# Real-Time and Robust Background Updating for Video Surveillance and Monitoring

Xingzhi Luo and Suchendra M. Bhandarkar

Department of Computer Science, The University of Georgia,
Athens, Georgia 30602-7404, USA

**Abstract.** Background updating is an important aspect of dynamic scene analysis. Two critical problems: sudden camera perturbation and the *sleeping person* problem, which arise frequently in real-world surveillance and monitoring systems, are addressed in the proposed scheme. The paper presents a multi-color model where multiple color clusters are used to represent the background at each pixel location. In the proposed background updating scheme, the updates to the mean and variance of each color cluster at each pixel location incorporate the most recently observed color values. Each cluster is assigned a weight which measures the time duration and temporal recurrence frequency of the cluster. The *sleeping person* problem is tackled by virtue of the observation that at a given pixel location, the time durations and recurrence frequencies of the color clusters representing temporarily static objects are smaller compared to those of color clusters representing the true background colors when measured over a sufficiently long history. The camera perturbation problem is solved using a fast camera motion detection algorithm, allowing the current background image to be registered with the background model maintained in memory. The background updating scheme is shown to be robust even when the scene is very busy and also computationally efficient, making it suitable for real-time surveillance and monitoring systems. Experimental results on real traffic monitoring and surveillance videos are presented.

## 1 Introduction

Separating foreground from background (also known as figure-ground discrimination) is an important though difficult problem in computer vision. Many background models have been proposed for surveillance systems. Based on the manner in which they are generated, existing background models can be categorized as off-line (static) background models [1,2,3] or online (adaptive) background models [4,5,6,7,8,9,10]. Due to the dynamic nature of the background, online adaptive background models are usually preferred in real time video surveillance and monitoring systems. Broadly speaking, there are two categories of online methods to model the background image. The first category models the background image using a single color value at each pixel location [4,5,6,7], whereas the second category uses multiple color values at each pixel location [8,9,10]. For dynamic scenes with high levels of random noise, a single value is not sufficient to represent the background color at a given pixel location. In the context of video surveillance and monitoring in outdoor scenes, the source of random noise could be the swaying of trees and the movement of grass due to breeze.

M. Kamel and A. Campilho (Eds.): ICIAR 2005, LNCS 3656, pp. 1226–1233, 2005.
© Springer-Verlag Berlin Heidelberg 2005

In this paper we propose an online background updating scheme for a real-time traffic surveillance and monitoring system. The background color at a pixel location is modeled as a mixture of Gaussian distributions as in [8,9,10]. In particular, we address two critical problems that are confronted by real-world surveillance and monitoring systems. The first problem is sudden camera perturbation, which occurs occasionally but causes typical background updating schemes to fail. In the context of traffic monitoring and surveillance, cameras mounted on bridges or overpasses are typically subject to structural vibrations caused by especially heavy moving vehicles, resulting in sudden and random camera perturbations. The second problem is the *sleeping person* problem [7] where a moving object stops in the scene and becomes motionless for some duration of time. A typical adaptive background model would result in the improper merging of the temporarily stationary object with the background image. The sleeping person problem arises frequently in the context of automated traffic monitoring when moving vehicles stop temporarily at traffic lights or intersections.

## 2 Proposed Background Model

In the proposed background model, the camera perturbation is modeled as a Euclidean transformation. A background image with dimensions greater than those of the actual image frame in the video stream is generated and continuously updated. The first image frame is aligned with the center of the background image. For each successive image frame, a fast algorithm is used to estimate the camera motion (perturbation) parameters and consequently determine the alignment of the new image frame with the stored background image. The background updating procedure is performed on those locations within the background image that overlap with the new image frame.

The color values at each pixel location in the background image are modeled as a multiple Gaussian mixture (MGM). A novel weight updating scheme for the color clusters is used to address the sleeping person problem. Once every $T$ frames, the number of color values that fall into each cluster is computed and stored in a counter. The corresponding cluster weight is updated once every $T$ frames based on the counter value and the previous history of the cluster. A new cluster may be created and an old cluster deleted after comparing all the cluster weights. The cluster weight evaluation scheme takes into consideration both the cluster duration and the recurrence frequency. The key idea behind the proposed approach is to use the cluster weight to approximate the cluster duration thus enabling one to decide whether or not to adapt a new cluster into the background model. A significant advantage of the proposed approach is that since the background updating is done once every $T$ frames, it is computationally very efficient and very well suited for real-time applications.

### 2.1 Camera Motion Estimation

As is common in most surveillance systems, the camera is assumed to be stationary. However, we account for sudden camera perturbations, which occur occasionally but nevertheless cause typical background updating schemes to fail. In the context of traffic monitoring and surveillance, cameras mounted on bridges or overpasses are typically subject to structural vibrations caused by especially heavy moving vehicles, resulting

in sudden and random camera perturbations. Since the range of the perturbations is typically small, we model them as a simple Euclidean transformation given by $x' = x\cos(\theta) - y\sin(\theta) + s_x$ and $y' = x\sin(\theta) + y\cos(\theta) + s_y$, where $s_x$ and $s_y$ are the translational parameters along the X axis and Y axis respectively, and $\theta$ is the angle of rotation assuming the first image frame in the video stream to be the reference image frame. We also assume that $s_x, s_y \in [-D_s, D_s]$ and $\theta \in [-D_\theta, D_\theta]$ where $D_s$ and $D_\theta$ are predefined bounds. The parameters $(s_x, s_y, \theta)$ are determined as follows:

1. For each triple $(s_x, s_y, \theta)$, compute the region of overlap between the image frames $f(0)$ and $f(t)$.
2. Compute a match metric between the two image frames within the rectangular region of overlap.
3. Output the parameters $(s_x, s_y, \theta)$ which optimize (minimize, in our case) the match metric between the two images.

Given the transformation parameters $(s_x, s_y, \theta)$, the region of overlap $R$ between $f(0)$ and $f(t)$ is computed. The match metric is given by $m = \sum_{(x,y)\in R} |f(x', y'; t) - f(x, y; 0)|/A$, where $A$ is the area of the region of overlap $R$. As an alternative to brute-force search in the parameter space $(s_x, s_y, \theta)$, a more efficient search algorithm is used. Since the camera is expected to be primarily static with a very small angle of rotation, a local search procedure is used to determine the translational parameters $(s_x, s_y)$ and $\theta$ is assumed to be 0. The algorithm is described as follows:

1. Begin with $(s_x, s_y) = (0, 0)$ and step size $\delta = \delta_0$.
2. Explore the four neighbors of $(s_x, s_y)$ in parameter space given by $(s_x + \delta, s_y + \delta)$, $(s_x + \delta, s_y - \delta)$, $(s_x - \delta, s_y + \delta)$, and $(s_x - \delta, s_y - \delta)$. If any of these neighbors results in a lower match metric, replace the current $(s_x, s_y)$ with the neighbor that results in the lowest match metric.
3. Repeat step 2 until there is no change in the match metric.
4. Update $\delta = \delta/2$, and repeat steps 2 and 3.
5. Repeat steps 2, 3 and 4 until $\delta = \delta_{min}$.

Based on empirical observations on real data, the coarse-to-fine tuning of the step size $\delta$ is seen to prevent the search from being trapped in a local minimum when the range of translational motion is small.

After the optimal translational parameters $(s_x, s_y)$ are determined, if the match metric $m$ is above a certain predefined threshold, then we search in the space of the rotational parameter $\theta$ in the range $[-D_\theta, D_\theta]$ to determine the optimal Euclidean transform parameters. If $m$ is less than the threshold, then no further search in the $\theta$ space is deemed necessary, i.e. $\theta = 0$. After the alignment of the new image frame with the background image is performed, the background updating is done at the corresponding locations in the background image.

### 2.2 Background Image Updating

A background color at a pixel location usually persists for a longer time duration than any foreground color and has a higher frequency of recurrence [1,8]. Thus, it is logical

to assign to each cluster center a weight that takes into account both, the time duration of the cluster and its recurrence frequency as an alternative to the simple weight updating scheme described in [10]. Consequently, in the proposed scheme, the weight assigned to each cluster is indicative of both, the time duration of the cluster and the cluster recurrence frequency. Thus each cluster is characterized by the following parameters:

- $C_i$: Centroid or mean of the $i$th color/gray level cluster.
- $\sigma_i^2$: Variance of the $i$th color/gray level cluster.
- $N_i$: Total number of colors/gray levels that have matched the $i$th cluster. Initially, $N_i = 1$ for all clusters.
- $tl_i$: The most recent time that the $i$th cluster has been updated. Initially, $tl_i = 0$ for all clusters.
- $n_i$: The number of colors/gray levels that have matched the $i$th cluster in recent history.

Given color $X_k$ at a certain pixel location in the current frame, we compare it to the existing cluster centroids associated with this pixel location. If $X_k \in [C_i - 2.5\sigma_i, C_i + 2.5\sigma_i]$ then $X_k$ is deemed to match the $i$th cluster. The centroid and covariance of the $i$th cluster are updated as follows:

$$C_i = C_i + \frac{1}{L}(X_k - C_i) \tag{1}$$

$$\sigma_i^2 = \sigma_i^2 + \frac{1}{L}((X_k - C_i)^2 - \sigma_i^2) \tag{2}$$

where $L$ is an integer representing the inverse of the learning rate. The advantage of using an integer $L$ instead of the learning rate $\alpha$ in equations (1) and (2), is that the need for floating point computation at each update is averted. For example, in equation (1) we can accumulate the difference $(X_k - C_i)$ and decrement $C_i$ by 1 if $(X_k - C_i) < -L$ and increment $C_i$ by 1 if $(X_k - C_i) > L$. If color $X_k$ does not match an existing cluster then a new cluster is created replacing an existing cluster $j$ with minimum weight $N_j$.

In order to efficiently compute the time duration and recurrence frequency of a cluster, we quantize the time series $x(t)$ into time slices of interval $T$. For all clusters in a given time slice, if the number of colors that have been assigned to cluster $i$ in that time slice is $n_i$ then the time duration of the cluster is updated as: $N_i = N_i + T$ if $n_i > T/2$; otherwise $N_i = N_i + n_i$. Thus, if a cluster at a given pixel location is assigned more than $T/2$ colors in a given time slice, then this cluster is deemed to dominate this time slice. Consequently, we reward this cluster by adding $T$ to $N_i$ else we update its time duration by adding the actual number of matched colors $n_i$ to $N_i$.

We also check for the recurrence frequency of clusters. If a cluster has not been matched for some period of time and then matched again, it is probable that the cluster does represent the real background. If this cluster has been deemed to be sufficiently exposed during the current time slice, i.e., $n_i \geq \delta$ then we increase its weight by increasing its value of $N$. On the other hand, if $n_i < \delta$ then the cluster is deemed insufficiently exposed and its recurrence frequency ignored during the current time slice. We measure the recurrence frequency of the $i$th cluster by checking the last time $tl_i$ that the cluster was matched. If $t - tl_i > 2T$, then $N_i = N_i + T/2$, that is $N_i$ is incremented by an extra duration $T/2$ to account for the cluster recurrence and $t_l$ is set to $t$. Checking for

recurrence frequency is useful when the dynamic scene is very busy. In this situation, the true background color/gray level may not persist at a given pixel location for a long time duration, however its recurrence frequency will typically be high. Increasing the value of $N$ to account for the high recurrence frequency increases the probability of this color/gray level to be considered as part of the background.

The clusters at each pixel location are ranked on the basis of their $N$ values, the higher the value of $N$, the higher the priority for the corresponding color/gray level of that cluster to be considered as part of the background. However, it is necessary to set an upper limit for the value of $N_i$ since too large a value of $N_i$ will make it difficult for an actual new background color cluster to be considered as part of the background. We set the upper limit of $N_i$ to $\Delta$. At any pixel location, if $N_{max} > 1.25\Delta$ where $N_{max} = \max_i(N_i)$, we scale down all the $N_i$ values by multiplying them by $4/5$. If $N_i = 0$ then the $i$th cluster is deleted. All the clusters which satisfy the condition $N_i > N_{max}/3$ are deemed to represent the valid background colors/gray levels. If a cluster has not been updated for a time period $\Delta$, then it is deleted. The background updating is performed once every $T$ frames.

The background updating algorithm is summarized as follows:

1. Given an observed color/gray level $X_k$ at a pixel, check all of the pixel's clusters. If $X_k$ matches cluster $i$, then update the centroid and the variance of cluster $i$ using equations (1) and (2). Set $n_i = n_i + 1$.
2. If there is no match, create a new cluster and replace an existing cluster with the smallest $N_i$ value. For the new cluster, set $C = X_k$, $N = 1$, $n = 1$, $\sigma^2 = \sigma_0^2$ and $tl = k$.
3. If $(t \mod T) \equiv 0$ and $t > 0$ then for each cluster $i$ at each pixel,
   (a) Check the value of $n_i$ and update $N_i$ as follows:
      i. If $n_i > T/2$, then $N_i = N_i + T$.
      ii. Otherwise, $N_i = N_i + n_i$.
   (b) Check for recurrence: If $n_i > \delta$ and $t - tl_i > 2T$, then $N_i = N_i + T/2$ and $tl_i = t$.
   (c) Reset all $n_i$ values to zero.
   (d) Check which clusters will be deemed as belonging to the background. All clusters $i$ such that $N_i > N_{max}/3$ where $N_{max} = \max_i(N_i)$ are considered to belong to the background.
   (e) If $N_{max} > 1.25\Delta$, then $N_i = N_i * 4/5$, for all $i$.
   (f) For any $i$, if $k - tl_i > \Delta$, then delete this cluster.

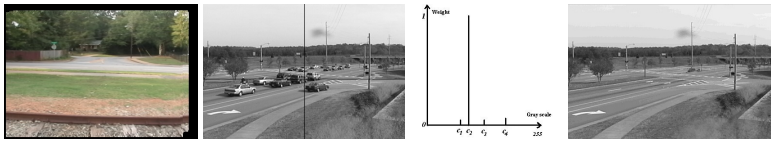
## 3 System Implementation

The background updating scheme described above was incorporated into a real-time traffic monitoring system and tested on color (RGB) and grayscale video sequences of real traffic scenes. All video sequences were sampled at a constant rate of 30 frames per second (fps). We chose values of $L = 1024$ and $k = 4$ in our implementation. Since the background is constantly refreshed, the value of $N_i$ for the cluster corresponding to the actual background eventually increases to $1.25\Delta$. In the context of traffic monitoring, we assumed that a vehicle stops temporarily for up to 2 minutes at a traffic light

or intersection, which amounts to 3600 frames at a sampling rate of 30 frames per second. In order to avert the *sleeping person* problem, this delay of 3600 frames must be less than $\Delta/3$ since all clusters for which $N_i \geq \Delta/3$ are considered to be part of the background. Since we need to choose a value of $\Delta > 3 \times 3600 = 10800$, we chose a value of $\Delta = 12000$. We also chose $T = 2$ seconds (60 frames) and $\delta = 20$. Thus, if a cluster that has not been updated for the past $2T = 120$ frames, has received more than 20 updates in the current time slice, then it is treated as an instance of a recurring background color/gray level and the $N_i$ value of the cluster is incremented by $T/2$.

## 4 Experimental Results

To simulate camera perturbation, the video streams were gathered while the tripod mount of the camera was being manually shaken. Figure 1(a) shows the background image from the moving camera.The actual size of the image frame is $360 \times 240$ pixels whereas the background image size is $400 \times 280$ pixels.




(a) Camera Motion (b) Frame3541 (c) Weights (d) Backgroung image

**Fig. 1.** Experimental Results

Experimental results on the captured videos captured show that the proposed scheme for camera motion compensation works well and that the background updating scheme is capable of recovering from abrupt and random camera motion, provided it is small.

A comparison between the proposed background updating scheme and that of Stauffer and Grimson [10] is performed using video streams, captured by a static camera, of a busy traffic scene containing several vehicles and with traffic lights present. Figures 1(b), 1(c) and 1(d) summarize the experimental results on a grayscale video sequence. Figure 1(b) shows a grayscale image frame at time $t = 3541$ where time is measured in terms of the frame number in the video sequence. In this frame it is evident that some cars have stopped at a traffic light. Figure 1(c) shows the weights $N_i$ of each of the $k = 4$ gray level clusters associated with pixel location $(180, 134)$ at time $t = 3541$. In Figure 1(c), the cluster centroids are denoted by $C_1$, $C_2$, $C_3$ and $C_4$ and the maximum cluster weight $N_{max}$ is marked as 1 with the other cluster weights scaled in proportion. In Figure 1(b), the pixel location $(180, 134)$ is marked by the intersection of the corresponding vertical and horizontal lines for the sake of clarity. It can be seen that although there is a stationary car at this pixel location for some length of time, the weight of the corresponding gray level cluster is small compared to that of the gray level cluster which denotes the actual background. Hence the gray level of the stationary car

(a) Our scheme (b) Grimson's scheme

**Fig. 2.** Background image and object segmentation at time 3841 with learning rate $\alpha = 1/1024$

at this pixel location is not treated as a background gray level. Figure 1(d) shows the background image generated with the proposed background model where, at each pixel location, the gray level displayed is the one corresponding to the cluster center with the maximum weight $N_{max}$. Since none of the stationary vehicles are merged with the background image, the background updating scheme can be seen to be robust in the face of the *sleeping person* problem. It is also adaptive since the cluster parameters are periodically updated. However, since the updating is done once every $T = 60$ frames (i.e., 2 seconds at 30 fps), the proposed scheme is also computationally efficient.

In an RGB color video sequence, each color cluster is represented as $C = \{c_r, c_g, c_b, \sigma^2\}$, that is, the same $\sigma^2$ value is used to represent the variance of the cluster along each of the R, G and B axes. Given a new observation $X = (r, g, b)$ and a cluster $C = \{c_r, c_g, c_b, \sigma^2\}$, if the condition $d < 2.5\sigma$ is satisfied, where $d = \max(d_r, d_g, d_b)$ and $d_r = |r - c_r|$, $d_g = |g - c_g|$, $d_b = |b - c_b|$, then $X = (r, g, b)$ is deemed to have matched the cluster $C$. Furthermore, the parameters of cluster $C$ are updated as follows: $c_r = c_r + (r - c_r)/L$, $c_g = c_g + (g - c_g)/L$, $c_b = c_b + (b - c_b)/L$ and $\sigma^2 = \sigma^2 + (1/L)(d_r^2 + d_g^2 + d_b^2 - 3\sigma^2)$.

Figures 2(a) summarize the results of our scheme for an RGB color video sequence. The image on the left in Figure 2(a) is the background image generated at time $t = 3841$, and the image on the right is the result of foreground object segmentation at the same time instant. It is evident that the stopped cars can be detected using background subtraction.

Figure 2(b) depicts the background image and the result of foreground segmentation at time $t = 3841$ using the background updating scheme proposed by Stauffer and Grimson [10] with the same learning rate of $1/1024$. It is evident that the objects that become temporarily motionless are partially merged into the background and cannot be extracted. More results on real traffic monitoring videos are available online at *http://www.cs.uga.edu/∼xingzhi/research/bkg/demo/*. From these videos it can be seen that, the proposed scheme suffers from the sleeping person problem initially when little information about the background is known. However, the proposed scheme is observed to eventually overcome this problem and converge to a stable background image. Stauffer and Grimson's scheme, in contrast, is observed to suffer from the sleeping person problem from time to time.

## 5 Conclusions

In this paper we proposed a background updating scheme for a real-time traffic monitoring system. Specifically, we addressed the camera perturbation problem and the *sleep-*

*ing person* problem. To make the background updating scheme adaptive to gradual changes in the background, we used a multi-color model where multiple color clusters were associated with each pixel location in the background image. The cluster parameters were updated periodically to adapt to gradual changes in the background. To make the proposed background updating scheme robust in the face of the *sleeping person* problem, the color clusters at each pixel location were assigned weights based on the observation that the clusters corresponding to the real background colors were likely to persist for a longer time duration and also have a higher recurrence frequency compared to the clusters that correspond to colors from temporarily motionless objects. The camera perturbation was modeled as a Euclidean transformation and the camera perturbation compensation procedure was modeled as a fast local search procedure in the perturbation parameter space to optimize an image match metric. Experimental results on grayscale and color video sequences obtained from real traffic scenes showed that the proposed background updating scheme could adapt to gradual or long-term changes in the background while ignoring short-term changes arising from the sleeping person problem. The proposed scheme was also shown to be computationally efficient since most of the computation was performed once every 60 frames (or 2 seconds at 30 fps). Moreover, the background update equations were optimized by greatly reducing the floating point computation, thus making the scheme well suited for real-time applications. Although the proposed real-time background updating scheme was specifically designed for a real-time traffic monitoring system, it is nevertheless applicable to most surveillance systems, in which the *sleeping person* problem is seen to occur but the time period for which a moving object is temporarily stationary has a definite upper bound.

## References


1. D. Farin, P.H.N. deWith and W. Effelsberg, Robust Background Estimation for Complex Video Sequences, *Proc. IEEE ICIP,* Barcelona, Spain, Sept. 2003, pp. 145-148.
2. M. Isard and J. MacCormick, BraMBLe: A Bayesian Multiple-Blob Tracker, *Proc. ICCV*, Vancouver, Canada, July 2001, Vol. 2, pp. 34-41.
3. M. Massey and W. Bender, Salient stills: Process and Practice, *IBM Sys. Jour.*, 1996, Vol. 35, Nos. 3&4, pp. 557-573.
4. D. Koller, J.W. Weber and J. Malik, Robust Multiple Car Tracking with Occlusion Reasoning, *Proc. ECCV,* Stockholm, Sweden, 1994, pp. 189-196.
5. C. Ridder, O. Munkelt and H. Kirchner, Adaptive Background Estimation and Foreground Detection Using Kalman Filtering, *Proc. of Intl. Conf. Recent Adv. Mechatronics*, Istanbul, Turkey, 1995, pp. 193-199.
6. S. Kamijo, Traffic Monitoring and Accident Detection at Intersections, *IEEE Trans. Intell. Transp. Sys.*, Vol. 1, No. 2, June 2000, pp. 108-118.
7. K. Tooyama, J. Krumm, B. Brumit, and B. Meyers, Wallflower: Principles and Practice of Background Maintenance, *Proc. ICCV*, Corfu, Greece, Sept. 1999, pp. 255-261.
8. D. Butler, S. Sridharan and V.M. Bove, Jr., Real-time Adaptive Background Segmentation, *Proc. IEEE ICME*, Baltimore, MD, July 2003.
9. P. KaewTraKulPong and R. Bowden, An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection, *Proc. Wkshp. Adv. Vision-based Surveillance Sys.*, Kingston, UK, Sept. 2001.
10. C. Stauffer and W.E.L. Grimson, Adaptive Background Mixture Models for Real-time Tracking. *Proc. IEEE Conf. CVPR*, Ft. Collins, CO, June 1999, pp. 246-252.