A Temporal Variance-Based Moving Target Detector*

Seong-Wook Joo and Qinfen Zheng
Center for Automation Research
University of Maryland
College Park, MD 20742
{swjoo, qinfen}@cfar.umd.edu

Abstract

We propose a new moving target detection method for stationary cameras. In this method, local temporal variance is used as a measure for characterizing object motion, along with a simple background modeling technique to remove an artifact resulting from using temporal variance. The algorithm is efficient both in computation time and required memory space. Performance evaluation using the PETS 2001 dataset shows that our approach gives a high detection rate while keeping a low false positive rate. Our method is also shown to outperform the kernel density estimation-based background subtraction method under temporary image degradation and rapidly changing illumination.

1. Introduction

Moving target detection and localization[1] is one of the most fundamental tasks in visual surveillance. Assuming that the video is taken from a stationary camera, moving target detection algorithms are mostly based on either some kind of image differencing or background modeling.

A simple moving target detection can be achieved by subtracting and thresholding two successive frames from a sequence. However, frame differencing alone is not robust enough for most applications. Jain and Nagel [1] used an accumulation of the thresholded difference images with respect to a reference frame. A Moving object detection method by Paragios and Tziritas [2] used the consecutive image difference in a Markov Random Field formulation. Rosin [3] compared several different criteria for choosing the threshold for change detection. A comprehensive survey on general change detection in images is given by Radke *et al.* [4].

Background modeling methods construct a model of the stationary background and then each pixel of a video frame is classified as a part of a moving object, if the pixel is not likely to be from the background. Stuaffer and Grimson [5] modeled each pixel value as a mixture of *K* Gaussians and used an approximate on-line algorithm to update the model. In [6], Elgammal *et al.* used kernel density estimation to model each pixel of the background and applied a threshold on the probability to obtain the foreground. False detection was suppressed by considering the local spatial information in the model. To achieve further robustness, results from two separate models, a short-term model with selective update and a long-term model with blind update are combined to produce the final result.

Background subtraction methods typically require some training period to construct the background model and are generally not robust to rapid changes in the background. We suggest a novel approach for moving object detection using the local temporal variance as the main criteria. In addition, a simple background model is used to enhance the detection. The algorithm not only requires minimal computation and memory but also quickly adapts to a changing background, eliminating the need for the training period. Our method is also robust to image degradation that is previously unobserved, and thus unmodeled.

To the best of our knowledge, temporal variance has not been used directly as the measure for target detection. In [7], the local temporal mean of difference images and adaptive background modeling was combined. The Dynamic Retina [8] used the local temporal mean as an intensity normalization factor to measure the intensity offset caused by camera jitter as well as object motion. Temporal variance of the spatial average of consecutive frame difference was used in [9] to determine when to update the threshold for background subtraction.

The following section describes the temporal variance measure in detail. Next, we show how a simple background model is utilized to remove the artifact coming from the use of a local window. Finally, we describe our performance metric based on the bounding boxes and centroids of the objects, followed by the evaluation results.

---

* This work was supported in part by the Advanced Sensors Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0008.

1 In this paper, moving target detection refers to both detection and localization.

2. The temporal variance measure

Since an object in a video generally occupies a small spatial area and almost always moves with a limited speed, the changes in the pixel values caused by the moving object are localized in the spatio-temporal domain. It is the pixel-wise temporal locality that we wish to exploit. Although consecutive two-frame differencing is highly adaptive to changes in the scene, it is also very sensitive to noise. Therefore, we use information from multiple frames for robustness. One natural way to measure the amount of change in some time interval is the variance. Further, we apply an exponentially decaying weight (window) to the pixel values to save computation and memory. This is easily computed by the recursive filter

$$\begin{aligned} m(t) &= \alpha\, m(t-1) + (1-\alpha)\, x(t) \\ m_2(t) &= \alpha\, m_2(t-1) + (1-\alpha)\{x(t)\}^2 \\ v(t) &= m_2(t) - \{m(t)\}^2 \end{aligned} \tag{1}$$

where $x(t)$ is the value of the pixel at time (frame) t, α is the decay rate, $v(t)$ is the variance, and t=1,2,… In order to avoid floating point operations, (1) can be rewritten as

$$\begin{aligned} m(t) &= ((N-1)\, m(t-1) + x(t))/N \\ m_2(t) &= ((N-1)\, m_2(t-1) + \{x(t)\}^2)/N \\ v(t) &= m_2(t) - \{m(t)\}^2 \end{aligned} \tag{2}$$

where $N = 1/(1-\alpha)$ is a measure of the size of the exponential window. The initial value $m(1)$ and $m_2(1)$ are respectively set as $x(1)$ and $\{x(1)\}^2$. Moving target detection can be achieved by thresholding this variance. Note that the variance measure is reduced to the consecutive frame differencing by using a uniform window of length 2

$$\begin{aligned} m(t) &= (x(t-1) + x(t))/2 \\ m_2(t) &= (\{x(t-1)\}^2 + \{x(t)\}^2)/2 \\ v(t) &= m_2(t) - \{m(t)\}^2 \\ &= \{x(t-1) - x(t)\}^2/2 \end{aligned} \tag{3}$$

This approach uses the variance *directly* as the measure whereas background subtraction methods *model* the background with the variance information (or probability distribution in general) and use the pixel values as the measure. This strategy is similar in spirit to the Resonant Retina [10] where the variance arising from camera jitter is actively used to collect useful information. The increase in variance of a pixel is caused by an object with a different intensity entering the pixel. Furthermore, since an object usually has texture or is non-rigid, as the object moves through, the pixel tends to have even larger variance. The advantage of using an exponential window, apart from its simplicity is that it is sensitive to the initial entry of the object and suppresses noise to some degree. Large coherent change in the signal is quickly amplified, while temporary noise is smoothed out. Assuming a simplified case of perfectly still background and a moving object of uniform intensity i.e., a square pulse temporal signal, the variance is expressed as

$$v(t) = \begin{cases} 0, & t \le 0 \\ A^2(1-\alpha^t)\alpha^t, & 0 < t \le T \\ A^2(1-\alpha^T)\alpha^{t-T}\left\{1-(1-\alpha^T)\alpha^{t-T}\right\}, & t > T \end{cases} \tag{4}$$

where the pulse starts right after time 0 and ends at time T, A is the intensity difference between the background and the object, and α is the window decay rate. The maximum value of $v(t)$ in the interval $0 < t \le T$ is

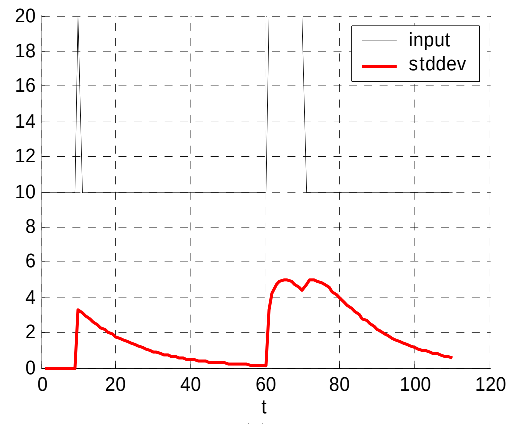
$$\max(v(t)) = \begin{cases} A^2(1-\alpha^T)\alpha^T, & T < -\log 2/\log\alpha \\ (A/2)^2, & T \ge -\log 2/\log\alpha, \end{cases} \tag{5}$$

meaning the peak in $v(t)$ is suppressed if the pulse duration T is short. Note that the critical point $t = -\log 2 / \log \alpha$ coincides with the half-life of the exponential window. As an example, Figure 1(a) illustrates two different pulse durations and the square-root of the variance. The short pulse at t = 10 results in a smaller peak variance than the case with the longer duration starting from t = 60. Figure 1(b) shows an example taken from a single pixel of a video with a person walking through it. Unfortunately, because the window has a long “tail”, the variance decreases too slowly over time. This results in the detection of the moving object with a long trail behind its trajectory (Figure 2(a)). Note that there is a tradeoff between the window size N and the robustness of detection: a small N will shorten the trail but cannot suppress the noise very well.

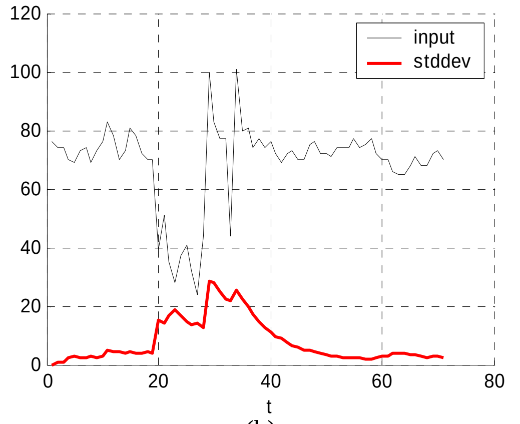
3. Removing the trail artifact

To eliminate the trail effect, a simple background subtraction is performed combining the result with the variance. The background is simply modeled by mean and variance, which are also obtained from recursive filtering.

$$\begin{aligned} m_{bg}(t) &= ((N_{bg}-1)\, m_{bg}(t-1) + x(t))/N_{bg} \\ m_{2bg}(t) &= ((N_{bg}-1)\, m_{2bg}(t-1) + \{x(t)\}^2)/N_{bg} \\ v_{bg}(t) &= m_{2bg}(t) - \{m_{bg}(t)\}^2 \end{aligned} \tag{6}$$



(a)



(b)

Figure 1: One-dimensional example of the variance measure

$m_{2bg}(t)$ is the background mean and $v_{bg}(t)$ is the background variance. The window size N_{bg} should be large compared to N so that the background model covers a longer period and is robust to noise. The background mean and variance are updated only if the pixel is not part of the foreground. In order to avoid including any part of the foreground to the background model, an enlarged foreground region is obtained by thresholding the temporal variance from (2) with a small value.

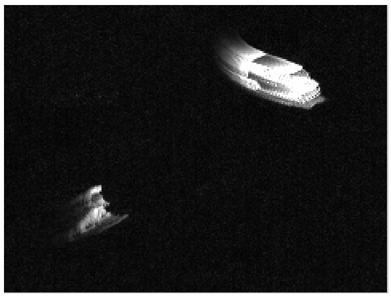
The background model is used to derive a confidence measure of the detected foreground. Assuming a Gaussian distribution for each background pixel, one measure of a given pixel being a foreground can be obtained by the error function

$$f_{conf}(t) = \tfrac{1}{\sigma}\sqrt{\tfrac{2}{\pi}}\int_0^d e^{-z^2/2\sigma^2}\,dz \tag{7}$$

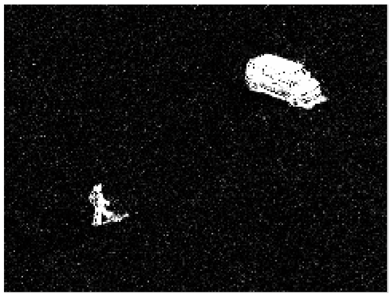
where $d = \left|x(t) - m_{bg}(t)\right|$ and $\sigma = \sqrt{v_{bg}(t)}$. The error function is close to linear near zero. However, we need a function that is very small near the background intensity so that it strongly suppresses false detection. Experience shows that the following sigmoidal function is adequate for our purpose.

$$f_{conf}(t) = \begin{cases} \frac{1}{2}\left(1-\cos\left(\frac{\pi d}{r\sigma}\right)\right), & 0 \le d \le r\sigma \\ 1, & d > r\sigma \end{cases} \tag{8}$$

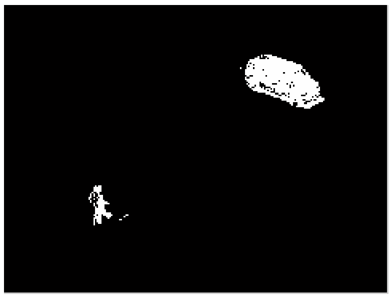
r is a scale factor that determines the range of the function having the transition from 0 to 1. Finally, the square-root of the variance[2] is multiplied by the confidence value and thresholded to obtain the detection mask. The threshold is defined as some factor of the average background



(a)



(b)



(c)

Figure 2: Results of combining variance and background subtraction

variance over all the pixels. Figure 2(a),(b), and (c) shows the variance, the confidence values, and the final detection mask, respectively.

By combining the variance measure and background subtraction, the algorithm retains the robustness of the variance measure while effectively removing false detection in the trail. If background subtraction happens to give a false positive, the variance measure is robust enough to suppress the error. False negatives from background subtraction are less critical since they tend to occur sparsely. Even when background subtraction fails overall, the result would at least be similar to using the variance alone.

4. Performance evaluation

4.1. Implementation

The temporal window size for our variance-based method was chosen as $N = 4$ and $N_{bg} = 8$, the range for the confidence function was fixed at $r = 9$. The sigmoidal function in (8) is discretized into a lookup table for faster computation. The algorithm implemented in Matlab can process 1.5 to 3 frames per second on a 1.7 GHz Pentium 4 processor, depending on the frame size. Another implementation of a simpler version of the algorithm written in C++ achieves real-time rate of 30 frames per second.

At a higher level, the location of the moving object is determined by finding the bounding box of the object and choosing the centroid of the detection mask inside the box as the center of the object. Instead of looking for connected components, which is computationally expensive, we use 1-dimensional projections of the detection image mask to find the bounding boxes. First the mask is projected on the vertical vector i.e., count the number of pixels in each horizontal scan line, and are segmented into chunks that form horizontal strips in the 2-dimensional mask. Each strip is projected on the horizontal vector to get the left and right sides of the box. A final projection to the vertical vector gives a tight top and bottom bound. The boxes that are close together are merged and boxes with sparse pixels are removed. Although there exist configurations where this algorithm fails, most common situations are handled correctly.

4.2. Performance metric

Three types of performance metrics were used for sequences that have bounding boxes as ground truth. “Detection rate” is defined as the fraction of the ground truth boxes that are successfully detected by the algorithm. By successful detection we mean the centroid of the detected object is inside the ground truth box. “False positive rate” is the total number of detection centroids that does not land on any of the ground truth boxes, divided by the number of frames. Lastly, “Multiple detection” refers to the average number of detection inside the ground truth box that is successfully detected. This indicates the amount of “broken up” detection of an object. The ground truth bounding boxes were not quite correct – occasionally part of the object protruded out a few pixels from the box. In our experiments, the ground truth boxes were enlarged by 5 pixels in all directions to correct this error.

4.3. The dataset

The PETS 2001 datasets were used to evaluate the performance since the ground truth information for some of the sequences is publicly available. However, the ground truth is in the form of bounding boxes for the objects. To accurately evaluate the performance of the detection algorithm, the sequence is cut and cropped so that only one moving object exists in each sequence, or two objects do not intersect each other. The reason for this is that when objects merge and split, the detected centroid does not always fall inside a ground truth bounding box. This error is caused by the bounding box algorithm, not from the main detection algorithm. Three sequences, 1, 2, and 3, respectively containing a walking person, a moving car, and another person were selected for testing (Figure 4). The characteristics of each sequence are summarized in Table 1. Units are in pixels. Object size is the average size of the bounding box over all frames. Travel distance refers to the distance of the box centers between starting and ending frames. All the images were converted to 256-level grayscale.

5. Results

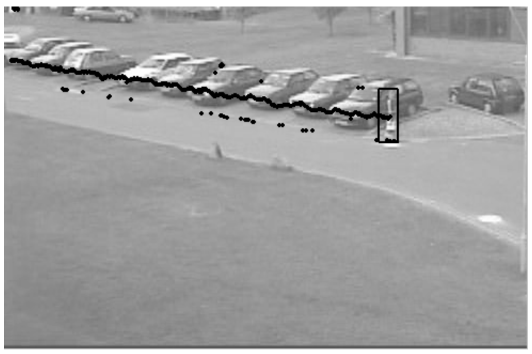
5.1. The effect of object speed

For the variance measure, the window size should be adjusted according to the speed of the object. A slow object requires a long window; otherwise, the interior of the object becomes hollow with small variance, especially when the object has little texture e.g., a solid colored object (see the vehicle in Figure 2(a)) or when an infrared sequence is used. Conversely, a shorter window is desirable for a fast object so that the length of the trail is minimized. For this experiment, the threshold factor was fixed at 5.

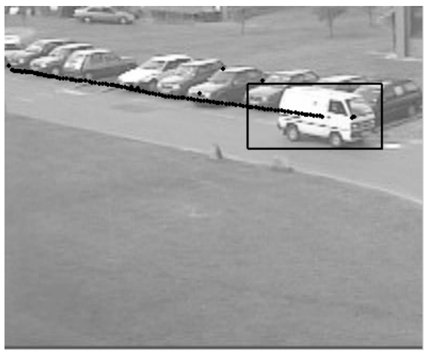
Various object speeds were simulated by skipping or

Table 1: Characteristics of the test sequences

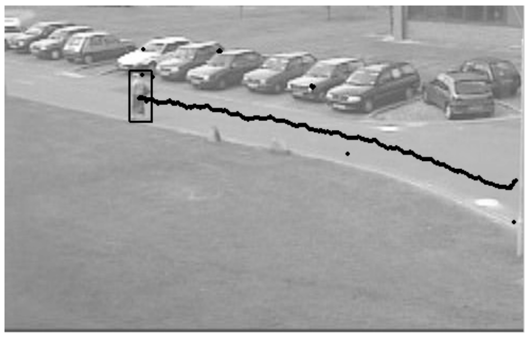
Seq.	Num. of frames	Frame size	Object size	Travel distance (x,y)
1	410	320×490	14.8×41.1	(470, 61)
2	139	392×322	71.7×50.6	(355, 53)
3	341	490×306	18.1×53.2	(413, 94)



(a)

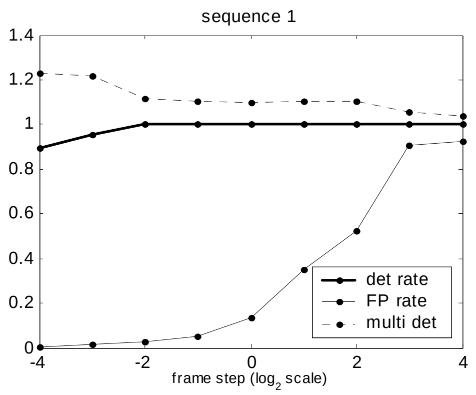


(b)

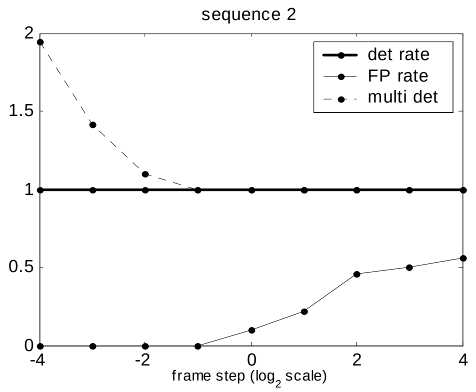


(c)

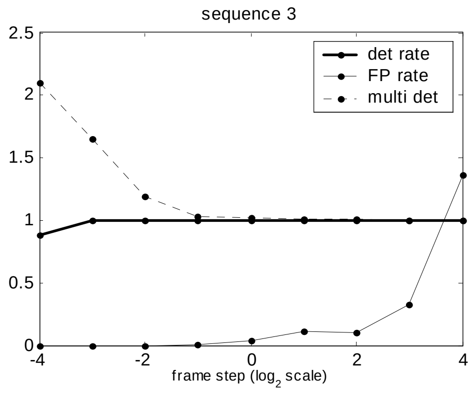
Figure 4: Sample frames from the test sequences and the detection results



(a)



(b)



(c)

Figure 3: The effect of object speed

interpolating the original frames. To speed up the object, larger frame steps were taken. Conversely to slow down the object, sub-frames were created by linear interpolation between consecutive frames, in which case the frame step was defined as a fraction. Figure 3 shows the detection performance with respect to the frame step. Faster object speed increased the false detection rate but did not degrade the detection rate. This is a consequence of using background subtraction: Even if the detection step based on variance had a long trail, background subtraction suppressed it, avoiding the bias that would have occurred towards the tail. At slower speeds, the detection rate slightly decreased and multiple detection increased. This is because the window size becomes relatively shorter compared to the object speed, and the detection is broken up into front and rear parts. This effect was amplified for larger objects (sequences 2 and 3). Multiple detection is considered relatively less critical in moving target detection applications and the current fixed window size appears to be sufficient. However, for some situation where the object speed is expected to be excessively fast or slow, the window size should be adjusted in advance for better performance.

5.2. Comparison with background subtraction

A background subtraction method using kernel density estimation (KDE) [6] was also implemented for comparison. For the training phase, a section of frames was used which contains only the background. The number of training frames used was set to be equal to the number of samples. Shadow detection was not implemented and the same bounding box algorithm was used.

The identical test sequences were given as input for both our variance-based and the background subtraction algorithm. For KDE, 40 samples[3] are used and only the "short term model" is used. For the variance-based algorithm, 5 consecutive frames consisting of only the background were added in front of the test sequence. The background-only frames were not required for our algorithm to work, but was included to remove the errors caused by inaccurate estimates of the parameters during the initial frames. This allows a fair evaluation of the long-term performance. The results are shown in Table 2. A set of different thresholds was used for each of the algorithms. For most thresholds, both methods achieved perfect detection. Since it is difficult to compare the two results objectively, the threshold that produced the lowest false positive was selected for each algorithm. It was observed that our algorithm tended to have lower false positives under similar multiple detections.

In order to compare the performance under a temporary image quality degradation, 20 frames from sequence 1 were selected with the last frame recompressed using JPEG compression quality rate of 50%. (The original frame was in JPEG format with a quite low compression ratio at 3.8%.) The resulting frame has a root mean squared error of 2.8966 and relative root

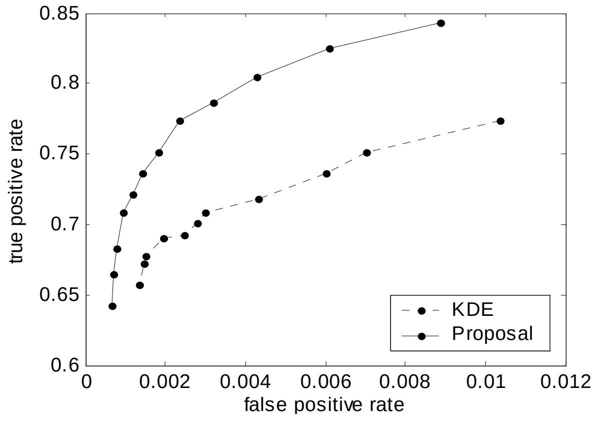
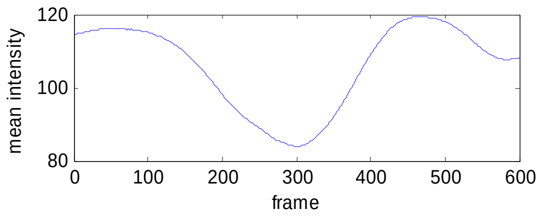


Figure 5: ROC curve for noisy frame

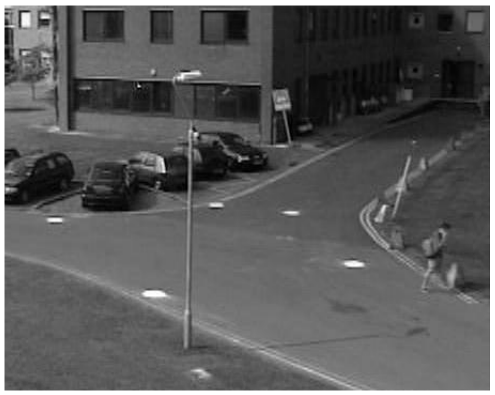
3 More samples did not significantly increase the performance for the test sequences used.

Table 2: Performance measurements for the lowest possible false positive rate obtained

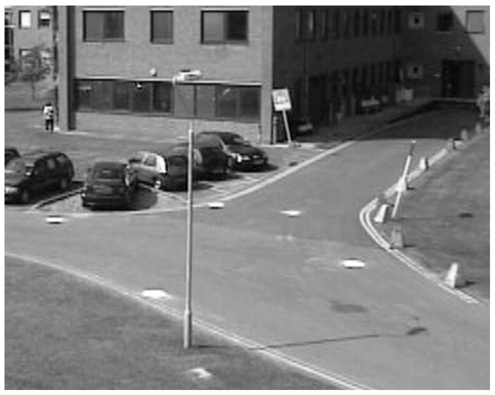
Seq	Detection rate		False Positive rate		Multiple detection	
	Proposal	KDE	Proposal	KDE	Proposal	KDE
1	1.0000	1.0000	0.0366	0.0512	1.1098	1.2415
2	1.0000	1.0000	0.0000	0.0863	1.0072	1.0000
3	1.0000	1.0000	0.0000	0.0528	1.0323	1.0469



(a)



(b)



(c)

Figure 6: Frames from sequence with illumination changes

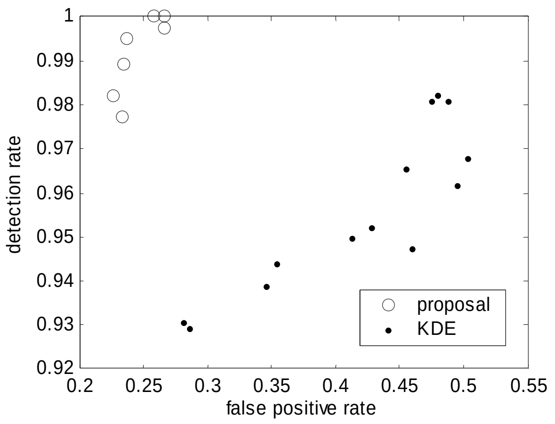


Figure 7: Performance under rapidly changing illumination

mean squared error (with respect to the spatial variance) of 0.0880. Pixel-wise ground truth mask was manually generated for the last frame. The two algorithms were tested with varying thresholds. The ROC curve for the detection on the last frame is shown in Figure 5. Our proposed algorithm consistently gave less false positive rate under equal true positive rate. Since background modeling approaches rely on the history of the pixel values, it is likely to give false positives for the values that exceed the modeled noise range. On the contrary, our variance measure does not rely on history and is effective in handling moderate spike noise as described in (5).

Next, performance under rapidly changing illumination was compared. The test frames are from another sequence of PETS 2001 dataset for which the ground truth is not available. We used manual selection and interpolation to generate a ground truth bounding box data. The number of frames is 600 with frame size 454×360. Figure 6 shows a plot of the average intensity for each frame and two frames with the minimum and maximum mean intensity. For the KDE method, in addition to the "short-term" model, the "long-term" model that covers the entire sequence was used to remove persistent false positives. Figure 7 plots the false positive rate vs. the detection rate, as defined in section 4.2. For our proposed method, only the threshold was varied whereas for KDE, various thresholds and sample sizes were used. The plot shows that our algorithm gives higher detection rates with low false positive rates. The KDE approach needs to keep a small number of samples to quickly adapt to the changing illumination but on the other hand requires large enough samples to be accurate. However, our approach does not require a large number of samples, therefore is more adaptive.

6. Conclusion

We have proposed a temporal variance-based approach for moving target detection. The variance measure is robust to noise and gives a low false detection rate. The trail artifact is greatly reduced by a simple background modeling method. Experiments show that a fixed window size can be used for a reasonable range of object speeds. Under normal conditions, our method performed as well as the KDE-based background subtraction approach. In addition, our approach performs better than KDE under temporary compression noise and rapidly changing illumination. The algorithm is simple and fast and is highly adaptive to changing conditions, making it suitable for a wide range of real-time surveillance applications.

References

[1] R. C. Jain, and H. H. Nagel, "On the Analysis of Accumulative Difference Pictures from Image Sequences of Real World Scenes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 1, No. 2, pp. 206-213, April 1979.

[2] N. Paragios and C. Tziritas, "Detection and location of moving objects using deterministic relaxation algorithms," in *Proceedings of the 13th International Conference on Pattern Recognition, vol.1*, 1996, pp. 201-205.

[3] P. L. Rosin, "Thresholding for Change Detection," *Computer Vision and Image Understanding* Volume 86, Issue 2, pp. 79-95, May 2002.

[4] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image Change Detection Algorithms: A Systematic Survey," *IEEE Trans. on Image Processing*, to appear.

[5] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1999, pp. 246-252.

[6] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proceedings of European Conference on Computer Vision, volume II*, 2000, pp. 751-767.

[7] S. Huwer and H. Niemann, "Adaptive Change Detection for Real-Time Surveillance Applications," in *Proc. of the Third IEEE International Workshop on Visual Surveillance*, July 2000, pp. 37-45.

[8] P. Prokopowicz and P. Cooper, "The Dynamic Retina: Contrast and Motion Detection for Active Vision", *International Journal of Computer Vision*, Volume 16, Issue 3, pp. 191-204, Nov. 1995.

[9] A. Albiol, C. Sandoval, V. Naranjo, J. M. Mossi, "Robust motion detector for video surveillance applications," in *Proc. of the International Conference on Image Processing*, 2003, vol.3, pp. II - 379-82.

[10] M.-O. Hongler, Y. L. de Meneses, A. Beyeler, J. Jacot, "The resonant retina: exploiting vibration noise to optimally detect edges in an image," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 9, pp. 1051- 1062, Sept. 2003.