Robust Real-Time Unusual Event Detection Using Multiple Fixed-Location Monitors

Amit Adam, *Member*, *IEEE*,
Ehud Rivlin, *Member*, *IEEE*,
Ilan Shimshoni, *Member*, *IEEE*, and
David Reinitz, *Member*, *IEEE*

Abstract—We present a novel algorithm for detection of certain types of unusual events. The algorithm is based on multiple local monitors which collect low-level statistics. Each local monitor produces an alert if its current measurement is unusual and these alerts are integrated to a final decision regarding the existence of an unusual event. Our algorithm satisfies a set of requirements that are critical for successful deployment of any large-scale surveillance system. In particular, it requires a minimal setup (taking only a few minutes) and is fully automatic afterwards. Since it is not based on objects' tracks, it is robust and works well in crowded scenes where tracking-based algorithms are likely to fail. The algorithm is effective as soon as sufficient low-level observations representing the routine activity have been collected, which usually happens after a few minutes. Our algorithm runs in real-time. It was tested on a variety of real-life crowded scenes. A ground-truth was extracted for these scenes, with respect to which detection and false-alarm rates are reported.

Index Terms—Video analysis, video surveillance, unusual events.

1 Introduction

In many public places (e.g., malls and airports), a large number of surveillance cameras have been installed. Due to the large number of cameras, it is impossible to have constant human monitoring of all the video streams arriving at the control center. Therefore, the video generated in current systems is being used mainly for investigative purposes after an event has happened, in contrast to using it as a mechanism for real-time alerts during an event.

Recently, there is an increasing interest in automatic analysis of a video stream in order to generate alerts in real time when an "unusual" event happens. Such algorithms may be used as an attention mechanism, which with proper detection and false-alarm rates, will enable a single operator to effectively "watch" a large number of cameras.

1.1 Operational Requirements

Before reviewing related work, it is worthwhile to consider algorithm-independent requirements which are necessary for successful deployment of such an unusual events detection mechanism in large scale surveillance projects. As we will see, all previous works fail to satisfy these requirements and are therefore unlikely to be suitable for deployment in large-scale surveillance projects.

The following is a list of requirements which are very important for successful deployment of a video analysis module in real-life projects:

- Tuning the algorithm for a given video stream should be simple and fast—this process is done by technicians and for many dozens of cameras at every installation site.
- The algorithm should be adaptive—the environment may change (different times of day for example).
- Robustness—many real-life scenes are crowded and cluttered.
- Fully automatic learning and operation, with a (desirably) short learning period.
- Low computational requirements—either the algorithm will run on a DSP card installed with the camera ("smart camera") or on a small number of PCs handling multiple cameras. It is unreasonable to require a PC for every camera.
- *Predictable* performance—which should be adequate in many operationally interesting scenes.

In contrast with most previous works reported, our algorithm satisfies all of these requirements (having of course its own limitations which will be discussed).

1.2 Related Work

Many of the works that have been published on event/gesture/action recognition are concerned with the recognition of a finite set of human actions in a specific, usually controlled domain (e.g., American Sign Language and ballet moves). The earlier works [29], [2], [4], [5] used a parametric representation of various observables in order to represent and recognize this set of actions.

Nonparametric representations of actions were used in [34], using the histograms of spatio-temporal gradients. More recent works that use the spatio-temporal volume to recognize specific actions are [16], [1]. In [28], a boosting-based classifier is learned for activity recognition.

Using spatio-temporal patches, Boiman and Irani [3] detect unusual behaviors with minimal learning requirements. However, the approach is demonstrated on (challenging) toy-examples and not on long-term real-life scenarios.

View-invariant recognition of specific actions is described in [23], [21] and also in [16]. The works in [23], [21] are based on tracking a hand's trajectory or human joints.

A lot of information is contained in the tracks that people or other objects follow in the scene [15], [11], [30], [17]. Videos from an airborne platform were considered in [18]. The works in [6], [31] use subspace constraints and shape theory to recognize deviation of people from a well-controlled path they should follow. In [19], [10], the authors also consider the interactions between objects in the scene. People usually move with a purpose, and this fact is used in [7] to determine if a person is moving normally (toward some goal) or inexplicably. The same authors addressed the question of evaluating algorithms for "abnormality" detection based on human judgment of abnormality [8]. Recently, in [24] additional information (among which is an extension to the classification of the type of movement as in [9]) was added to the tracking-based information, and applied to analysis of tennis videos. Porkili and Haga [22] added other object and frame-based information to the trajectories and showed abnormality detection in synthetic and simple real-life scenes.

The main drawback of tracking-based approaches is their limitation to scenes where tracking is possible and the many opportunities for false alerts due to tracking failure or creation of false targets (shadows, clouds, fragmentation of targets, etc).

Specific complex events involving a number of objects are identified in [14], [20]. These works are limited to recognition of well structured scenarios. In [12], sequences of events were used to model the activities—the extraction of objects and actions from the video was semiautomatic.

Approaches which use low-level information without the need for tracking were described in [32], [33], [35]. Low-level features may be extracted reliably over time and, therefore, these approaches are more robust than tracking approaches. In addition, in [32], [33] model selection techniques are employed toward achieving fully automatic operation. However, these works which are based on clustering video segments have a more offline nature.

- *A. Adam and E. Rivlin are with the Department of Computer Science, Technion—Israel Institute of Technology, Haifa 32000 Israel. E-mail: {amita, ehudr}@cs.technion.ac.il.*
- *I. Shimshoni is with the Department of Management Information Systems, University of Haifa, Haifa 31905 Israel. E-mail: ishimshoni@mis.haifa.ac.il.*
- *D. Reinitz is with Rafael Ltd., POB 2250 Haifa Israel. E-mail: rntz@rafael.co.il.*

Manuscript received 4 Aug. 2006; revised 19 June 2007; accepted 1 Oct. 2007; published online 14 Nov. 2007.
Recommended for acceptance by H. Sawhney.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0578-0806.
Digital Object Identifier no. 10.1109/TPAMI.2007.70825.

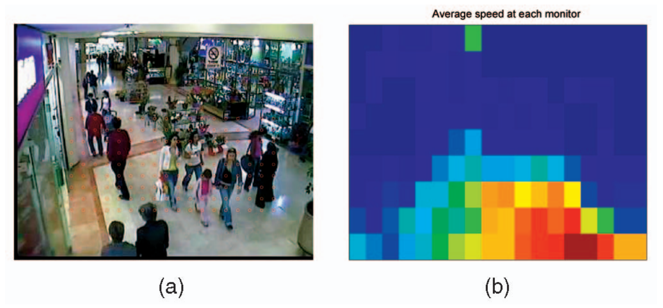

Fig. 1. (a) A typical scene and the positions of the multiple low-level monitors chosen for this scene. (b) The long-term average flow magnitude at each monitor of the scene. Note that monitors closer to the camera indeed observe larger flows.

Moreover, the complexity of these algorithms precludes predicting their suitability for a given scene.

1.3 Our Contribution

This paper presents several contributions to the discussion on unusual events detection in video surveillance:

- First, we evaluate previous work and our own with respect to *applicability* to real-life scenarios. This is achieved by critically reviewing previous work and our own in light of the operational requirements we have listed in Section 1.1.
- Moreover, we use long, real-life videos for experimental evaluation. Compared with previous works, e.g., [13], [22], [3], our videos are much more cluttered and dense and of much longer durations. These videos are much more representative of the types of videos a real-life surveillance system has to cope with.
- Third, we demonstrate that simple low-level cues are useful for reliable detection of operationally interesting events, in cluttered and complex scenes.

Zooming into the details, our algorithm is based on statistical monitoring of low-level observations at multiple spatial locations. It differs from previous works on abnormality detection in a number of important aspects. It is these differences that make it applicable, in contrast with previous works.

First, we extract low-level measurements at fixed spatial locations instead of extracting object-based measurements—in other words, we do not track objects. As a figure of speech, this is like the Lagrangian versus Eulerian approaches in tracking curve evolution. This "Eulerian" approach we use allows our algorithm to operate robustly for long periods, in a variety of scenes *including crowded ones*.

Second, our algorithm is computationally nondemanding, allowing it to perform real-time, online analysis of the video stream while running on a DSP card.

Next, the algorithm is highly adaptive to a changing environment. Moreover, since we model relatively simple entities, the learning period is usually very short (in fact, there is no learning period but until a sufficient number of observations have been collected, the algorithm may be less effective).

Finally, our algorithm is simple and intuitive. This simplicity has a number of important practical consequences. First, we can directly mark on the screen where the abnormality exists. This is in contrast with [35], for example, which finds a video segment which is abnormal and requires further analysis in order to pinpoint the abnormality. Second, due to its intuitiveness, the algorithm's performance is usually *predictable* which is important for a commercial application.

Our limitations. Having said that, the main limitation of our approach is that it cannot detect events that are characterized by an unusual sequence of short-term normal actions. For example, our approach cannot currently detect a loitering person [19] or a person not passing his card in a card reader before entering a secure place [32].

Organization. The rest of this paper is organized as follows: In the next section, we present an overview of our local-monitors-based algorithm. In Section 3, we describe in detail the approximated optical flow measurements our local monitors extract. Next, we present our experimental results. Finally, Section 5 concludes the paper.

TABLE 1
Effectiveness of Temporal Integration for Filtering False Alerts

K alerting frames out of last Y frames	Detections	False Alerts
1 out of 1	8/8	dozens
2 out of 2	8/8	5
3 out of 3	7/8	0
4 out of 6	8/8	0

2 Algorithm

2.1 Overview

We detect unusual events by monitoring the scene with multiple, local, low-level feature monitors. An example is shown in Fig. 1a, where the red circles mark the positions of the local monitors which were setup for this particular scene.

Each monitor is an object which extracts local low-level observations from the video stream. For example, the observation could be the current direction of optical flow at the monitor's location, or the local flow's magnitude (see details in the next section).

The monitor has a cyclic (fixed length) buffer where the extracted observations are stored. Given a new observation, the monitor computes the likelihood of this observation with respect to the probability distribution of the observations currently stored in the buffer. This is done by building a histogram of the samples and checking the new observation's likelihood. We have also implemented an option using kernel density estimation based on the buffered observations with automatic choice of bandwidth, but we found the improvement not significant. Once the likelihood falls below a preset threshold, the monitor outputs an alert.

The monitor inserts each new observation into the buffer and removes the oldest observation. This allows our monitor to adapt to a changing environment. Note that alerting observations are rare and there is no risk in inserting these observations into the buffer.

2.2 Integrating Multiple Monitors

Each monitor uses only local information and decides whether to alert or not. The algorithm uses a simple integration rule as follows in order to decide whether to produce an "unusual event" alert to the user or not: First, if the number of alerting monitors in the current frame is at least Z (we always used $Z = 1$), then this frame is considered an "alarming frame." Next, if in the last Y frames at least K frames were "alarming frames," then the algorithm produces an alert to the user.

Table 1 gives an example of the effectiveness of this integration procedure. We ran the algorithm on the bus terminal scene (see Section 4) and varied the values of K and Y. One may see that indeed false alerts are filtered without compromising the detection rate.

Intuitively, the number Y is related to the expected duration of an unusual activity. The longer the expected duration, the higher Y can be. K may and should be chosen lower than Y to give some robustness to occlusions or missing observations (the values we use are usually $K = 7$ and $Y = 10$). Note that Y is also related to the delay in producing the alert to the user, although this is usually insignificant (typical values of less than 1 second of delay).

Although other reasonable tests like proximity of alerting monitors may also filter out local false alerts, we found that the abovementioned temporal integration rule works well in all scenes that we tested.

2.3 Density of Monitors

The choice of Z (required number of alerting monitors) above and the density of the monitors placed in the image are related and application-dependent. The basic issue is how many monitors will observe an "unusual" observation as a result of an object in the scene acting in an unusual manner. This depends on the spatial extent of the object and on the density of the monitors—basically, through the number of monitors covered by the object. For redundancy (occlusions, aperture problem), we usually choose the monitor density to be such that objects will typically be covered by at least five monitors. We then require $Z = 1$ alerting monitors giving us a lot of slack to handle occlusions or nonobservation (see the next section). As described previously, the inevitable sporadic false alerts are filtered by temporal integration.

3 Approximated Optical Flow Monitors

We now describe the specific low-level monitors which we implemented. We chose to monitor the velocities in the scene. Recall that we do not track objects and, therefore, we need to extract measurements that are actually optical flow. However, we extract the measurements in fixed spatial positions. Therefore, it is very likely that we will encounter the aperture problem (since we do not compute the flow at "good features to track" [27]).

Flow probability matrix. In order to overcome this problem, we first compute an explicit representation of the ambiguity in flow due to the aperture problem. Taking a template window around the monitor's pixel, we compute the SSD error matrix corresponding to discrete shifts in a certain window surrounding the monitor pixel. We then transform this SSD matrix into a probability distribution function by

$$P(u,v) = K \cdot e^{-\alpha \cdot SSD(u,v)} \qquad (1)$$

(a similar model was used in, e.g., [25], [26]). We now have a flow distribution matrix $P(\cdot,\cdot)$ in which the entry (u,v) represents the "probability" that (u,v) is the current optical flow at the monitor. In Figs. 2c and 2d, we show the computed flow distribution at the point in the center of the template region shown in Figs. 2a and 2b.

We note that a flow matrix with respect to the previous frame is computed in every frame for every monitor. Computationally, this is the most demanding part of the algorithm.

Binning. Basically, we can now aggregate the flow probability in P either by angular bins (when we want to compute flow direction) or by radial bins (when we want to compute flow magnitude). By aggregating the flow into angular or radial bins, we will a get a new probability distribution $\{(O_i, P(O_i))\}$. Here, O_i are the various possible observations—either a direction or a flow magnitude. An example of the distribution obtained by aggregating the flow matrix into angular bins is shown in parts Fig. 2e and 2f.

Replacing with most-likely and ambiguity test. Although the distribution $\{(O_i, P(O_i))\}$ is the "correct" measurement which we can extract, we simplify the measurement process by extracting the direction or magnitude of the most likely flow we found. Before we take this observation as the current measurement, we verify that the actual ambiguity in this observation is not too large: Let $O_{M.L.}$ be the observation at the most likely flow pixel (O is either magnitude or direction) and let $d(O_i, O_j)$ be a measure of the amount of difference between the two possible observations O_i and O_j (for example, angular difference or radial distance). We accept $O_{M.L.}$ as our observation only if

$$\sum_i d(O_{M.L.}, O_i) P(O_i) < T, \qquad (2)$$

where T is the allowed angular or radial ambiguity.

Note that this criterion is a "goal-oriented" criterion. If we used, for example, the standard check on eigenvalues of the Lucas-Kanade matrix, then an edge-type ambiguity (e.g., like the one shown in the first example in Fig. 2) would not result in a valid observation, although it may happen to be nonambiguous at all if for example we are interested in direction monitoring.

If the aperture problem is such that the ambiguity exceeds T, then the monitor simply does not produce an observation at the current frame.

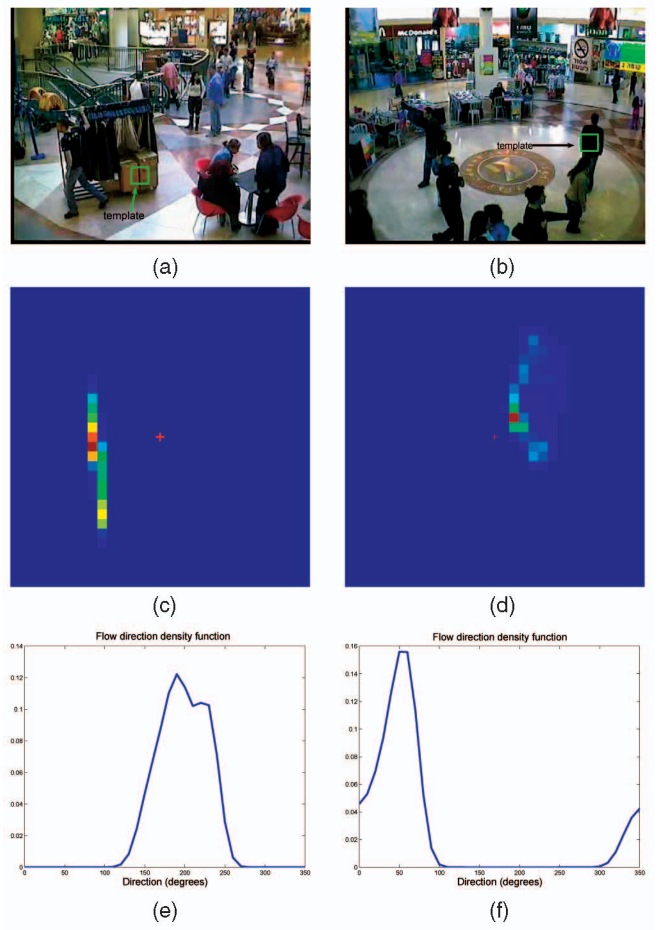

Fig. 2. Example of explicit representation of flow ambiguity. (a) and (b) Two example scenes with a marked template. (c) and (d) The corresponding flow distribution matrix $P(\cdot,\cdot)$ obtained by SSD template matching. (e) and (f) Binning the flow distribution matrix into angular bins, yielding a distribution on flow directions.

As an example of the collected features, Fig. 1b shows the average flow magnitude (observed over a certain period) at each of the monitors shown in Fig. 1a. Note that the monitors located at the bottom of the image (which are closer to the camera) indeed observe higher speeds than the monitors which are further away from the camera.

3.1 Choice of Parameters and Additional Remarks

A number of choices have to be made in the algorithm described above. We now elaborate on these choices.

Template and search region size. The SSD in (1) is computed over a certain window size. This size is application and scene dependent. For example, in monitoring pedestrians, the window should be large enough to capture the full body's motion as opposed to capturing waving hands motion. Likewise, the search region size (possible (u,v) values in (1)) should be large enough to cover the expected possible displacements between two frames. In order to handle large flows at a reasonable computational burden, we work on downsampled images.

Ambiguity threshold. When monitoring directions, we set the threshold T in (2) to 20 degrees. When monitoring speeds over a range of 0 to 10 pixels (possibly downsampled) the value of T was set to 1.5 pixels. These values were chosen empirically and were fixed across all our tests, demonstrating relative insensitivity to their exact value.

Monitoring speeds or directions. Computationally, the most demanding part of the algorithm is the extraction of the flow distribution matrices. Once these matrices are computed, it is possible to monitor both speeds and directions with little additional computational cost. However, in many operational scenarios and, in

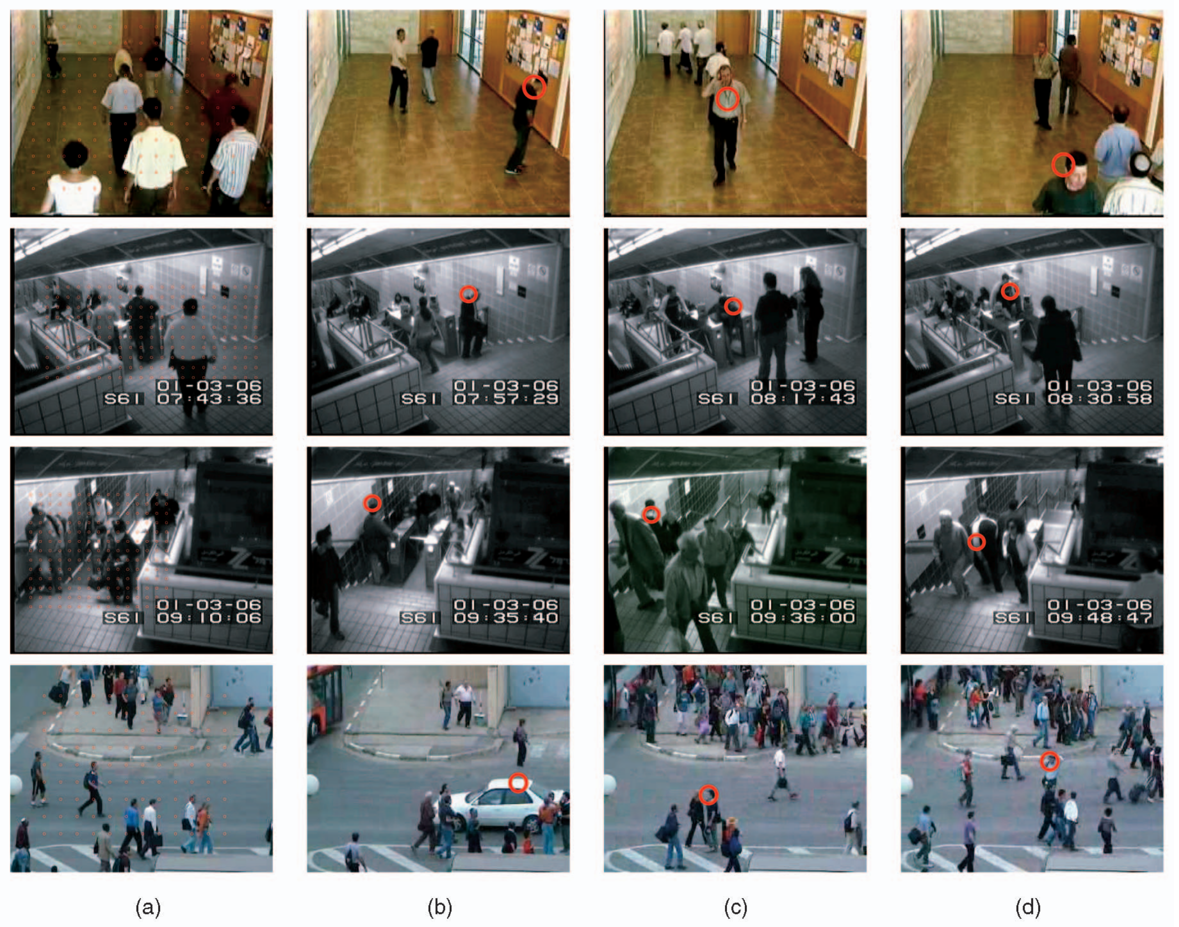

Fig. 3. Please view electronically—in color and enlarged. Column (a) shows the typical activity in each scene and the placement of the monitors. Columns (b), (c), and (d) show examples of detected unusual events. We mark the median position of alerting monitors. In these scenes, the detected events gave rise to flows in an unlikely direction. See Table 2 for actual statistics.

particular, in the ones we have considered, it makes sense to monitor either speed or direction but not both. In the experiments we ran, the user selected which observable is to be monitored.

No observation. A monitor returns no observation at a given frame if either the most likely flow is (0, 0), or if the ambiguity threshold T in (2) is crossed. We remark that while it is not rare for a monitor not to produce an observation due to ambiguity, in practice, we still obtain enough observations for the algorithm to achieve good performance.

4 Results

We present results on videos from eight cameras in five different sites. Except for the dining hall and bus terminal sites, where we recorded with our own camera, all videos were obtained by direct recording from site-installed surveillance cameras.

We watched all the videos and prepared a "ground truth" list of all the unusual events which would interest a security guard watching these videos [8] (e.g., people running in the mall). We then counted the detections and false alarms of our algorithm with respect to this "ground truth."

Please refer to Figs. 3 and 4 when reading the following description of each site. The leftmost column describes the typical activity in each scene and the positions of the local monitors placed in each scene. Columns (b), (c), and (d) show examples of detected events, where the median position of the alerting monitors is marked (e.g., third image in second row in Fig. 4, where two children were detected running, and the median position is between them).

In the dining hall site (first row in Fig. 3), we recorded video during the beginning of lunch time. Most people go into the dining hall with their back facing the camera. Observing the flow directions, our algorithm, in this case, detected people coming out of the dining hall or turning right to the restrooms (this is of course of no interest to a security guard, but it simulates a one-directional passage).

Two cameras were recorded in an underground train station. The first was pointed toward the entrance platform (second row in Fig. 3) where the normal routine is to go down through the turnstiles and enter the platform (with the back toward the camera). The second camera observed the exit platform where people go up facing the camera (third row in Fig. 3). Sample detections are shown in Fig. 3—please view the images in enlarged mode in the PDF viewer.

We shot a few minutes of video in a bus terminal where hundreds of people generally walk toward the right. Our algorithm detected a car and a few people going in unusual directions in the fourth row of Fig. 3.

Next, we recorded over five hours of video from a surveillance camera viewing an (outdoor) one-way exit lane from an industrial facility. During this period, around 800 vehicles (including cars, trucks, and vans) exited from the plant. In order to simulate a car trying to break into the facility, we were authorized to drive the wrong way several times. In addition, a number of pedestrians were walking on the lane toward the plant. All five of the "breaking in" car events were detected and nine out of 11 of the pedestrians were detected. There was one false alarm during these five hours of video.

Again, note that in all of these examples the algorithm has automatically acquired the likely flow directions at every monitor. No directions were specified in advance as "abnormal."

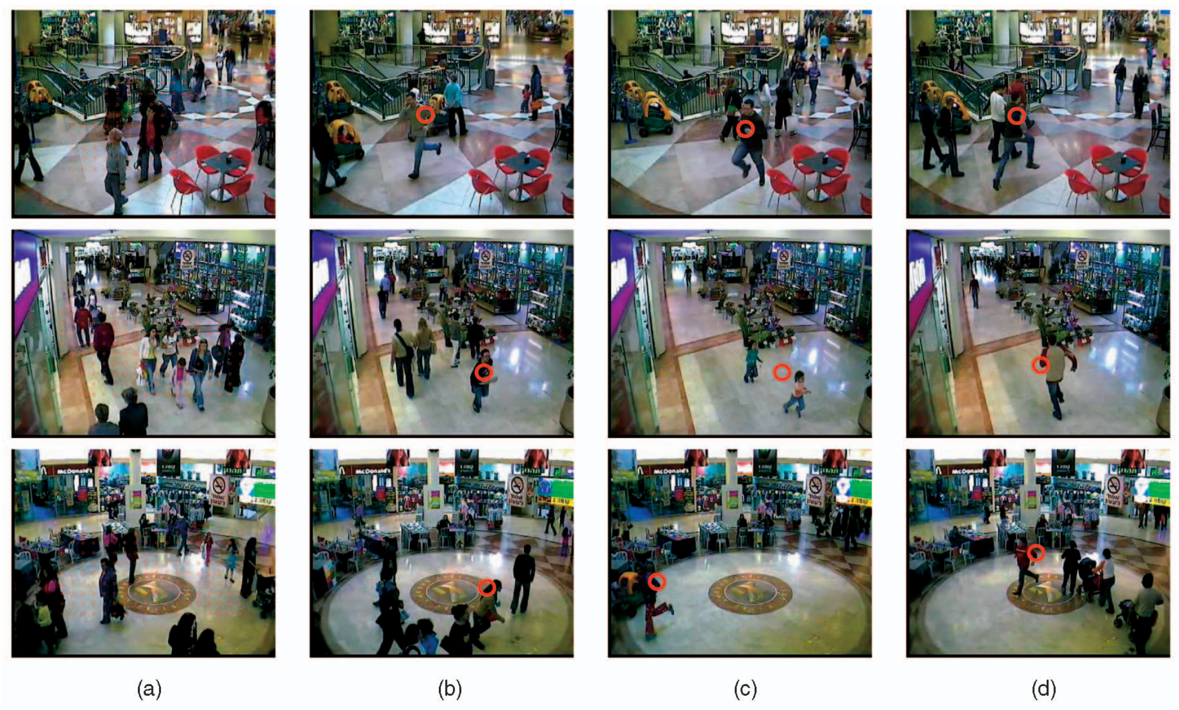

Fig. 4. Please view electronically—in color and enlarged. Column (a) shows the typical activity in each scene and the placement of the monitors. Columns (b), (c), and (d) show examples of detected unusual events. We mark the median position of alerting monitors. In these scenes, the detected events gave rise to flows with an unlikely magnitude. See Table 2 for actual statistics.

TABLE 2
Performance Statistics Obtained

Site	Row in Figure	Video duration	Amount of activity	Detection rate	False alarms
Dining Hall	1, Fig. 3	11 minutes	dozens of people	91% (10/11)	none
Underground entrance platform	2, Fig. 3	1 hour 36 minutes	dozens of people	81% (17/21)	4
Underground exit platform	3, Fig. 3	43 minutes	dozens of people	100% (9/9)	2
Bus Terminal	4, Fig. 3	2 minutes 20 seconds	hundreds of people	100% (8/8)	none
Exit Lane	unavailable	5 hours 20 minutes	around 800 vehicles	81% pedestrians (9/11) 100% vehicles (5/5)	1
Mall camera 1	1, Fig. 4	55 minutes	hundreds of people	95% (19/20)	1
Mall camera 2	2, Fig. 4	40 minutes	hundreds of people	100% (17/17)	6
Mall camera 3	3, Fig. 4	60 minutes	hundreds of people	95% (20/21)	4

Finally, we recorded video from three cameras at a large local mall, on a rather busy day (there was at least one observation in 55-65 percent of the frames, depending on the camera). In these scenes (Fig. 4), we monitored the flow magnitudes (since directions are unconstrained). Four of our colleagues ran in the mall at various speeds and directions in order for us to be able to obtain enough detection statistics at a reasonable time. We again emphasize that our algorithm automatically acquired the "usual" flow magnitudes and alerted when "unusual" speeds (running) were present.

Table 2 summarizes the durations, amounts of activity, and detection and false alarms rates that we obtained. Next to the detection rate, we listed also the actual number of detected events out of the actual number of events that occurred. It may be noted that in most scenes we had near perfect detection at the cost of a relatively small number of false alarms.

In all cases, the setup of the algorithm parameters was minimal (placement of the monitors, choice between direction or magnitude monitoring, working resolution—nearly all other parameters were fixed across the scenes) and took no longer than a few minutes. After this short setup, the algorithm ran fully automatically. We believe these results show that our algorithm obtained very good detection rates while keeping the false alarm rates to a reasonable level. This is established on a wide variety of scenes and on durations long enough to suggest actual long term performance.

5 Summary

For a video-analysis algorithm to be a suitable for deployment in large-scale surveillance projects, certain requirements have to be satisfied. In this paper, we have identified these requirements for algorithms intended for unusual events detection. In short, these requirements call for a low computational cost algorithm, which is fully automatic after possibly a minimal setup, and which works well in cluttered and crowded environments. In addition, the algorithm should be able to adapt to a changing environment and should be such that a field technician will be able to know where the algorithm is suitable and where not (predictable performance).

We presented a novel algorithm which satisfies all of these requirements. Instead of trying to track objects, our algorithm monitors low-level measurements in a set of fixed spatial positions.

In this work, we monitored integral-pixel approximations of optical flow, using a goal-directed criterion for ignoring unreliable observations due to the aperture problem.

The algorithm was tested extensively and rigorously. Very good detection rates at a reasonable false alarm rate were obtained in all of the tested scenes.

Most if not all of the previous works we have reviewed fail to satisfy some of the mentioned requirements. Therefore, their application to real-life large-scale surveillance projects is limited. For example, all approaches relying on objects' tracks will fail to work in the types of scenes which we have considered in this paper. These scenes are very typical of the scenes that are operationally interesting in real-life surveillance projects and any system for detection of unusual events should be able to perform well in these types of scenes.

We discussed the main limitation of our algorithm—namely, the lack of sequential monitoring.

Finally, we see this work as presenting a baseline of achievable performance in these types of scenes. Our data sets are available for public use upon request.

Acknowledgments

The work was done while A. Adam was with Rafael Ltd.

References

[1] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.

[2] A. Bobick and J. Davis, "The Recognition of Human Movement Using Temporal Templates," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 3, Mar. 2001.

[3] O. Boiman and M. Irani, "Detecting Irregularities in Images and in Video," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.

[4] C. Bregler, "Learning and Recognizing Human Dynamics in Video Sequences," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 1997.

[5] O. Chomat and J. Crowley, "Probabilistic Recognition of Activity Using Local Appearance," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 1999.

[6] A. Chowdhury and R. Chellappa, "A Factorization Approach for Activity Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.

[7] H. Dee and D. Hogg, "Detecting Inexplicable Behaviour," *Proc. British Machine Vision Conf.,* 2004.

[8] H. Dee and D. Hogg, "Is It interesting? Comparing Human and Machine Judgements on the Pets Dataset," *Proc. Performance and Evaluation of Tracking Systems,* 2004.

[9] A. Efros, A. Berg, G. Mori, and J. Malik, "Recognizing Action at a Distance," *Proc. IEEE Int'l Conf. Computer Vision,* 2003.

[10] A. Galata, A.A. Cohn, D. Magee, and D. Hogg, "Modeling Interaction Using Learnt Qualitative Spatio-Temporal Relations and Variable Length Markov Models," *Proc. European Conf. Artificial Intelligence,* 2002.

[11] A. Galata, N. Johnson, and D. Hogg, "Learning Behaviour Models of Human Activity," *Proc. British Machine Vision Conf.,* 1999.

[12] R. Hamid, A. Johnson, S. Batta, A. Bobick, C. Isbell, and G. Coleman, "Detection and Explanation of Anomalous Activites: Representing Activities as Bags of Event N-Grams," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2005.

[13] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A System for Learning Statistical Motion Patterns," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 9, pp. 1450-1464, Sept. 2006.

[14] Y. Ivanov and A. Bobick, "Recognition of Visual Activities and Interactions by Stochastic Parsing," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, Aug. 2000.

[15] N. Johnson and D. Hogg, "Learning the Distribution of Object Trajectories for Event Recognition," *Image and Vision Computing,* vol. 14, no. 8, pp. 609-615, 1996.

[16] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient Visual Event Detection Using Volumetric Features," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.

[17] D. Makris and T. Ellis, "Spatial and Probabilistic Modelling of Pedestrian Behaviour," *Proc. British Machine Vision Conf.,* 2002.

[18] G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia, "Event Detection and Analysis from Video Streams," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 8, pp. 873-889, Aug. 2001.

[19] R. Morris and D. Hogg, "Statistical Models of Object Interaction," *Int'l J. Computer Vision,* vol. 37, no. 2, pp. 209-215, 2000.

[20] R. Nevatia, T.T. Zhao, and S. Hongeng, "Hierarchical Language-Based Representation of Events in Video Streams," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.

[21] V. Parameswaran and R. Chellappa, "View Invariants for Human Action Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.

[22] F. Porkili and T. Haga, "Event Detection by Eigenvector Decomposition Using Object and Frame Features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2004.

[23] C. Rao, A. Yilmaz, and M. Shah, "View Invariant Representation and Recognition of Events," *Int'l J. Computer Vision,* vol. 50, no. 2, 2002.

[24] N. Robertson and I. Reid, "Behaviour Understanding in Video: A Combined Method," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.

[25] Y. Rosenberg and M. Werman, "Representing Local Motion as a Probability Distribution Matrix and Object Tracking," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 1997.

[26] J. Shi and J. Malik, "Motion Segmentation and Tracking Using Normalized Cuts," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 1998.

[27] J. Shi and C. Tomasi, "Good Features to Track," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 593-600, 1994.

[28] P. Smith, N. Lobo, and M. Shah, "Temporalboost for Event Recognition," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.

[29] T. Starner and A. Pentland, "Visual Recognition of American Sign Language Using Hidden Markov Models," *Proc. Int'l Workshop Automatic Face and Gesture Recognition,* 1995.

[30] C. Stauffer and W. Grimson, "Learning Patterns of Activity Using Real-Time Tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 747-758, Aug. 2000.

[31] N. Vaswani, A. Chowdhury, and R. Chellappa, "Activity Recognition Using the Dynamics of the Configuration of Interacting Objects," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.

[32] T. Xiang and S. Gong, "Video Behaviour Profiling and Abnormality Detection without Manual Labelling," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.

[33] T. Xiang and S. Gong, "Beyond Tracking: Modelling Activity and Understanding Behaviour," *Int'l J. Computer Vision,* vol. 67, no. 1, pp. 21-51, 2006.

[34] L. Zelnik-Manor and M. Irani, "Event-Based Analysis of Video," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2001.

[35] H. Zhong, J. Shi, and M. Visontai, "Detecting Unusual Activity in Video," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2004.