Comparison of Target Detection Algorithms using Adaptive Background Models

D.Hall[1], J. Nascimento[2], P. Ribeiro[2], E. Andrade[3], P. Moreno[2], S. Pesnel[1], T. List[3], R. Emonet[1], R.B. Fisher[3], J. Santos Victor[2] and J.L. Crowley[1]

INRIA Rhône-Alpes, France[1] IST Lisbon, Portugal[2] University of Edinburgh, UK[3]

Abstract

This article compares the performance of target detectors based on adaptive background differencing on public benchmark data. Five state of the art methods are described. The performance is evaluated using state of the art measures with respect to ground truth. The original points are the comparison to hand labelled ground truth and the evaluation on a large database. The simpler methods LOTS and SGM are more appropriate to the particular task as MGM using a more complex background model.

1. Introduction

The video surveillance domain has a great demand for real time image processing systems that operate reliably 24 hours a day, 7 days a week. Ef cient and reliable algorithms are required for this kind of system. Adaptive background differencing techniques have become a widely used solution, since they can incorporate illumination changes as they occur in outdoor scenes during the day. In this article, we compare ve state of the art adaptive background differencing techniques. The detectors are evaluated on the same public benchmark dataset which allows a fairer and more extensive comparison. Furthermore we show an architecture where such an detector can be combined with a Kalman lter . The performance increase is demonstrated on an example.

This article gives insight into performance, computation time and usability of these methods. The same techniques were previously tested on the PETS 2001 dataset [9]. Ground truth for the PETS 2001 data set was generated by a semi-automatic technique, where the tracking results were monitored and corrected by a human. The data set used here is much larger and we compare the results with respect to independently hand labelled ground truth. This shows in addition to the previous comparison the performance compared to a human.

The rst method is a basic background subtraction algorithm *(BBS)*. This is the simplest algorithm and it provides a lower benchmark for the other algorithms which are more complex but based on the same principle.

The second algorithm is denoted as *W4* and operates on gray scale images. Three parameters are learned for each pixel to model the background: minimum intensity, maximum intensity and maximum absolute difference in consecutive frames. This algorithm incorporates the noise variations into the background model.

The third method is used in *P nder* [13] denoted here as *SGM* (Single Gaussian Model). This method assumes that each pixel is a realization of a random variable with a Gaussian distribution. The rst and second order statistics of this distribution are independently estimated for each pixel.

The fourth method is an adaptive mixture of multiple Gaussians *(MGM)* as proposed by Stauffer and Grimson in [12]. Every pixel of the background is modeled using a mixture of Gaussians. The weights of the mixture and the parameters of the Gaussians are adapted with respect to the current frames. This method has the advantage that multimodal backgrounds (such as moving trees) can be modeled. Among the tested techniques, this is the one with the most complex background model.

The fth approach *(LOTS)* proposed by Boult in [2] is an ef cient method designed for military applications that presumes a two background model. In addition, the approach uses high and low per-pixel thresholds. The method adapts the background by incorporating the current image with a small weight. At the end of each cycle, pixels are classi ed as false detection, missed detection and correct detection. The original point of this algorithm is that the per-pixel thresholds are updated as a function of the classi cation.

The article is organized as follows. Sections 2 to 7 describe the technical details of the approaches. Section 8 describes the database, the evaluation metrics and the experimental results. We nish with conclusion and an outlook.

2. Basic Background Subtraction

This method detects targets by computing the difference between the current frame and a background image for each color channel RGB. A thresholding operation is performed to classify each pixel as foreground if

$$|I^t(\phi) - B^t(\phi)| > n_c, \quad (1)$$

where $I^t(\phi)$ is a 3-dimensional vector representing the intensity values of the three color channels at image position

0-7803-9424-0/05/$20.00 ©2005 IEEE.

ϕ. $B^t(\phi)$ is the mean color (background) of the pixel, n_c is the noise threshold. The operation in (1) is performed for all image pixels ϕ.

Segmentation of objects from the background can be achieved by connected component analysis (e.g., using 8 - connectivity criterion). This step is performed after morphological ltering with a 3x3 mask (dilation and erosion) that eliminates isolated pixels. The same connected component analysis is performed in the methods *W4* and *SGM* (Section 3 and 4).

To take into account slow illumination changes which is necessary to ensure longterm tracking, the background image is subsequently updated by

$$B^{t+1}(\phi) = \alpha I^t(\phi) + (1-\alpha)B^t(\phi) \quad (2)$$

with the learning rate α. In the experiments we use $\alpha = 0.15$ and noise threshold $n_c = 0.2$ since there were found to be good values in a previous experiment [10]. These parameters stay constant during the experiment.

3. W4 method

This algorithm was proposed by Haritaoglu in [5]. The background scene is modelled by representing each pixel by three values; minimum intensity (Min), maximum intensity (Max), and the maximum intensity difference (D) between consecutive frames during the training period. These values are estimated over several frames and are periodically updated for background regions. In the experiments, 100 target free images are selected for parameter learning. No parameters need to be set by hand.

Foreground objects are computed in four steps: i) thresholding, ii) region based noise cleaning, iii) morphological ltering and iv) object detection. Each pixel is classi ed as background or foreground using following equation. Giving the values of Min, Max and D, a pixel $I(\phi)$ is considered as foreground pixel if

$$|\text{Min}(\phi) - I(\phi)| > D(\phi) \text{ or } |\text{Max}(\phi) - I(\phi)| > D(\phi) \quad (3)$$

The resulting thresholded image usually contains a signi - cant amount of noise. A region based noise cleaning algorithm is applied that is composed of an erosion operation followed by a connected component analysis that allows to remove regions with less than 50 pixels. The result is a set of bounding boxes that contain the targets. The morphological operations dilation and erosion are now applied to the foreground pixels that are inside the bounding boxes. The nal bounding boxes are computed and returned as targets.

4. Single Gaussian Model

In this section we describe the Single Gaussian Model algorithm *(SGM)* proposed by Wren in [13]. In this method, the intensity and color of each pixel is represented by a vector $[Y, U, V]^T$. We assume only slow scene changes. The mean $\vec{\mu}(\phi)$ and covariance $U(\phi)$ of each pixel ϕ can be recursively updated as follows

$$\vec{\mu}^t(\phi) = (1-\alpha)\vec{\mu}^{t-1}(\phi) + \alpha I^t(\phi), \quad (4)$$

$$U^t(\phi) = (1-\alpha)U^{t-1}(\phi) + \alpha\vec{\nu}(\phi)\vec{\nu}(\phi)^T \quad (5)$$

where $I^t(\phi)$ is the pixel of the current frame in YUV color space, α is the learning rate and $\vec{\nu}(\phi) = I^t(\phi) - \vec{\mu}^t(\phi)$.

After background adaptation, we compute for all image positions ϕ the log likelihood $l(\phi)$ of the difference $\vec{\nu}(\phi)$ between current image and background. This value gives rise to a classi cation of individual pixels as background or foreground

$$l(\phi) = -\frac{1}{2}\vec{\nu}(\phi)^T(U^t)^{-1}\vec{\nu}(\phi) - \frac{1}{2}\ln|U^t| - \frac{3}{2}\ln(2\pi) \quad (6)$$

A pixel ϕ is classi ed as foreground if $l(\phi) < n_c$ else it is background. Then SGM detects targets by computing connected components from the foreground pixels. In the experiments we use $\alpha = 0.005$ and $n_c = -300$. After a series of tests, these parameters produced the best results.

5. Multiple Gaussian Model

In recent years time-adaptive per pixel mixtures of Gaussians background models have been a popular choice for modelling complex and time varying backgrounds [6]. In this work we have implemented the original version of the adaptive mixture of multiple Gaussians background model *(MGM)* for motion tracking described in [12].

5.1. Algorithm

In this algorithm the pixel process is considered a time series of vectors for colour images. The history of a particular pixel ϕ is given by:

$$\{X_1, ...X_t\} = \{I(\phi, i) : 1 \le i \le t\} \quad (7)$$

where I is the image sequence. The algorithm models the recent history of each pixel as a mixture of K Gaussian distributions. Thus the probability of observing the current pixel value is:

$$P(X_t) = \Sigma_{i=1}^{K}\omega_{i,t} * \eta(X_t, \vec{\mu}_{i,t}, U_{i,t}) \quad (8)$$

where K is the number of distributions, $w_{i,t}$ is the weight estimate of the ith Gaussian in the mixture at time t, $\vec{\mu}_{i,t}$ and $U_{i,t}$ are the mean value and covariance matrix of the ith Gaussian at time t, and η is the Gaussian probability density function.

$$\eta(X_t, \vec{\mu}, U) = \frac{1}{(2\pi)^{\frac{n}{2}}|U|^{\frac{1}{2}}}e^{-\frac{1}{2}(X_t-\vec{\mu})^T U^{-1}(X_t-\vec{\mu})} \quad (9)$$

For computational simplicity the covariance matrix is assumed to be of the form $U_{k,t} = \sigma_k^2 \mathbf{I}$ avoiding a costly matrix inversion at the expense of some accuracy. The algorithm assumes that red, green and blue channels are independent and that the pixel process is non-stationary. These assumptions result in an on-line k-means approximation algorithm for the mixture model. In the on-line approximation, every new pixel X_t is checked against the K existing Gaussian distribution. A match is found if the pixel value is within $L = 2.5$ standard deviation of a distribution. This is effectively a per pixel per distribution threshold and can be used to model regions witrh periodically changing lighting conditions. If the current pixel value matches none of the distributions the least probable distribution is updated with the current pixel values, a high variance and low prior weight. The prior weights of the K distributions are updated at time t according to:

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \tag{10}$$

where α is the learning rate and $M_{k,t}$ is 1 for the model which matched the pixel and 0 for the remaining models. After this approximation the weights are renormalised. The changing rate in the model is de ned by $1/\alpha$. The parameters $\vec{\mu}$ and σ for the unmatched distributions remain the same. The parameters for the matching distribution are updated as follows:

$$\vec{\mu}_t = (1 - \rho)\vec{\mu}_{t-1} + \rho X_t \tag{11}$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(X_t - \vec{\mu})^T(X_t - \vec{\mu}) \tag{12}$$

$$\rho = \alpha\eta(X_t|\vec{\mu}_k, \sigma_k) \tag{13}$$

For change detection a heuristic searches for the learnt distributions which have more supporting evidence. The Gaussians are ordered based on the ratio of ω/σ. This increases as the Gaussian's weight increases and its variance decreases. The rst B distributions accounting for a proportion T of the observed data are de ned as background.

$$B = \arg\min_b \left(\Sigma_{k=1}^b \omega_k > T\right) \tag{14}$$

In the current implementation the original algorithm [12] is modi ed to update the background model only for pixels detected as background in the previous frame. This decreases the absorption rate of stationary objects (dropped bags, immobile people) into the background model. Several sequences contain such immobile targets that should be detected by the algorithm. This modi cation increases therefore the detection performance. After foreground detection the pixels are morphologically ltered (noise reduction) and labelled. Connected components smaller than T_c pixels are discarded.

5.2. Parameterization for the experiments

The model is initially trained with 13 training sequences. No modi cation in the update rules are used for training. The algorithm parameters are set to $K = 5$ Gaussians, learning rate $\alpha = 0.002$ and $L = 2.5$ standard deviations to look for matching Gaussians. The detection performance is tested using the 14 test sequences.

For testing the initial 20 frames of each sequence receive a full adaptation, after that, only pixels classi ed as background have their distributions updated. For detection, the additional algorithm parameters are set to $T = 0.97$, corresponding to the percentage of pixels observed accounted for by the most stable distributions (background). This results in multi-modal distribution for the background. $L = 4.5$ allows larger deviations from the original background model. $T_c = 25$ pixels to remove small (noisy) connected components. After several trials with different parameters, these settings produced the best results.

6. LOTS

The target detector proposed in [2] operates on gray scale images. It uses two background images and two per-pixel thresholds. The two backgrounds model periodic changes such as moving trees. The per-pixel threshold image can treat each pixel differently, allowing the detector to be robust to localized noise in low-size image regions. The per-pixel threshold evolves according to a pixel label provided by a Quasi Connected Components analysis (QCC). This is a light version of the traditional connected component analysis that is also used to provide the target's bounding boxes.

6.1. Algorithm

The steps of the algorithm can be summarized as:

1. **Background and threshold initialization**. Set the background models B_1, B_2, and the threshold values T_L (low threshold), T_H (low threshold). The values in B_1 and B_2 are the lower and higher "non-target" pixel values in the scene, considering some temporal window. The per-pixel threshold T_L, is then initialized to the difference between the two backgrounds:

$$T_L(\phi) = |B_1(\phi) - B_2(\phi)| + \mathcal{U}(\phi) \tag{15}$$

where $\mathcal{U}$ represents noise with an uniform distribution in $[1, 10]$, and ϕ an image pixel. A higher threshold T_H is computed by:

$$T_H(\phi) = T_L(\phi) + \mathcal{V} \tag{16}$$

where $\mathcal{V}$ is the sensitivity of the algorithm.

2. **Detection and Labeling** First we create D, which contains the minimum of the differences between the new image I, and the backgrounds B_1 and B_2:

$$D(\phi) = \min_j |I(\phi) - B_j(\phi)|, j = 1, 2. \quad (17)$$

D is then compared with the threshold images. Two binary images D_L and D_H are created. The active pixels of D_L and D_H are the pixels of D that are higher than the thresholds T_L and T_H respectively.

QCC computes for each thresholded image D_L and D_H images Ds_L and Ds_H with 16 times smaller resolution. Each element in these reduced images, Ds_L and Ds_H, has the number of active pixels in a 4x4 block of D_L and D_H respectively. Both images are then merged into a single image that labels pixels as *detected*, *missed* or *insertions*. This process labels every pixel, and also deals with targets that are not completely connected, considering them as only one region.

A 4-neighbor connected components is then applied to this image and, the regions with less than $\mathcal{A}$ pixels are eliminated. The remaining regions are considered as detected. The *detected* pixels are the ones from D_L that correspond to detected regions. The *insertions* are the active pixels in D_L, but do not correspond to detected regions, and the *missed* pixels are the inactive pixels in D_L.

3. **Backgrounds and threshold adaptation**. The backgrounds are updated as follows:

$$B_i^{t+1}(\phi) = \begin{cases} (1-\alpha')B_i^t(\phi) + \alpha' I^t(\phi), & \phi \in \textit{detected} \\ (1-\alpha)B_i^t(\phi) + \alpha I^t(\phi), & \\ & \phi \in \textit{missed} \cup \textit{insertions} \end{cases} \quad (18)$$

Generally α' is smaller than α and only the background corresponding to the smaller difference D is updated. Using the pixel labels, thresholds are updated as follows:

$$T_L^{t+1}(\phi) = \begin{cases} T_L^t(\phi) + 10 & \phi \in \textit{insertions} \\ T_L^t(\phi) - 1 & \phi \in \textit{missed} \\ T_L^t(\phi) & \phi \in \textit{detected} \end{cases} \quad (19)$$

This adaptation procedure decreases slowly the threshold for *missed* pixels. This is done until their label changes. If they become *detected* the threshold is maintained constant. If they become *insertions*, due to noisy pixels or if the threshold is too low, it is increased. This way the system has a constant rate of inserted pixels that can be chosen changing the increasing and decreasing steps.

6.2. Parameterization for the experiments

The first step of the algorithm, performed only once, assumes that during a period of time there are no targets in the image. In this ideal scenario, the two backgrounds are computed easily. The image sequences used in this work do not have target-free images, so another approach was used. B_1 is initialised as the mean of K consecutive frames. $B_2 = B_1 + u$ with u additive noise of $\mathcal{N}(\mu = 10, \sigma = 20)$. The thresholds are initialized as follows: (i) initialize T_L randomly, $T_L = \mathcal{N}(\mu = 10, \sigma = 20)$, and (ii) run the sequence from the end to the beginning and adapt the threshold. Then the resulting threshold was used to start testing the sequence. After the initialization, the detection and labeling step is performed every frame. The adaptation step is performed every N frames, where N is related to the background adaptation constant (see [2] for details).

In the experiments we use the parameters $\alpha = 0.000306$ and $\alpha' = \frac{\alpha}{4}$ as proposed in [2]. The sensitivity, $\mathcal{V} = 40$, is chosen from Receiver Operating Characteristics [10] and the minimum area, $\mathcal{A} = 100$ pixels, from the working scenario. The system runs at more than 130Hz on a 1.6GHz processor using images of 384x288 pixels.

7. Real-time tracking system

In this section we describe a modular architecture for a real time tracking system and use it to increase the performance of the detection. For demonstrating the performance gain by enhancing a detector with a Kalman filter using this architecture, we add to the experiments the evaluation of the real time tracking system using the *BBS* method for target detection.

The tracking system is composed of a central supervisor that calls subsequently the video demon, the robust tracking module and the target detection module (Figure 1). The supervisor manages the data flow between the modules. The detection module detects new targets that are added to the target list. The tracking module provides robust tracking of the current targets using a Kalman filter . The architecture is modular and allows experiments with different implementations of the detection module.

An interesting point is that the Kalman filter can serve to reduce the image surface that needs to be processed by computing a region of interest (ROI) of the targets. This approach allows to reduce the processing time. For further speedup, we restrict the appearance of new targets to manually defined detection regions [11].

The robust tracking system returns events in form of vectors composed of centroid and width and height of the target regions $\vec{y}(t_i) = (x_c, y_c, w, h)^T$. These values are computed from the foreground pixel in the target ROI. This operation is a short cut that allows to obtain the target position and extent directly without need for a costly computation of

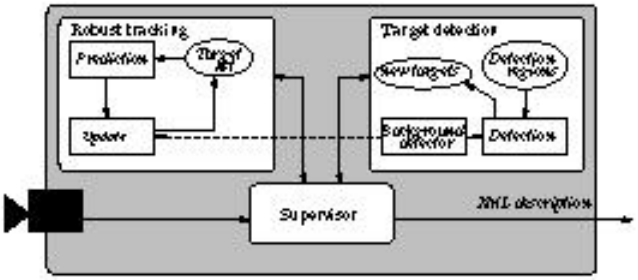

Figure 1: Architecture of the tracking and detection system controlled by a supervisor.

Figure 2: Example frame of the evaluation database.

connected components. The tracking system depends on a number of parameters such as detection threshold d_c (minimum size of targets), noise threshold n_c (pixel energy below this threshold is considered as noise) and parameters that control split and merge of targets (these parameters determine how close targets need to be for merging or splitting).

7.1. Robust Tracking

Robust tracking is achieved by a first order Kalman filter that propagates the target positions and extents in time and updates them by measurements from the detection module. A target is deleted only when the detected pixels in the region do not exceed the detection threshold for 10 subsequent frames. This reduces the number of target losses within a track.

The robust tracking module operates on the list of current targets. For each target, a search region and a Gaussian mask centered on the most likely position are determined. The targets are then updated by collecting data from the detection module that processes the search region and computes first and second moments of the energy image weighted by the Gaussian mask. The Gaussian mask makes the tracking robust to outliers and reduces the error introduced by the above shortcut.

After the update step of each target, the module manages split and merge of targets. The distance between close targets is measured. If this distance is smaller than the merge threshold, the targets are merged.

For splitting of targets, the module performs a connectivity analysis of the pixels within the bounding box. The ROIs are general small. For this reason, the connected component computation does not slow down too much the processing. If there are several components, their distance is evaluated and if this distance is greater than the split threshold, the target is split into its components.

7.2. Parameterization

The performance of the tracking system depends on the correct choice of the parameters. In many systems, these parameters are set manually. In this article, we use an automatic parameter regulation technique [4] that selects the best parameter setting with respect to an output quality metric. The system can monitor 5 entry regions and track robustly up to 8 targets in images of 384 × 288 pixels at 30Hz on a 2 GHz processor.

8. Experiments

In this section we evaluate the performance of the methods *BBS* (Section 2), *W4* (Section 3), *SGM* (Section 4), *MGM* (Section 5), and *LOTS* (Section 6) on the same data set. In addition we evaluate the tracking system described in Section 7 in the same way. This is to demonstrate the increase in performance when a detector is enhanced with temporal filtering.

The CAVIAR entry hall sequences [3] (27 sequences) are partitioned into 14 sequences for testing (13692 frames, 21217 boxes) and 13 sequences for training (12023 frames, 18411 boxes). This database contains people interacting in an entry hall of a public building at different times of the day (see Figure 2). The light regions are close to the saturation point of the camera and move within the sequence. These are indoor sequences, but we need to deal with typical problems of outdoor scenes.

The comparison is based on manually annotated ground truth. The annotaters were instructed to draw a bounding box around each individual and also around each group of individuals. Individuals are labelled only once they start moving. The receptionist is considered as background even when moving. Groups are defined as two or more individuals that interact. Groups can not be determined only by analysing the spatial relations between individuals which makes the detection and tracking very difficult for artificial systems. For this reason, we decide to restrict the evaluation only to individual bounding boxes.

Comparison with the hand labelled ground truth causes several problems that we want to mention here and which

may help when interpreting the results. The detectors are based on background differencing and require a reference image of the background for initialisation. Ideally, this background image should represent the empty scene. Due to the mobile light region, we need a different background image for each sequence. In several sequences, people are present in the rst frame. Taking the rst frame as a reference would incorporate those people into the background. Each detector requires a different type of background initialisation. For this reason, we de ned individual protocols for background initialisation that does not penalise the detector.

The detectors perform a connected components analysis to segment the targets from the background. Two individuals walking side by side may be detected as one target. There are 1656 such cases in the test sequences. Without higher level recognition and interpretation, it is impossible to separate the two individuals. The reader should be aware that this causes additional missed and inserted targets which reduces the numerical value of the detection rate. On the other hand, the comparison between the detectors is still valid, since all detectors compute connected components.

The ground truth labelling depends on the subjective judgement of the annotator concerning when an individual starts moving and also which objects to annotate. For more details please see [7] that evaluates the ground truth of a sequence produced by three independent annotators.

The system performance is measured for each method by the measures proposed by Black, Ellis and Rosin in [1]. Since we are comparing the performance of detectors, we are missing the temporal coherence of tracks. For this reason we use only a subset of the performance measures, namely the Tracker Detection Rate (TRDR) and the False Alarm Rate (FAR). These values are obtained by determining for each frame the best matching pairs of detected and ground truth bounding boxes.

$$\text{TRDR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad \text{FAR} = \frac{\text{FP}}{\text{TP} + \text{FP}} \quad (20)$$

with TP correct (true positive) FP insertion (false positive), FN missed (false negative).

TRDR and FAR are given for a particular overlap threshold T (see eq 21). In addition we compute the area under the curve (AUC) for TRDR and FAR as in [8]. The AUC in this article is computed by the mean of the values sampled between [0.0,1.0] with steps of 0.001. AUC is a comparison measure with the advantage that it is independent of a particular overlap threshold. A perfect system would have AUC of TRDR of 1.0 and AUC of FAR of 0.0.

Figure 3 and Figure 4 display respectively TRDR and FAR of the different methods evaluated on the test sequences. Table 1 shows the TRDR and FAR with an overlap

Method	Tracking Detection Rate (TRDR)		
	% at $T = 50\%$	abs values	AUC
BBS	42.5	9024/21217	0.379
W4	11.7	2473/21217	0.209
SGM	42.8	9075/21217	0.380
MGM	38.2	8097/21217	0.373
LOTS	47.9	10161/21217	0.375
Track	44.4	9425/21217	0.348
	False Alarm Rate (FAR)		
	% at $T = 50\%$	abs values	AUC
BBS	72.4	23710/32734	0.754
W4	92.1	28921/31394	0.858
SGM	54.0	10636/19711	0.591
MGM	63.3	13984/22081	0.642
LOTS	40.3	6851/17012	0.533
Track	35.2	5111/14536	0.493

Table 1: Comparison of TRDR and FAR of the methods evaluated on the test sequences.

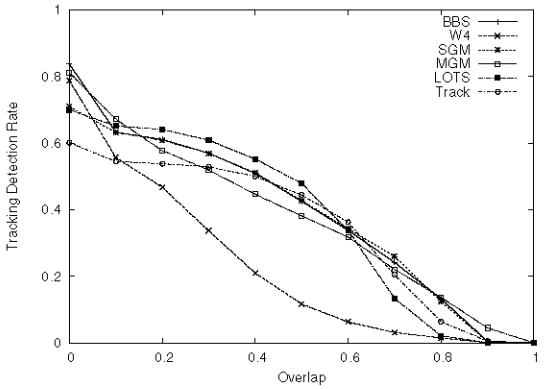

Figure 3: Comparison of TRDR with respect to overlap.

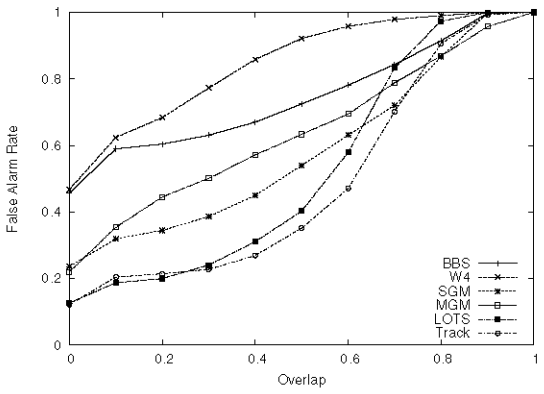

Figure 4: Comparison of FAR with respect to overlap.

requirement of 50%. A correct match is registered, when the bounding boxes of the targets A_{obs} and A_{truth} overlap at least $T = 50\%$.

$$\frac{A_{obs} \cap A_{truth}}{A_{obs} \cup A_{truth}} \geq T \quad (21)$$

with

$$A(x, y, w, h) = [x - \frac{w}{2}, x + \frac{w}{2}] \times [y - \frac{h}{2}, y + \frac{h}{2}] \quad (22)$$

All methods (except *W4*) have similar TRDR (for an overlap of 60%). The method *W4* seems not to be appropriate for the task. In general, we observe TRDR values between 35% and 80% as a function of overlap. These values seem to be quite low for a detection system, but at the same time show the dif culty of the sequences. The main reason for these values are related to the ground truth labelling discussed above. The method *MGM* has the best TRDR for high overlap thresholds, which means that the size estimates are quite good. This is due to the connected component analysis of the methods for bounding box estimation.

Another important value for performance evaluation is the False Alarm Rate (FAR). The method *LOTS* produces the best (lowest) values. The next best performing algorithms are *SGM* followed by *MGM*. The simple method *BBS* has a good TRDR, but at the same time a very high number of false detections increase the FAR which makes the method less suitable.

In the experiments, several problems occurred. Groups of persons are perceived as single targets by the detectors. For small overlap requirements, these cases are counted as one correct match and one missed target. For large overlap requirements (since the bounding box covering both persons is larger), these cases are reported as two missed targets and one insertion.

All detectors build a background model from several frames. Anything that is not part of this learned background is detected. The ground truth labels targets only once they start moving. This causes a lot of insertions by the detectors, since persons waiting but not moving and the person at the reception desk are systematically detected, but not labelled in the ground truth. To remove the false detections of the receptionist, we declared a region of no detection around the reception desk.

The hand labelling chooses the smallest bounding box that covers the target. The method *LOTS* uses the result of the QCC algorithm for bounding box computation. The QCC is a faster but less precise version of a connected components algorithm. This explains the stronger decrease of TRDR for high thresholds.

To demonstrate the performance increase when a detector is combined with temporal ltering, we choose the *BBS* method as detector for the real time tracking method *Track*. The main result is that the method achieves similar TRDR rates, but the temporal ltering and the use of detection regions reduces signi cantly the FAR. The use of detection regions also causes all targets that are within the scene at the beginning of the sequence to be missed. Despite these design decisions with respect to speed, the FAR is half that of the method *BBS*. The enhancement of the other detectors with temporal ltering may also reduce the FAR and allow optimization of computation speed.

Table 2 shows additional statistics for the targets with an overlap of 50%. All methods have equal position errors. The observation concerning the size is con rmed with these statistics. *Track* and *LOTS* are much faster than the other approaches with a small decrease of the quality of the size estimate. The use of detection regions in *Track* increases the time lag for initial target detection which is compensated by very good results in continuously tracking targets once they are detected (small number of dropped frames).

9. Conclusions

We presented ve adaptive background differencing techniques with background models of different complexity. These approaches were evaluated on indoor sequences with dif cult lighting conditions with respect to manually labelled ground truth. The methods LOTS and SGM produce better results than MGM that uses a more complex background model. The same result has been observed previously [9] and may be related to the type of the database.

Several problems occurred in the experiments which reduces the numerical value of the TRDR and FAR. These problems are mainly related to the comparison with respect to the ground truth. The detectors can not distinguish individuals of groups and detect also immobile targets whereas those targets are considered as background. This fact significantly reduces the detection and false alarm rate of the detectors. The comparison between the detectors is still valid, because all detectors encounter the same dif culties.

We demonstrated the reduction of FAR by combining the *BBS* method with temporal ltering within a real time tracking architecture. An important goal of this architecture is also to reduce computation time. Several design decisions were made with respect to speed that reduce the quality of the tracking. Nevertheless, the system obtains the same TRDR rate and signi cantly reduces the FAR. This combination should be seen as an example and the proposed architecture allows other detector types to be enhanced with temporal ltering.

Error Metric [unit]	BBS mean (std.)	W4 mean (std.)	SGM mean (std.)	MGM mean (std.)	LOTS mean (std.)	Track mean (std.)
position [pix]	4.9 (4.5)	5.0 (4.8)	5.0 (4.4)	5.1 (5.8)	5.0 (5.3)	5.2 (5.8)
size [%]	3.1 (24.2)	5.6 (23.6)	-1.6 (18.5)	-0.2 (24.5)	33.5 (33.3)	11.7 (98.2)
entry [frames]	34.4 (47.5)	38.6 (126.7)	34.4 (48.2)	40.1 (45.2)	39.1 (131.1)	107.9 (193)
dropped [frames/100]	20.4 (26.5)	41.5 (27.5)	21.7 (26.8)	28.0 (29.0)	14.4 (22.4)	8.8 (14.5)
processing time [Hz]	8.3	16.7	4.5	2.8	130	70

Table 2: Additional error metrics at 50 % overlap.

Acknowledgements

This research is supported by the CAVIAR project, funded by the EC's Information Society Technology's programme project IST 2001 37540.

References

[1] J. Black, T. Ellis, and P. Rosin. A novel method for video tracking performance evaluation. In *International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 125–132, 2003.

[2] T.E. Boult, R.J. Micheals, X. Gao, and M. Eckmann. Into the woods: Visual surveillance of noncooperative and camou aged targets in complex outdoor settings. *Proceedings of the IEEE*, 89(10):1382–1402, October 2001.

[3] R.B. Fisher. The PETS04 surveillance ground-truth data sets. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, Prague, Czech Republic, May 2004.

[4] D. Hall. Automatic parameter regulation for a tracking system with an auto-critical function. In *International Workshop on Computer Architecture for Machine Perception*, pages 39–45, Palermo, Italy, July 2005.

[5] I. Haritaoglu, D. Harwood, and L. S. Davis. W^4: real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, August 2000.

[6] M. Harville. A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models. In *European Conference on Computer Vision*, pages 543–560, May 2002.

[7] T. List, J. Bins, J. Vasquez, and R.B. Fisher. Performance evaluating the evaluator. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, Bejing, China, October 2005.

[8] J. Min, M. Powell, and K.W. Bowyer. Automated performance evaluation of range image segmentation algorithms. *IEEE Transactions on Systems Man and Cybernetics - Part B - Cybernetics*, 34(1):263–271, 2004.

[9] J. Nascimento and J.S. Marques. New performance evaluation metrics for object detection algorithms. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, Prague, Czech Republic, May 2004.

[10] R. J. Oliveira, P. Canotilho Ribeiro, J. dos Santos Salvador Marques, and J. M. Lemos. A video system for urban surveillance: Function integration and evaluation. In *International Workshop on Image Analysis for Multimedia Interactive Systems*, 2004.

[11] J.H. Piater and J.L. Crowley. Multi-modal tracking of interacting targets using gaussian approximations. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, 2001.

[12] C. Stauffer and W.E.L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, August 2000.

[13] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. P nder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, July 1997.