Department of Electrical
and
Computer Systems Engineering

Technical Report
MECSE-15-2005

SACON: A Consensus Based Model for Background Subtraction

Hanzi Wang and David Suter

SACON: A Consensus Based Model for Background Subtraction

Hanzi Wang and David Suter
Department of Electrical and Computer Systems Engineering,
Monash University, Clayton Vic. 3800. Australia.

Abstract

Statistical background modeling is a fundamental and important part for many visual tracking systems and other computer vision applications. This paper presents an effective and adaptive background modeling method for detecting foreground objects in both static and dynamic scenes. The proposed method computes SAmple CONsensus (SACON) of the background samples and estimates a statistical model per pixel. SACON exploits both color and motion information to detect foreground objects. SACON can deal with complex background scenarios including non-stationary scenes (such as moving trees, rain, and fountains), moved/inserted background objects, slowly moving foreground objects, illumination changes etc. Numerous experiments on both indoor and outdoor video sequences show that the method is robust to various types of background scenarios and, compared with several state-of-the-art methods, can achieve very promising performance.

1. Introduction

Background modelling is an important and fundamental part for many computer vision applications such as real-time tracking [1, 2, 3, 4, 5], video/traffic surveillance [6, 7] and human-machine interface [8, 9]. After the background is modelled, one commonly performs "background subtraction" to differentiate foreground objects (those parts are of interest to track or recognize) from the background pixels. The result of background modelling significantly affects the final performance of these applications.

Generally speaking, a good background model should be able to achieve the following desirable properties:

- accurate in shape detection (i.e., the model should be able to ignore shadow, highlight, etc.);
- reliable in different light conditions (such as a light switched on/off, gradual illumination changes) and to the movement of background objects (e.g., if a background object is moved, that object should not be labelled as a foreground object);
- flexible to different scenarios (including both indoor and outdoor scenes);
- robust to different models of the background (i.e., a time series of observation at a background pixel can be either uni-modal or multiple-modal distributed) and robust in the training stage even if foreground objects exist;
- accurate despite camouflage (e.g., if a foreground object has similar color to the background) and foreground aperture (if a homogeneously colored object moves, many of the interior pixels of the object may not be detected);
- efficient in computation.

Unfortunately, none of the existing background models can achieve desirable performance on all of the abovementioned criteria.

In this paper, we propose a robust and efficient background modelling method, SAmple CONsensus (SACON), and we apply it to background subtraction. SACON gathers background samples and computes sample consensus to estimate a statistical model at each pixel. SACON is easy to perform but highly effective in background modelling and subtraction. Numerous quantitive experiments show the advantages of SACON over several other popular methods in background modelling/subtraction.

The organization of the remainder of this paper is as follows: in section 2, we present a short review of the previous related work. We present the SACON method in section 3 and a framework for applying SACON to background subtraction in section 4. Experiments showing the advantages of our method are provided in section 5. We investigate the influence of the parameters of SACON on the results in section 6 and summarize in section 7.

2. Related Work

There are numerous background models appearing in the literature in recent years, [2, 3, 7, 8, 9, 10, 11, 12, 13, 14]. A simple background model usually assumes that the background pixels are static over time. The foreground objects can then be obtained by subtracting the current frame from the background image.

Prominent examples are Pfinder [9] and W^4 [7]. Pfinder assumes that the pixels over time window at a particular image location are single- Gaussian distributed. W^4 models the background by maximum and minimum intensity values, and the maximum intensity difference between consecutive frames in the training stage. Although they can deal with small or gradual changes in the background and work well if the background includes only a static scene, they may fail when background pixels are multi-modal distributed (e.g., waving trees) or widely dispersed in intensity.

Several methods have been proposed to deal with multiple-model distributed background pixels. Wallflower

[10] employs a linear Wiener filter to learn and predict background changes. Wallflower works well for periodically changing pixels. However, when background pixels change dramatically or the movement of background pixels are less periodical, Wallflower is less effective in learning and predicting background changes.
Other examples in include Tracey [11] which models foreground and background by codebook vectors; and [15], where "cooccurrence" of image variations at neighbouring image blocks is employed for modelling a dynamic background.
The pixel-level Mixture of Gaussians (MOG) background model [2, 16] is effective in modeling multiple-modal distributed backgrounds. The basic idea of MOG is to assume that the observations at an image pixel can be modeled by a mixture of K Gaussians (K is usually set from 3 to 5). Let x_t be a pixel value at time t. Thus, the probability that the pixel value x_t is observed at time t is [2]:

$$P(x_t) = \sum_{i=1}^{K} \frac{w_{i,t}}{(2\pi)^{\frac{d}{2}} \left|\Sigma_{i,t}\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t-\mu_{i,t})^T \Sigma_{i,t}^{-1}(x_t-\mu_{i,t})} \qquad (1)$$

where $w_{i,t}$ is a weight, $\mu_{i,t}$ is the mean value, and $\Sigma_{i,t}$ is the covariance matrix for the ith Gaussian distribution at time t. Each channel of the color space is assumed to be independent from the other channels.
If a new observation x_t belongs to the ith Gaussian distribution, the parameters of the ith Gaussian distribution at time t will be updated as follows:

$$\begin{aligned} \mu_{i,t} &= (1-\alpha)\mu_{i,t-1} + \alpha x_t \\ \sigma_{i,t}^2 &= (1-\alpha)\sigma_{i,t-1}^2 + \alpha(x_t-\mu_{i,t})^T(x_t-\mu_{i,t}) \\ w_{i,t} &= (1-\alpha)w_{i,t-1} + \alpha \mathrm{M}_{i,t} \end{aligned} \qquad (2)$$

where α is learning rate; $\mathrm{M}_{i,t}$ is 1 when the observation matches the ith Gaussian distribution, and 0 otherwise.
MOG can adapt to a change of the background (such as gradual light change, etc.). However, there still are some limitations of MOG: for example, in the training stage, MOG usually employs a K-mean algorithm to initialize the parameters, which is slow and may be inaccurate. When the background involves a wide distribution in color/intensity, modelling the background with a small number of mixtures of Gaussian distributions is not efficient. It is also hard to set the value of the learning rate α.
A lot of variants of the MOG background model have been proposed [2, 13, 17]. Elgammal et. al. [13] presented a non-parametric background model which can handle situations where the background contains small motions such as tree branches and bushes. The method gathers recent samples per pixel and computes the non-parametric model (i.e., kernel density estimation using a Gaussian Kernel). The PDF is written as:

$$PD(x_t) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}} \left|\Sigma_{i,t}\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t-x_i)^T \Sigma_{i,t}^{-1}(x_t-x_i)} \qquad (3)$$

where N is the number of samples at a pixel.
Since the cost to compute Equation (3) at each pixel is high, several pre-calculated lookup tables for the kernel function values, given different (x_t-x_i) and the kernel bandwidth, are used to reduce the burden of computation of the algorithm. Since the kernel bandwidth is estimated by using the median absolute deviation over sample for consecutive intensity values of the pixel, the method requires all the samples to be consecutive in time, and the bandwidth estimate may be inaccurate if the distribution of the background samples is multi-model.

3. The Method of SACON

We now define a background model Sample Consensus inspired by RANSAC [18]. Following previous works (such as [1, 2, 3, 10, 12, 13]), we assume that:

1) the camera is stable during recording the video sequence. We do not assume a stable background, i.e., the background can include dynamic scenes such as moving trees, sea waves etc.
2) the different color channels are independent of each other. Previous work (e.g. [2, 13]) and our experiments validate this assumption.
3) the movements of dynamic background objects should be repetitive and should be seen during the training stage.
4) when the foreground objects or inserted/moved background objects remain static for a long time, they should be adopted into the background. It is hard to differentiate static foreground objects from inserted/moved background objects by low level processing.

Let N be the number of background samples at each pixel. For an observation at pixel m at time t—$x_t(m)$, we need to classify it into either a background pixel or a foreground pixel according to the background samples $\{x_i(m) \mid i = 1,\ldots,N\}$ at that pixel location m. Each observation $x_t = (x_t^{C_1},\ldots,x_t^{C_k})$ has k channels (e.g., in RGB color space, each observation is expressed by three channels of R, G, B). Let:

$$\Gamma_i^c(m) = \begin{cases} 1 & if \left|x_i^c(m) - x_t^c(m)\right| \le T_r \\ 0 & otherwise \end{cases} \qquad (4)$$

For each observation at time t, we form a binary mask B_t *defining the sample consensus classification* (with "one" for a background pixel and "zero" for a foreground pixel):

$$B_t(m) = \begin{cases} 1 & \sum_{i=1}^{N} \Gamma_i^c(m) \ge T_n \ \forall c \in \{C_1,\ldots C_k\} \\ 0 & otherwise \end{cases} \qquad (5)$$

where T_n is a value thresholding the number of data points that are within the error tolerance T_r of a mode.
T_n should reflect the sample size N: the larger value N is, the larger value T_n should be set; the less N is, the less T_n

should be set. Likewise, T_n should also reflect the error tolerance T_r: the larger value T_r is set, the larger value T_n should be; vice versa. Thus, T_n can be approximately set to $\tau\, T_r\, N$, where τ is a constant and is chosen empirically.

4. Framework

Having a background modelling method is not sufficient for effective background subtraction. In practice, many issues such as shadows, illumination changes, dynamic scenes etc. should be considered in designing a complete system. In this section, describe such a system employing SACON as a core.

4.1 Overview of the Framework

The major components are shown in Figure 1.

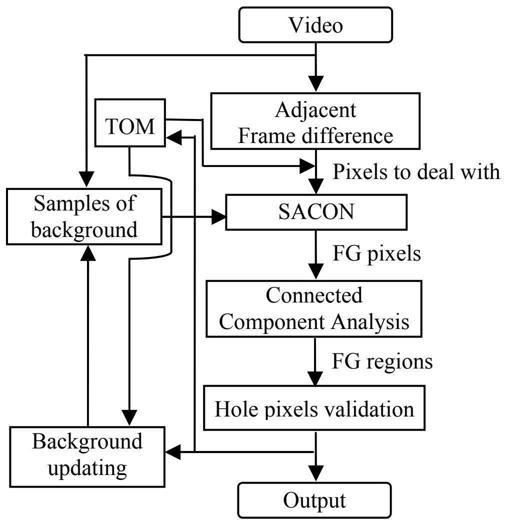


Figure 1. Block diagram of the complete framework.

Key additional elements include a Foreground Mask (FM) and a Time Out Map (TOM - see section 4.2.4). FM is used to mark foreground (FG) pixels. TOM is used to record the consecutive times that a pixel is marked as a FG pixel. As Figure 1 shows, the proposed framework mainly contains three phases. In the first phase, the adjacent frame difference method [10] is employed to extract possible foreground pixels. However, if the background includes dynamic parts, these pixels may belong to background. This issue will be solved in the next phase. In the second phase, we feed the possible foreground pixels, and also the pixels whose TOM values are larger than a value, as well as the background samples, to SACON. The output is the detected foreground (FG) pixels. Only the FM of the pixels from the first stage is updated at this stage. However, there may be holes inside the foreground regions (see section 4.2). In the third phase, we validate the pixels inside the holes of the detected FG regions, and we update the background samples and the TOM. The details of the procedure are given in the following sub-sections.

4.2 Shadow Removal and Related Issues

RGB color space is sensitive to the change of illumination thus employing RGB color space may cause incorrect labelling of shadows as foreground pixels.

When shadows appear, although the illumination part may change dramatically, it is usually assumed that the chromaticity part at the pixel is not significantly changed. Normalized color has been used in many background modelling methods, such as in [3, 5, 13, 14], to take advantage of this. The normalized chromaticity coordinates can be written as:

$$\begin{aligned} r &= R/(R+G+B) \\ g &= G/(R+G+B) \\ b &= B/(R+G+B) \end{aligned} \tag{6}$$

(Note: we scale r, g, b to the range [0, 255], assuming the 8 bit image value in each channel is used).

However, the loss of the intensity information can be a problem so some papers (such as [5, 13, 14]) promote (r, g, I) coordinates

Let (r_b, g_b, I_b) be the observed value of a background pixel x_b and (r_t, g_t, I_t) be the observed value at this pixel (i.e., x_t) in frame t. If the background is totally static, we can expect $\beta \le I_t / I_b \le 1$ when the pixel is covered by shadow and $1 \le I_t / I_b \le \gamma$ when the pixel is highlighted by strong light. Thus, the shadow can be suppressed if the following two conditions hold:

$$\begin{cases} \left| x_t^c - x_b^c \right| \le T_r & \forall c \in \{r, g\} \\ \beta \le x_t^c / x_b^c \le \gamma & c \in \{I\} \end{cases} \tag{7}$$

where β, γ are constant and are chosen empirically and T_n, is set as described in section 4.2.2.

If the background is dynamic and includes multiple modes at pixel m, we compute the sample consensus by checking how many data points of the N background samples satisfy Equation (7), if the number is larger than T_n, we label $B_t(m)$ in Equation (5) with value 1.

However, there are three problems related to Equation (7):

1) When the intensity I is small, the estimated normalized color (r, g, b) can be very noisy. This is because of the nonlinear transformation from the RGB space to the normalized rgb color space in Equation (6);
2) Deciding how to set an effective value for T_r, which is also effective for other image pixels, is hard.
3) When the chromaticity component of the foreground pixel is similar to that of the background pixel, but the intensity component difference is relatively large (e.g., the absolute difference of I_t and I_b is large, however $\beta \le I_t / I_b \le \gamma$ is still satisfied), the foreground pixel may be wrongly labelled.

We address these issues in the following subsections.

4.2.1 Normalized Color Noise

Figure 2 (a) shows one frame of the image sequence "Light Switch" (LS) in the Wallflower dataset [10]. We selected three hundred frames (frame 1001 to 1300) of LS where the light was switched off. Except for rolling interference bars on the screen, the rest of the pixels remain static. From Figure 2 (b) and (c), we can see that when the intensities of image pixels in Figure 2 (a) are low, the estimated standard variances of both *r* channel and *g* channel are high (corresponding to bright pixels in Figure 2 (b) and (c)). This means that the estimated *r* and *g* values are very noisy when the intensity values are low.

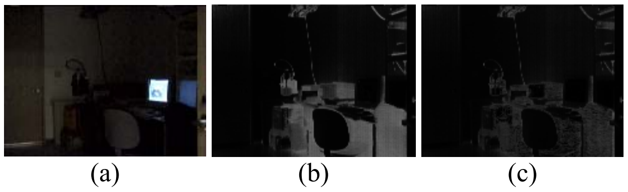


Figure 2. (a) One frame of LS; Images showing the standard variance of the red channel (b) and the green channel (c) in the normalized *rgb* color space over 300 frames of LS.

To solve this problem, we only use *I* when the intensity *I* of the pixel is lower than a threshold (*Itd*):

$$x = \begin{cases} (r, g, I) & if\ I \geq Itd \\ (I) & if\ I < Itd \end{cases} \quad (8)$$

4.2.2 Setting the Value of T_r

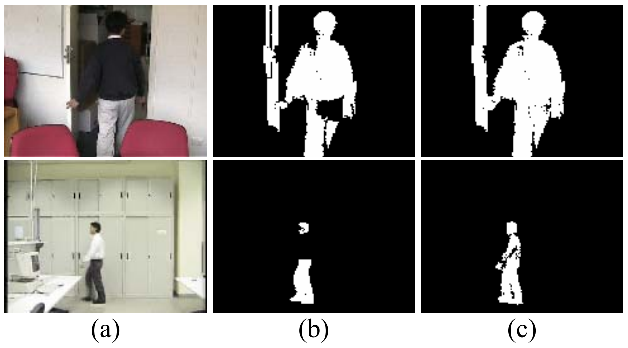


Figure 3. (a) One frame of the videos; the detected foreground pixels by (b) setting a global value to T_r and by (c) setting the values of T_r for each pixel according to Equation (9).

There are two possible ways to set the value of T_r. The first is to empirically set a global value of T_r for all pixels. To obtain an efficient value of T_r for all image pixels is hard. The second way is to estimate the standard variance σ and set T_r equal to $\eta\sigma$, (where η is usually set as 2.5 or 3). However, the estimated σ may be overestimated when the data is multi-model distributed.

We set T_r by combining the above two ways as follows:

$$T_r = \min(T_1, \eta\sigma) \quad (9)$$

where T_1 is a constant (We will discuss the influence of T_1 on the results in section 6).

From Figure 3, we can see that when we set a global T_r, some parts (e.g., part of the trousers of the person in the first row and the shirt of the person in the second row) of the foreground objects are not successfully detected. In contrast, when we set the various values of T_r for each pixel according to Equation (9), most of the foreground pixels are correctly detected (Figure 3c).

4.2.3 Validation of Pixels inside Holes

When a foreground object has similar color to the background scene, there may be holes in the detected foreground regions (i.e., the foreground pixels inside the holes are wrongly labelled as background pixels). Let us consider Equation (7), although $\beta \leq x_t^I / x_b^I \leq \gamma$ can be used to suppress shadows, the intensity information is also "damaged" to some extent. If the chromaticity component of foreground pixels is similar that of background pixels, the difference of the intensity part is large but still within the range of $\beta \leq x_t^I / x_b^I \leq \gamma$(this is notable especially when x_b^I is large), the pixels are wrongly marked. For these pixels, however, we can not simply use some hole filling techniques to remove the holes, because these holes may also be caused by the structure of the foreground object or the posture of a human being. Thus, we use a validation procedure to recheck the pixels inside the holes. For these pixels inside the holes, we use $\left|x_t^I - x_b^I\right| \leq T_I$(where T_I is a threshold for intensity channel), i.e., if the condition is not satisfied, we mark the pixels of the holes as foreground pixels; otherwise, we mark the pixels as background pixels.

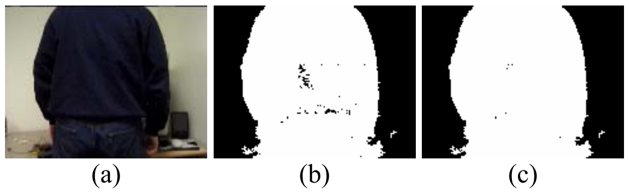


Figure 4. An example showing foreground hole pixel validation. One frame of C (a); The results without (b) and with (c) the validation procedure.

Although the validation can not correct wrong labels of foreground pixels when the color of these pixels is very similar to the background, it can improve the results obtained by Equation (7). Figure 4 shows us an example. One frame of the image sequence "Camouflage" (C) in the Wallflower dataset is shown in Figure 4 (a). The person walked into the room and stood in front of a monitor which has similar color (on the screen) to the person's dress. Figure 4 (b) shows the result without the validation procedure. We can see there are a number of holes inside the foreground object, which are wrongly marked as background pixels. Figure 4 (c) shows the result after applying the validation procedure. From Figure 4 (c) we can see that most pixels inside the holes are correctly marked as foreground pixels.

4.2.4 Updating Background Samples

When the background scene changes, the background samples should be updated to reflect the change of the background scene. Generally speaking, the background samples should be updated so that the background model can:

- adapt to light condition changes such as gradual illumination changes;
- adapt to moved or inserted background objects to the background scene.
- adapt the foreground objects, which remain static for a long time, to the background scene (e.g. in [19]).

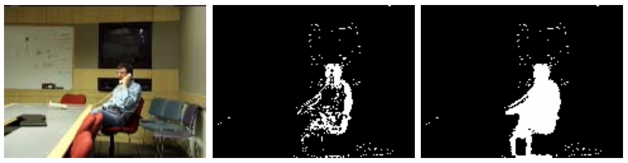
There are several methods to update the background samples [13, 20]. The simplest method is to blindly add each pixel of the current frame to the background scene. However, this method also adds the foreground pixels to the background samples. Another simple but more efficient method is to selectively add only pixels marked as background pixels to the background model while neglecting the foreground pixels. This method is efficient in gradual illumination changes. However, the method also causes some problems: for example, if a background object is moved to a new place, or if a new background object is inserted to the background scene, the method can not adaptively add the corresponding pixels of the background object to the background model.

We use a selective update mechanism to update the background samples. To incorporate the moved/inserted background object or static foreground object *t* into the background model, we use a Time Out Map (TOM). Let $TOM_t(m)$ be the time out map at pixel *m* at frame *t*. We have:

$$\begin{cases} TOM_t(m) = TOM_{t-1}(m) + 1 \ \ if\ B_t(m) = 0 \\ TOM_t(m) = 0 \qquad\qquad otherwise \end{cases} \tag{10}$$

Equation (10) shows that the TOM is used to record how long (how many frames) a pixel is continuously classified as a foreground pixel. Once the pixel is classified as a background pixel, the TOM value of that pixel is set to zero. When the value of TOM at a pixel is larger than a threshold T_{TM}, that pixel will be assigned to the background (the pixel of the object has remained in place too long).

However, we find, in some cases, pixels of moving objects, are also incorporated to the background. Figure 5 (b) shows such an example. In the image sequence "Moved Object" (MO) in the Wallflower dataset, a person entered into a room, moved the chair and sat in the chair. While he made phone call, he turned around in the chair. There is an overlapped region around the center of his body. The TOM values of the pixels of the region keep increasing because the pixels keep being marked as foreground pixels at each frame. Thus, when the TOM values are increased to be larger than T_{TM} (i.e., after a long time), these pixels are added to the background model.



(a) (b) (c)

Figure 5. (a) A frame of MO; Results obtained by updating at pixel level (b) and combination of pixel and region level(c).

To further improve the above modification, we update the background samples at both pixel level and region level. For pixels of moved background or static foreground objects, who are connected to large numbers of such pixels., we treat these pixels at region level instead of at pixel level (using Equation 10).

We judge if an object is moving or static by two criteria: the center of the object and the number of the pixels of the object. If either changes by an amount larger than a threshold, we judge the object is moving. Otherwise, we judge the object is static. If an object is judged static, we increase the TOM value of all pixels of that object by one; if an object is judged moving, we set the TOM value of the pixels of that object to zero. If the TOM value of an object is higher than T_{TM}, we add the all pixels of the object to the background samples. Figure 5 (c) shows the result obtained by background sample update at both pixel and region level. We can see the pixels at the center part of the person are correctly marked as foreground pixels.

5. Experimental Comparisons

Toyama et. al. [10] benchmarked their algorithm "Wallflower" using a set of image sequences where each sequence presents a different type of difficulty that a practical task may meet. The performance is evaluated against hand-segmented ground truth. In this section, we will evaluate our method using these image sequences and compare the performance of our method with that of five state-of-the-art methods.

A brief description of the Wallflower image sequences follows:

Moved Object (MO): A person enters into a room, makes a phone call, and leaves. The phone and the chair are left in a different position. **Time of Day (TOD):** The light in a room gradually changes from dark to bright. Then, a person enters into the room and sits down. **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. **Waving Trees (WT):** A tree is swaying and a person walks in front of the tree.

Camouflage (C): A person walks in front of a monitor, which has rolling interference bars on the screen. The bars include color similar to the person's clothing. **Boostrapping (B):** The image sequence shows a busy cafeteria and each frame contains people. **Foreground Aperture (FA):** A person with uniformly colored shirt wakes up and begins to move slowly.

For the evaluation of performance against each image sequence, we use three terms: False Positive (FP), False Negative (FN), and total error for that image sequence (*te*). FP is the number of background pixels that are wrongly marked as foreground; FN is the number of foreground pixels that are wrongly marked as background; *te* is the sum of FP and FN for each image sequence. For the evaluation of overall performance, we use TE (the sum of total error for all seven image sequences) and TE* (the sum of total error excluding the light switch image sequence).

For each result image, we eliminated the foreground pixels whose 4-connected foreground pixels number less than 8. Figure 6 and Table 1 show the results obtained by SACON, and five other state-of-the-art methods. From Figure 6 and Table 1, we can see that our method achieves the most accurate overall performance on TE and TE* among the six competitive methods. SACON also obtains the best results of total error (*te*) for image sequences of MO, TOD, WT, C and B. For MO sequence, the Eigen background model achieves the most inaccurate result, while other methods obtain same (or similar) result. For LS sequence, the Wallflower (with maintaining background at frame level) achieves the least total error. For FA sequence, the Wallflower achieves the most accurate result. However, the authors of [10] used a region-level processing as a post-processing step for Wallflower. Although SACON uses a step of validation of pixels inside the foreground holes, it is at pixel level.

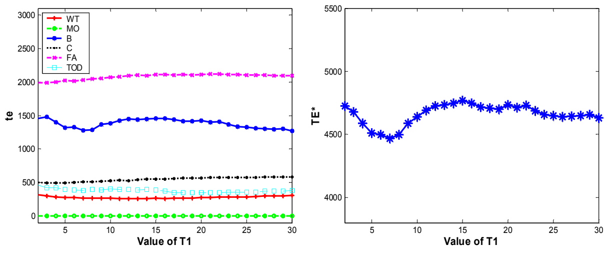
6. The Influence of the Parameters

There are two parameters that are crucial to SACON: the value of T_I (in Equation 9) and the number of the background samples at each pixel. In this section, we investigate the influence of these two parameters on the results of SACON. We evaluate the results by TE* and corresponding total error (*te*) for each of the image sequences.

6.1 The Influence of T_I on the Results of SACON

We change the value of T_I from 2 to 30, with interval 1.

We test the total error of each image sequence for different T_I value. From Figure 7, we can see that when the value of T_I varies, the influence on the results for various types of image sequences is different: for example, with the increase of T_I value, the total error (*te*) increases for FA and C, remains stable for MO, and fluctuates for WT, B, and TOD image sequences. The fluctuation of the *te* for B image sequence is relatively large, while relatively small for other image sequences. For the overall performance (see Figure 7 (b)), we can see that TE* fluctuates with the increase of T_I. Moreover, TE* is relatively stable when T_I is lager than eight. However, the fluctuation is within a reasonable range: even the highest TE* value is still less than that of the other five comparative methods.

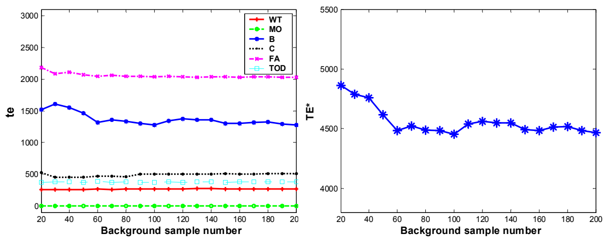


(a) (b)

Figure 7. Plot of (a) total error (*te*) and (b) TE* vs different T_I values.

6.2 The Influence of Background Sample Number on the Results of SACON

Because our method is based on sample consensus, the number of the background samples N is crucial to the performance of our method. It is desirable to investigate the influence of the background sample number N on the results of SACON.



(a) (b)

Figure 8. Plot of (a) total error (*te*) and (b) TE* vs different background sample number.

We use the same set of image sequences as those used in the last subsection. We use various numbers of background samples, and N changes from 20 to 200, with interval 10.

From Figure 8 (a), we can see that the influence of the sample number is small on most image sequences, except for image sequence B, where there is relatively large fluctuation in total error (*te*). Figure 8 (b) shows the overall performance on various N. We can see that when N is less that 50, TE* increases with the decrease of N. However, when N is larger than 50, TE* remains relatively stable.

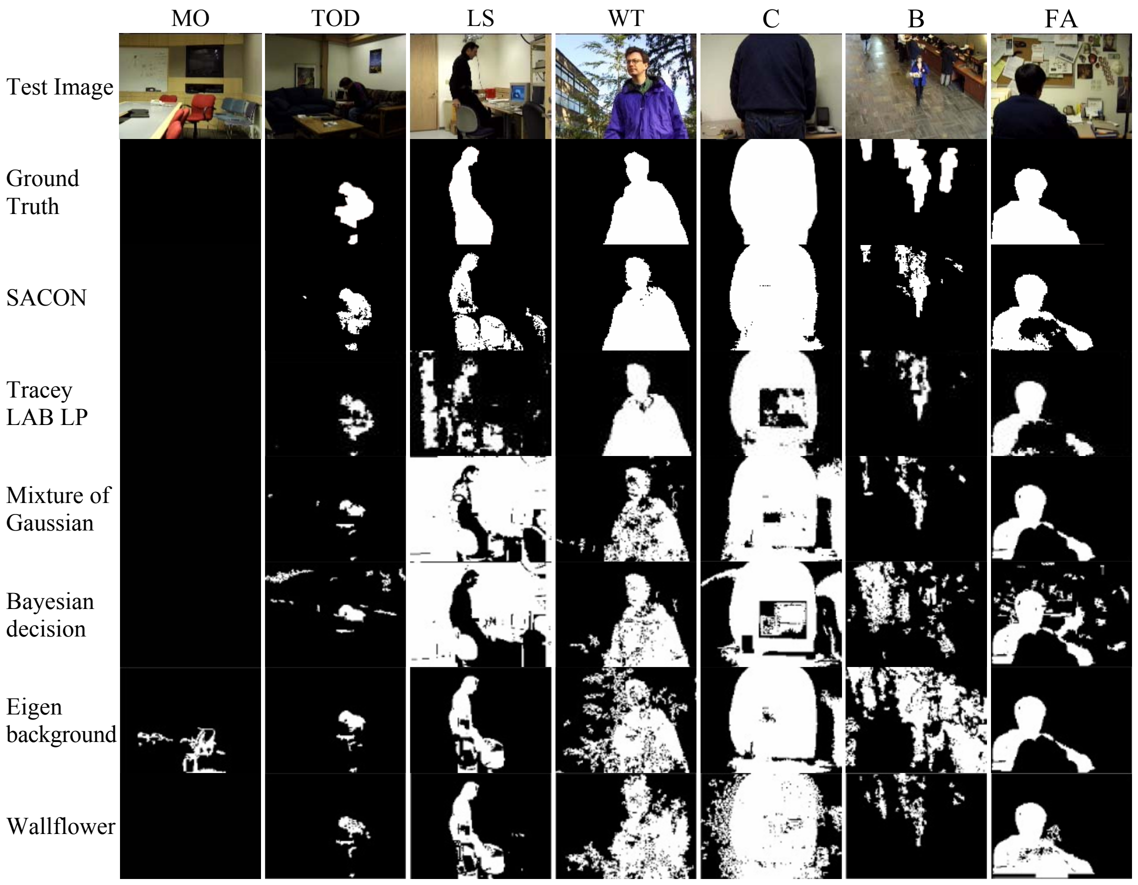


Fig. 6: Experimental results by several methods on the seven canonical background problems of the Wallflower benchmarks. The top row shows the evaluated frames of each image sequences; the second row shows the hand-segmented ground truth; the third row shows the results of SACON. The fourth row shows the results of Tracey reported in [11]]; the fifth to the eighth rows show the results reported in [10]].

Methods	ET	MO	TOD	LS	WT	C	B	FA	TE	TE*
SACON	f. neg.	0	236	589	41	47	1150	1508	6087	4467
	f.pos.	0	147	1031	230	462	125	521		
	te	0	383	1620	271	509	1275	2029		
Tracey LAB LP	f. neg.	0	772	1965	191	1998	1974	2403	12035	8046
	f. pos.	1	54	2024	136	69	92	356		
	te	1	826	3989	327	2067	2066	2759		
Mixture of Gaussian	f. neg.	0	1008	1633	1323	398	1874	2442	27053	11251
	f. pos.	0	20	14169	341	3098	217	530		
	te	0	1028	15802	1664	3496	2091	2972		
Bayesian decision	f. neg.	0	1018	2380	629	1538	2143	2511	31422	15603
	f. pos.	0	562	13439	334	2130	2764	1974		
	te	0	1580	15819	963	3668	4907	4485		
Eigen-background	f. neg.	0	879	962	1027	350	304	2441	17677	16353
	f. pos.	1065	16	362	2057	1548	6129	537		
	te	1065	895	1324	3084	1898	6433	2978		
Wallflower	f. neg.	0	961	947	877	229	2025	320	11478	10156
	f. pos.	0	25	375	1999	2706	365	649		
	te	0	986	1322	2876	2935	2390	969		

Table 1: Experimental results by different methods on Wallflower benchmarks.

6.3 The Time Complexity

The processing time of our method is affected to a relatively large extent by the number of the background samples *N*. Also, it is affected by the type of image sequences, the type of computing language, etc. Here, we provide a rough estimation the processing time of our method. We perform our method in MATLAB language (interfaced with C MEX) on a Laptop with Pentium M processor 1.6MGHZ. The averaged processing time for the seven Wallflower image sequences (120x160 color images) is about 10 frames per second when we set *N* equal to 100, and 6 frames per second when *N* is set to 200. We notice that a relatively large part of time is used to transfer data (mainly background samples) between MATALB and C MEX. Programming in complete C code with optimization will make the method faster.

7. Conclusion

In this paper, an effective and robust background modeling method (SACON) is proposed. The method may be applied in many practical environments and is effective in modeling a dynamic background. An effective framework is also proposed to apply SACON to background subtraction. The proposed method has been tested and validated by a significant number of experiments. SACON has proved to be robust in various environments (including indoor and outdoor scenes) and different types of background scenes such as dynamic or static scenes. We also numerically evaluate the performance of SACON with the Wallflower benchmarks and compare its results with those of five other popular background modelling methods. The comparisons show that SACON achieves very promising results.

References

1. A. Senior. "*Tracking with Probabilistic Appearance Models,*" in *ECCV workshop on Performance Evaluation of Tracking and Surveillance Systems*: p. 48-55, 2002.

2. C. Stauffer and W.E.L. Grimson. "*Adaptive Background Mixture Models for Real-time Tracking,*" in *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition*: p. 246-252, 1999.

3. S.J. McKenna, et al., "*Tracking Groups of People,*" Computer Vision and Image Understanding. **80**: p. 42-56, 2000.

4. R. Cucchiara, et al., "*Detecting Moving Objects, Ghosts, and Shadows in Video Streams,*" IEEE Trans. Pattern Analysis and Machine Intelligence. **25**(10): p. 1337- 1342, 2003.

5. A. Mittal and L.S. Davis, "*M_2 Tracker: A Multi-View Approach to Segmenting and Tracking People in a Clutter Scene,*" International Journal of Computer Vision. **51**(3): p. 189-203, 2003.

6. D. Gutchess, et al. "*A Background Model Initialization Algorithm for Video Surveillance,*" in *IEEE Int'l Conference on Computer Vision*, 2001.

7. I. Haritaoglu, D. Harwood and L.S. Davis, "*W4: Real-Time Surveillance of People and Their Activities,*" IEEE Trans. Pattern Analysis and Machine Intelligence. **22**(8): p. 809-830, 2000.

8. N. Oliver, B. Rosario and A. Pentland. "*A Bayesian Computer Vision System for Modeling Human Interactions,*" in *International Conference on Computer Vision Systems*: p. 255 - 272, 1999.

9. C.R. Wren, et al., "*Pfinder: real-time tracking of the human body,*" IEEE Trans. Pattern Analysis and Machine Intelligence. **19**(7): p. 780-785, 1997.

10. K. Toyama, et al. "*Wallflower: Principles and Practice of Background Maintenance,*" in *7th International Conference on Computer Vision*, Greece: p. 255-261, 1999.

11. D. Kottow, M. Koppen and J. Ruiz-del-Solar. "*A Background Maintenance Model in the Spatial-Range Domain,*" in *2nd Workshop on Statistical Methods in Video Processing*, Prague, Czech Republic, 2004.

12. M. Harville. "*A Framework for High-Level Feedback to Adaptive, Per-Pixel, Mixture-of-Gaussian Background Models,*" in *7th European Conference on Computer Vision*, Copenhagen, Denmark: p. 543-560, 2002.

13. A. Elgammal, D. Harwood and L.S. Davis. "*Non-parametric Model for Background Subtraction,*" in *6th European Conference on Computer Vision*, Dublin, Ireland: p. 751-767, 2000.

14. A. Mittal and N. Paragios. "*Motion-Based Background Subtraction using Adaptive Kernel Density Estimation,*" in *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition*, Washington, DC: p. 302-309, 2004.

15. M. Seki and T.F. Wada, H.Sumi, K. "*Background Subtraction Based on Cooccurrence of Image Variations,*" in *Computer Vision and Pattern Recognition*: p. 65-72, 2003.

16. N. Friedman and S. Russell. "*Image Segmentation in Vedio Sequences: A Probabilitic Approach,*" in *In Proc. Thirteenth Conf. on Uncertainty in Artificial Intelligence*, San Francisco, CA: p. 175-181, 1997.

17. O. Javed, K. Shafique and M. Shah. "*A Hierarchical Approach to Robust Background Subtraction using Color and Gradient Information,*" in *IEEE Workshop on Motion and Video Computing*, Orlando: p. 22-27, 2002.

18. M.A. Fischler and R.C. Rolles, "*Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,*" Commun. ACM. **24**(6): p. 381-395, 1981.

19. J. Connell, et al. "*Detection and Tracking in the IBM PeopleVision System,*" in *IEEE ICME*, 2004.

20. K.-P. Karmann and A.v. Brandt, *Moving Object Recognition Using and Adaptive Background Memory*, in *Time-Varying Image Processing and Moving Object Recognition*. 1990, Elsevier Science Publishers B.V.