

A Consensus Based Method for Tracking: Modelling Background Scenario and Foreground Appearance

Hanzi Wang* and David Suter

Department of Electrical and Computer Systems Engineering

Monash University, Clayton Vic. 3800, Australia.

Abstract

Modelling of the background (“uninteresting parts of the scene”), and of the foreground, play important roles in the tasks of visual detection and tracking of objects. This paper presents an effective and adaptive background modelling 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 of the background, 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.

However, it is one thing to detect objects that are not likely to be part of the background; it is another task to track those objects. Sample consensus is again utilized to model the appearance of foreground objects to facilitate tracking. This appearance model is employed to segment and track people through occlusions. Experimental results from several video sequences validate the effectiveness of the proposed method.

* Corresponding author

Tel: +61-3-9905-5751; Fax: +61-3-9905-9602.

Email Address: hanzi.wang@eng.monash.edu.au (Hanzi Wang).

Keywords: Background modelling, background subtraction, sample consensus, visual tracking, segmentation, foreground appearance modelling, occlusion.

1. Introduction

Background modelling is an important and fundamental part for many computer vision applications such as real-time tracking [2, 20, 21, 25, 26], video/traffic surveillance [9, 10] and human-machine interface [23, 29]. 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 multi-modal distributed) and robust in the training stage *even if foreground objects exist in all training examples*;
- 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 as moving);

- efficient in computation.

In the first part of 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. Quantitative experiments show the advantages of SACON over several other popular methods in background modelling/subtraction. Such background modelling can be useful in its own right but this paper goes on to tackle the problem of tracking/segmenting people through video sequences.

Tracking people is one of the most challenging tasks in computer vision. Human motion is non-rigid because when people walk towards or away from the video camera, both the shape and the size of the images of those people change. People can also merge to form a group, occlude each other, or split from each other. A visual tracker needs to cope with such complex interactions.

In the second part of this paper, we again use a form of sample consensus, this time to model the appearance of human bodies. We use the obtained appearance model to segment and track people despite occlusions. We exploit both the spatial and color information of the human bodies in our method. We show experimental results in several video sequences to validate the effectiveness of the proposed method.

The main contributions of this paper can be summarized as follows:

- We exploit the notion of "*sample consensus*" to construct an effective and adaptive *background modelling* method (SACON) for detecting foreground objects in both static and dynamic scenes;
- We present a new sample consensus based method for *modelling human appearance* and handling occlusion in human segmentation and tracking tasks.

- The consensus based background modelling and consensus based appearance modelling are combined to form an effective tracking system.

Experiments are presented to show that the proposed methods can achieve promising performance in background subtraction (including handling both static and dynamic background scenes) and foreground appearance modelling (including tracking/segmenting people through occlusions).

The organization of the remainder of this paper is as follows: in section 2, we first present a short review of previous related work on background modelling. Then, we present the SACON concept and a framework for applying SACON to background subtraction. In section 3, experiments showing the advantages of our background modelling method over several popular methods are provided. We also investigated the influence of the parameters of SACON on the results. In section 4, we describe how to use sample consensus to model the foreground appearance. Experimental results of segmenting and tracking people with occlusions by the proposed tracking method are also provided. We conclude the paper in section 5.

Part I: Background Modelling

2. Sample Consensus in Background Modelling – SACON

In this section, we define a novel Sample Consensus (SACON) method for modelling background scenario, and a framework that employs SACON as a core for background subtraction. We begin with an overview of related work on background modelling (section 2.1). We then introduce the concept behind SACON (section 2.2) described for conceptual simplicity in terms of RGB colour space. Various modifications to the concept are then introduced (section

2.3): to cope with shadows, normalized colour is used (sections 2.3.1 and 2.3.2); treatment of isolated holes in foreground objects (section 2.3.3); the setting of the essential threshold controlling the notion of consensus (section 2.3.4) and the use of a Time Out Map to control which pixels are added to update the background samples (section 2.3.5). These components are put together into an overall background modeling framework (section 2.3.6).

2.1 Related Work of Background Modelling

Numerous background modelling studies have appeared in the literature in recent years, [3, 10, 11, 17, 20, 22, 23, 26, 27, 29]. 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. Realistically, background models have to allow for a *distribution* of background pixel values as lighting changes etc. For example, to capture the allowed variation in background values, W^4 [10] models the background by maximum and minimum intensity values, and the maximum intensity difference between consecutive frames in the training stage. Other techniques assume a statistical model: Pfister [29] assumes that the pixels over a time window at a particular image location are single Gaussian distributed. Although these methods can deal with small or gradual changes in the background and they 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 multi-modal distributed background pixels. Wallflower [27] employs a linear Wiener filter to learn and predict background changes. Wallflower works well for periodically changing pixels. However, when the background pixels change dramatically or the movement of those background pixels are less periodical, Wallflower is less effective in learning and predicting background changes.

Other examples include Tracey [17] which models foreground and background by codebook vectors; [16] which quantizes and compresses background samples at each pixel into codebooks; and [24], where “cooccurrence” of image variations at neighboring image blocks is employed for modelling a dynamic background.

The pixel-level (i.e., ignoring spatial relationship between neighbors) Mixture of Gaussians (MOG) background model [8, 26] is popular and effective in modelling multi-modal distributed backgrounds. Augmented with a simple method to update the Mixture of Gaussian parameters, 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-means algorithm to initialize the parameters, which is slow and may be inaccurate. When the background involves many modes, modelling the background with a small number of Gaussians per pixel is not efficient. It is also hard to set the value of the learning rate.

A lot of variants of the MOG background model have been proposed [3, 13, 26]. Elgammal et. al. [3] chose to replace the MOG Probability Density Function (PDF) with a Kernel-based density estimation method and showed it was effective in handling situations where the background contains small motions such as tree branches and bushes. Since the cost to compute the kernel density estimate at each pixel is very high, several pre-calculated lookup tables are used to reduce the burden of computation of the algorithm. Moreover, because the kernel bandwidth is estimated by using the median absolute deviation over samples of consecutive intensity values at the pixel, the bandwidth estimate may be inaccurate if the distribution of the background samples is multi-modal.

Instead of explicitly choosing a background PDF model, we propose to use the more simple notion of "consensus" to classify a pixel value as foreground or background. We believe such and

approach is more robust (to deviations from the arbitrarily assumed background PDF model), easier to compute, and, as we show experimentally, performs very well.

2.2 Sample Consensus in Background Modelling - SACON

We now define a background model Sample Consensus inspired by the work of RANSAC [7] (which was designed for parametric model fitting).

We keep a cache (or history) of N background samples at each pixel, so that at time t we have $\{x_i(m) | i = 1, \dots, N, N < t\}$ where $x_i(m)$ is an observation at pixel m at time t . Note: each observation $x_t(m) = (x_t^{c_1}(m), \dots, x_t^{c_k}(m))$ has k channels (e.g., in RGB color space, each observation is expressed by three channels of R, G, B). For each sample in this cache, we define a binary label capturing the notion of “agreement” between the channel values at the current sample and past channel values at that pixel:

$$\Gamma_i^c(m, t) = \begin{cases} 1 & \text{if } |x_i^c(m) - x_t^c(m)| \leq T_r \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where T_r is a threshold value related to the residual $r_i(m, t) = |x_i^c(m) - x_t^c(m)|$ (the choice of T_r will be discussed in section 2.3.4).

The sample consensus is simply formed by counting the number of times previous samples “agree” with the current sample:

$$B_t(m) = \begin{cases} 1 & \sum_{i=1}^N \Gamma_i^c(m, t) \geq T_n \quad \forall c \in \{C_1, \dots, C_k\} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where T_n is a value thresholding the number of data points that are within the error tolerance T_r of a mode. B_t is a binary value with “one” for a background pixel and “zero” for a foreground pixel.

It is clear that the value of the threshold T_n should be influenced by the sample size N : the larger N is, the larger T_n should be. Likewise, T_n should also reflect the error tolerance T_r : the larger T_r is, the larger value T_n should be. Thus, T_n can be effectively set to $\tau T_r N$, where τ is a constant and is chosen empirically.

In contrast to the MOG-based background model, parameters such as the number of modes, and the weight, mean, and covariance of each mode are not required. In contrast to the kernel-based background model, SACON is more computationally efficient and no pre-calculated lookup tables are used.

2.3 Building on SACON: A Background Modelling Framework

In this section, using SACON as a core step, we present a complete framework for background subtraction. Each component will be discussed in turn before we present the overall framework.

2.3.1 Shadow Removal and Related Issues

RGB color space is sensitive to changes of illumination. For example, employing RGB color space may cause incorrect labelling of shadows as foreground pixels. Normalized color has been used in many background modelling methods, such as in [3, 20, 21, 22], to minimise the effects of brightness changes. The normalized chromaticity coordinates can be written as:

$$\begin{aligned} r &= R/(R+G+B) \\ g &= G/(R+G+B) \\ b &= B/(R+G+B) \end{aligned} \tag{3}$$

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

However, the complete loss of the intensity information can be a problem so we use (r, g, I) coordinates [3, 21, 22].

To employ these normalized coordinates we reformulate the test in equation (3). Let (r_b, g_b, I_b) be the sample value at a background pixel x_b and (r_t, g_t, I_t) be the sample value at this pixel (i.e., x_t) in frame t . If the background is totally static, we can expect the pixel in shadow to be dark (but not totally black): $\beta \leq I_t/I_b \leq 1$ and, conversely, when the pixel is highlighted by strong light $1 \leq I_t/I_b \leq \gamma$. Thus, we express the tolerance for consensus on the intensity channel in terms of the ratio: $\beta \leq I_t/I_b \leq \gamma$ i.e., equation (1) is replaced by :

$$\Gamma_i^c(m, t) = \begin{cases} 1 & \text{if } |x_t^c - x_b^c| \leq T_r, \forall c \in \{r, g\} \text{ and } \beta \leq x_t^c / x_b^c \leq \gamma \quad c \in \{I\} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where β, γ are constant and are chosen empirically (in our case, we set $\beta=0.6; \gamma=1.5$) and T_r is set as described in section 2.3.4.

However, there are still some problems remaining:

- 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 (3). We address this issue in section 2.3.2.
- 2) When the chromaticity component of the foreground pixel is similar to that of the background pixel, the ratio test we have just defined can fail - see section 2.3.3.

2.3.2 Normalized Color Noise

Figure 1 (a) shows one frame of the image sequence ‘‘Light Switch’’ (LS) in the Wallflower dataset [27]. 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 1 (b) and (c), we can see that when the intensities of image pixels in Figure 1 (a) are low, the estimated standard variances of both the r channel and the g channel are high

(corresponding to bright pixels in Figure 1 (b) and (c)). This illustrates that the estimated r and g values are very noisy when the intensity values are low.

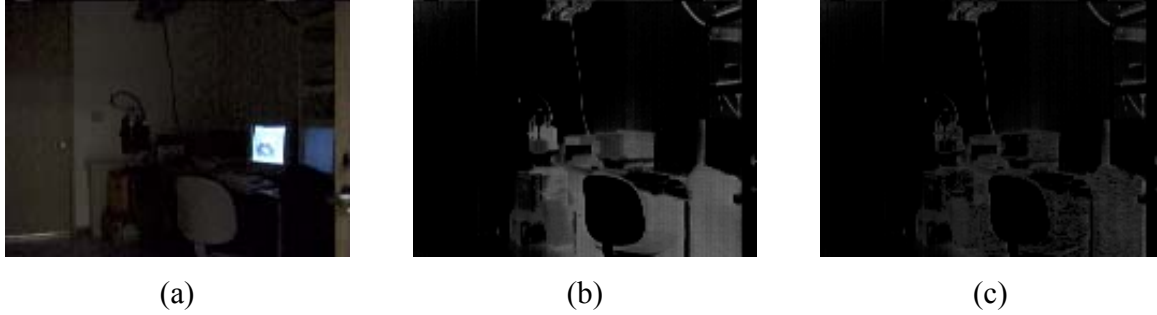


Figure 1. (a) One frame of LS; Images showing the standard variance of the red channel (b) and the green channel (c) in the normalized rgb color space over 300 frames of LS.

To solve this problem, when the intensity value I is high, both r and g values are reliable. We use $x = (r, g, I)$ as the color channels. However, when the intensity I of the pixel is lower than a threshold I_{td} (which is experimentally set to 7), r and g values are not reliable. In this case, we use only the intensity I (we then only have one color channel).

$$x = \begin{cases} (r, g, I) & \text{if } I \geq I_{td} \\ (I) & \text{if } I < I_{td} \end{cases} \quad (5)$$

2.3.3 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 re-consider Equation (4), although $\beta \leq x'_t / x'_b \leq \gamma$ can be used to suppress shadows, the intensity information has been disregarded to some extent. If the chromaticity component of foreground pixels is similar that of background pixels, the difference of the intensity part may be large but still within the range of $\beta \leq x'_t / x'_b \leq \gamma$ (this is notable especially when x'_b is large), so in such cases the pixels are wrongly marked. For these pixels,

however, we cannot simply use some hole filling technique 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, for example. Thus, we use a validation procedure to recheck the pixels inside the holes (note: foreground objects are represented by connected components). For pixels inside the holes, we use $|x'_i - x'_b| \leq T_i$ (where T_i is a special threshold, we experimentally set it to 7, for the intensity channel applied only to pixels of the holes, as a post-processing step, 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).

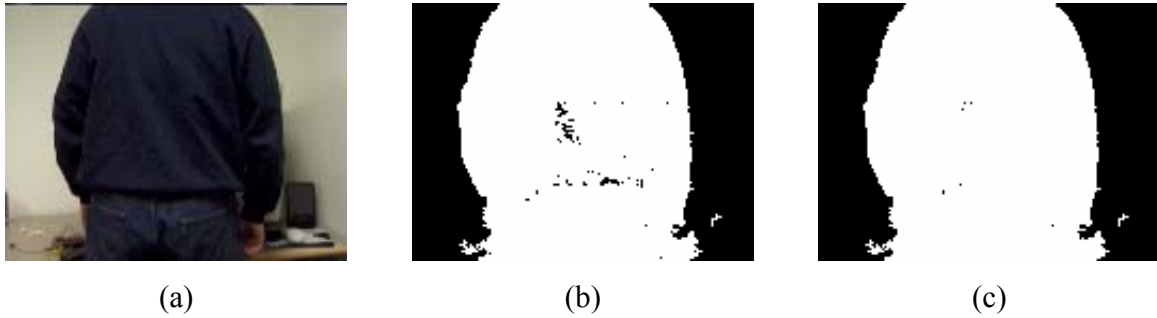


Figure 2. An example showing foreground hole pixel validation. One frame of “Camouflage” (a); The results without (b) and with (c) the validation procedure.

Although the validation cannot 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 (4). Figure 2 shows us an example. One frame of the image sequence “Camouflage” (C) in the Wallflower dataset is shown in Figure 2 (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 clothes. Figure 2 (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 2 (c) shows the result after applying the validation procedure. From Figure 2 (c) we can see that most pixels inside the holes are correctly marked as foreground pixels.

2.3.4 Setting the Value of T_r

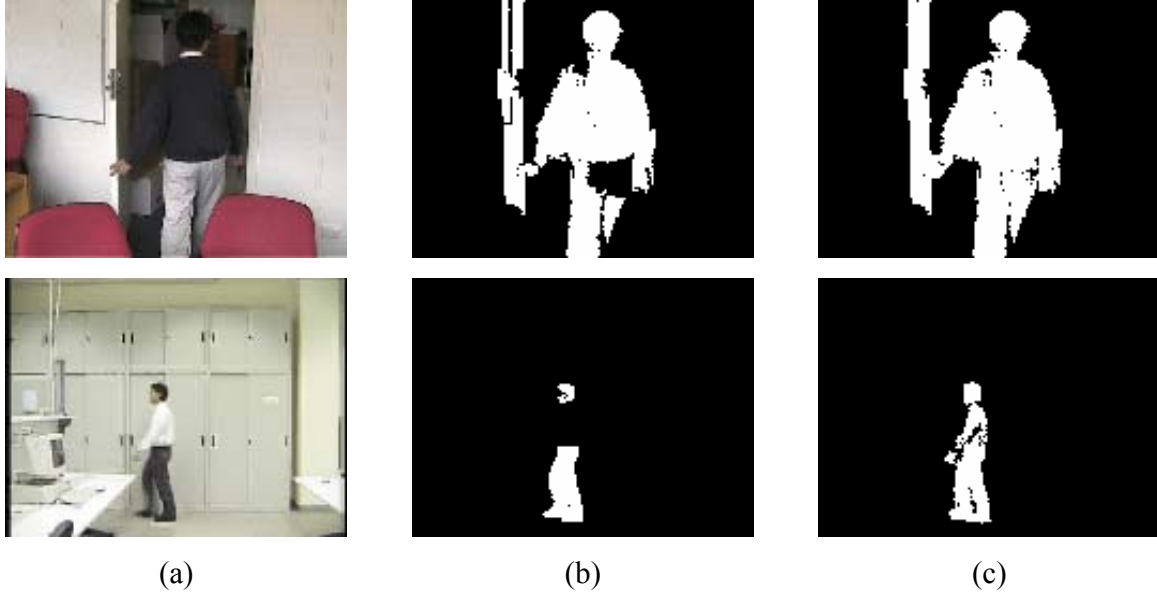


Figure 3. (a) One frame of the videos; the detected foreground pixels by (b) setting a global value to T_r and by (c) setting the values of T_{ri} for each pixel according to Equation (6).

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. However, to obtain an effective value of T_r for all image pixels is hard. The second way is to estimate the standard variance σ_i at each image pixel and set T_r equal to $\eta\sigma_i$, (where η is usually set as 2.5 or 3). However, σ_i may be overestimated when the data is multi-modal distributed.

We set T_{ri} for each image pixel by combining the above two ways as follows:

$$T_{ri} = \min(T_l, \eta\sigma_i) \quad (6)$$

where T_l is a constant (We will discuss the influence of T_l on the results in subsection 3.2.1).

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_{ri} for each pixel according to Equation (6), most of the foreground pixels are correctly detected (Figure 3c).

2.3.5 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 (classifying these as background).
- adapt the foreground objects, which remain static for a long time, (classifying these as background, e.g. in [1]).

There are several methods to update the background samples [3, 14]. 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.

In our method, we use a selective update mechanism to update the background samples. We update the background samples at both pixel level and blob level described in the following subsections. Our method can incorporate the moved/inserted background object into the background model.

2.3.5.1 Updating Background at Pixel Level

To incorporate the moved/inserted background object into the background model, we use a Time Out Map (TOM) (similar to [10] - see below). A simple counter, $TOM_t(m)$ is created for the time

out map at pixel m at frame t . This map is simply incremented at every frame the pixel has been classified as foreground:

$$\begin{cases} TOM_t(m) = TOM_{t-1}(m) + 1 & \text{if } B_t(m) = 0 \\ TOM_t(m) = 0 & \text{otherwise} \end{cases} \quad (7)$$

In other words, 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} (which we experimentally set as 45), that pixel will be assigned to the background (the pixel of the object has remained in place too long).

Our TOM is similar to the Detection Support Map (DSM) in [10] as both work as a counter. The differences between TOM and DSM are in that: (a) TOM represents the times a pixel is classified as a foreground pixel while DSM is used to record how long a pixel is classified as a background pixel; (b) when a pixel is classified as a background pixel, the corresponding TOM value is set to zero; In contrast, when a pixel is classified as a foreground pixel, the DSM value at that pixel is unchanged.

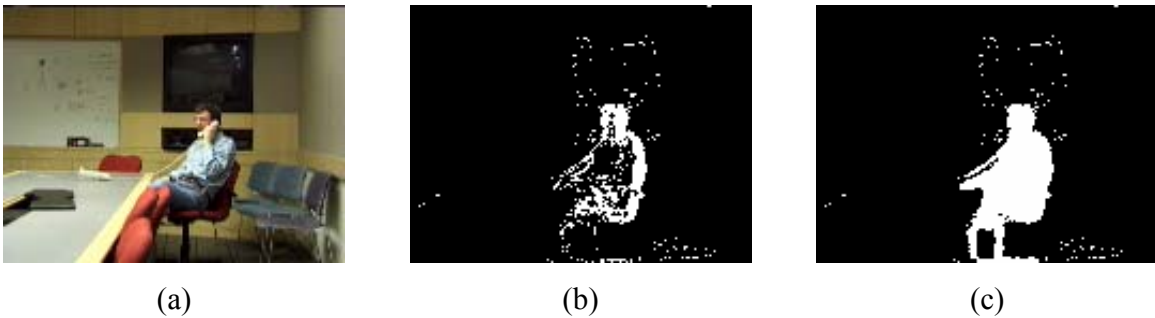


Figure 4. (a) A frame of MO; Results obtained by updating at pixel level (b) and combination of pixel and blob level(c).

We find the proposed updating method at pixel level works in most cases. However, in some cases, pixels of moving objects can be incorporated to the background. Figure 4(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 a phone call, he turned around in the chair. There is an overlapped region around the center of his body that remains about the same brightness and color despite the underlying movement. The TOM values of the pixels of that 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. To tackle this issue, we must update the background samples at both the pixel level and the blob level (see next section).

2.3.5.2 Updating Background at Blob Level

Connected groups of foreground pixels below a certain size have their TOM updated "at the pixel level" as described in the previous section. Groups of pixels of size above this threshold are updated "at Blob Level" as we now describe.

First we decide whether a blob is "static". 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 (compared with the values of the nearest blob in the previous frame) by a large amount, we judge that the object is moving (otherwise it is static - “static foreground”).

Note: an object which belongs to the background (such as the telephone, and the chairs in Figure 4) that then moves (“moved background”) is not distinguished from a foreground moving object (both the "hole" it leaves and the new region it occupies will typically be treated as foreground objects that have "moved"). For a blob Ω (a large connected region) the TOM values of the pixels in the blob are updated as follows:

$$\begin{cases} TOM_t(m') = TOM_{t-1}(m') + 1 & \text{if } \Omega \text{ is static} \\ TOM_t(m') = 0 & \text{otherwise} \end{cases} \quad (8)$$

i.e., if a blob is judged static, we increase the TOM value of all pixels of that blob 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 (an object has remained stationary long enough to now be considered as background).

Figure 4 (c) shows the result obtained by background sample update at both pixel and blob level. We can see the pixels at the center part of the person are correctly marked as foreground pixels.

2.3.6 The Complete Framework of the Proposed Method

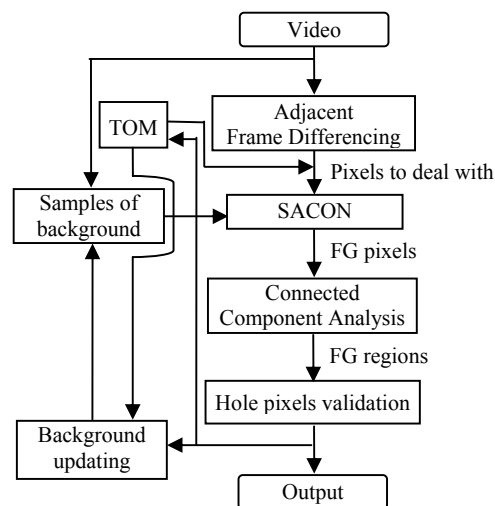


Figure 5. Block diagram of the complete framework.

The major components are shown in Figure 5. The proposed framework mainly contains three phases: extracting all possible foreground pixels, running the SACON algorithm, and validating the pixels inside the foreground holes. In the first phase, we use adjacent frame differencing [27] to extract possible candidate foreground pixels. The computational speed is improved by this as typically only a few pixels (from moving foreground objects) will be dealt with in the second phase. However, if the background includes dynamic parts, some of the extracted candidate

foreground pixels may belong to the background. This issue will be solved in the next phase. In the second phase, we feed the candidate foreground pixels, their as well as their corresponding background samples to SACON. Also, the pixels whose TOM values are larger than a value (i.e., the pixels are labeled as foreground pixels for a long time) will be sent to SACON. SACON is then run (using equation (4) - modified to single channel as in equation (5) as appropriate). The pixels from the dynamic background scene are suppressed by SACON and output of the second phase is the detected foreground (FG) pixels. In the third phase, we form connected components and where there are holes inside the foreground regions we validate these pixels (see section 2.3.3). After the third phase, we update the background samples and the TOM using the way described in section 2.3.5 and go on the next frame.

3. Experimental Validation of the Background Modelling

In this section we first show that the background method can be competitive with other contemporary techniques. We then investigated the effect of variations in the parameters that the method employs. We also illustrate the computational cost/complexity of the method. *Except for the case where we investigate the influence of parameters on the proposed method, all parameters are held fixed for all experiments.*

3.1 Demonstration of the Background Modelling Capability

Toyama et. al. [27] 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 size of each frame of all the Wallflower image sequences is 160x120 pixels. The sample rate of each image sequence is 4Hz. The performance is evaluated against hand-segmented ground truth. In this section, we will evaluate our background modelling method using these image sequences

and compare the performance of our method with that of five state-of-the-art background modelling 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 into 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.
- **Bootstrapping (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.


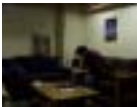













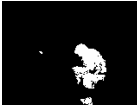





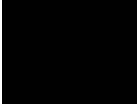


































	MO	TOD	LS	WT	C	B	FA
The Number of Frames	300	1851	226	248	252	300	230
Foreground Region Size	0	1548	3247	5881	10565	2925	5022
Test Image							
Ground Truth							
SACON							
Tracey LAB LP							
Mixture of Gaussian							
Bayesian decision							
Eigen background							
Wallflower							

Figure 6: Experimental results by several methods on the seven canonical background problems of the Wallflower benchmarks. The first row shows the number of frames that are used for each image sequence; The second row shows the size of the foreground regions (in pixels); The third row shows the evaluated frames of each image sequences; the fourth row shows the hand-segmented ground truth; the fifth row shows the results of SACON. The sixth row shows the results of Tracey reported in [17]; the seventh to the tenth rows show the results reported in [27].

For the evaluation of performance against each image sequence, we use three terms (as in [27]): 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). Because many methods do not work well for the LS sequence, we also use TE* (the sum of total error excluding the light switch image sequence) for the evaluation. For each result image, we eliminated isolated foreground pixels: those whose 4-connected foreground pixel number is less than 8.

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.

Figure 6 and Table 1 show the results obtained by SACON, and five other state-of-the-art methods. There are some isolated examples and measures where some methods have relative advantages. For the LS sequence, the Wallflower method and the Eigen-based method achieve less total error. For the FA sequence, the Wallflower method achieves the most accurate result based on the measure te . For MO sequence, except for the Eigen-based background model, almost all other methods achieve accurate results. However, 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 the five image sequences of MO, TOD, WT, C and B.

3.2 The Influence of the Parameters

There are two important parameters that are crucial to SACON: the value of T_l (in Equation 6) 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.

3.2.1 The Influence of T_l on the Results of SACON

In this experiment, we changed the value of T_l from 2 to 30, with interval 1.

From Figure 7, we can see that when the value of T_l varies, the influence on the results for various types of image sequences is different. 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_l . The shape of TE* in Figure 7 (b) is roughly the same as the shape of the te of B in Figure 7 (a). This is because it is the sum of several quantities which, except for the "Bootstrapping", remain relatively flat. Moreover, TE* is

relatively stable when T_I 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.

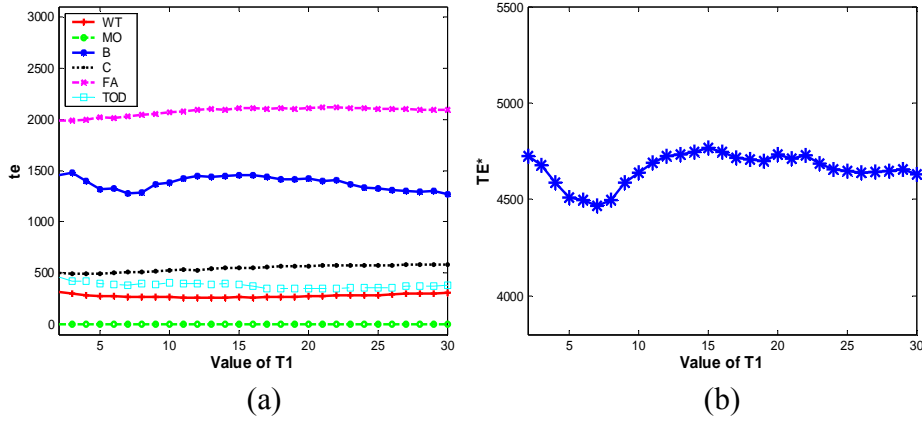


Figure 7. Plot of (a) total error (te) and (b) TE^* vs different T_I values.

3.2.2 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.

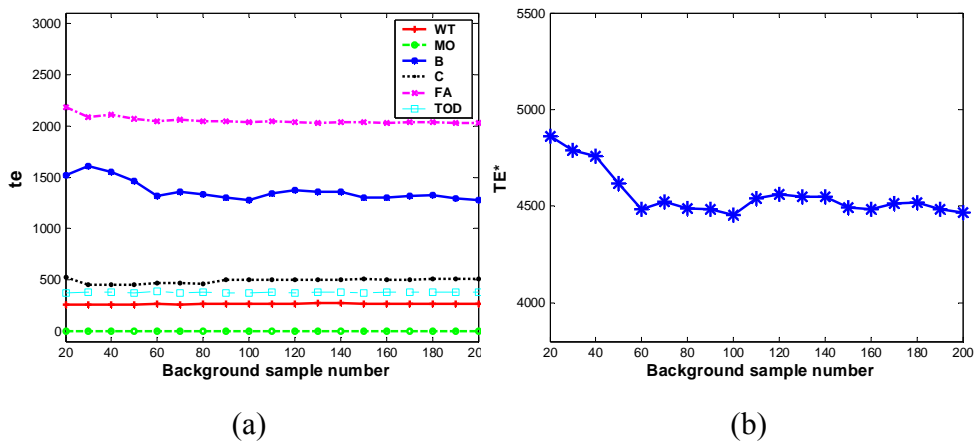


Figure 8. Plot of (a) total error (te) and (b) TE^* vs different background sample number.

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 our experience, $N = 20$ is the minimum that may be used in most practical tasks.

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 a relatively large fluctuation in the total error (te). Figure 8 (b) shows the overall performance with various N . We can see that when N is less than 50, TE^* increases with the decrease of N . However, when N is larger than 50, TE^* remains relatively stable.

3.3 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, by the type of computer processor, the type of computing language, etc. Here, we provide a rough estimation the processing time of our method. We implemented our method in MATLAB language (interfaced with C language) 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 the time is used to transfer data (mainly background samples) between MATLAB and C. Programming in complete C code with optimization will make the method faster.

Part II: Segmentation and Tracking

4. Segmenting and Tracking People with Occlusions by a Sample Consensus-based Method

We now describe an application of the previously described background modelling approach. We choose to address the problem of tracking people. This is one of the most difficult problems in visual tracking because people can meet, form groups, cross-over etc. It is clear that simple connected component methods [18] that extract several isolated regions and label each region as a person will not be adequate in such situations. An appearance model, even if simple, should be used in more successful approaches.

A lot of work has been done in modeling human appearance [4, 5, 15, 20, 19, 25]. We refer [12, 28] for survey. In [19] and [15], the authors employed a mixture of Gaussians to estimate the color distributions of human bodies. Although some appealing results have been obtained, choosing the right number of Gaussians is a challenging problem. The authors of [20] employed color histograms to model the color distributions of human bodies. However the spatial information of human bodies is ignored. Indeed two people may have the same color histograms though they may dress in different ways. Another problem of color histograms is that they require the sample size to be large enough to ensure statistical efficiency. In [25], a probability mask (i.e., an appearance template) is used to model each pixel of human bodies. Because this method records the appearance at each pixel, it requires a large memory to store the templates. Non-parametric methods using Gaussian kernel density are employed in [4, 5] to model the human appearance. Although such is an improvement over [20] and [15], the biggest problem of this method is its high computational cost. Although a fast kernel density estimation method was proposed in [6], there are still no satisfactory solutions.

In the previous sections, we have successfully applied the SAmple CONsensus (SACON) to model backgrounds involving dynamic/static scenes. In this section, we will extend that work to use sample consensus to construct a simple but efficient model for the appearance of human bodies. This model utilizes both color and spatial information of human bodies. We then use this appearance model to segment and track people through occlusions.

Our main aim is to track people with occlusion. In order to simplify the situation, we consider the occlusion between only two people. However, our method can be easily extended to segment and track multiple people (see Figure 13). We use two tracking modules in the framework of the proposed method: box-based tracking and appearance-based tracking. When people are separated and do not occlude each other, we use the box-based tracking module similar to [25]; when people are occluded, or meet and form a group, we use the appearance-based tracking module.

4.1 Box-Based Tracking

In order to track people, we need an effective model to approximate the shape of humans. There are two widely-used human shape models in the literature: ellipse [5] [12] and box (or rectangle) [20, 25]. Although an ellipse is more accurate than a box in modelling the shape of a human body, we find the box is effective in most cases. We take the state of the box as: $S = (c_x, c_y, h, w)$, where c_x , and c_y is the center of the box, h and w is the half height and half width of the box. We update the state of all tracked boxes at each frame. When we detect two people occlude each other (i.e., when two tracked boxes merge into one bigger box), we only update the centers of each corresponding box (we assume that the h and w of each box remain unchanged during occlusion).

To compute the distance between two boxes A and B, we use:

$$D_{box} = \max(0, d_x) + \max(0, d_y) \quad (9)$$

where:

$$d_x = \begin{cases} c_{xA} - w_A - c_{xB} - w_B, & \text{when } c_{xA} \geq c_{xB} \\ c_{xB} - w_B - c_{xA} - w_A, & \text{when } c_{xA} < c_{xB} \end{cases}$$

and similar for d_y .

From the above equation, we can see that: when $d_x < 0$ (or $d_y < 0$), it means that the projections of the two boxes onto the x-axis (or y-axis) overlap; and that when $d_x > 0$ (or $d_y > 0$), there is no overlapping area for the projections of the two boxes onto the x-axis (or y-axis). Moreover, when the projections of the two boxes onto both the x-axis and y-axis overlap, the value of D_{box} is equal to zero.

4.2 Appearance-Based Tracking

It is necessary to initialize the appearance of people when they enter a scene or separate from a group, and to differentiate each person from the others when they meet to form a group or occlude each other. We assume that the poses are upright and their appearance models do not change dramatically when they group or occlude each other. As in our background modelling, we employ the normalized chromaticity r , g and intensity I as color space (see Subsection 2.3.2)

4.2.1 Modelling the Foreground Appearance

We now define a foreground model by Sample Consensus. People are modeled as a blob (connected component of foreground pixels in a rectangular bounding box). The difficulty occurs when people form a group or occlude each other, which we detect by touching or overlapping boxes as described in the previous section.

Suppose k persons P_j ($j=1,\dots,k$) form a group or occlude each other: we need to classify each pixel p_i ($i=1,\dots,n$) contained within the "blob" for the group into one of the labels $\{P_j\}$. Let A_j be the number of pixels of the j th person mask P_j in the frame prior to occlusion (we assume this remains fixed through the grouping/occlusion and is not updated during occlusion - a similar assumption to that made when treating the dimensions of the bounding boxes of people, as described in the previous section), and $\{S_j(m)\}_{m=1,\dots,A_j}$ be the samples from the j th person before occlusion. A pixel p_i within the group can be classified as belonging to the k th person by:

$$p_i \in P_k, k = \arg \max_k \prod_{c=r,g,l} \frac{\sum_m \Gamma_k^c(p_i, m)}{A_k} \quad (10)$$

where $\Gamma_k^c(p_i, m) = \begin{cases} 1 & \text{if } |p_i^c - S_k^c(m)| \leq T_r \\ 0 & \text{otherwise} \end{cases}$, T_r is a threshold value, $S^c(j)$ is the value of the

sample in the c channel ($c = r, g, l$). In words, we count up how many times the pixel to be classified agrees (consensus) with pixels in the k -th person blob in the frame immediately before, normalised so that the size of that person (blob) does not bias the outcome.

4.2.2 Exploiting Spatial Information

Spatial information of data (position within the person blob) is clearly important but is absent from Equation (10), so we now take spatial information into account.

We consider the vertical direction (y-axis) and the horizontal direction (x-axis) separately. For the vertical direction, because we assume that the poses are upright, we do not want samples of the feet of a person, which may have similar color, to contribute to the classification of a pixel at the head location of the person. Thus, Equation (10) is revised as:

$$p_i \in P_k, k = \arg \max_k \prod_{c=r,g,l} \frac{\sum_{m'} \Gamma_k^c(p_i, m')}{A_k'} \quad (11)$$

where m' is a set of samples of P_k , whose y values are close to that of p_i (in our case, we choose the samples m' whose y distances to that of p_i are less than 7); A_k' is the number of the set of samples of person k in that range (indexed by $\{m'\}$).

For the horizontal direction, we take the median of the x values of the samples of P_k as c_x when P_k is occluded by other persons or in a group. Because c_x of P_k may change when the person is in the group, or with occlusion, it may cause error if we use c_x from the previous frame (i.e., at $t-1$ frame, thus, in contrast to most other parameters describing a person, we must update c_x). This is especially noticeable when, for example, two persons exchange position in the horizontal direction.

In order to use the horizontal spatial information, we use a two-step method:

- (1) We use Equation (11) to get the initial classification of pixels within the mask of the group;
- (2) We compute $c_x(P_k)$ of the pixels classified as P_k . Then, we use Equation (12) to reclassify each pixel within the group mask:

$$p_i \in P_k, k = \arg \max_k \frac{\prod_{c=r,g,l} \frac{\sum_{m'} \Gamma_k^c(p_i, m')}{A_k'}}{|x(p_i) - c_x(P_k)|} \quad (12)$$

where $x(p_i)$ is the x value of p_i . (We note that $|x(p_i) - c_x(P_k)|$ can be zero. Thus, we set a minimum threshold to avoid dividing zero).

Figure 9 show the results obtained by sample consensus model without and with using spatial information. From Figure 9 we can see that, compared with the results obtained by the proposed

method without using spatial information or using only vertical information, the results using both vertical and horizontal spatial information are the most accurate.

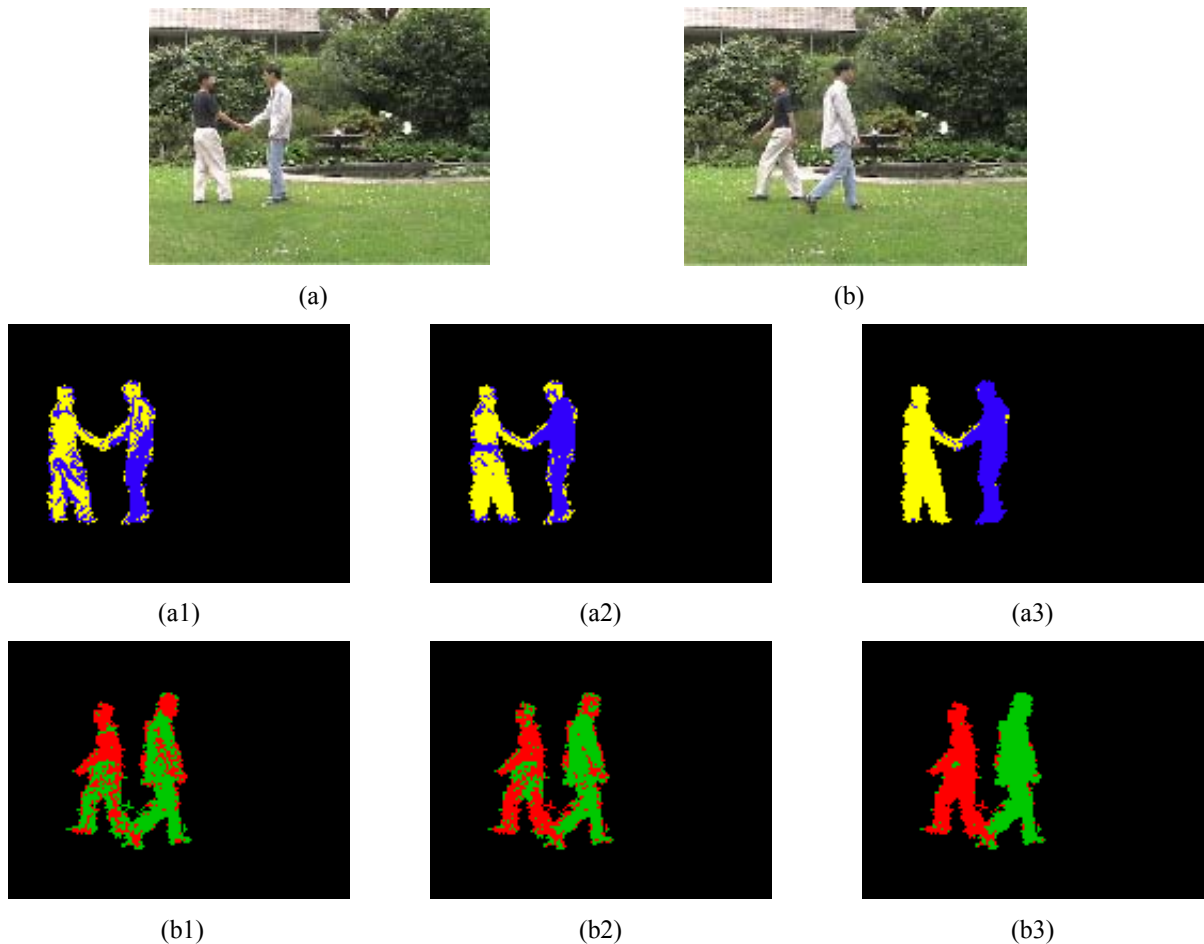


Figure 9. (a) and (b) Original images; (a1) and (b1) segmentation results without using spatial information; (a2) and (b2) segmentation results using only vertical spatial information; (a3) and (b3) segmentation results using both vertical and horizontal spatial information.

4.3 Experiments on Segmenting and Tracking People with Occlusions

To segment and track people in video sequences, first, we apply our proposed SACON background modelling method (described in section 2.2 and 2.3) to extract the foreground regions from the video sequences. Then, we input the foreground regions (which should correspond to

humans), to the algorithm proposed in section 4.1 and section 4.2. We initialize the human appearance models $\{P_j\}_{j=1,\dots,k}$ after people have been detected for several frames (i.e., when they are stable) and afterwards, we update the human appearance models at each frame until they occlude each other or merge to a group (when we freeze most of the parameters for the k -th person - as described in the previous section).



Figure 10. Segmentation and tracking results according to Equation (12). The first row: original images (frame numbers from the left column to the right column are 227, 228 and 229); The second row: human segmentation results; The third row: tracking results.

Figure 10 shows the segmentation results of humans by the proposed method. From Figure 10, we can see that, although there are some pixels misclassified in the segmented results, most pixels of the two persons are correctly classified and the persons are successfully tracked even when they are subject to occlusions.

Figure 11 and Figure 12 show more segmentation and tracking results by the proposed method. In Figure 11, one person passed by and almost completely occluded the other person. However, the proposed method successfully tracked both persons when they pass across and occluded each other.

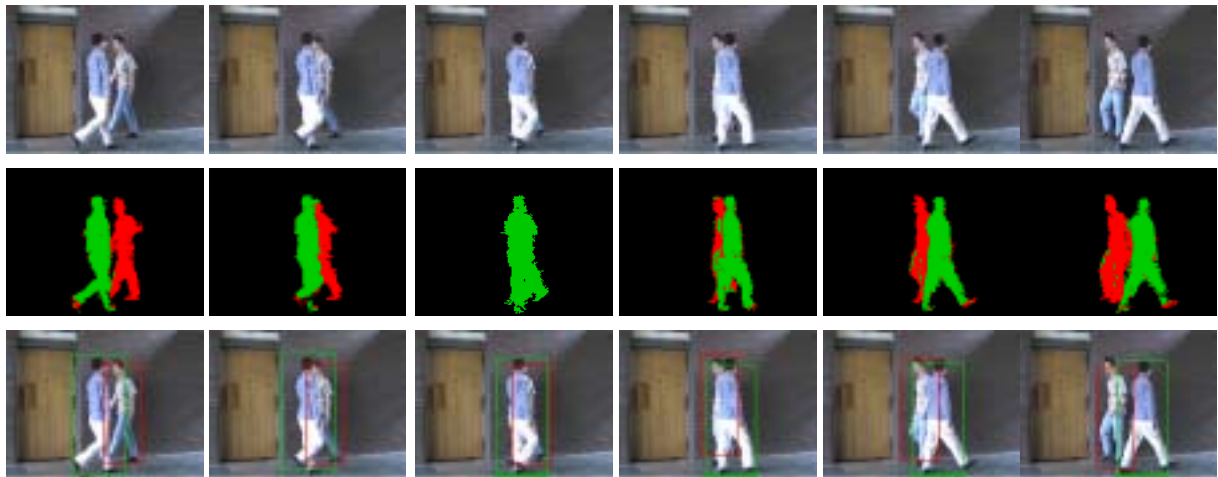


Figure 11. Segmentation and tracking results. The first row: original images (frame numbers from the left column to the right column are 103 to 108); The second row: human segmentation results; The third row: tracking results.

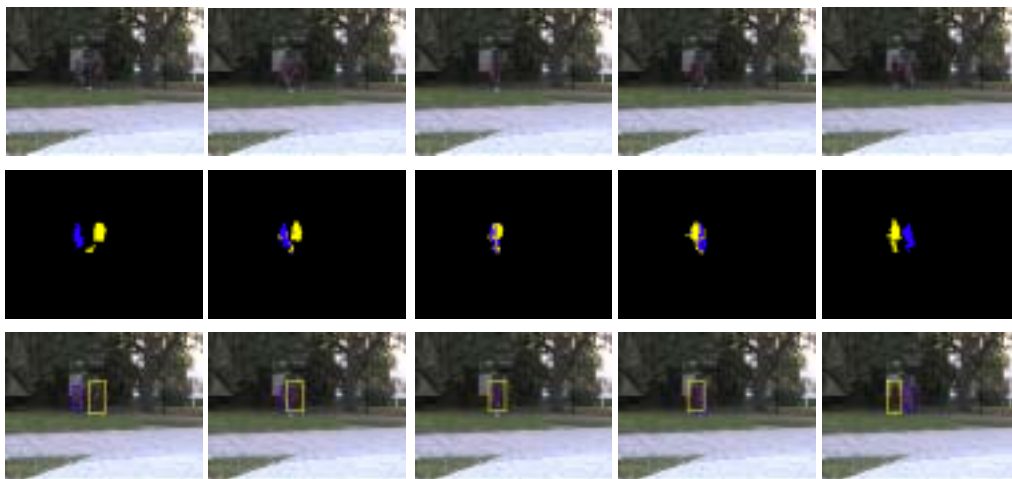


Figure 12. Segmentation and tracking results. The first row: original images (frame numbers from the left column to the right column are 236 to 240); The second row: human segmentation results; The third row: tracking results.

As shown in Figure 12, the color of the people's clothes is close to that of the background scene and the appearance of the people is similar to each other. These similarities increase the difficulties in not only background subtraction but also in modeling foreground appearance. The people are also small. Despite these facts, the proposed method successfully tracked both persons.

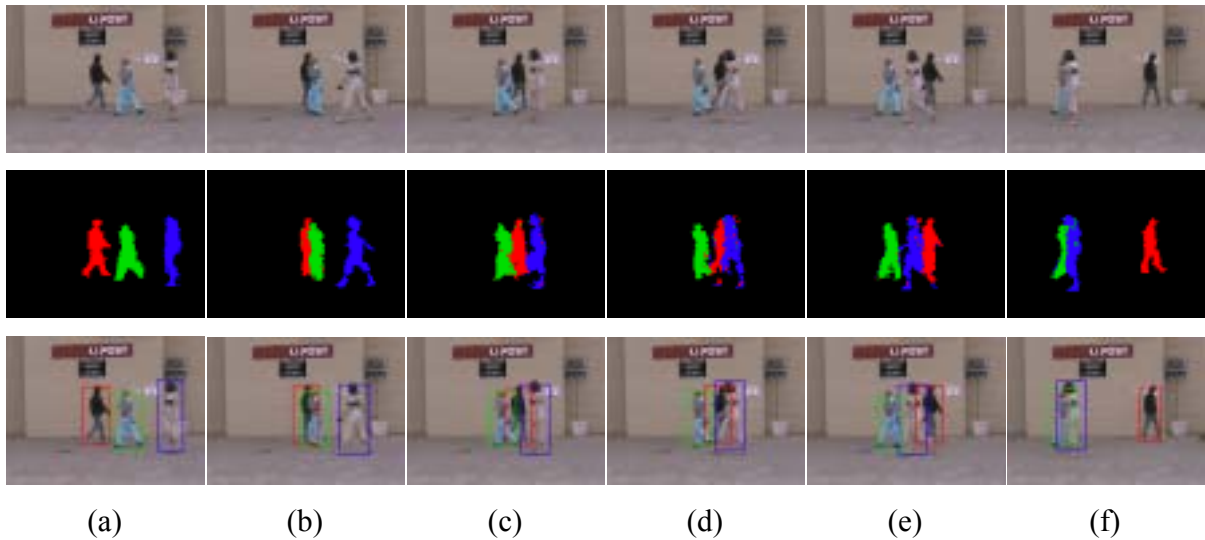


Figure 13 Segmentation and tracking of three people. The first row: original images (frame numbers: 114,121,128,130,136, and 151); The second row: human segmentation results; The third row: tracking results by the proposed method.

Figure 13 shows an example that the proposed method is effective to segment and track multiple people with occlusions. There are three almost complete occlusions happening in the video sequence. From Figure 13 (c)-(e), we can see that the three persons form a group and occlude one another. The proposed method effectively segments the three persons and correctly tracks all the three persons throughout the sequence.

5. Conclusion

In this paper, we propose an effective and robust background modeling method (SACON). The method may be applied in many practical environments and is effective in modeling a dynamic

background. We also proposed an effective framework 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 in background modelling.

In the second part of the paper, we exploit our background modelling approach to build a system that segments and tracks people even in the presence of occlusions. In doing so, we present a new sample consensus based method for modelling human appearance and handling occlusion problem in human segmentation and tracking. Both color and spatial information are employed in our human appearance model. We show that the proposed method can successfully segment and track people through occlusion, using both outdoors and indoors video sequences. Promising results have been achieved.

However, we must acknowledge that the proposed method has some limitations and restrictions. For example, we assume that the poses of people are upright. Neither the foreground appearance nor the vertical spatial distribution of people experience dramatic changes when occlusions occur. We will work on the solution in our future work.

Acknowledgements

This work is supported by ARC grant DP0452416. The work was carried out within the Monash University Institute for Vision Systems Engineering. We thank Dr. Kentaro Toyama and Dr. John Krumm for their kind help in providing the Wallflower image sequences.

References

1. Connell, J., et al. Detection and Tracking in the IBM PeopleVision System. *IEEE ICME*. 2004. p. 1403 - 1406.
2. Cucchiara, R., et al., Detecting Moving Objects, Ghosts, and Shadows in Video Streams. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2003. **25**(10): p. 1337- 1342.
3. Elgammal, A., D. Harwood, and L.S. Davis. Non-parametric Model for Background Subtraction. *6th European Conference on Computer Vision*. 2000. p. 751-767.
4. Elgammal, A. and L.S. Davis. Probabilistic Framework for Segmenting People Under Occlusion. *Proc. IEEE 8th Int. Conf. Computer Vision*. 2001. p. 145–152.
5. Elgammal, A., et al., Background and Foreground Modeling using Non-parametric Kernel Density Estimation for Visual Surveillance. *Proceedings of the IEEE*, 2002. **90**(7): p. 1151-1163.
6. Elgammal, A., R. Duraiswami, and L.S. Davis, Efficient Kernel Density Estimation Using the Fast Gauss Transform for Computer Vision. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2003. **25**(11): p. 1499-1504.
7. Fischler, M.A. and R.C. Rolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 1981. **24**(6): p. 381-395.
8. Friedman, N. and S. Russell. Image Segmentation in Video Sequences: A Probabilistic Approach. *Proc. Thirteenth Conf. on Uncertainty in Artificial Intelligence*. 1997. p. 175-181.
9. Gutchess, D., et al. A Background Model Initialization Algorithm for Video Surveillance. *IEEE Int'l Conference on Computer Vision*. 2001. p. 733-740.
10. Haritaoglu, I., D. Harwood, and L.S. Davis, W4: Real-Time Surveillance of People and Their Activities. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2000. **22**(8): p. 809-830.
11. Harville, M. A Framework for High-Level Feedback to Adaptive, Per-Pixel, Mixture-of-Gaussian Background Models. *7th European Conference on Computer Vision*. 2002. p. 543-560.
12. Hu, W., et al., A Survey on Visual Surveillance of Object Motion and Behaviors. *IEEE Trans. Systems, Man, and Cybernetics—Part C*, 2004. **34**(3): p. 334-352.
13. Javed, O., K. Shafique, and M. Shah. A Hierarchical Approach to Robust Background Subtraction using Color and Gradient Information. *IEEE Workshop on Motion and Video Computing*. 2002. p. 22-27.

14. Karmann, K.-P. 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.
15. Khan, S. and M. Shah. Tracking People in Presence of Occlusion. *Asian Conference on Computer Vision*. 2000. p. 263-266.
16. Kim, K., et al., Real-time Foreground-background Segmentation Using Codebook Model. *Real-Time Imaging*, 2005. **11**(3): p. 172-185.
17. Kottow, D., M. Koppen, and J. Ruiz-del-Solar. A Background Maintenance Model in the Spatial-Range Domain. *2nd Workshop on Statistical Methods in Video Processing*. 2004. p. 141-152.
18. Lumia, R., L. Shapiro, and O. Zungia, A New Connected Components Algorithm for Virtual Memory Computer. *Computer Vision, Graphics, and Image Processing*, 1983. **22**(2): p. 287-300.
19. McKenna, S.J., Y. Raja, and S. Gong, Tracking Color Objects Using Adaptive Mixture Models. *Image Vision Computing*, 1999. **17**(3): p. 225-231.
20. McKenna, S.J., et al., Tracking Groups of People. *Computer Vision and Image Understanding*, 2000. **80**: p. 42-56.
21. Mittal, A. and L.S. Davis, M₂ Tracker: A Multi-View Approach to Segmenting and Tracking People in a Clutter Scene. *International Journal of Computer Vision*, 2003. **51**(3): p. 189-203.
22. Mittal, A. and N. Paragios. Motion-Based Background Subtraction using Adaptive Kernel Density Estimation. *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition*. 2004. p. 302-309.
23. Oliver, N., B. Rosario, and A. Pentland. A Bayesian Computer Vision System for Modeling Human Interactions. *International Conference on Computer Vision Systems*. 1999. p. 255 - 272.
24. Seki, M. and T.F. Wada, H.Sumi, K. Background Subtraction Based on Cooccurrence of Image Variations. *Computer Vision and Pattern Recognition*. 2003. p. 65-72.
25. Senior, A. Tracking with Probabilistic Appearance Models. *Performance Evaluation of Tracking and Surveillance Systems*. 2002. p. 48-55.
26. Stauffer, C. and W.E.L. Grimson. Adaptive Background Mixture Models for Real-time Tracking. *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition*. 1999. p. 246-252.
27. Toyama, K., et al. Wallflower: Principles and Practice of Background Maintenance. *7th International Conference on Computer Vision*. 1999. p. 255-261.

28. Wang, L., W. Hu, and T. Tan, Recent Developments in Human Motion Analysis. *Pattern Recognition*, 2003. **36**(3): p. 585-601.
29. Wren, C.R., et al., Pfinder: real-time tracking of the human body. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1997. **19**(7): p. 780-785.