Controlling Background Subtraction Algorithms for Robust Object Detection

Anh-Tuan Nghiem, François Bremond, Monique Thonnat

Controlling Background Subtraction Algorithms for Robust Object Detection

A.T. Nghiem, F. Bremond, M. Thonnat

Project PULSAR, INRIA Sophia Antipolis France

Keywords: Updating background, background subtraction algorithms, adapting parameters.

Abstract

This paper presents a controller for background subtraction algorithms to detect mobile objects in videos. The controller has two main tasks.

The first task is to guide the background subtraction algorithm to update its background representation. To realize this task, the controller has to solve two important problems: removing ghosts (background regions misclassified as object of interest) and managing stationary objects. The controller detects ghosts based on object borders. To manage stationary objects, the controller cooperates with the tracking task to detect faster stationary objects without storing various background layers which are difficult to maintain.

The second task is to initialize the parameter values of background subtraction algorithms to adapt to the current conditions of the scene. These parameter values enable the background subtraction algorithms to be as much sensitive as possible and to be consistent with the feedback of classification and tracking task.

1 Introduction

Detecting mobile objects is an important task in many video analysis applications such as video surveillance, people monitoring, video indexing for multimedia. Among various object detection methods, the ones based on adaptive background subtraction [12, 9] are the most popular. However, the background subtraction algorithm alone could not easily handle various problems such as adapting to changes of environment, removing noise, detecting ghosts (defined in section 2) etc. To help background subtraction algorithms to deal with these problems we have constructed a controller for managing object detection algorithms. Being independent from one particular background subtraction algorithm, this controller has two main tasks:

- Supervising background subtraction algorithms to update their background representation.
- Adapting parameter values of background subtraction algorithms to be suitable for the current conditions of the scene.

The article is organized as follows. Section 2 describes the problems the controller has to solve to realize the above tasks. This section also presents the state of the art related to these problems. Section 3 describes how the controller detects ghosts. Section 4 details the method to manage stationary objects. Section 5 presents our method to adapt the parameter values of the background subtraction algorithms to the current conditions of the scene. Several experiments are presented in section 6. The conclusion is presented in section 7.

2 Problem statement and related work

For object detection algorithms which work with very long video sequences, they must be able to adapt to various changes of the scene. To do this, background subtraction algorithms should update their background representation regularly. Therefore, after a certain number of frames, the changes of the scene are absorbed into the background representation and these changes do not occur in the detection results again. However, in scenes like the one in figure 1, a person can often stay at the same place for a long time. Consequently, if we do not distinguish this person from the changes of the scene, after a while, the person will be absorbed into the background representation and the background subtraction algorithm will not be able to detect this person. In [7], Harville et al propose a simple solution for this problem. In their framework, whenever the classification task detects a person from the segmentation results, the background subtraction algorithm does not update the corresponding region. However, the classification task may be wrong, i.e. it misclassifies a background region as a person. For example, in figure 2, at the beginning, the person sits in the chair and the background subtraction algorithm does not have the background corresponding to the chair region occupied by the person. As a result, when this person moves to the table, the background at the chair is viewed by the camera and the classification task classifies this region as a sitting person (called a ghost). In this case, the updating strategy of Harville et al will not update the newly observed region and the "ghost" person (the background region classified as person) remains in the detection results forever.

To distinguish ghosts from real mobile objects, some works employ object borders (i.e. edges). In [4], Connell et al compute the edge energy along the border of each detected object. A foreground region is considered as a ghost if it does not have a sufficient amount of edges. In [10], Lu et al use the

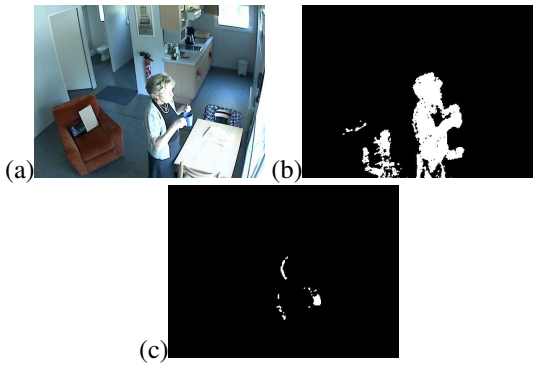

Figure 1. The effect of uniform updating. Figure (a) is the original image. The background subtraction algorithm detects the person in this frame correctly (figure (b)). However, after 80 frames, the person is absorbed into background and the background subtraction algorithm cannot detect the person any more (figure (c)).

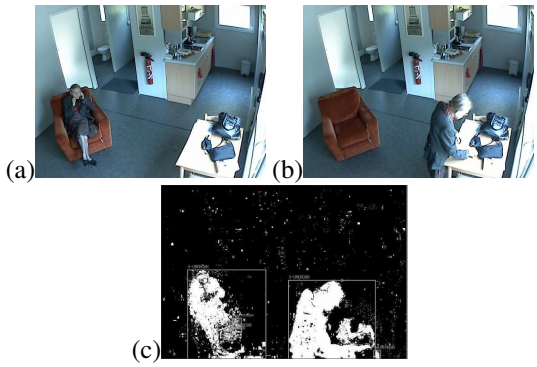

Figure 2. The "ghost" problem of selective updating strategy. At the beginning (figure (a)), the person sits in the chair and the background subtraction algorithm does not see the background occupied by the person. Then, when the person moves to the table (figure (b)), the background at the chair can be observed and the classification task classifies it as a person. Therefore, the background subtraction algorithm does not update the corresponding region. Then the "ghost" person stays forever in the detection results.

in-painting algorithm to fill the region corresponding to each detected object. If it is a ghost, it does not have strong border and the algorithm could fill a large part or the whole object. In general, these techniques can have good results if background subtraction algorithms can correctly detect the real object border. However this requirement cannot always be satisfied, for example in case of shadows.

Our method for detecting ghosts is also based on border detection. Nevertheless, the proposed method does not require the background subtraction algorithm to detect the object border precisely. This method is also fast enough to be included into a real time object detection system.

Beside detecting ghosts, the controller should also handle stationary objects. For example in figure 3, the background subtraction algorithm has to detect both people and cars. Therefore, the background subtraction algorithm should not integrate the regions of detected people and cars into background. However, when a car stops and a person gets out of the car, the classification task is unable to distinguish the people from the car given only the detection results as in figure 3. This problem is called stationary object problem. To solve this problem, in [5], the authors create a temporal background layer containing the pixel values of the car. Then when people passing in front of the car, the background subtraction algorithm uses this temporal background layer to extract these people from the car detection. When the car moves again, the background subtraction algorithm removes the corresponding temporal background layer. In general, these algorithms perform well when the illumination of the scene does not change much. When such a change happens, these algorithms could have difficulties in maintaining the lighting consistency of the background layers. Moreover, these multi-layer frameworks cannot be applied directly to many background subtraction algorithms such as Gaussian Mixture Model (GMM).

In this article we propose a method to solve the stationary object problem. Unlike the above methods, the proposed

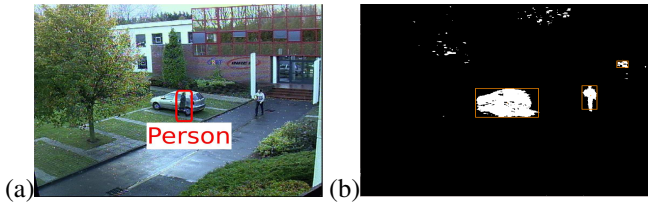

Figure 3. The stationary object problem. Figure (a) is the original frame. Figure (b) shows the detection results of this frame. The background subtraction algorithm should detect both cars and people. Therefore it should not update the regions corresponding to detected cars or people. However, given only the detection result of the background subtraction algorithm, the classification task cannot distinguish the person from the car.

method employs only one one single background representation (single layer) to better adapt to illumination changes. The management of stationary objects is delegated to the tracking task. This task will guide the background subtraction algorithm to update the background representation when necessary. Therefore the proposed method does not have to maintain multiple background layers.

Another problem of background subtraction algorithms is to adapt the parameters of the background subtraction algorithms to the scene conditions. For example, with a background subtraction algorithm such as GMM in [12], the parameter T is a sensitive parameter and its value depends on the type of the background. A small value of T is suitable for static background and a high value of T is suitable for background containing motion such as waves, wind in trees, etc. Selecting the wrong value of T may degrade the performance of GMM seriously as in figure 6. Therefore, a parameter adapting algorithm

is necessary for background subtraction algorithms. In the literature, there are two main approaches to address this problem: online and offline adaptation.

In the offline approach [11, 3], the adaptation algorithm first collects a set of reference videos representing every possible conditions of the scene with ground truth information on the mobile objects. After that, optimization algorithms are used to find optimal parameter value for each reference video. Then for the current scene, the adaptation algorithm finds a similar reference video and applies the optimized parameter value of the reference video to the background subtraction algorithm to process the current scene. The method in [11] is an example of this approach which uses a large amount of ground truth and reference videos illustrating all variations within 24 hours of recording.

In the online approach, to avoid the reference videos which are difficult to construct, the adaptation algorithm uses the current video with the feedback from higher level tasks to evaluate the object detection results. For example, in [6], Hall evaluates object detection results based on the similarity between the trajectory and size of detected mobile objects to the reference model (clusters of trajectories and sizes of mobile objects, learned over long sequences). This method is complicated and the parameter optimization must be run on a separated process (on a different computer in the network for example). Moreover, this controlling algorithm is only able to find a global parameter value set for the whole image since trajectories spread over the scene throughout the video. This global value is not desirable especially if the scene is complex and each region needs a different parameter value. This global value corresponds also to a compromise between all the trajectories detected within the last period of time, thus this parameter value is not necessarily optimal.

In this article, we propose an online parameter adaptation algorithms. Our approach evaluates directly the detection results based on the feedback from the classification and tracking task. Because this method only selects the parameter values among several predefined values, it is fast enough to be included in the object detection system. Finally, it is capable of finding good parameter values for each individual pixel in the image.

3 Removing ghosts

The algorithm for detecting ghosts is based on the heuristic that if we draw a horizontal line cutting a real object, this line (called cut) will intersect with the object border at at least 2 points. Moreover, the distance between these 2 points must be large compared to the object model. For example, with the calibrated camera, in case of people, the 3D distance between these two border points must be larger than 0.2 meter (in case of neck or leg).

Based on this heuristic, the following algorithm is proposed to detect ghosts:

1. Take n horizontal cuts from current object
2. For each cut, verify if it satisfies the heuristic. If yes, increase the number of valid cuts by 1
3. If there are more than $n - k$ valid cuts, the object is considered as a real object. Otherwise, it is classified as a ghost.

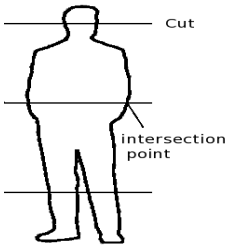

Figure 4. If a detected person is a real person (not a ghost), each line (cut) should cross the person border at at least two points (intersection points).

To check the validity of each cut, the algorithm must be able to find the intersection points with border. At the border, there is often a large difference between the values of adjacent pixels. Formally, a pixel at position (x, y) can be classified as a Vertical Border Point (VBP) as follow:

$$VBP(x,y) = \begin{cases} 1 & \text{if } \exists i \in (R,G,B), \\ & abs(I^i_{(x,y)} - I^i_{(x+1,y)}) > \tau \\ & \vee abs(I^i_{(x,y)} - I^i_{(x-1,y)}) > \tau \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $I^i_{(x,y)}$ is the channel i value at position (x, y), τ is the threshold for the intensity difference, $VBP(x, y) = 1$ means the pixel at the position (x, y) is a possible border pixel.

Among possible border pixels, the algorithm takes the left most and the right most pixels as the real border pixels to verify the lower bound of the distance between border points according to the reference model.

4 Managing stationary objects

As presented earlier, the background subtraction algorithm cooperates with the tracking task to manage stationary objects. To simplify the presentation, we suppose that the type of the stationary objects is car. To detect stationary cars, we use two lists: $TrackedCarPos$ and $CurrentCarPos$. Each element of the $TrackedCarPos$ list contains the bounding box of a car which can become stationary in association with a counter which stores the number of consecutive frames when this car does not move. The $CurrentCarPos$ list contains the bounding box of cars detected in the current frame. At the beginning, both of these lists are empty. The controller tracks the stationary cars as follows:

- In the current frame, whenever a car is detected, its bounding box is pushed into the $CurrentCarPos$ list. The background inside this bounding box is not updated.

- For each element in the $TrackedCarPos$ list:
  - If the bounding box of this element matches exactly one bounding box in the $CurrentCarPos$ list (i.e. the car has not moved), remove the corresponding bounding box from the $CurrentCarPos$ list and increase the stationary counter of the element by 1. If the stationary counter is higher than a threshold, a stationary car is detected.
  - If not, (i.e. when the car moves or when the car has stopped but it is merged with another moving object) decrease the stationary counter. If this counter is still greater than 1, perhaps the car has stopped, so keep this element in the $TrackedCarPos$ list. Otherwise remove this element.
- Insert the bounding boxes which still exist in the $CurrentCarPos$ list into the $TrackedCarPos$ list with stationary counter equal to 1.

A bounding box is represented by two corner points: top left corner (x^{TL}, y^{TL}) and bottom right corner (x^{BR}, y^{BR}). Then a bounding box R_1 matches exactly another one R_2 if:

$$ExactMatch(R_1, R_2) = \begin{cases} 1 & \text{if } abs(x^{TL}_{R_1} - x^{TL}_{R_2}) < \tau_d \\ & \wedge abs(y^{TL}_{R_1} - y^{TL}_{R_2}) < \tau_d \\ & \wedge abs(x^{BR}_{R_1} - x^{BR}_{R_2}) < \tau_d \\ & \wedge abs(y^{BR}_{R_1} - y^{BR}_{R_2}) < \tau_d \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

When the controller detects a stationary car, it orders the background subtraction algorithm to absorb immediately the region inside the car bounding box into background. Therefore, the background subtraction algorithm can detect other mobile objects passing in front of the stationary car. At the same time, it informs the tracking task that there is a stationary car at the location of the bounding box. When the stationary car moves again, a ghost car can occur in the detection results. The tracking task in this case compares the location of the ghost with the list of stationary cars. If it can find a corresponding car, the tracking task removes the ghost as well as the stationary object from its output and informs the background subtraction algorithm to integrate the ghost into the background. As a result, the background subtraction algorithm does not have to take care of storing and updating the old background if the illumination of the scene has changed as this is the case in [5].

5 Tuning parameter for background subtraction algorithms

The tuner aims at evaluating and changing current background subtraction algorithm parameters for every pixel in the image.

To tune the parameters of background subtraction algorithms, the tuner has to evaluate their detection results. However, the background subtraction algorithm alone cannot evaluate its own detection results. Therefore, we employ the feedback information from the classification and tracking task. For example, the classification task may classify a detected foreground region as noise if the size of this region is too small. Then, if the noise level at one particular pixel is too high, we can order the background subtraction algorithms to change its parameter value at this pixel to reduce the noise. On the other hand, if the noise level is too low, perhaps the background subtraction algorithm may be not sensitive enough for detecting mobile objects. Therefore, we tune the background subtraction algorithm to change its parameter value to increase the sensitivity in detecting mobile objects while still maintaining an acceptable noise level.

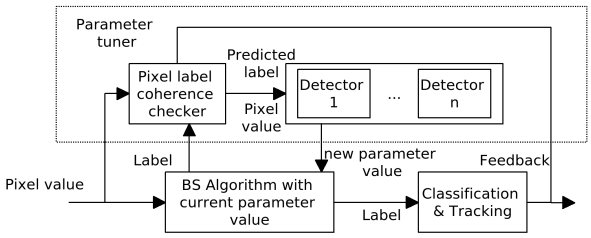

Figure 5. Tuning parameter values for single pixel

The tuner takes as input the current pixel value, the corresponding background subtraction algorithm results (i.e. the pixel label background or foreground), and the feedback of the classification and tracking tasks. The output of the tuner is the most suitable parameter value for each pixel in the image. This parameter value enables the background subtraction algorithm to be as much sensitive in detecting mobile objects as possible and more consistent with the feedback from classification and tracking.

As described in figure 2, the algorithm for parameter tuning consists of two main components: pixel label coherence checker and a group of n detectors. These detectors are the same background subtraction algorithm with different parameter values. The tuning algorithm is composed of 3 steps:

1. The tuner verifies the consistence of the current pixel label provided by the current background subtraction algorithm (called current label) with the feedback information from classification. If the current label is foreground (FG) and the classification task indicates that this pixel value is noise, the pixel value is labeled as background for the next step. If the classification task indicates that this pixel belongs to a detected object, this pixel value is discarded and not used for next step. Therefore, the next step contains only pixel values with label background (BG).

2. The tuner feeds n detectors with the current pixel value and builds a statistical information on the number of time the detector classifies pixel values as BG. This information is called Statistic for Local Evaluation (SLE).

3. The tuner chooses the parameter value of the best detector as the tuned parameter value depending on an evaluation function using SLE and the parameter value.

The evaluation function is:

$$ES = (SLE) + \alpha f(p) \quad (3)$$

Where α is the importance of the parameter value, $f(p)$ is the function in the range $[0, 1]$ quantifying the user's preference of parameter value given the parameter value p. For example in case of Gaussian Mixture Model, we prefer low values of T because they make the algorithm more sensitive in detecting mobile objects. Therefore, $f(T)$ can be defined as $f(T) = 1 - T$ with $T \in [0, 1]$.

SLE is defined as:

$$SLE = \frac{nCoherence}{|Window|} \quad (4)$$

where $nCoherence$ is the number of times the detector classifies the pixel values as background and $|Window|$ is the size of temporal window we use to construct SLE.

To make the tuning algorithm suitable for the real time requirement, instead of tuning parameters for every pixel in the image, the tuner only work on m pixels distributed as a grid over the image. After finding the suitable parameter value for these pixels, the tuner generalizes the tuned parameter values to the other pixels in the image. Particularly, to find a suitable parameter value for pixel A, the tuner firsts find the 4 closest pixels (using Euclid distance) on the grid. Then among these 4 pixels, the tuner finds the pixel (called pixel B) whose the background representation is closest to the background representation of pixel A. If the distance between the two background representation at pixel A and B is small enough, the tuner assigns the tuned parameter value of pixel B to the same parameter of pixel A. The background representation distance depends on the underlying background subtraction algorithm. For example, in case of GMM, the distance $D(BR_1, BR_2)$ between two background representations BR_1, BR_2 can be computed as follows:

$$D(BR_1, BR_2) = \sum_{i=0}^{n} |w_i^1 - w_i^2| \quad (5)$$

where n is the number of Gaussian components corresponding to background in BR_1 and w_i^j is the normalized weight of components i in BR_j. According to this formula, two background representations are said to be close if they have a similar form (similar number of background components, similar ratios between different component weight).

6 Experimental results

To assess the performance of the controller, we perform two experiments. The first experiment validates the effectiveness of the controller in guiding the background update. In this experiment, the background subtraction algorithm is GMM [12] and the testing video is one of the videos from homecare project [2]. Figure 1 shows a sample image from this video. This video shows a person living in an apartment. The video is more than 1h long (40800 frames). The ground truth has been constructed by annotating a frame every other 200 frames and by drawing bounding boxes around the person inside that frame. The metric M.1.2.1. of ETISEO project [1] is used to measure the algorithm performance. This metric counts the number of physical objects in the ground truth that are detected by the background subtraction algorithm. To match the detected objects with the ground truth objects, the metric uses the bounding box. The Precision (P), Sensitivity (S), and F-Score are computed based on True Positive (TP), False Positive (FP), and False Negative (FN) as shown below:

$$P = \frac{TP}{TP + FP}, S = \frac{TP}{TP + FN}, F - Score = \frac{2 \times P \times S}{P + S} \quad (6)$$

Detection results	Performance indexes		
	P(%)	S(%)	F-Score(%)
GMM_1	77	13	23
GMM_2	56	75	64
GMM_3	72	78	75

Table 1. Detection results of the GMM algorithm alone (GMM_1), of the GMM algorithm with an updating controller without ghost removal (GMM_2), and of the GMM algorithm with an updating controller with ghost removal (GMM_3). P: Precision, S: Sensitivity, F-Score: balance between P and S.

Table 1 shows the detection results of the GMM algorithm alone, the GMM algorithm with an updating controller but without ghost removal, and the GMM algorithm with an updating controller including ghost removal. In this sequence, the person often stays at the same place for a long time. Therefore, the GMM algorithm alone updates the regions corresponding to the person and after several frames, it cannot detect the person any more (sensitivity = 13%). On the other hand, with the help of updating controller not including ghost removal, the GMM algorithm can increase its sensitivity (from 13% to 75%) but at the expense of reducing precision (from 77% down to 56%) because there are many ghosts in the detection results. With the updating controller including a ghost removal, the GMM algorithm increases the sensitivity (from 13% to 78%) without reducing much the precision. However the sensitivity is still low because either the GMM cannot detect mobile objects or the classification task cannot classify detected foreground regions as person.

The second experiment verifies the performance of the parameter tuner for the parameter T of GMM with the video of a waving tree in [8]. As shown in figure 6, the parameter tuner can successfully set up a suitable T value for each pixel depending on the distribution of this pixel: the background region corresponding to static background (building, sky) is assigned a low T value and the background region corresponding to the background containing motions (tree) is assigned a high T value. As a result, the GMM with adaptive T value reduces the noise inside the tree region.

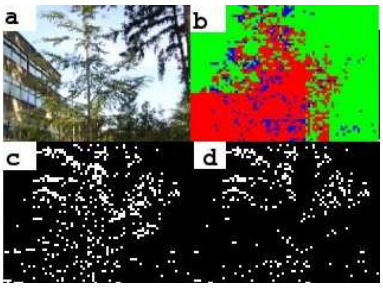

Figure 6. (a): Sample image from the video of a waving tree in [8], (b) the image illustrating the value of T for each pixel computed by the parameter tuner. T value is represented by different colors (green: Low, blue: Medium, red: High). The tuning algorithm has correctly assigned high T value to the region of the tree which has multimodal distribution. (c) Detection result of the GMM with constant T value (Medium for every pixel). (d) Detection result of the GMM with adaptive T value. The GMM with adaptive T value has reduced the noise inside the tree regions

7 Conclusion

In this paper, we present a controller that helps background subtraction algorithms to update their background representation and to initialize the parameter values to adapt to the current conditions of the scene. To guide the background subtraction algorithms to update the background representation, we propose methods to solve 2 important problems: detecting ghosts and managing stationary objects. The algorithm to detect ghosts relies on object borders (i.e. gradients around the foreground region) but it does not need to detect precise object borders. Moreover, the proposed algorithm is efficient because it does not examine the whole object borders. The algorithm to manage stationary objects cooperates with the tracking task to faster detect stationary objects and to avoid the maintenance of various background layers. To tune the parameter values of background subtraction algorithms, we select the parameter value which enables background subtraction algorithms to be as much sensitive in detecting mobile objects as possible and at the same time more consistent with feedback from classification and tracking tasks. However the effectiveness of the tuning algorithm relies heavily on the quality of the feedback from the classification and tracking tasks. When these tasks are wrong, the tuning algorithm cannot find a good parameter value for background subtraction algorithms. Therefore the proposed algorithm can serve as a complement for the offline tuning algorithms as in [11, 3] when we do not have training video whose conditions are similar to the current conditions of the scene. The experiments have shown that the controller has improved the performance of the background subtraction algorithms.

In the future, we will extend the ghost detection algorithm by taking into account the relative distance between object border points. For the parameter tuner, we will study the possibility of combining current method with methods of offline approach to learn local pixel distribution and to have a more optimal parameter value.

Acknowledge: This work is support by Project SIC - Securisation des Infrastructures Critiques, France.

References

[1] Project ETISEO-http://www-sop.inria.fr/orion/ETISEO.

[2] Project Gerhome - http://gerhome.cstb.fr.

[3] M. Thonnat B. Georis, F. Bremond. Real-time control of video surveillance systems with program supervision techniques. 2007.

[4] J. H. Connell, A. W. Senior, A. Hampapur, Y.L. Tian, L. M. G. Brown, and S. Pankanti. Detection and tracking in the IBM PeopleVision system. In *International Conference on Multimedia and Expo*, 2004.

[5] H. Fujiyoshi and T. Kanade. Layered Detection for Multiple Overlapping Objects. In *International Conference on Pattern Recognition*, 2002.

[6] D. Hall. Automatic parameter regulation of perceptual systems. In *Image and Vision Computing*, 2006.

[7] A. Harville. A Framework for High-Level Feedback to Adaptive, Per-Pixel, Mixture-of-Gaussian Background Models. 2002.

[8] B. Brumitt K. Toyama, J. Krumm and B. Meyers. Wallflower: Principles and Practice of Background Maintenance. In *IEEE International Conference on Computer Vision*, 1999.

[9] Kyungnam Kim, T.H. Chalidabhongse, D. Harwood, and L. Davis. Background modeling and subtraction by codebook construction. In *International Conference on Image Processing*, 2004.

[10] S. Lu, J. Zhang, and D. Feng. An efficient method for detecting ghost and left objects in surveillance video. In *IEEE International Conference on Advanced Video and Signal based Surveillance*, 2007.

[11] V. Martin and M. Thonnat. Learning Contextual Variations for Video Segmentation. In *International Conference on Computer Vision System*, 2008.

[12] Chris Stauffer and W.E.L. Grimson. Adaptive Background Mixture Models for Real-Time Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.