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 quantitative 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...
The PDF is written as:

\[ PD(x_t) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}}} e^{-\frac{1}{2}(x_t - \mu_i)^T \Sigma^{-1}(x_t - \mu_i)} \]  

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 \((x_i - \mu)\) 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 \( t \rightarrow x_t(m) \), we need to classify it into either a background pixel or a foreground pixel according to the background samples \[ \{x_i(m) | i = 1, ..., N\} \] at that pixel location \( m \). Each observation \( x_t = (x_t^R, x_t^G, x_t^B) \) has \( k \) channels (e.g., in RGB color space, each observation is expressed by three channels of R, G, B). Let:

\[ \Gamma^i(m) = \begin{cases} 1 & \text{if } |x_t^i(m) - x_t^i(m)| \leq T_r \\ 0 & \text{otherwise} \end{cases} \]  

For each observation at time \( t \), we form a binary mask \( B \) defined as the sample consensus classification (with “one” for a background pixel and “zero” for a foreground pixel):  

\[ B(m) = \begin{cases} 1 & \sum_{i=1}^{N} \Gamma^i(m) \geq T_n \forall c \in \{C_1, ..., C_c\} \\ 0 & \text{otherwise} \end{cases} \]  

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.
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 \( \tau \), \( N \), where \( \tau \) 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.

![Block diagram of the complete framework.](image)

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 foreground 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:

\[
\begin{align*}
  r &= R / (R + G + B) \\
  g &= G / (R + G + B) \\
  b &= B / (R + G + B)
\end{align*}
\]

(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 \((r, g, l)\) coordinates

Let \((r_b, g_b, l_b)\) be the observed value of a background pixel \(x_b\) and \((r, g, l)\) be the observed value at this pixel (i.e., \(x_i\)) in frame \(i\). If the background is totally static, we can expect \(\beta \leq l / l_b \leq 1\) when the pixel is covered by shadow and \(1 \leq l / l_b \leq \gamma\) when the pixel is highlighted by strong light. Thus, the shadow can be suppressed if the following two conditions hold:

\[
\begin{align*}
  |x'_c - x'_c| &\leq T_r \quad \forall c \in \{r, g\} \\
  \beta &\leq x'_c / x'_c \leq \gamma \quad c \in \{l\}
\end{align*}
\]

(7)

where \(\beta, \gamma\) 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 \(B\) 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 \(T_n\), 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 \(I_c\) and \(I_b\) is large, however \(\beta \leq I_c / I_b \leq \gamma\) 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.

To solve this problem, we only use $I$ when the intensity $I$ of the pixel is lower than a threshold ($I_{td}$):

$$I = \begin{cases} (r, g, I) & \text{if } I \geq I_{td} \\ (I) & \text{if } I < I_{td} \end{cases}$$

(8)

4.2.2 Setting the Value of $T_r$

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 $\sigma$ and set $T_r$ equal to $\eta \sigma$, (where $\eta$ is usually set as 2.5 or 3). However, the estimated $\sigma$ may be overestimated when the data is multi-model distributed.

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

$$T_r = \min(T_r, \eta \sigma)$$

(9)

where $T_i$ is a constant (We will discuss the influence of $T_i$ 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 $\beta \leq x_t^i / x_s^i \leq \gamma$ 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 $\beta \leq x_t^i / x_s^i \leq \gamma$ (this is notable especially when $x_s^i$ 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 $|x_t^i - x_s^i| \leq T_i$ (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.

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.

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 cannot 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 \( TOM_m(t) \) be the time out map at pixel \( m \) at frame \( t \). We have:

\[
TOM_m(t) = \begin{cases} 
TOM_{m-1}(t) + 1 & \text{if } B_m(t) = 0 \\
TOM_{m-1}(t) & \text{otherwise}
\end{cases}
\] (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).

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.

Light Switch (LS): A room scene begins with the lights on. Then a person enters the room and sits down. A phone call is made, and the chair is turned around.

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. Boosting (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 $T_1$ (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 $T_1$ on the Results of SACON

We change the value of $T_1$ from 2 to 30, with interval 1. We test the total error of each image sequence for different $T_1$ value. From Figure 7, we can see that when the value of $T_1$ varies, the influence on the results for various types of image sequences is different: for example, with the increase of $T_1$ 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 $T_1$. Moreover, TE* is relatively stable when $T_1$ is larger 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.

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.

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

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

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 sequence; 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].
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