Time Dependent On-line Boosting for Robust Backgroundmodeling*

Helmut Grabner, Christian Leistner, and Horst Bischof

Institute for Computer Graphics and Vision
Graz University of Technology
{hgrabner, leistner, bischof}@icg.tugraz.at

Abstract. In modern video surveillance systems change and outlier detection is of highest interest. Most of these systems are based on standard pixel-by-pixel background modeling approaches. In this paper, we propose a novel robust block-based background model that is suitable for outlier detection using an extension to on-line boosting for feature selection. In order to be robust our system incorporates several novelties for previous proposed on-line boosting algorithms and classifier-based background modeling systems. We introduce time-dependency and control for on-line boosting. Our system allows for automatically adjusting its temporal behavior to the underlying scene by using a control system which regulates the model parameters. The benefits of our approach are illustrated on several experiments on challenging standard datasets.

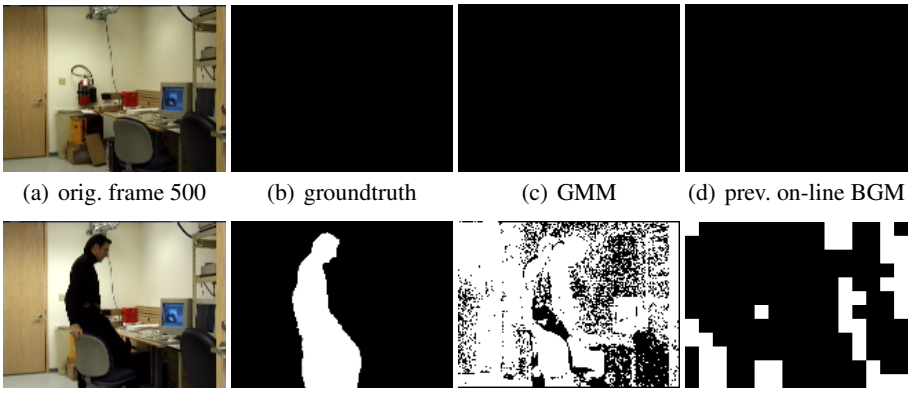
1 Introduction

For most video surveillance systems the detection of moving or intruding objects is of crucial importance. Frequently simple background subtraction methods are applied, as they are easy to implement and fast to compute. There objects are detected by blobs of pixels which do not correspond to the background model.

In its simplest form the background model (BGM) is solely one image, called the background image. Having this image, pixels are marked as foreground if they do not fit to it, *i.e.*, are more than a certain threshold above or below each pixel value. For realistic applications and in order to handle different environmental conditions (*e.g.*, changing lightening conditions or foreground models moving to background and vice versa), more sophisticated, multi-modal statistical models such as Mixture of Gaussians (GMM) [1] or Eigenbackgrounds [2] are often used. More efficient systems analyzing foreground models [3] have been proposed. In order to further improve the robustness some recent approaches exploit the spatial correlation between pixels arranged in blocks (*e.g.*, [4]), *e.g.*, by describing statistics within one block using features [5], the entire block is decided to be either background or foreground, which significantly improves the detection reliability.

Furthermore, several adaptive methods for estimating a pixel-based BGM have been proposed which update the existing image with respect to the current input frame (*e.g.*,

* This work has been supported by the Austrian Joint Research Project Cognitive Vision under projects S9103-N04 and S9104-N04 and the FFG project EVis under the FIT-IT program.



(a) orig. frame 500 (b) groundtruth (c) GMM (d) prev. on-line BGM

(e) orig. frame 1865 (f) groundtruth (g) GMM (h) prev. on-line BGM

Fig. 1. Comparison of BGM based on Gaussian Mixture Models (GMM) [1] and an on-line classifier grid [9] at a challenging Wallflower scene [10]. The first row depicts the results after 500 frames, where both adaptive approaches still perform well. From frame 829 to 1844 the lights are switched off. The second line depicts the result at frame 1865. As can be seen, both approaches fail because the GMM does not take neighboring pixels into account and the on-line classifier drifts away due to false updates during the lights-off phase.

running average [6], temporal median filter [7], approximate median filter [8]). For block-based BGMs adaptiveness can be achieved by describing each block with an on-line classifier [9]. This, additionally, allows to adapt to continuous changes (*e.g.*, illumination changes in outdoor scenes) while observing the scene.

Although being effective for describing highly dynamic scenes, yet, the main drawback (see also Section 2) of this on-line learner-based BGM is that its update strategy is similar to a self-learning method since the model updates are *directly* based on its own classifier predictions and can therefore end up in a catastrophic state (*i.e.*, the model drifts away). Due to this dependency on its own predictions, the model performs quite well for a relatively short period of time but finally tends to learn foreground objects very quickly without offering any control on its temporal behavior (see also Figure 1). Furthermore, the cumbersome update strategy is highly scene dependent and, therefore, has to be hand-tuned every now and then.

To sum up, previously proposed background models based on on-line classifiers show high performance potentials but are only tediously, if not impossible, to be applied in practice. In this paper we extend the classifier-based background model by using an alternative version of the on-line learning algorithm which is controlled via temporal aspects and is resistant to outliers. Furthermore, our proposed BGM is based on fixed and simple update rules which yield stable results over a long period of time. Finally, the period of time necessary for modeling regular and periodical scene behaviors does not have to be hand-tuned but autonomously adapts to the underlying problem.

The rest of the paper is organized as follows. In Section 2 we extend the on-line boosting approach to temporal aspects while in Section 3 we illustrate how these classifiers can be used for modeling backgrounds and present a general framework. In Sec-

tion 4 we (a) show illustrative examples and (b) evaluate the background model on public available sequences. Section 5 concludes the paper and gives some ideas for further research.

2 Time Dependent On-line Boosting

For off-line learning all labeled training samples have to be given in advance. Contrary, on-line algorithms see each sample only once. They are necessary when (a) the whole data does not fit into memory at once, (b) data becomes available only over time and (c) the data generation process changes over time. Especially, for the last point the algorithm has to be adaptive since the training data is not sampled always from one fixed distribution. In such applications the control of time (forgetting old, irrelevant information) is an important parameter.

Since in this work we use on-line boosting as learning algorithm , we start this section with a short review of on-line boosting for feature selection. We then proceed by extending it and introduce the time-dependent approach.

2.1 On-line Boosting for Feature Selection

In general, boosting (see [11] for a good introduction) is a widely used technique in machine learning for improving the accuracy of any given learning algorithm. In fact, boosting converts a weak learning algorithm into a strong one. Therefore, for an input sample $\mathbf{x}$ a strong classifier $h^{strong}(\mathbf{x})$ is computed as linear combination of a set of N weak classifiers $h_n^{weak}(\mathbf{x})$:

$$h^{strong}(\mathbf{x}) = \text{sign}(conf(\mathbf{x})) \quad \text{where} \quad conf(\mathbf{x}) = \frac{\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x})}{\sum_{n=1}^{N} \alpha_n} \tag{1}$$

A weak classifier is a classifier which has to perform only slightly better than random guessing, *i.e.*, for a binary decision task, the error rate must be less than 50%. The weak classifiers are trained by adapting the weights (initizalies equally) of the training samples. As $conf(\mathbf{x})$ is bounded by $[-1, 1]$ it can be interpreted as a confidence measure. The higher the absolute value, the more confident is the result. Further, boosting can be used for feature selection (Tieu and Viola [12]) where each feature (*e.g.*, a Haar-Wavelet) corresponds to a single weak classifier h_n^{weak}.

Standard boosting as described above is an off-line learning procedure. In order to be adaptive, we apply an on-line version of boosting for feature selection similar to [13]. Here, the main idea is to introduce "selectors" and to perform boosting on these selectors and not directly on the weak classifiers. Each selector $h^{sel}(\mathbf{x})$ holds a set of M weak classifiers $\{h_1^{weak}(\mathbf{x}), \ldots, h_M^{weak}(\mathbf{x})\}$. Training/Updating a selector means, that each weak classifier in it gets updated and than it selects one of them according to an optimization criterion. In fact we select the weak classifier h_i^{weak} with the lowest estimated error

$$e_i = \frac{\lambda_i^{wrong}}{\lambda_i^{corr} + \lambda_i^{wrong}} \tag{2}$$

where λ_i^{corr} and λ_i^{wrong} are sums of importance weights λ for correct and incorrect seen samples so far. For updating the weak classifier two distributions of the feature responds $f(\mathbf{x})$ are estimated by two Gaussians where the parameter are determined via a Kalman filtering technique.

$$P(+1|f(\mathbf{x})) \approx \mathcal{N}(\mu^+, \sigma^+) \quad \text{and} \quad P(-1|f(\mathbf{x})) \approx \mathcal{N}(\mu^-, \sigma^-) \tag{3}$$

for the positive and negative samples, respectively. They are then used to build a simple hypothesis which corresponds to a weak classifier. Moreover, the importance/difficulty λ (initialized by 1) of a sample is estimated by propagating it through the set of selectors. There, it is used to update each weak classifier and for selecting one of those.

Concerning robust adaptiveness, this approach has several limitations: *First*, the sample weights contribute forever to the entire model statistics and are only unlearned (*i.e.*, getting less important) by updating the system with new ones. Hence, there is yet no control how fast information "fades" away and new one is gained. *Second*, samples with high errors get a much higher weight assigned than low-error samples. When sampling always from the same (static) distribution this is perfect, since boosting focuses on the difficult examples. Nevertheless, this makes the system extremely sensitive to label noise and, especially when dealing with an adaptive learning problem, this assumption makes no sense anymore.

2.2 Time Dependent On-line Boosting for Feature Selection

Our first change to on-line boosting considers the basic assumption that the examples are all drawn from a fixed distribution. As we aspire fading memory we propose to use exponential forgetting of the examples over time. Therefore, we define the following update rule for the estimation of the value $\hat{w}_t$ using the previous value $\hat{w}_{t-1}$ and a new measurement w

$$\hat{w}_t = K\hat{w}_{t-1} + (1-K)w \quad \text{where} \quad K = \sqrt[\Delta t]{0.5} \tag{4}$$

gives the factor of how much previous information should be kept. In particular, it determines that half the information is kept in the time interval Δt. Once defined, this rule can now be easily used for all dynamic elements of the system. Especially, we incorporate this update rule for estimating the probability density functions (replacing the Kalman filtering like updates of mean and variance in Equation 3) of the weak classifier as well as the estimated errors , *i.e.*, both λ^{corr} and λ^{wrong} in Equation 2.

The second change limits the effect of label noise. Each new incoming sample $\langle \mathbf{x}_t, y_t \rangle$ for the first selector is initialized with the importance $\lambda = 1$. This means if it is well predictable its importance decreases by propagating through all the selectors otherwise increases. Assuming label noise a single noisy example can get very high importance and can change the entire model statistics rapidly. Therefore, we propose to keep the importance of the samples at the end of the ensemble constant. We first propagate the example through the set of selectors and obtain the value of λ_n without doing an update. The actual update is done with the initial value set to $\lambda_1 = \frac{1}{\lambda_n}$ which clearly results in keeping it at one at the last selector. This simple modification ensures trusting the model more than the examples which means that the system, inherently, has

some kind of outlier detection implemented. Please note, that this modification does not change the overall boosting process, it rather gives the example a prior importance.

Finally, we introduce a soft-selector, in order to limit the hard switching effect within the selectors. In contrast to taking the best weak classifier (*i.e.*, that with the smallest error) for each selector it uses the information of all weak classifiers combined (which we have anyway). Although every arbitrary classifier fusion rule might be applied, we chose to use the simple sum-rule which in practice yields good results [14]. As a result, all weak classifiers, the errors and therefore the importance weight λ as well as the voting α_i are changing continuously.

3 Background modeling

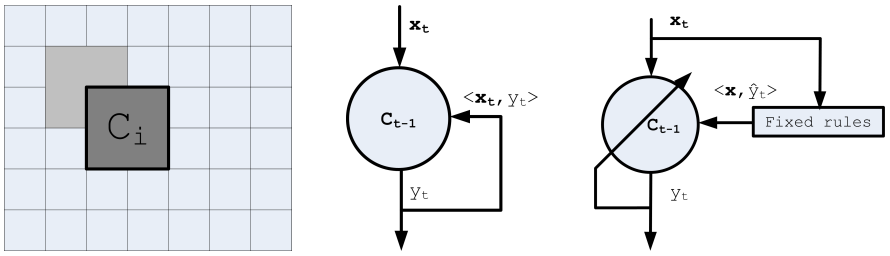
First, we start with a review of the block-based classifier background model. Second, we show how to extend this approach by introducing a fixed and simple update strategy in combination with the extended time-dependent on-line boosting approach.

3.1 Classifier-Based Backgroundmodel

In order to use classifiers for background modeling [9], we partition the image into a grid of small, highly overlapping rectangular blocks (patches). For each of them a separate classifier is computed by combining simple image features (*e.g.*, Haar features) which are selected using on-line boosting for feature selection. This is depicted in Figure 2(a). The main idea is to learn if the underlying image patch is predictable using the classifier. If so, this is considered as "allowed" background and otherwise it is considered as unknow (therefore foreground). Since boosting is used to train the classifier, which is a discriminative learning method, normally both positive and negative labeled samples are required in order to learn a decision boundary. Contrary, the task of background modeling is a one class classification problem, meaning only positive samples (the observed images) are given. Therefore, for each feature the negative distribution is estimated directly without learning (*e.g.*, assuming each pixel is a random variable which is normal distributed or by using statistics of natural images [15]). From this distribution simple learning rules are used to get a hypothesis for the weak classifier (for more details see [9]).

The system has two phases: First, an initial learning stage where a separate classifier is built for all image patches assuming that all input images are positive examples (*i.e.*, correspond to allowed background variations). Later on, in order to be adaptive to the scene, new input images are analyzed and the background model is updated according to a given, yet not totally traceable, policy. Then, it ends up with three different thresholds, which have to be hand-tuned as well as some higher update policy, for instance, that neighboring patches are inhibited to be updated for a certain time (yet another irreproducible variable) when the current patch is considered as foreground. Further on, due to its analogy to self-learning which relies on a *direct* feedback of its own predictions, the approach tends to drift and ends up in not predictable states when running for a long time (*e.g.*, 24 hours a day, 7 days a week).

3.2 Robust Classifier-Based Background Model



(a) classifier grid (b) self-learning (c) our approach: fixed-learning with adaptive parameters

Fig. 2. (a) The background model is formed by a highly overlapping grid of classifiers. Updating the former approach [9] uses self-learning (b). Our proposed method (c) does not directly take the classifier responds $y_t = C_{t-1}(\mathbf{x}_t)$ into account and thus does not suffer from the drifting problem.

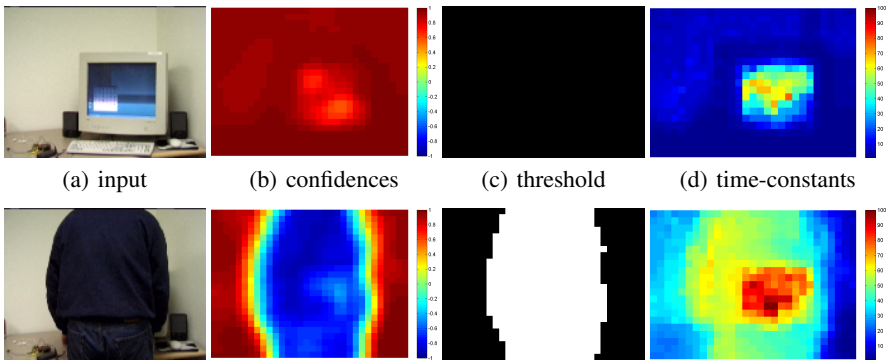
In order to get rid of the self-learning update strategy (Figure 2(b)), we propose a fixed yet simple update strategy. Each classifier C_i with the correspondent patch $\mathbf{x}_{i,t}$ incorporates every new upcoming frame t as a positive example $\langle \mathbf{x}_{i,t}, +1 \rangle$.

For automatically updating[1] the parameter Δt we choose a simple dynamic control system, taking the following simple observation into account: The time constant should be large enough in order to model dynamic behavior but still as small as possible in order to be highly sensitive to small background changes. If observing a static scene then every movement or change should be considered as foreground. On the other hand, observing a dynamic behavior, *e.g.*, leafs in the wind, this should be modeled, and therefore we have to increase the time constant up to a limit where it is still possible to model these dynamic backgrounds. Furthermore, we assume that the time constant should move smoothly. For each individual on-line classifier having its own Δt we use the following estimator

$$\Delta t_t = K_i \Delta t_{t-1} + K_p \hat{\Delta} t, \qquad \hat{\Delta} t = 1 - 0.5\,(1 + conf(\mathbf{x}))\,, \tag{5}$$

where $K_i \in [0 \quad 1]$ assumes the smoothness constraint and $K_p > 0$ is multiplied by the current estimate $\hat{\Delta} t$, which is considered to be proportional to the confidence of the patch $\mathbf{x}$. As soon as the confidence changes dramatically, *i.e.*, a totally unknown intruder enters the scene, our control system tremendously increases Δt, which yields the implicit result that the harder the underlying scenario changes, the higher the controller sets Δt and, thus, the longer it takes for a new object to "fade" into the background. Yet, smooth changes result in only small changes of Δt which let the system adapt to small

[1] Please note, that of course one can also specify the time constant by hand in order to get a predefined background model for a specific application.



(a) input (b) confidences (c) threshold (d) time-constants

Fig. 3. The first row depicts the test scene after $t = 100$ and the second row at $t = 251$, respectively. The second column shows the yielded confidences. As we assume the flickering monitor to be background, the confidences are quite high. In order to achieve such results our control system autonomously sets the Δt in the flickering area higher than in the non-dynamic rest of the scene.

background changes, *e.g.*, slightly changing lighting conditions. This allows us to autonomously model different dynamic movements and periods for each classifier patch without drifting into unpredictable states since only Δt and λ_1 as model parameters are changed but not the model itself (see Figure 2(c)).

The confidence of the classifier corresponds to the likelihood that the example correspond to the background. In fact, we robustly detect outliers and mark them as unknown foreground objects. Note, that all these would not be possible without the changes we proposed in the previous section.

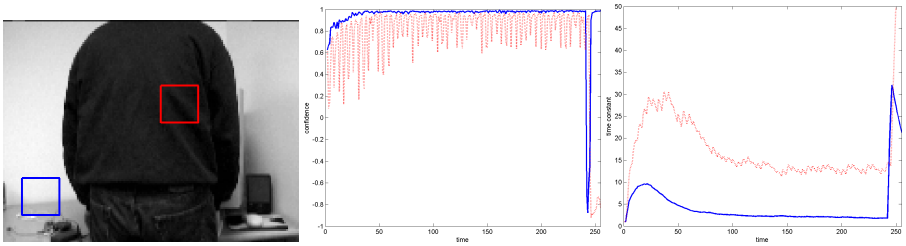
4 Experiments

In the following we demonstrate the benefits of our approach compared to existing methods. Therefore, we split the experiments into two main parts. First, we give a detailed evaluation on a sample sequence. Second, to show that we obtain state-of-the-art results the proposed method is applied on public available benchmark data sets.

For all of our experiments we use a classifier grid with a patch-size of 20×20 with an overlap of 75%. To compute the classifier we use only 15 selectors each using a set of 30 weak classifiers. Haar-like features are used because they can be evaluated very fast using the integral data structure. The thus obtained grid of detectors is evaluated and updated whenever a new frame arises. In order to set the time constant Δt for each classifier we set $K_i = 0.95$ and $K_p = 10$.

4.1 Detail Experiments

Figure 3 shows the behavior of our proposed background model. The rows correspond to two different times and the columns show the input image and corresponding internal results, which are the archived confidence map as well as the segmented foreground object, which is a simple threshold on the confidence map, and, in addition, the estimated



(a) image (b) confidences (c) time constant

Fig. 4. Confidences (b) and estimated time-constants (c) for two different patches (a) over time. As can be seen, the blue patch covers a non-dynamic scene, reliably detects the intruder and immediately recovers its state. The second patch (red) has to handle dynamic flickering background. Hence, the time-constant also stabilizes around a higher value than for the blue patch. However, when the scene changed dramatically the time-constant raised very high making it very difficult for the intruded object to fade into the background.

time constant for each grid element. The sequence is taken from [10] which shows a sequence of a flickering screen. At the end a person enters and fully occludes the monitor. In the first row each patch is able to model the background quite well using the on-line learned classifier (high confidence). In the region where the screen is located, the time-constant is automatically set to a higher value. This allows the patches located around the screen to model the flickering while still being able to detect the intruding object.

In addition, Figure 4 depicts some more detail results of two specific patches. The color of the plots corresponds to the color of the rectangles. In particular, the red patch is located on the screen which is flicker during the scene. Therefore its time constant is automatically set to a higher value so that the classifier is able to describe it. Note, at the end of the sequence there is a correct break down by the confidence value, because a person enters the scene.

4.2 Wallflower Test Sequences

In [10], a test set for evaluating background subtraction methods was presented. It consists of seven video sequences, each addressing a specific canonical background subtraction problem. In the same paper, 10 different methods were compared using the test set. We tested our method against this test set and achieved the results shown in Fig. 5. Comparing the results in their paper as well as from the recently proposed grid based classifier [5] we achieve comparable performance. In contrast to the method using locally binary patterns [5] we are able to handle the the Light Switch problem since we use features which are illumination invariant and is achieved without use of any higher level processing that could be used to detect sudden changes in background.

In addition to the segmentation, which is achieved by thresholding the classifier output by zero we depict as well the confidence map. This map profiles include more information which can be included in further analysis in the image processing pipeline (*e.g.*, more sophisticated methods such as a mean shift-based clustering can be applied).

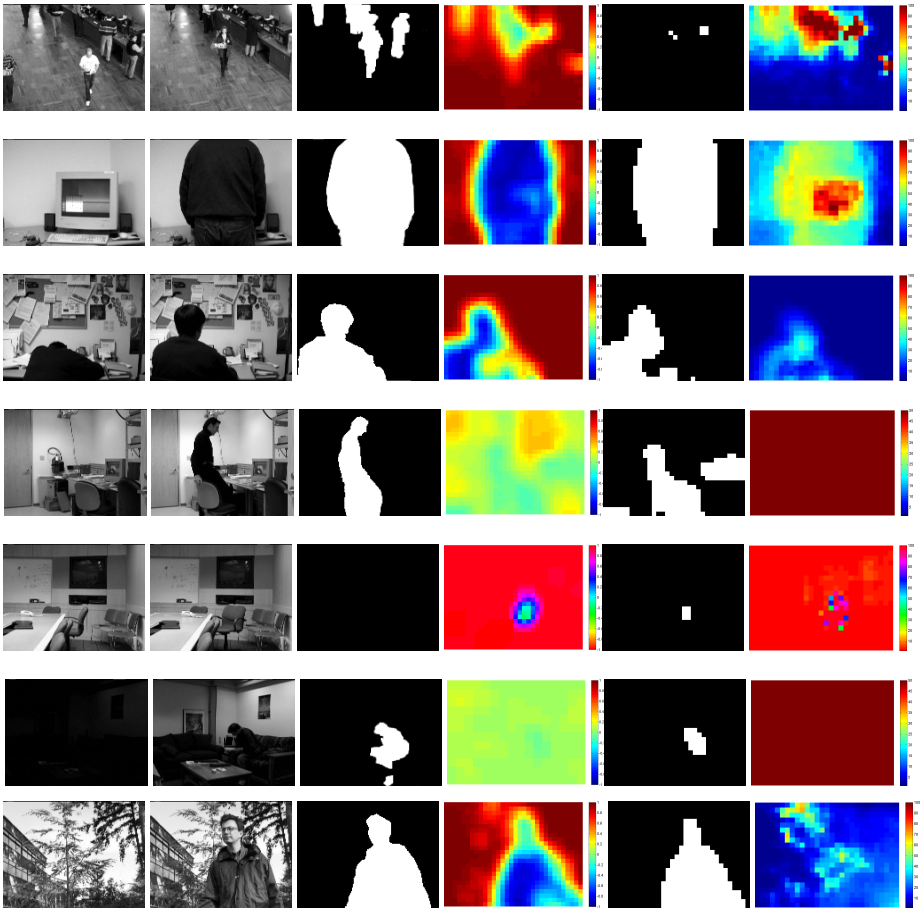


Fig. 5. Detection results of our method for the test sequences presented in [10]. The first column shows the initial frame of each sequence, second column the test frame and the third column the hand segmented ground truth. In the last three columns our results (the real valued confidence map, a binary segmentation achieved be zero thresholding as well as the time constants) are depicted. Note that the performance could be easily increased by analyzing the confidences more closely.

5 Conclusions

In this paper we introduced controllable time dependency into on-line boosting. This was achieved by incorporating several inter-dependent changes into the on-line algorithm. First, our system uses a fading memory strategy in order to forget old information and acquire new one. Second, each new example is incorporated into the model with equal importance. Additionally, soft-switching selectors further allow for keeping the model statistics more robust. This results in both a higher noise invariance and the ability to smoothly adapt to new problems. Together with a fixed yet simple update rule, which would not be feasible without controllable fading memory, this yields a very powerful classifier highly suitable for the task of background modeling of dynamic scenes. The main advantage is that the model cannot drift and does neither have to be hand-tuned for each new scene. This is achieved by incorporating a simple automatic control system which seeks to adjust the model time-period.

We are confident that the changes in the on-line boosting method are also highly suitable for other on-line learning applications, such as tracking and improving object detection.

References

1. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: Proc. CVPR. Volume II. (1999) 246–252
2. Oliver, N.M., Rosario, B., Pentland, A.: A bayesian computer vision system for modeling human interactions. PAMI **22** (2000) 831–843
3. Tian, Y.L., Lu, M., Hampapur, A.: Robust and efficient foreground analysis for real-time video surveillance. In: Proc. CVPR. Volume 1. (2005) 1182 – 1187
4. Russell, D., Gong, S.: A highly efficient block-based dynamic background model. In: Proc. Conf. on Advanced Video and Signal Based Surveillance. (2005) 417–422
5. Heikkilä, M., Pietikäinen, M.: A texture-based method for modeling the background and detecting moving objects. PAMI **28** (2006) 657–662
6. Koller, D., Weber, J., Huang, T., Malik, J., Ogasawara, G., B.Rao, Russell, S.: Towards robust automatic traffic scene analysis in real-time. In: Proc. ICPR. Volume I. (1994) 126–131
7. Lo, B., Velastin, S.A.: Automatic congestion detection system for underground platforms. In: Proc. IEEE Intern. Symposium on Intelligent Multimedia , Video and Speech Processing. (2001) 158–161
8. McFarlane, N.J., Schofield, C.P.: Segmentation and tracking of piglets. Machine Vision and Applications **8** (1995) 187–193
9. Grabner, H., Roth, P., Grabner, M., Bischof, H.: Autonomous learning a robust background model for change detection. In: Proc. PETS. (2006) 39–46
10. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: Principles and practice of background maintenance. In: Proc. ICCV. (1999) 255–261
11. Freund, Y., Schapire, R.: A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence **14** (1999) 771–780
12. Tieu, K., Viola, P.: Boosting image retrieval. In: Proc. CVPR. (2000) 228–235
13. Grabner, H., Bischof, H.: On-line boosting and vision. In: Proc. CVPR. Volume 1. (2006) 260–267
14. Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. PAMI **20** (1998) 226–239
15. Huang, J., Mumford., D.: Statistics of natural images and models. In: Proc. CVPR. Volume 1. (1999)