K(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right) \tag{2}$$

where σ is the kernel bandwidth parameter. You can find further information about kernel methods in Refs. 16 and 33. The basic idea in KLMS algorithm is to perform the linear Least Mean Square (LMS) algorithm[45] in the kernel feature space.

Figure 1 shows the input vector $\boldsymbol{u}_n$ being transformed to the infinite feature vector $\boldsymbol{\Phi}(\boldsymbol{u}_n)$, whose components are then linearly combined by the infinite dimensional weight vector Ω_n to generate output y_n.

In this figure, d_n is the desired output and $e_n = d_n - y_n$. Notice that depending on the choice of the kernel, the feature space can be infinite dimensional. When Gaussian kernel is used, the feature space corresponds to an infinite dimensional Hilbert space. The estimated output y_n is calculated by

$$y_n = \langle \Omega_n, \boldsymbol{\Phi}(\boldsymbol{u}_n) \rangle \tag{3}$$

where Ω_n is the weight vector in feature space at time n.

The cost function is defined as $J_n = E[(d_n - y_n)^2]$ which can be minimized with respect to Ω. This results in an update rule of weight vector as below

$$\Omega_{n+1} = \Omega_n + 2\eta e_n \boldsymbol{\Phi}(\boldsymbol{u}_n) \tag{4}$$

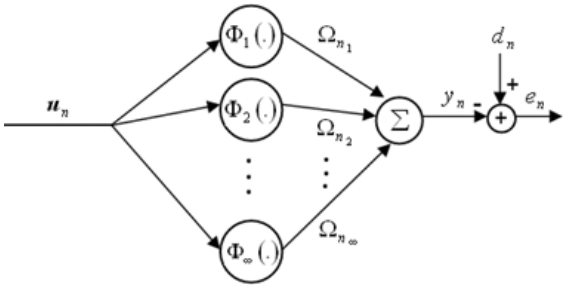


Fig. 1. Filtering scheme in the feature mapped space.

where η is the step size of the learning algorithm. Non recursive type of (4) can be written as

$$\Omega_n = \Omega_0 + 2\eta \sum_{i=0}^{n-1} e_i \boldsymbol{\Phi}(\boldsymbol{u}_i) \tag{5}$$

By choosing $\Omega_0 = 0$ (hence, $e_0 = d_0$), the final expression for Ω_n becomes

$$\Omega_n = 2\eta \sum_{i=0}^{n-1} e_i \boldsymbol{\Phi}(\boldsymbol{u}_i) \tag{6}$$

Based on (3) and (6) the output at n is given by

$$y_n = \langle \Omega_n, \boldsymbol{\Phi}(\boldsymbol{u}_n) \rangle = \left\langle 2\eta \sum_{i=0}^{n-1} e_i \boldsymbol{\Phi}(\boldsymbol{u}_i), \boldsymbol{\Phi}(\boldsymbol{u}_n) \right\rangle = 2\eta \sum_{i=0}^{n-1} e_i \langle \boldsymbol{\Phi}(\boldsymbol{u}_i), \boldsymbol{\Phi}(\boldsymbol{u}_n) \rangle \tag{7}$$

And finally by exploiting the kernel trick in the above equation, it yields

$$y_n = \eta \sum_{i=0}^{n-1} e_i K(\boldsymbol{u}_i, \boldsymbol{u}_n) \tag{8}$$

Equation (8) is named KLMS algorithm. More details about this algorithm can be found in Ref. 27.

3. The Proposed Weighted Kernel-Based Learning Algorithm

Function estimation is used to model a desired function or an input−output relation from a set of input−output sample data that unfortunately often suffers from noise and outliers in real systems. To overcome this issue, this paper presents a new estimator using a Weighted Kernel-based Learning Algorithm (WKLA) for the approximation of nonlinear functions even with a given input data set including noisy samples.

In the proposed approach, the concept of the TSK (Takagi−Sugeno−Kang) fuzzy modeling approach[26] is adopted. Unlike conventional modeling approaches, where a single model is used to describe the global behavior of a system, TSK modeling is essentially a multimodel approach in which simple submodels are combined to describe the global behavior of the system. The concept of the TSK fuzzy models is simply discussed as follows. Generally, a TSK fuzzy model consists of a set of if-then rules of the following form

$$\begin{aligned} R^i: \text{ if } x^1 \text{ is } A_1^i, \quad x^2 \text{ is } A_2^i, \ldots, x^q \text{ is } A_q^i \\ \text{then } h^i = f_i(\boldsymbol{x}; \boldsymbol{a}^i) = a_0^i + a_1^i x^1 + \cdots + a_q^i x^q \end{aligned} \tag{9}$$

for $i = 1, 2, \ldots, C$, where C denotes the number of rules, A_j^i is the corresponding fuzzy set (i.e. membership function), and $\boldsymbol{a}^i = (a_0^i, \ldots, a_q^i)$ is the parameter set in the

consequent part. The predicted output of the fuzzy model is inferred as

$$\hat{y} = \frac{\sum_{i=1}^{C} w^i h^i}{\sum_{i=1}^{C} w^i} \tag{10}$$

where $h^i = f_i(\boldsymbol{x}_T; \boldsymbol{a}^i) = a_0^i + a_1^i x_T^1 + \cdots + a_q^i x_T^q$ and is the output of the rule R^i for a test sample x_T and the weight w^i is obtained by $A_j^i(\boldsymbol{x}_T)$. Both the parameters in the premise parts and the consequent parts for the TSK fuzzy model are required to be identified. Besides, the number of rules should be specified.[5] For those parameters in the TSK fuzzy models, some methods are proposed in Ref. 26. With such concept, we present a new kernel-based estimator namely WKLA which models the learning input data set as a set of fuzzy rules. Equivalently, we can say that WKLA generates a fuzzy inference system (FIS). It includes three main stages which are explained as follows.

Stage 1: Data partitioning and generation of a weight assigning mechanism
In this stage, we should divide input data into training subsets. We can use an unsupervised learning algorithm like fuzzy c-means (FCM) clustering algorithm for this division. Let $\{(x_i, y_i), i = 1, \ldots, N\}$ be input data set. By applying FCM algorithm to the data of $x_i (i = 1, \ldots, N)$, a matrix of membership values is obtained as below

$$U = [u'_{ij}], \quad i = 1, \ldots, N, \quad j = 1, \ldots, C \tag{11}$$

where C is the number of clusters or partitions, u'_{ij} is the degree of membership of x_i in the cluster j, and x_i is the ith training data (for more details about FCM algorithm, see Ref. 26). Based on the obtained membership values, input data points are split into training subsets φ_k by

$$\varphi_k = \{(x_i, y_i) | k = \arg\max_j u'_{ij} \text{ and the corresponding } y_i\}$$
$$\text{for } i = 1, \ldots, N, \quad j = 1, \ldots, C \tag{12}$$

As we already mentioned, WKLA generates a set of fuzzy if-then rules such that each rule models one of the training subsets. By identifying the unknown functions in the "if" and "then" parts in the first and second stages of WKLA, respectively, the proposed fuzzy model is completely learnt. In this paper, we use Gaussian function as the membership function (MF) of the premise part (i.e. "if" part) of each rule. The Gaussian MF is represented as below.

$$\text{Gaussian } (x, \mu, s) = \exp\left(-\frac{(x-\mu)^2}{2s^2}\right) \tag{13}$$

where μ and s are center and standard deviation parameters of the Gaussian membership function, respectively. So, by obtaining the parameters of each Gaussian function in each partition, (i.e. μ_j and s_j for $j = 1, \ldots, C$), the learning of the "if"

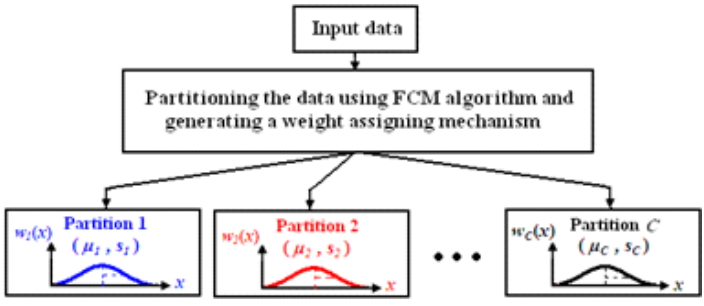


Fig. 2. First stage of the proposed WKLA.

parts of all rules is terminated. Obviously, the parameters μ_j and s_j are acquired by calculating the mean and standard deviation of the samples x_i which belong to the partition φ_j. Figure 2 shows the procedure of the first stage of WKLA.

On the other hand, we can say that a weight assigning mechanism is generated at this stage by introducing these MFs in the premise parts. However the testing process of this mechanism is performed at stage 3 of WKLA.

Stage 2: Learning a kernel-based estimator (KE) in each partition

In this stage each available training subset for any partition is an input for the KLMS algorithm like in Fig. 3. The output is a function which can be expressed as the equation below

$$\mathrm{KE}^j = \eta \sum_{l=1}^{n_j} e_l K(x, x_l), \quad x \in \varphi_j, \quad j = 1, \ldots, C \tag{14}$$

where n_j denotes the number of training data in the jth training subset and $\{e_l | l = 1, \ldots, n_j\}$ is a set of the resulting error values during training procedure for jth partition. These values are obtained for each partition based on (15) and (16) below.

$$e_l = y_l - \hat{y}_l, \quad e_1 = y_1 \tag{15}$$

$$\hat{y}_i = \eta \sum_{l=1}^{i-1} e_l K(x_i, x_l) \tag{16}$$

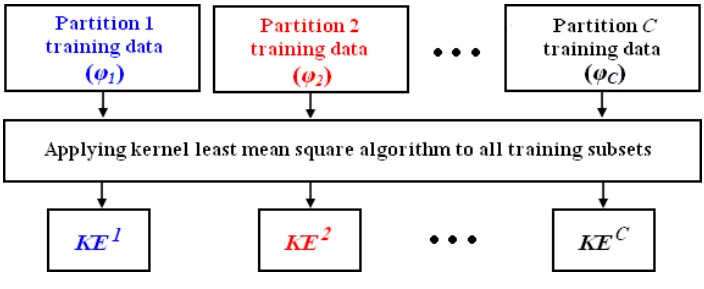


Fig. 3. Second stage of the proposed WKLA.

If the popular Gaussian radial basis function (RBF) is used as (2), then (14) can be rewritten as

$$\mathrm{KE}^j = \eta \sum_{l=1}^{n_j} e_l \exp\left(-\frac{\|x - x_l\|^2}{2\sigma^2}\right), \quad x \in \varphi_j, \quad j = 1, \ldots, C \tag{17}$$

As above, each KE is expressed in the form of the summation of nonlinear functions in each partition. It can be said that KE^j is the identified function in the "then" part (i.e. in the output) of the jth rule.

By performing the second stage, the learning of the "then" parts of all rules has ended. Now, a set of if-then rules similar to TSK fuzzy model is available which is used in the testing stage. This set of fuzzy rules can be written as

$$\begin{cases} R^1: \text{ if } x \text{ is in partition 1 then } f^1(x) = \eta \sum_{l=1}^{n_1} e_l K(x, x_l), \\ R^2: \text{ if } x \text{ is in partition 2 then } f^2(x) = \eta \sum_{l=1}^{n_2} e_l K(x, x_l), \\ \cdots \qquad\qquad \cdots \qquad\qquad \cdots \\ R^j: \text{ if } x \text{ is in partition } j \text{ then } f^j(x) = \eta \sum_{l=1}^{n_j} e_l K(x, x_l), \\ \cdots \qquad\qquad \cdots \qquad\qquad \cdots \\ R^C: \text{ if } x \text{ is in partition } C \text{ then } f^C(x) = \eta \sum_{l=1}^{n_C} e_l K(x, x_l). \end{cases} \tag{18}$$

where R^j denotes the jth rule, the term "*partition* j" is characterized with the parameters μ_j and s_j, and f^j is the same as KE^j in (17).

Stage 3: Testing procedure

The third stage is the testing procedure (Fig. 4). In this stage, each test sample should be applied to the fuzzy rules of the learnt FIS in stages 1 and 2, i.e. the FIS represented in (18). For example, let us suppose that the test sample x_T is applied to the jth rule (R^j) in (18). The "if" part of this rule results in the weight $w_j(x_T)$ which corresponds to the degree of membership of x_T in the jth partition. So, by considering Eq. (13) for the partition, the weight $w_j(x_T)$ is calculated as

$$w_j(x_T) = \exp\left(-\frac{(x_T - \mu_j)^2}{2s_j^2}\right) \tag{19}$$

It can also be said that the obtained weight ($w_j(x_T)$) demonstrates the firing strength of jth rule activated by the sample x_T. Moreover, when the test sample is

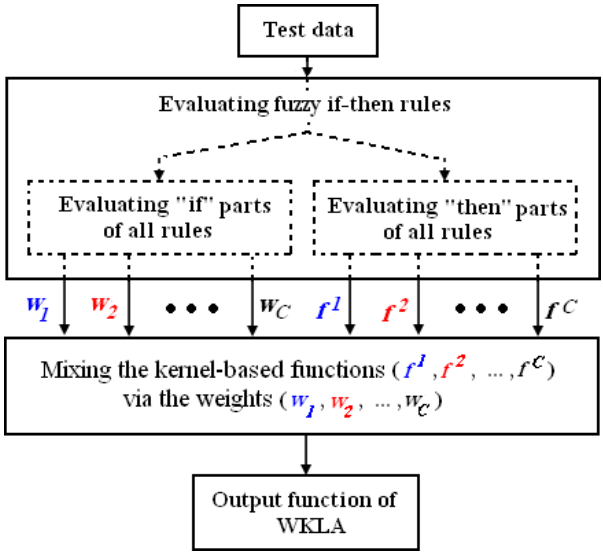


Fig. 4. Third stage of the proposed WKLA.

applied to the kernel-based function in the "then" part of the jth rule, it yields

$$f^j(x_T) = \eta \sum_{l=1}^{n_j} e_l K(x_T, x_l) \tag{20}$$

Notice that in the above equation, the sets of error values $\{e_l\}$ and learning samples $\{x_l\}$ where $l = 1, \ldots, n_j$, for each partition are available from stage 2. Finally, the overall output of the proposed WKLA is calculated by mixing the kernel-based functions $f^j(x_T)$ via the weights $w_j(x_T)$ as

$$\mathrm{WKLA}(x_T) = \frac{\sum_{j=1}^{C} w_j(x_T) f^j(x_T)}{\sum_{j=1}^{C} w_j(x_T)} \tag{21}$$

At the end of this section, two examples are illustrated to show the performance of the proposed WKLA in approximating functions.

Example 1. In the first example, a function is defined as

$$y = e^{\left(-\frac{x}{3}\right)} \sin(2x) \quad \text{with} \quad x \in [0, 10] \tag{22}$$

In this case, we use these values for the parameters of WKLA: $\eta = 0.25$, $\sigma = 1$, $C = 4$ and signal-to-noise ratio (SNR) $= 10$ dB. Moreover, 501 noisy data points are considered as learning samples for WKLA. In the literature (e.g. Refs. 2 and 5), σ is proposed as $(0.1 \sim 0.5)$ * range (x), where range (x) represents the input range of the training/test data. As it can be seen in Fig. 5, the proposed method has estimated the main signal well, despite noisy data as learning samples. The number of the used test data and mean square error (MSE) for this example are 2001 and 0.0002, respectively.

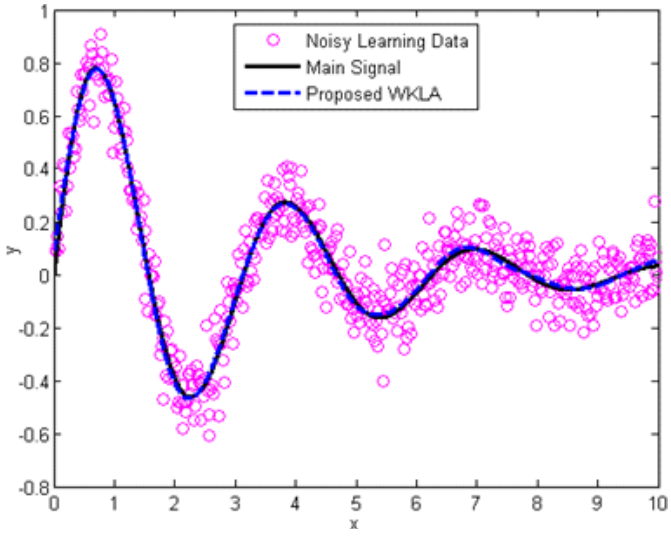


Fig. 5. Obtained result for noisy data of the function given in (22).

Example 2. In this example, the sinc function is considered and defined as

$$y = \frac{\sin(x)}{x}, \quad x \in [-10, 10] \tag{23}$$

For this case, these values of WKLA parameters are used: $\eta = 0.25$, $\sigma = 2$ and $C = 3$. Despite generating 101 data points from (23), seven artificial outliers are also added into the learning sample data pool, as shown in Fig. 6.

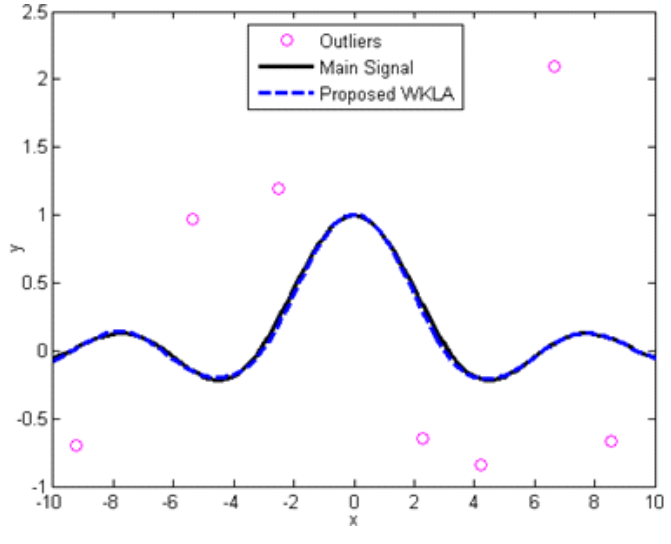


Fig. 6. Obtained result for data of a sinc function including outliers.

Notice that we here suppose that there is a process which is capable of separating outliers from normal samples (we will present an outlier detector in Sec. 4 which is applicable to background estimation). Then, we use only sample data points in the training of WKLA which are not outliers. Finally, we apply 201 data points in the test stage. Clearly, WKLA has estimated desirable values in positions where outliers have occurred. MSE, for this example, is 0.0004. In fact, with this example, we would show if outliers are removed from the learning data, it results in a good estimation of the main signal. This issue will be realized for our main purpose i.e. background estimation, in the following section.

4. WKLA-Based Background Estimation

As it can be seen from Sec. 3, WKLA can well estimate functions in the presence of noise. It suppresses effects of the noisy samples and outliers and generates desirable estimations of them. This issue motivates using WKLA in the estimation of background function from observations of each pixel. Clearly, each sample can belong to the foreground or background class. In order to estimate background using WKLA, we interpret foreground samples as outliers relative to the background samples. So, the proposed WKLA-based background estimation approach is capable of suppressing foregrounds and estimating desirable values of them using background samples.

Additionally, WKLA-based estimator would also consider most of the background observations as the noisy samples. For example, pixel observations of a static background which are usually corrupted with flicker noises and observations of a pixel in a moving background area are some types of the noisy samples. So, it is also expected that applying WKLA to the pixel observations results in suppression of flicker noises and dynamic properties in the background samples.

As already mentioned, in order to achieve a desirable estimation using WKLA, it must be trained with normal samples not outliers. Thus, we need to have an outlier detection algorithm before training of WKLA which adapts to our interpretation of outliers. For this purpose, we propose an outlier detector that is applicable to various sample functions of different pixels to separate set including outliers from the set of normal samples. Moreover, it results in a vector of membership values for each pixel which is used in both background estimation and foreground detection. It is noted that all three stages of WKLA (presented in Sec. 3) are applied to a window with the width of N video frames. Also, corresponding to the pairs (x_i, y_i) in previous section, we use pairs (t, x_{ij}^t) as input to our background estimation algorithm. t is the observation time ($t = 1, \ldots, N$) and x_{ij}^t is an observation (intensity value or RGB vector) of pixel (i, j) in frame (time) t. Throughout the paper, for simplicity, we usually remove subscript ij for sets, sequences and arrays related to the pixel (i, j). The proposed WKLA-based background estimation algorithm is shown in pseudo-code format in Fig. 7. It consists of four major parts: initialization, training, testing and adaptation. In what follows, we first present the comprehensive discussions about the parts of training, testing and adaptation, respectively. Then we also

1. ***Initialization*** $(C, \sigma, \alpha_1, \varepsilon_1, \varepsilon_2, N, \eta)$
2. ***Training step***

   process on each pixel

   $[S_{normal_{ij}}, S_{outlier_{ij}}, \boldsymbol{M}_{ij}] \leftarrow$ *Fuzzy Outlier Detector* $\left(\boldsymbol{X}_{ij}\right)$;

   $[e_{Training_{ij}}, T_{Training_{ij}}] \leftarrow$ *Background Training* $\left(T_{normal_{ij}}, S_{normal_{ij}}\right)$;
3. ***Testing step***

   process on each frame[a]

   process on each pixel

   $[B^{tt}_{f_{ij}}] \leftarrow$ *Background Estimation* $\left(t, x_{ij}, m^t_{ij}, e_{Training_{ij}}, T_{Training_{ij}}\right)$;

   $[Mask^t_{ij}] \leftarrow$ *Foreground Detection* $\left(x^{tt}_{ij}, B_{f_{ij}}, m^t_{ij}\right)$;
4. ***Adaptation step***

   Shift WKLA window $\left(e.g., \text{ as much as } (2/3).N\right)$;

   Return to step 2 ;

Fig. 7. The proposed WKLA-based background estimation.[a]

give some guidelines for how to select the proper parameter values of the method, at the end of the section.

4.1. *Training step*

In this step, the necessary information for estimating the background in each pixel is extracted from two successive stages:

(a) Applying the proposed fuzzy outlier detector to all N observations of each pixel,
(b) Applying stages 1 and 2 of WKLA for training of background model in each pixel by using detected normal samples of the pixel.

(i) *Fuzzy outlier detector in background estimation*

A simple representation of the algorithm of Fuzzy Outlier Detector (FOD) is shown in Fig. 8. The input of the algorithm is the vector/matrix of pixel observations which is denoted as $X_{ij} = [x^1_{ij}, x^2_{ij}, \ldots, x^N_{ij}]$ and its outputs are two sets including normal

[a]All frames of current window of WKLA should be processed in the testing step which have not been estimated until this stage.

Input: *pixel observations* $\left(\boldsymbol{X}_{ij}\right)$,

Output: *sets including normal samples* $\left(S_{normal_{ij}}\right)$ *and outliers* $\left(S_{outlier_{ij}}\right)$,

vector of membership values $\left(\boldsymbol{M}_{ij}\right)$.

---

process on a pixel :

-Applying a fuzzy 3 *- means clustering algorithm to pixel observations*,

-Calculation of a defined temporal criterion for each cluster,

-Determining the two sets $S_{normal_{ij}}$ *and* $S_{outlier_{ij}}$ *according to* (24) *and* (26),

-Assigning a vector consisting of membership values according to (27) *and* (28).

Fig. 8. The proposed fuzzy outlier detector algorithm.

samples and outliers which are indicated as S_{normal} and S_{outlier}, respectively, and a vector consisting of defined membership values ($\boldsymbol{M}_{ij}$) for each pixel. FOD algorithm itself includes some subsections which are expressed as follows. Also notice that, for simplicity, we present our approach in this section based on intensity observations of each pixel. However, extension of the approach to the RGB color space is straightforward and will be indicated in the subsequent section.

(a) *Applying a fuzzy* 3*-means clustering algorithm to pixel observations*

In the first stage of the proposed FOD algorithm, we divide pixel observations in the window of WKLA into some clusters.

Since the true background includes usually both static pixels and moving background pixels, and foreground may include several moving objects with different intensity values relative to the background, we use a fuzzy 3-means clustering algorithm (FCM clustering algorithm with three clusters). In other words, we believe that there is usually one cluster including only background samples, static or dynamic. The two other clusters also consist of static/dynamic background observations or/and foreground ones. Moreover, the two clusters may have less intensity values or/and more intensity ones than the first cluster. The output of this stage is a matrix as (11) with N observations and three clusters. Each observation (sample) can belong to the cluster which results in maximum membership value for the observation. Hereinafter, we denote three obtained clusters by $C_k (k = 1, 2, 3)$.

(b) *Calculation of a defined temporal criterion for each cluster*

In the FOD approach, for determining clusters corresponding to the background observations (normal samples) and clusters relevant to the foregrounds (outliers), first two features are extracted from each cluster. These features are expressed as size and maximum negative run length (MNRL). The first feature is clearly equivalent to

the number of samples of each cluster and the second one is defined as the maximum interval of time that the cluster has not recurred during the period of N frames. These are denoted as n and λ, respectively. The latter has been used in Ref. 20 as an extracted feature of each codeword considered for each pixel. Based on the fact that a pixel is most of the times enumerated as a background pixel, the cluster having a large size and a small MNRL is mostly relative to a background event. On the other hand, one having a small size and large MNRL could be a foreground event. So, we define a temporal criterion of combination of MNRL and size in the form of λ/n. This criterion is calculated for each cluster.

(c) *Determining the two sets* S_{normal} *and* S_{outlier}

In this stage of FOD algorithm, we classify three obtained clusters based on their temporal criterions into two more general sets S_{normal} and S_{outlier}. Since, it is necessary to have some normal samples for training of WKLA; we first specify the cluster with minimum criterion of λ/n as an element of set S_{normal}. So, we have

$$S_{\text{normal}} = \{C_k | k = \arg\min_m \lambda_m/n_m\}, \quad m = 1, 2, 3 \tag{24}$$

where λ_m/n_m is the temporal criterion corresponding to the cluster m. Clearly, $N = n_1 + n_2 + n_3$. Then it can be determined which of the two remaining clusters may include outliers. For this purpose, we define a model for the cluster consisting of outliers as

$$S_{\text{outlier}} = \{C_m | \lambda_m \geq T_\lambda \wedge n_m \leq T_n \wedge C_m \notin S_{\text{normal}}\}, \quad m = 1, 2, 3 \tag{25}$$

We set T_λ and T_n equal to half the number of frames in a window of WKLA, i.e. $N/2$. So, the above equation can be rewritten as

$$S_{\text{outlier}} = \{C_m | \lambda_m/n_m \geq 1 \wedge C_m \notin S_{\text{normal}}\}, \quad m = 1, 2, 3 \tag{26}$$

This means any of the remaining clusters matching the model, should be considered as a cluster including outliers and be classified as an element of set S_{outlier}, otherwise it should be clearly merged into the set of normal samples (i.e. S_{normal}). Obviously, S_{outlier} may become an empty set but S_{normal} includes at least the members of the cluster with minimum criterion λ/n.

(d) *Assigning a vector consisting of membership values* ($\boldsymbol{M}_{ij}$)

To take into account the uncertainty of the fuzzy clustering algorithm, we propose to assign a vector of membership values to each pixel. This vector is denoted by $\boldsymbol{M}_{ij}$ as

$$\boldsymbol{M}_{ij} = [m_{ij}^1, m_{ij}^2, \ldots, m_{ij}^N] \tag{27}$$

In the above equation, m_{ij}^t is expressed as below

$$m_{ij}^t = u'_{tl} \tag{28}$$

where $l = \arg_k \max u'_{tk}$ and $C_k \in S_{\text{normal}}$, for $k = 1, 2, 3$ and $t = 1, \ldots, N$.

In other words, m_{ij}^t (called a membership value) is the degree of membership of sample x_{ij}^t in the closest cluster of set S_{normal} to this sample. Notice that the values u'_{tk} have been obtained after performing the first stage of the FOD approach.

(ii) *Training of background using normal samples* (*stages* 1 *and* 2 *of WKLA*)

In this stage, training of background is performed for each pixel using its normal samples (we name this stage as "Background Training" in pseudo-code shown in Fig. 7). In fact, this stage is the same as stages 1 and 2 of WKLA which were presented in Sec. 3.

It is important to note that, few samples of foreground (few outliers) may lie in the background cluster (the cluster of normal samples) even after implementing the FOD approach. Although the proposed algorithm suppresses the effect of these outliers up to some extent, the existence of them in the training of the background may result in undesirable background estimation. Based on the fact that the outliers usually take membership values (m_{ij}^t) less than the normal samples, we define a subset S_{Training} of set S_{normal} as below

$$S_{\text{Training}} = \{x_{ij}^t | x_{ij}^t \in S_{\text{normal}} \wedge m_{ij}^t \geq Th_{ij}\} \tag{29}$$

where S_{Training} is a refined version of set S_{normal} and with the purpose of removing few outliers which probably exist in S_{normal} (even after implementing FOD) is defined. Th_{ij} is the used threshold to choose the proper samples in training of the background for the pixel (i, j) and takes a value between 0 and 1. In this stage, using a global threshold for all pixels may not be a good choice, because it is possible that the sets S_{Training} with the very small number of training samples for some pixels are obtained. Obviously, this problem causes an incorrect training process for the pixels. So, we use an adaptive threshold for each pixel as

$$\text{Th}_{ij} = \text{median}(m_{ij}^t), \quad \text{for } t \in T_{\text{normal}} \tag{30}$$

where T_{normal}, is the set of observation times corresponding to the samples of set S_{normal} and *median* represents the median value of the membership values m_{ij}^t for $t \in T_{\text{normal}}$. With this choosing, there are always at least half the samples of set S_{normal} which are used in the training of background for each pixel. Thus, by using set S_{Training} in the training procedure instead of S_{normal}, a more efficient training is accomplished and thereby we can obtain more accurate estimations of background for each pixel. Notice that the set S_{Training} is used along with the set of its corresponding observation times i.e. set T_{Training} in training process.

Like stage 2 of WKLA, the necessary outputs of this stage include sequences/sets of training errors and observation times which were denoted as e_{Training} and T_{Training}, respectively, in pseudo-code in Fig. 7. These outputs along with vector $\boldsymbol{M}_{ij}$ obtained by FOD algorithm are the main inputs of subsequent step (i.e. testing step).

4.2. *Testing step*

In this step, for each frame, a testing procedure including estimation of background and detection of foreground is performed on its pixels. These two stages are called "Background Estimation" and "Foreground Detection" in Fig. 7 which are explained as follows.

4.2.1. *Estimation of background*

First, by applying stage 3 of WKLA to the pixel in the current frame and its output sequences of training step, background value of the pixel is estimated which is denoted as $B^t_{\mathrm{WKLA}_{ij}}$. As we already mentioned, besides suppressing of foreground samples (outliers) by WKLA-based estimator, it would also suppress background samples. Apparently, the suppressing of foregrounds is desirable, but it is not good for backgrounds, because it causes the simultaneous detection of moving background pixels and foreground ones in the foreground detection stage. So, we can use a learning coefficient α_l as a constant value in the [0, 1] interval and apply it to the estimated background in any time when the sample of S_{normal} occurs. So

$$B^t_{f_{ij}} = \begin{cases} (1-\alpha_l)B^t_{\mathrm{WKLA}_{ij}} + \alpha_l x^t_{ij} & \text{if } x^t_{ij} \in S_{\mathrm{normal}} \\ B^t_{\mathrm{WKLA}_{ij}} & \text{otherwise} \end{cases} \tag{31}$$

where $B^t_{f_{ij}}$ is the final value of the estimated background in pixel (i, j) in frame t. Equation (31) can estimate the background well, if all samples of set S_{normal} are corresponding to the real normal samples. But, this condition will not always hold as we already mentioned. An alternative of (31) can be written as below

$$B^t_{f_{ij}} = \begin{cases} (1-\alpha_l)B^t_{\mathrm{WKLA}_{ij}} + \alpha_l x^t_{ij} & \text{if } x^t_{ij} \in S_{\mathrm{Training}} \\ B^t_{\mathrm{WKLA}_{ij}} & \text{otherwise} \end{cases} \tag{32}$$

This means, only for the samples of set S_{Training}, the learning coefficient α_l is applied to the initial estimated background ($B^t_{\mathrm{WKLA}_{ij}}$). Also, the final value of estimated background for other remaining samples (i.e. $x^t_{ij} \in S_{\mathrm{outlier}}$ and $x^t_{ij} \in S_{\mathrm{normal}} - S_{\mathrm{Training}}$) takes the suppressed value $B^t_{\mathrm{WKLA}_{ij}}$. Although it is a desired value for the samples of S_{outlier}, it is not necessarily good for the samples of set $S_{\mathrm{normal}} - S_{\mathrm{Training}}$. The reason for this issue can be expressed as follows. Almost in all cases, the number of real normal samples in set S_{normal} is much more than the number of probable outliers in the set. So, it is very likely that, besides the few outliers, some real normal samples are also removed after refining set S_{normal} according to (29) and (30). In fact, set $S_{\mathrm{normal}} - S_{\mathrm{Training}}$ usually includes some real normal samples along with the few outliers. Apparently, the values of estimated background from (32) are not the proper values for the real normal samples in set $S_{\mathrm{normal}} - S_{\mathrm{Training}}$. So, for solving the limitations presented in the estimation of background using (31) and (32), we use the vector $\boldsymbol{M}_{ij}$ including membership values, in the final estimation value of the background instead of the crisp estimations in those equations. Therefore, the final

equation of background estimation for all observations in the window of WKLA is written as

$$B^t_{f_{ij}} = m^t_{ij}((1-\alpha_l)B^t_{\mathrm{WKLA}_{ij}} + \alpha_l x^t_{ij}) + (1-m^t_{ij})B^t_{\mathrm{WKLA}_{ij}} \tag{33}$$

where m^t_{ij} is obtained by FOD algorithm and according to (28). With some attention to Eq. (28), it can be found that m^t_{ij} takes the relatively large value (close to 1) for normal samples and small value for those which are in set S_{outlier}. In Ref. 10, a fuzzy approach for background subtraction has been proposed. In that work, membership value of each pixel to the foreground and background class was used to update the background value of the pixel.

4.2.2. *Foreground detection*

It is conventional for detecting of foreground that the values or models of each pixel be compared to its values or models in the background, and if this deviation is larger than a heuristically selected threshold, it is selected as a pixel of foreground. In our approach, although using (33) makes considerable improvement in the estimated background especially in the moving backgrounds, there still exists little suppression of real background in the final estimation. So, it can be found that only a thresholding operation on the difference of current value and estimated background in each pixel does not work quite well. To solve this problem, we suggest using membership value m^t_{ij} in detection of foreground pixel too. Thus, an incoming pixel must meet two conditions until it is classified as foreground: (1) The absolute value of the difference between pixel value in current frame and estimated background image is more than the detection threshold ε_1 and (2) the membership value for the pixel in current frame is less than the detection threshold ε_2. Obviously, if each of above conditions is not satisfied, the respective pixel is classified as background. We show the foreground map in pixel (i, j) of frame t as Mask^t_{ij}.

$$\mathrm{Mask}^t_{ij} = \begin{cases} 1 & \text{if } |x^t_{ij} - B^t_{f_{ij}}| > \varepsilon_1 \wedge m^t_{ij} < \varepsilon_2 \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

As for the other parameters of the method, we explain how to select the proper values for these two detection thresholds, i.e. ε_1 and ε_2, in Sec. 4.4.

4.3. *Adaptation step*

We suppose that the scene illumination may undergo gradual changes all the time. So, in order to update the model for each pixel to adapt the system to gradual changes, we slide the window of WKLA on video frames such that each two consecutive windows have some overlaps on each other. In all experiments, we shift the window as much as two thirds of the width of the window (i.e. $2N/3$). In other words, let us assume $N = 300$, every 200 frames, each pixel in the scene undergoes a retraining process. This retraining process similar to the original training can be

performed with the samples of S_{Training} in the window. When the retraining is completed its results are used in the testing step. Clearly, after each stage of retraining process, it is sufficient that only 2/3 of samples at the end of the window be used for testing, because the estimation results of 1/3 at the start of the window were obtained in the previous testing step. However, in order to decrease the cost of the computation, this adaptation process is not performed at each frame.

4.4. *Selection of parameter values*

In this subsection, some guidelines for how to select values for the parameters of the method are given.

It is remembered from Sec. 3 that training data is split into C subsets or partitions in WKLA. Then, a kernel-based estimator is learned for each partition. This learning is accomplished based on the normal samples of each partition. Now, it is supposed that outliers occur for a relatively long time. If many partitions are used, it is possible that a partition is placed on the time interval of occurrence of outliers. In this case, there are not any normal samples in the interval and thereby the accurate estimation of background may not be obtained. So, we can use a few partitions like 2 or 3 in all experiments.

The kernel bandwidth parameter (σ) is selected based on the term represented in Example 1 in Sec. 3 (i.e. $(0.1 \sim 0.5) * \text{range}(x)$). According to this term, the kernel bandwidth parameter is appropriately selected to reflect the input range of training/test data. Now, in our background estimation problem, if this range is considered equal to the range of the training data, it is probable that inaccurate estimations of the background are obtained in the intervals of occurrence of outliers, especially when they occur for a relatively long time. So, we choose the range (x) equal to the width of the window of WKLA. Clearly, since this value (i.e. the width of the window) is fixed, it is independent of the time interval of occurrence of outliers. According to the above description, the problem of existing foregrounds at the start of the WKLA window or nonempty background problems can be addressed by the proposed approach as it can be seen in Figs. 10, 13, and the sixth row of Fig. 14.

According to Eq. (33) using a very large value of α_l may make some foreground samples appear in the estimated background. On the other hand, a very small value for this parameter ensures the suppression of dynamic backgrounds in the estimated background using WKLA. Since the background is usually static in most indoor scenes, we use a relatively small value of α_l in such environments. This certifies a good removal of moving foregrounds. However, removing moving objects is an essential task in background estimation, but maintaining moving backgrounds in dynamic scenes is important too. So, a larger value of the learning coefficient is chosen in these scenes. In other words, a proper value for the parameter α_l is dependent on the scene being modeled. According to the experiments, values between 0.6 and 0.7 of this parameter give good results for all of our test sequences with

Table 1. WKLA algorithm parameter values adopted for results shown in Figs. 9–14.

	WS	SM	FT	CAM	HW	MO	ToD	LS	WT	C	B	FA
N	300	300	300	300	120	300	300	300	287	300	800	450
σ	90	90	90	90	36	90	90	90	86	90	240	135
α_l	0.6	0.1	0.6	0.7	0.1	0.2	0.2	0.2	0.7	0.6	0.2	0.2
ε_1	0.02	0.02	0.02	0.03	0.02	0.02	0.02	0.03	0.03	0.03	0.02	0.03

dynamic background. The proper value of the parameter lies between 0.1 and 0.2 in the scenes with little changes of background and most on indoor environments.

In all experimental results of Sec. 5, we will choose detection threshold ε_1 between 0.02 and 0.03. Note that we first map the possible values between 0 and 255 of the absolute value in (34) to values between 0 and 1, and then use threshold ε_1 from its proposed interval. For example, when we use $\varepsilon_1 = 0.02$ (i.e. $\approx 5/255$), all pixels in which the absolute value of the difference between the current intensity value and the estimated one is more than 5 of 255 possible values, are nominated for detection operation. With this choosing of ε_1, good foreground detection may be obtained if background is estimated with high accuracy. Also, we have tested different values of ε_2 and by experimentation, best value for this parameter lies between 0.6 and 0.8.

The width of the WKLA window is denoted by the parameter N. It is important that there be no sudden illumination changes in N pixel observations in the window. So, the proposed algorithm can show a good performance in all video sequences which do not have sudden changes. However, for the sequences in which the motion speed of the foreground objects is very slow, this parameter should be set to a sufficiently high value in order to prevent the learning of the background model by the foreground observations. According to the experiments, in most cases, the proper value for this parameter is set to 300. To be specific, if the total number of available frames is less than this value, it can be set to the value equal to the number of the frames.

Finally, note that, based on various experiments, a proper value for step size of KLMS algorithm (η) is between 0.1 and 0.4.

According to the above considerations, WKLA algorithm parameter values that are not common to all experiments are detailed in Table 1. Moreover, we use $C = 2$, $\varepsilon_2 = 0.7$, and $\eta = 0.25$ as the same parameter values for all of the reported experiments in the subsequent section.

5. Experimental Results

The proposed method has been tested over many indoor and outdoor sequences.[b] In this section, we evaluate the performance of the proposed method using several examples of the different standard sequences. The chosen examples present the most

[b] All the experiments have been performed on a Pentium 4 PC, 2.54 GHz, using Matlab 6.5.

difficult issues from the background estimation and foreground detection viewpoint in video sequences.

The first five sequences are available in Ref. 17. There exist 20 manually generated ground truths for each of these sequences. So, background estimation and foreground detection in the sequences are performed for some randomly selected frames whose ground truths are available. Other remaining sequences are selected from the well-known *wallflower* dataset.[42] Since only one ground truth has been defined for each sequence of this dataset, we process a test frame of each sequence of the set.

For each sequence, the background is estimated over the selected frame(s) using the proposed method. Moreover, we compare the performance of our method in detection of foreground objects to other existing algorithms such as MoG,[48] KDE,[11] and Codebook (CB) algorithm[20] in terms of both qualitative results and quantitative evaluations.

The method of Zivkovic[48] is an improved MoG background subtraction scheme. As we pointed out in part(ii) of Sec. 1.1, it is an online algorithm in which recursive equations are used to continuously update the MoG parameters and also to simultaneously select the appropriate number of components for each pixel based on the multimodality of the pixel. In this method, a pixel is classified as foreground if that does not match any of the background Gaussian components.

As we studied in the introduction of this paper, the background pdf is given as a sum of kernels (like Gaussian kernels) centered in the most recent background values in kernel-based methods. In MoG each Gaussian describes a main mode of the pdf and is updated over time; whereas in kernel methods such as KDE, each Gaussian describes just one sample data. In fact, this method can easily deal with multimodality in background pixel distributions without specifying the number of modes in the background. In KDE, a pixel is considered as foreground if the background pdf in that is less than a global threshold.

In CB algorithm, reported in Ref. 20, the sample values of each pixel quantize into the group of codewords constituting a codebook for each pixel. Each pixel might have a different codebook size based on its sample variations. In the approach, the difference of the current image from the background model is tested with respect to color and brightness differences. If an incoming pixel meets two conditions, it is classified as background: (1) the color distortion to some codeword is less than the detection threshold, and (2) its brightness lies within the brightness range of that codeword. Otherwise, it is classified as foreground.

For all of the three compared algorithms (i.e. Refs. 11, 20 and 48), we experimented with different settings of adjustable parameters until the results seemed optimal over the entire sequence.

Also, the same post-processing (i.e. the morphological smoothing and small region elimination) is applied to all methods.

Note that all results of this section are obtained by implementing the proposed method to the color pixel observations. For this purpose, x_{ij}^t is considered as a three-dimensional vector with color components R, G and B. So, FCM algorithm is applied

to the array of three-dimensional vector observations. That is, clustering of observations of each pixel is performed in RGB color space instead of intensity space. Also, it should be considered that the error data for each sample has a vector form. However, since kernel calculation is done on time axis, it still takes a scalar form as when intensity values are used.

The visual examples and quantitative evaluations of the experiments are described in the following two subsections, respectively.

5.1. *Examples on various sequences*

Here, for each chosen sequence, we first introduce the nature and the causes of complexity of background changes in the sequence. Then the qualitative results are shown for the sequence. Notice that the discussion is limited and the reader is referred to Ref. 20 for a more in-depth treatment about the compared methods.

5.1.1. *Water Surface* (*WS*)

The sequence includes water waves as a moving background which has nonperiodic and irregular motion (see Fig. 9). As expected, the proposed method would suppress

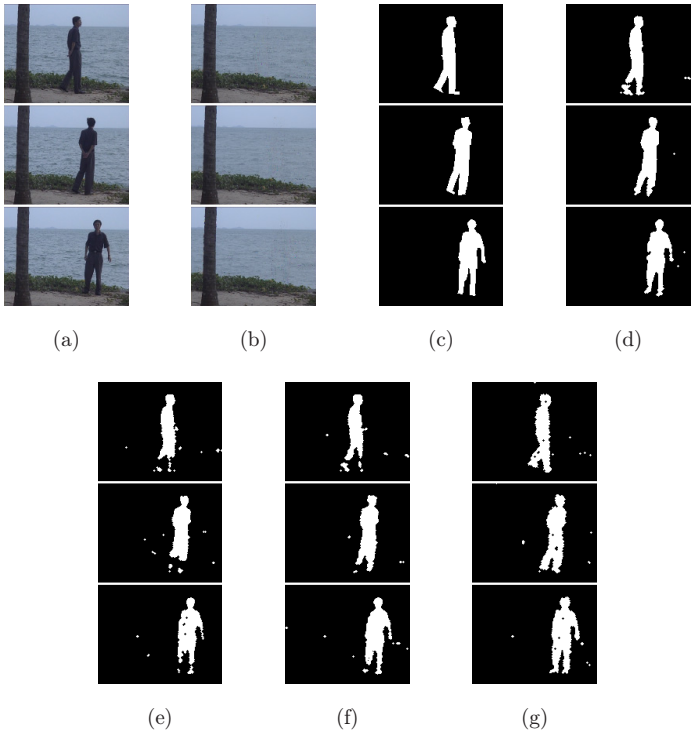


Fig. 9. Experimental results on a sequence of a water surface (WS): (a) test frames (they are frames 523, 575, and 624); (b) WKLA estimated backgrounds; (c) ground truths; (d) WKLA detection results; (e) MoG results; (f) KDE results; (g) CB results.

water waves. But, by considering membership vector $\boldsymbol{M}_{ij}$ and choosing a proper value for learning coefficient α_l, this suppression is weakened and dynamic properties of water waves are preserved in all frames as much as possible. So, only a few pixels of the water surface to the right of some foreground images of the proposed approach were misdetected as foreground pixels. However, it can be seen that the foreground has been entirely removed from the estimated background using the proposed approach and also the extracted object is very similar to the ground truth.

5.1.2. *Shopping Mall* (*SM*)

This sequence is an indoor busy scene which includes many people walking most of the time. From the results in Fig. 10, one can see that the proposed method has removed the effect of all persons from the background. Moreover, it has obtained satisfactory results in foreground detection apart from where the parts of the shadows have been detected in this environment. It should be noted that we do not use any special model for removing shadows and the results for detected foregrounds in all sequences are only based on the two-condition process (i.e. Eq. (34)) for each pixel.

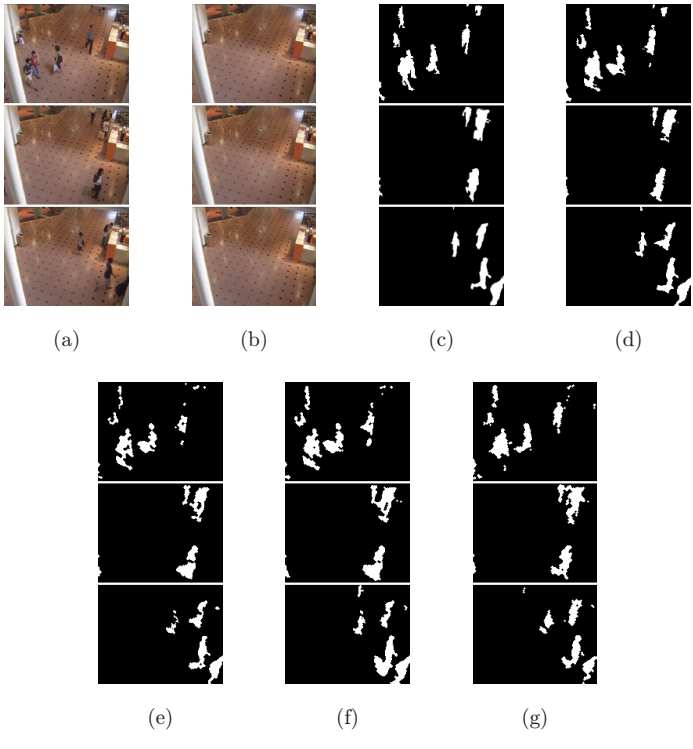


Fig. 10. Experimental results on a sequence of a shopping mall (SM): (a) test frames (they are frames 544, 672, and 980); (b) WKLA estimated backgrounds; (c) ground truths; (d) WKLA detection results; (e) MoG results; (f) KDE results; (g) CB results.

5.1.3. *Fountain* (*FT*)

In this example, the performance of the method is tested over a scene containing both moving background (in our example, a large fountain) and some objects with various color covers (Fig. 11). We observe that the proposed method has well generated a dynamic background. Moreover, by comparing the results of detection with ground truth one can see that accurate shape information of the foreground objects has been obtained by the proposed method. However, the shadows of the people on the ground surface were also misdetected as the foregrounds in the method.

5.1.4. *Campus* (*CAM*)

The example displayed in Fig. 12 comes from a video containing moving tree branches. The great motion of tree branches was caused by strong winds which can be observed from the waving yellow flag to the left of the frames. In addition, the three example frames contain the people with few pixels relative to the total size of frame and vehicles with different colors. So, it can be said that this sequence is an almost complex scene from the background estimation point of view. But, according

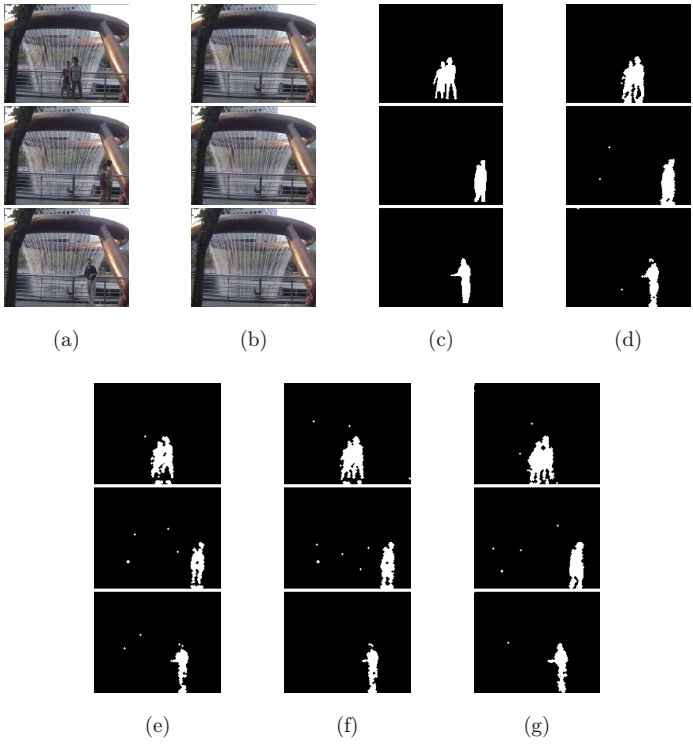


Fig. 11. Experimental results on a sequence of a big fountain (FT): (a) test frames (they are frames 184, 202, and 494); (b) WKLA estimated backgrounds; (c) ground truths; (d) WKLA detection results; (e) MoG results; (f) KDE results; (g) CB results.

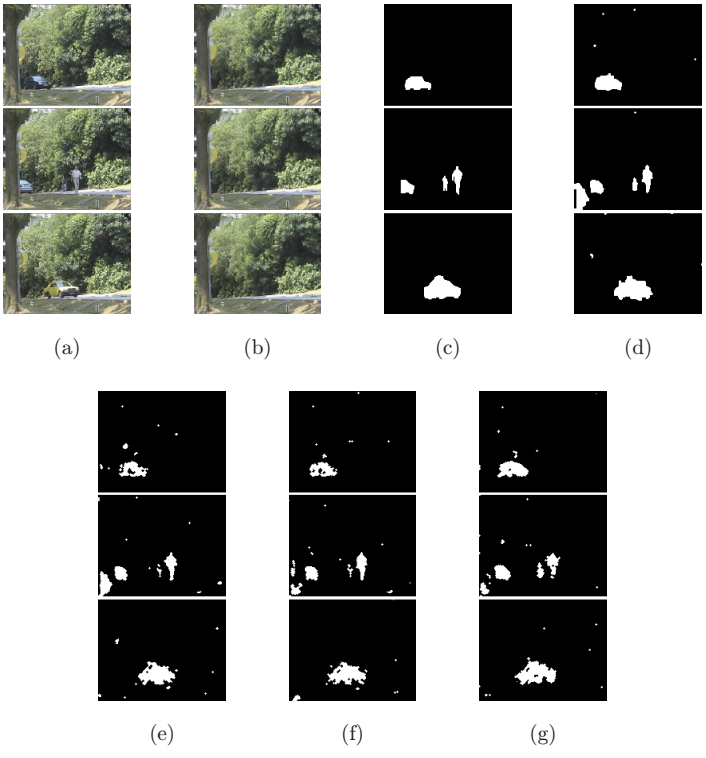


Fig. 12. Experimental results on a sequence of a campus (CAM): (a) test frames (they are frames 695, 831, and 1019); (b) WKLA estimated backgrounds; (c) ground truths; (d) WKLA detection results; (e) MoG results; (f) KDE results; (g) CB results.

to the results, the proposed method has estimated the background well and is successful in modeling the waving trees as background.

Also the humans and vehicles have been almost perfectly detected. However, a few pixels of moving flag and branches were considered as foreground pixels. In addition, due to sudden illumination changes over the tree to the left of frames, some pixels of the tree were misdetected as foreground. Since more intense background changes in this sequence compared to other dynamic tested sequence exist, choosing larger value of α_l and ε_1 can generate more accurate results of background estimation and foreground detection.

5.1.5. *Highway* (*HW*)[c]

The sequence is taken of a highway with cars running across the screen from the beginning to the end (Fig. 13). Besides, the camera is not stationary and has small vibrations in this sequence. These vibrations can be interpreted as noises which are

[c]For highway video, only a sequence including 120 frames with 15 frames per second is available. We manually generated some ground truths for this sequence.

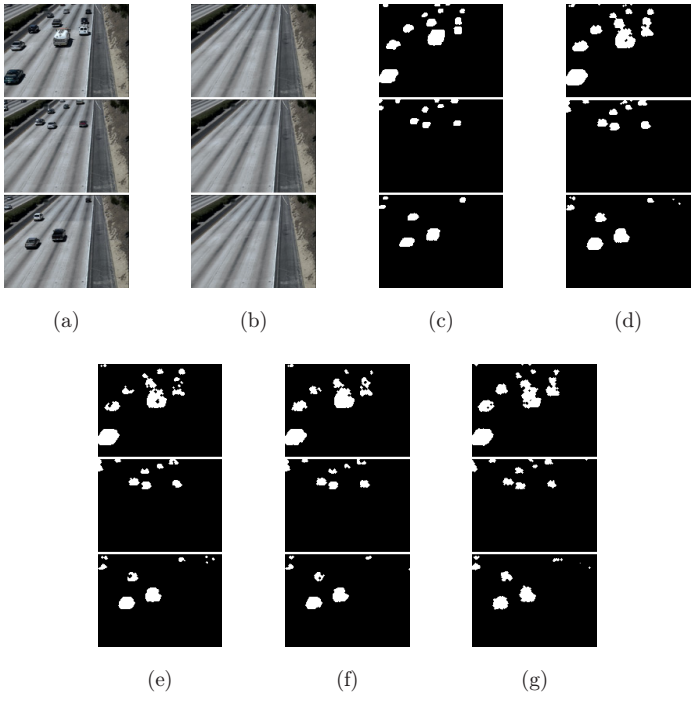


Fig. 13. Experimental results on a sequence of a highway (HW): (a) test frames (they are frames 29, 69, and 118); (b) WKLA estimated backgrounds; (c) ground truths; (d) WKLA detection results; (e) MoG results; (f) KDE results; (g) CB results.

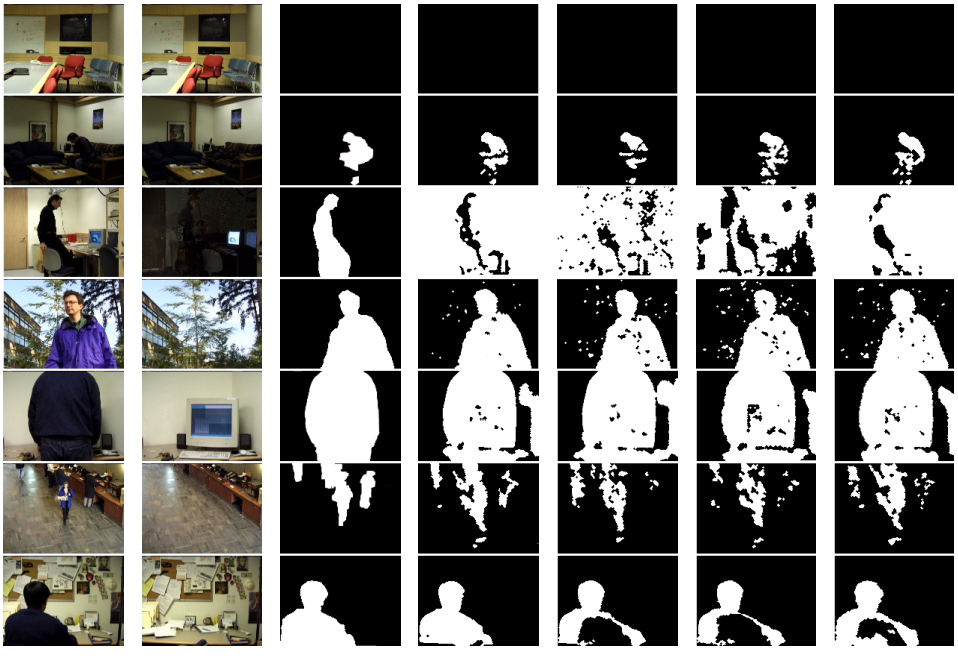
added to the values of the pixels of the scene. Obviously, using membership vector $\boldsymbol{M}_{ij}$ and even a small value of α_l, our proposed method can model these generated partial changes in the background well. Although camera vibrations exist in this sequence, we can observe that detection accuracy by the proposed approach is quite appreciable, especially, for the cars on top of the tested frames.

5.1.6. *Moved Object* (*MO*)

In the sequence, a person enters into a room, makes a phone call, and leaves. The phone and the chair are left in a different position. A frame of this sequence has been tested in the first row of Fig. 14. We can see that WKLA is able to handle the moved objects by absorbing the phone and the chair into the background when they regain stationary state. It is obvious that there is no difference between the estimated background by WKLA and the test frame and so detected foreground is similar to the ground truth.

5.1.7. *Time of Day* (*ToD*)

We process a frame of sequence ToD in the second row of Fig. 14. The sequence consists of an empty room, where light gradually brightens, simulating the moving



(a) (b) (c) (d) (e) (f) (g)

Fig. 14. Experimental results on all sequences of the *Wallflower* dataset: (a) test frames; (b) WKLA estimated backgrounds; (c) ground truths; (d) WKLA detection results; (e) MoG results; (f) KDE results; (g) CB results.

sun; finally, a man enters into the room and sits on a couch. As we already pointed out, the gradual illumination changes can be easily handled by the shifting of the WKLA window on pixel observations. So it can be observed that the proposed approach has estimated the background well. However, since the test image is very dark and the man has colored features which tend to the ones of the couch, the middle parts of the man's body were missing in detected foregrounds.

5.1.8. *Light Switch* (*LS*)

A room scene begins with the lights on. Then a person enters the room and turns off the lights for a long period. Later, a person walks in the room, switches on the light, and moves the chair, while the door is closed. The camera sees the room with lights both on and off during the training stage. The third row of Fig. 14 has been assigned to this problem. Like most of the other methods, our method is not capable of handling the Light Switch problem. This is because we do not employ any higher level processing that could be used to detect sudden changes in background.

5.1.9. *Waving Trees* (*WT*)

This outdoor sequence includes trees moving in the background and a man passing in front of the trees. A frame of this sequence has been tested in the fourth row of Fig. 14. It can be observed that the proposed method has estimated the waving trees well and so has generated a desirable foreground mask which is almost similar to the ground truth. This issue justifies the ability of the WKLA approach to cope with moving backgrounds as it can be already seen from the obtained results from sequences WS, FT and CAM. Note that the number of available frames for this sequence is 287. So, according to the pointed considerations about parameter N in Sec. 4.4, this parameter has been set to 287 in Table 1.

5.1.10. *Camouflage* (*C*)

In the sequence, a person walks in front of a monitor, which has rolling interference bars on the screen (see the fifth row of Fig. 14). Moreover, the bars include color similar to the person's clothing. So, some small regions of the clothing were misclassified as background pixels in our method. However, the shadow of the man on the wall surface was also misdetected as the foreground in the method. It should be mentioned that although Camouflage is an indoor sequence, due to existing intense changes on the screen of the monitor, we used a relatively large value for the learning coefficient α_l in the sequence (see Table 1).

5.1.11. *Bootstrapping* (*B*)

The sequence displays a busy restaurant where each frame contains walking people. A test frame of the sequence has been processed in the sixth row of Fig. 14. Significant shadows of moving persons cast on the ground surface from different directions can be observed in the sequence. On the other hand, most of the time, the person continuously walks near the desk to the left of the video frames. This issue can cause some parts of the persons' bodies to be misclassified as background regions in the detected foreground. However, according to the expressed point in Sec. 4.4, we chose a high value for the parameter N of WKLA in this sequence. This selection could adequately improve the obtained results by the proposed method. This means that the negligible effects of the moving foreground near the desk may be seen in the estimated background. Also, many parts of the foreground were detected by the proposed method compared to the other existing methods. This sequence, SM, and HW are usually called nonempty background problems.

5.1.12. *Foreground Aperture* (*FA*)

Finally, in the last row of Fig. 14, a frame of the sequence is tested using the methods. In the sequence, a person with uniformly colored shirt wakes up and begins to move slowly. One can see that the person has been perfectly removed from the estimated background by WKLA. So, the interior pixels of the foreground were explicitly

detected by the proposed two-condition foreground detection process. However, some small foreground zones at the bottom of the frame with colors similar to the background were missed in the detected foreground by the proposed algorithm.

According to Figs. 9−14, we can observe that MoG has missed some foreground pixels with colors similar to the grass at the bottom of the frames of sequence WS, the foreground pixels similar to the water fountain in sequence FT, and the parts of humans' bodies in sequences SM, B and ToD. Moreover, it also fails when sudden illumination changes occurs like in sequence LS. However, since the used MoG is an adaptive method which uses the appropriate number of components for each pixel, it shows a good performance in multimodal scenes such as WT and CAM.

Based on the fact that there is no need to assume any parametric form for the underlying probability density of pixels in KDE, this method usually gives better performance compared to the parametric density estimation techniques, i.e. MoG. The obtained results in Figs. 9−14 show this issue. However, the KDE method needs large number of training frames to generate a good model for the background in the scenes with slowly changing backgrounds or backgrounds whose changes are not periodic, such as WS. Also, using only a global threshold for all pixels in all frames is the main problem in KDE which can have an effect in decreasing the accuracy in detected foreground by this approach. As already mentioned, the KDE is used by assuming that the color features used for each pixel are independent, while the observations show that color dependencies exist in RGB color space and should be considered. In the proposed method, any assumption is not considered over independence of color channels from each other and color image processing is accomplished based on the input samples as vectors of RGB components. Moreover, KDE is not also suitable for scenarios with sudden illumination changes.

In the CB algorithm, it is not assumed that backgrounds are multimode Gaussians unlike MoG. Also, in contrast to KDE, it does not store raw samples to maintain the background model. The CB algorithm usually has a better performance in modeling of mixed backgrounds compared to KDE and MoG methods which can be seen in Figs. 9−14.

However, the CB algorithm needs an additional model called cache after training of background model and assumes that the background modeling is performed over a long period of time. In that approach, multiple information of each codeword corresponding to the codebook of each pixel should be extracted to ensure an accurate model of background. Also, an updating process of codewords of each pixel is performed in order to adapt the model to changing backgrounds. These issues cause an increase in computational complexity and a decrease in processing speed of the algorithm compared to the proposed method. It is evident that the nominal motion of the camera causes degradation in performance of compared methods, specifically, on the top of the frames in sequence HW where the cars include few pixels of the images.

Finally, it can be concluded from the results shown in Figs. 9−14 that the proposed method has outperformed the compared methods in these selected critical situations. It is clear that all these methods could not address Light Switch problem.

However, there are other approaches which have been also tested on *wallflower* dataset, beside the ones evaluated in Fig. 14. For example, a background subtraction method based on joint pixel-region analysis has been proposed in Ref. 6. The method is able to automatically select the sampling rate with which pixels in different areas are checked out, while adapting the size of the neighborhood region around each pixel is considered. It showed a good performance in sequences WT and C of *wallflower* set.

In Ref. 23, a set of local spatial-range codebook vectors is used to model the video scene and several mechanisms are also employed that update the vectors belonging to the background and foreground. The method could well address Waving Trees and Time of Day problems.

The presented approach in Ref. 44 gathers background samples and computes sample consensus to estimate a statistical model at each pixel. This model is followed by a set of morphological operations in order to solve the background subtraction issues. The best results of the method have been obtained for image sequences of WT, C, and ToD. The superiority of the proposed WKLA approach can be also justified, if one compares its shown results of *wallflower* dataset in Fig. 14 with the ones reported in Refs. 6, 23 and 44.

5.2. *Quantitative evaluations*

For measuring the accuracy of the proposed method, quantitative evaluation of the method and comparison with the existing methods are also performed in this study. We evaluate these methods using the measure used in Ref. 25. If F is a detected region and G the corresponding ground truth then the similarity measure between regions F and G is defined as

$$S(F, G) = (F \cap G)/(F \cup G) \tag{35}$$

This measure is increasing with the similarity of the detected region to the ground truth with values between 0 and 1. It reaches the maximum value of 1 if these two regions are the same.

Table 2 shows the similarity measures obtained by WKLA and the compared methods for the sequences displayed in Figs. 9−14. As already mentioned, for each sequence shown in Figs. 9−13, ground truth has been defined for twenty test frames

Table 2. Comparison of similarity values obtained by the proposed method, MoG,[48] KDE,[11] and CB.[20]

	WS	SM	FT	CAM	HW	MO	ToD	LS	WT	C	B	FA
WKLA	0.8660	0.7674	0.7388	0.6955	0.6575	1.0000	0.7214	0.1059	0.9149	0.8296	0.5706	0.9128
MoG	0.7782	0.6054	0.6715	0.6143	0.5651	1.0000	0.5984	0.1351	0.8694	0.7848	0.3398	0.4947
Kernel	0.8239	0.6265	0.7307	0.6578	0.6161	1.0000	0.6455	0.1548	0.8560	0.7982	0.4409	0.4543
CB	0.8290	0.6928	0.7134	0.6636	0.6183	1.0000	0.6809	0.1098	0.8715	0.8510	0.5318	0.6545

randomly chosen along the sequence. So, the reported results for these sequences in Table 2 have been obtained by averaging of the similarity measures evaluated over twenty frames. But for sequences belonging to Ref. 42, shown in Fig. 14, ground truth is available as a binary detection mask for only one reference frame. So, the similarity measure for one frame of each sequence in Fig. 14 is given in Table 2. The quantitative evaluation agrees with the conclusions from the visual observation of the experimental results in Figs. 9−14.

According to the quantitative results in Table 2, it is evident that the detection of objects' shadows in foreground mask makes the similarity between detected foreground and the corresponding ground truth be reduced in sequences like B, HW and SM. In the case of sequence ToD, the very low illumination level and the consequent low contrast between the moving person and the background lead the pixel-based measure values to be not very high. The existence of global strong illumination changes in sequence LS causes the similarity measures to take the minimum values in Table 2 for this sequence. In contrast, maximum similarity values have been obtained for sequence MO. Moreover, due to the occurrence of local sudden illumination changes in CAM the similarity values for this sequence are low. However, the quantitative results for sequences WT, WS and FT truly show that WKLA is successful in coping with complex background variations. Also, two hard issues such as FA and C have been satisfactorily handled by WKLA with suitable choices of parameter values for N in FA and α_l in C, leading to proper background models.

Finally, regarding the given results in Table 2, we can conclude that the proposed method (i.e. WKLA) has given the larger values of similarity measure compared to the other existing methods for almost all reported sequences.

6. Conclusions and Future Work

In this paper, we proposed a novel method in video background estimation based on a Weighted Kernel-based Learning Algorithm (WKLA). The proposed WKLA includes a weighted version of Kernel Least Mean Square (KLMS) algorithm which can well approximate the functions in the presence of noise. WKLA obtains a set of fuzzy rules by partitioning input data, generating a weight assigning mechanism, and learning a kernel-based estimator in each partition. Then final estimation of WKLA is achieved by applying fuzzy rules to the test samples. It is concluded that WKLA must be trained only with normal samples to result in an accurate estimation. So, we first proposed a Fuzzy Outlier Detector (FOD) by interpreting foreground as outlier relative to the background. Besides separating outliers set from the normal samples, FOD results in a defined vector of membership value for each pixel. Background estimation is then achieved by applying the results of FOD to WKLA approach. We also used an approach based on two complementary conditions in foreground detection. Moreover, in order to adapt the proposed system to gradual illumination changes in the scene, an adaptation process was performed by shifting the window of the WKLA on the pixel observations and retraining the window. We explained the

effect of enforcing membership values in the improvement of the estimated background and detected foreground. We also discussed the reasonable tuning of the parameters of the algorithm. The qualitative and quantitative results on different indoor and outdoor sequences demonstrated the effectiveness of the proposed approach which proves also its robustness to the critical situations of the scene like moving backgrounds, gradual illumination changes, nonempty backgrounds, and camera vibrations.

Future directions of this work are stated as follows. We are investigating a more efficient outlier detector which increases the possibility of identifying real outliers and works with less computational time. Also we plan to improve the method to handle the scenes that include sudden illumination changes and the situation where foreground objects may remain so for a long time. Further research consists of fusing intensity/color features with edges or motion features and using tracking information in order to obtain more accurate background estimation and detection results.

Acknowledgments

The authors would like to acknowledge the support provided by Education and Research Institute for ICT (ERICT), Iran.

References

1. F. C. Cheng and Y. K. Chen, Effective $\Sigma-\Delta$ background estimation for video background generation, *IEEE Asia-Pacific Services Computing Conf.* (2008), pp. 1315–1321.
2. V. Cherkarsky and Y. Ma, Practical selection of SVM parameters and noise estimation for SVM regression, *Neural Netw.* **17**(1) (2004) 113–126.
3. S. C. Cheung and C. Kamath, Robust techniques for background subtraction in urban traffic video, *Proc. EI-VCIP* (2004), pp. 881–892.
4. S. Cheung and C. Kamath, Robust background subtraction with foreground validation for urban traffic video, *J. Appl. Sign. Process* (2005).
5. C. C. Chuang, Fuzzy weighted support vector regression with a fuzzy partition, *IEEE Trans. Syst. Man and Cybernet.–Part B: Cybernet.* **37**(3) (2007) 630–640.
6. M. Cristani and V. Murino, Background subtraction with adaptive spatio-temporal neighborhood analysis, *Proc. VISAPP, Int. Conf. Computer Vision Theory and Applications* (2008), Madeira, PT.
7. R. Cucchiara, M. Piccardi and A. Prati, Detecting moving objects, ghosts and shadows in video streams, *IEEE Trans. Patt. Anal. Mach. Intell.* **25**(10) (2003) 1–6.
8. 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.* **18**(6) (2007) 1614–1627.
9. F. El Baf, T. Bouwmans and B. Vachon, Fuzzy integral for moving object detection, *IEEE Int. Conf. Fuzzy Systems* (*FUZZ-IEEE 2008*) (2008).
10. F. El Baf, T. Bouwmans and B. Vachon, A fuzzy approach for background subtraction, *ICIP* (2008), pp. 2648–2651.
11. A. Elgammal, R. Duraiswami, D. Harwood and L. S. Davis, Background and foreground modeling using nonparametric kernel density estimation for visual surveillance, *Proc. IEEE* **90**(7) (2002) 1151–1163.

12. S. Elhabian, K. El-Sayed and S. Ahmed, Moving object detection in spatial domain using removal techniques — state-of-art, *Recent Patents on Computer Science* **1**(1) (2008) 32−54.
13. N. Friedman and S. Russell, Image segmentation in video sequences: A probabilistic approach, *Proc. Conf. Uncertainty in Artificial Intell.* (1997), pp. 175−181.
14. N. Ganguly, B. K. Sikdar, A. Deutsch, G. Canright and P. P. Chaudhuri, A survey on cellular automata, Technical Report, Centre for High Performance Computing, Dresden University of Technology, (2003), pp. 1−30.
15. I. Haritaoglu, D. Harwood and L. Davis, W4: Real-time surveillance of people and their activities, *IEEE Trans. Patt. Anal. Mach. Intell.* **22**(8) (2000) 809−830.
16. R. Herbrich, *Learning Kernel Classifiers* (Cambridge, MIT Press, MA, 2002).
17. http://profsite.um.ac.ir/∼seyedin/svbm.zip.
18. P. KaewTraKulPong and R. Bowden, An improved adaptive background mixture model for real-time tracking with shadow detection, *Proc. European Workshop Advanced Video Based Surveillance Systems* (2001).
19. K. Karmann and A. von Brandt, Moving object recognition using an adaptive background memory, *Time-Varying Image Process. Moving Object Recogn.* (1990), pp. 289−296.
20. K. Kim, T. H. Chalidabhongse, D. Harwood and L. S. Davis, Real-time foreground-background segmentation using codebook model, *Real-Time Imag.* **11** (2005) 172−185.
21. T. Kohonen, *Self-Organization and Associative Memory*, 2nd ed. (Springer-Verlag, Berlin, Germany, 1988).
22. D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao and S. Russel, Towards robust automatic traffic scene analysis in real-time, *Proc. ICPR* (1994), pp. 126−131.
23. D. Kottow, M. Koppen and J. Ruiz-del-Solar, A background maintenance model in the spatial-range domain, *ECCV Workshop SMVP* (2004), pp. 141−152.
24. S. D. Lee, Effective Gaussian mixture learning for video background subtraction, *IEEE Trans. Patt. Anal. Mach. Intell.* **27**(5) (2005) 827−832.
25. L. Li, W. Huang, I. Y. H. Gu and Q. Tian, Statistical modeling of complex backgrounds for foreground object detection, *IEEE Trans. Imag. Proc.* **13**(11) (2004) 1459−1472.
26. C. T. Lin and C. S. George Lee, *Neural Fuzzy Systems* (Prentice-Hall, Englewood Cliffs, NJ, 1996).
27. W. Liu, P. P. Pokharel and J. C. Principe, Kernel least mean square algorithm, *IEEE Trans. Sign. Proc.* **56**(2) (2008) 543−554.
28. L. Maddalena, A. Petrosino and A. Ferone, Object motion detection and tracking by an artificial intelligence approach, *Int. J. Patt. Recogn. Artif. Intell.* **22**(5) (2008) 915−928.
29. N. J. B. McFarlane and C. P. Schofield, Segmentation and tracking of piglets in images, *Mach. Vis. Appl.* **8** (1995) 187−193.
30. C. Montacie, M. J. Caraty and C. Barras, Mixture splitting technic and temporal control in a HMM-based recognition system, *Proc.* (*ICSLP*) (1996), pp. 977−980.
31. M. Piccardi, Background subtraction techniques: A review, in *Proc. IEEE Int. Conf. Systems, Man, Cybernetics* (2004), pp. 3099−3104.
32. J. Rittscher, J. Kato, S. Joga and A. Blake, A probabilistic background model for tracking, *Proc. 6th Eur. Conf. Computer Vision* (2000), pp. 336−350.
33. B. Schölkopf and A. Smola, *Learning with Kernels* (Cambridge, MIT Press, MA, 2002).
34. D. W. Scott, *Multivariate Density Estimation* (Wiley Interscience, New York, 1992).
35. M. Shakeri, H. Deldari, H. Foroughi, H. Saberi and A. Naseri, A novel fuzzy background subtraction method based on cellular automata for urban traffic applications, *ICSP 2008*, pp. 899−902.

36. M. Sigari, N. Mozayani and H. Pourreza, Fuzzy running average and fuzzy background subtraction: Concepts and application, *IJCSNS* **8**(2) (2008) 138−143.
37. C. Stauffer and W. Grimson, Adaptive background mixture models for real-time tracking, in *Proc. IEEE Conf. Vis. Patt. Recogn.* (1999), pp. 246−252.
38. C. Stauffer and W. Grimson, Learning patterns of activity using real-time tracking, *IEEE Trans. Patt. Anal. Mach. Intell.* **22** (2000) 747−757.
39. A. Tavakkoli, M. Nicolescu and G. Bebis, Automatic statistical object detection for visual surveillance, in *Proc. IEEE Southwest Symp. Image Analysis and Interpretation* (2006), pp. 144−148.
40. A. Tavakkoli, M. Nicolescu and G. Bebis, Non-parametric statistical background modeling for efficient foreground region detection, *Int. J. Mach. Vis. Appl.* (2008), pp. 1−16.
41. S. Toral, M. Vargas, F. Barrero and M. G. Ortega, Improved sigma−delta background estimation for vehicle detection, *Electron. Lett.* **45**(1) (2009) 32−34.
42. K. Toyama, J. Krumm, B. Brumitt and B. Meyers, Wallflower: Principles and practice of background maintenance, in *Proc. IEEE Int. Conf. Computer Vision* (1999), pp. 255−261.
43. V. Vapnik, *Statistical Learning Theory* (Wiley Interscience: New York, 1998).
44. H. Wang and D. Suter, Background subtraction based on a robust consensus method, in *ICPR '06: Proc. 18th Int. Conf. Patt. Recogn.*, Washington, DC, USA, IEEE Computer Society (2006), pp. 223−226.
45. B. Widrow, *Adaptive Filters I. Fundamentals* (*TR 6764-6*), Stanford Electronics Laboratories, Stanford, CA, 1966, Technical Report.
46. C. Wren, A. Azarbayegani, T. Darrell and A. Pentland, Pfinder: Real-time tracking of the human body, *IEEE Trans. Patt. Anal. Mach. Intell.* **19** (1997) 780−785.
47. H. Zhang and D. Xu, Fusing color and texture features for background model, *Third Int. Conf. Fuzzy Systems and Knowledge Discovery* **4223**(7) (2006) 887−893.
48. Z. Zivkovic, Improved adaptive Gaussian mixture model for background subtraction, *Int. Conf. Pattern Recogn.* (*ICPR 2004*) **2** (2004) 28−31.

Hamidreza Baradaran Kashani received the B.S. degree in robotic engineering from Shahrood University of Technology, Shahrood, Iran, in 2007, and then he received the M.S. degree in communication engineering from Ferdowsi University of Mashhad, Mashhad, Iran, in 2010.

His research interests include image and video processing, and pattern recognition. In these fields, he is especially interested in background modeling and moving object detection and tracking in dynamic video scenes.

Hadi Sadoghi Yazdi received the B.S. degree in electrical engineering from Ferdowsi University of Mashhad in 1994, and then he received to the M.S. and Ph.D. degrees in electrical engineering from Tarbiat Modarres University of Tehran, Iran, in 1996 and 2005 respectively. He works in the Computer Science Department as an associate professor at Ferdowsi University of Mashhad.

His research interests include pattern recognition, optimization in signal processing.

Seyed Alireza Seyedin received the B.S. degree in electronics engineering from Isfehan University of Technology, Isfehan, Iran in 1986, and the M.S. degree in control and guidance engineering from Roorkee University, Roorkee, India in 1992, and the Ph.D. degree from the University of New South Wales, Sydney, Australia in 1996. He has been an Associate Professor with the Department of Electrical Engineering, the University of Mashhad (Ferdowsi), Mashhad, Iran.

His research interests include image processing, computer vision, signal processing, and pattern recognition. In these fields, he is especially interested in image analysis, motion detection and estimation in image sequences, autonomous vehicles, and diverse applications of the Radon transform.