Extensible Video Surveillance Software with Simultaneous Event Detection for Low and High Density Crowd Analysis

Anuruddha L. Hettiarachchi, Heshani O. Thathsarani, Pamuditha U. Wickramasinghe,
Dilranjan S. Wickramasuriya and Ranga Rodrigo
Department of Electronic and Telecommunication Engineering, University of Moratuwa, Sri Lanka
Email: 090184v, 090518c, 090560v, 090561b, ranga@ent.mrt.ac.lk

***Abstract*—Manual analysis of large volumes of video surveillance footage stemming from the widespread deployment of security cameras is error prone, expensive and time consuming. Despite the commercial availability of software for automated analysis, many products lack third party extensibility, the capability to perform simultaneous event detection and have no provision for anomaly detection in highly dense crowded scenes. We present a plugin based software system for video surveillance applications addressing these shortcomings and achieve realtime performance in typical crowded scenes. Core parameters are computed once per frame and shared among plugins to improve performance by eliminating redundant calculations. A novel multiple pedestrian tracking algorithm is incorporated into the framework to achieve the expected performance. We also propose an improvement to anomaly detection in densely crowded scenes using non-trajectory based dominant motion pattern clusters that can enhance the detection capability of the state-of-the-art.**

***Keywords*—Video surveillance; computer vision; anomaly detection**

I. Introduction

The widespread deployment of surveillance cameras coupled together with the heightened emphasis placed on security has made the need for automated video analysis and suspicious activity tagging a pressing need. At present, there exists only a handful of organizations that develop commercial software for this purpose. Most of these products have limited simultaneous event detection capabilities. For instance, a user may only be provided the option of enabling virtual fencing alarms or loitering detection at a time, but not both. Moreover, the products are proprietary and do not facilitate extensibility or added customization. In certain instances where the activity selected involves significant computations, a live video feed is not displayed on the Graphical User Interface (GUI) of the software but rather only relevant numbers, anomaly types etc. appear on an activity log. Additionally, commercial products are unable to detect suspicious behaviour in highly dense crowds as they rely on conventional pedestrian detection and tracking paradigms. These methods become unreliable due to severe occlusions and clutter that characterize densely crowded scenes such as the video feed recorded from a train station or central bus terminal at rush hour.

978-1-4799-4598-6/14/$31.00

We present an extensible video surveillance software platform featuring simultaneous event detection capability in low and high density crowds. A plugin based architecture is utilized where a host application performs a set of operations commonly required by a majority of event detection tasks implemented as plugins. Event detection in low density crowds primarily relies on tracking pedestrians in a scene. Here, we propose a novel multiple pedestrian tracking (MPT) algorithm capable of processing an online video stream without introducing lags between observations and output results. For densely crowded scenes, analysis is based on the extraction of low level information such as optical flows and foreground regions. This low level information is provided by the host application and the built-in crowd anomaly detection plugin utilizes it to identify suspicious crowd activity. Identifying a set of common core operations which are handed to the host application enables simultaneous multiple event detection in realtime. The plugin architecture enables third party developers to add new features further extending the capabilities of the software.

II. Literature Survey

A. Multiple Pedestrian Tracking

MPT is still an active research topic in computer vision. Since a complete literature review is beyond the scope of this paper, we choose to focus on areas relevant to the proposed methodology. This involves both the detection and tracking of pedestrians. Information regarding foreground objects has also been used. Results from all three methods are combined to determine pedestrian trajectories.

The most widely used pedestrian detector today utilizes the method based on Histograms of Oriented Gradients (HOG) developed by Dalal and Triggs [1]. Although methods with higher accuracies have been proposed since then, most of them suffer from excess running times. However, certain algorithms have been able to achieve both higher levels of accuracy and faster execution times. For instance, Dollar *et al.* [2] use sparsely sampled image pyramids and achieve faster processing and a lower miss rate than in [1].

Object tracking algorithms can be divided into three primary categories - point tracking, kernel tracking and silhouette tracking. Here, we focus on kernel based tracking. Kernel

based methods perform tracking by following a particular object region described by an appearance model. The Mean Shift Tracker by Comaniciu *et al*. [3] and the Kanade-Lucas-Tomasi (KLT) tracker by Shi and Tomasi [4] belong to this category. The low resource consumption of the KLT tracker makes it an ideal choice for this application.

A successful combination of all observations is required to obtain the final trajectories. The models proposed in [5], [6], [7], [8] and [9] report pedestrian tracking with superior accuracies. However, except for [5] and [9], a majority of these models fail to achieve a frame rate of at least 3 frames per second. A faster processing rate is achieved in [9] and its design is intended for offline video processing. Similarly, a fixed time delay exists between the computation results and the observations in [5] due to the sliding window approach utilized. Though the duration of the latency is only a few seconds, it is nevertheless critical in a security application such as the one we propose. The proposed multiple pedestrian tracker was designed after considering these issues.

B. Crowd Anomaly Detection

Video anomaly detection algorithms can be broadly divided into two categories. Here we name them as the non-traditional (modern) approach and the traditional approach. We primarily focus on the modern approach.

The non-traditional approach is characterized by most low level feature extraction based on optical flow calculation [10], [11], [12]. Among the different approaches, one which gained recent popularity involved the modeling of pedestrians in crowds as liquid particles and the use of descriptors of fluid flow dynamics to detect anomalous behaviour. Mehran *et al*. [10] used particle advection and streaklines for representing crowd flow and anomaly detection. Similarly, Hu *et al*. [11] and Wu *et al*. [12] used particle flow maps and particle trajectories respectively. After descriptor calculations, some methods employ algorithms for dimensionality reduction/noise effect suppression. For instance, GaussianART is used in [11]. High-level features extracted in these methods include Space-Time Markov Random Field, Finite Time Lyapunov Exponent field (FTLE) and representative trajectories. Finally, classification techniques are utilized to detect anomalies deviating from the learnt model of normal crowd behaviour.

III. Methodology

We overcome the existing limitations of video surveillance software, namely - the lack of support for extensibility and simultaneous event detection capability, by incorporating a plugin based architecture where common services required by plugins are provided by a host application. Information rendered through these common services is calculated only once resulting in a significant reduction in computational cost. The MPT algorithm achieves realtime performance in low density crowded scenes providing pedestrian trajectories while the crowd anomaly detection plugin detects the presence of anomalies in highly dense crowds. This section presents the implementation details of the application software, MPT algorithm and crowd anomaly detection algorithm.

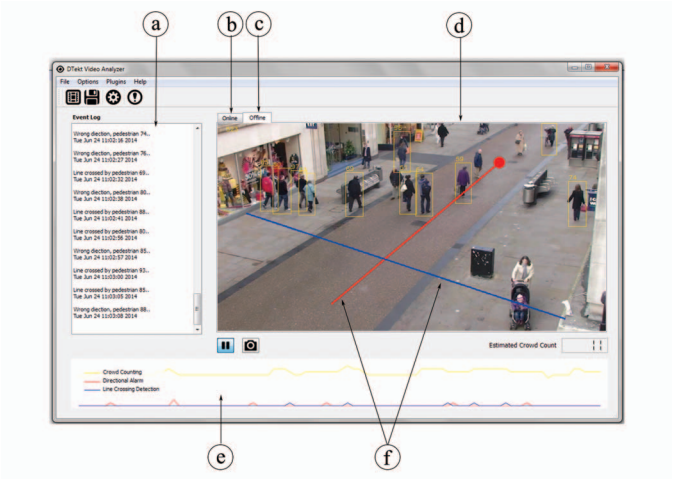


Fig. 1: GUI of the software [(a) Event log, (b) Online mode, (c) Offline mode, (d) Display widget, (e) Activity graph, (f) User inputs]

A. Plugin-based Application Software

A host application running on Microsoft Windows 8 platform (Figure 1) was developed to support plugins and also provide a GUI for users to configure plugins, playback video feed, and view anomalous events. Implementation of both host application and plugins is based on the Qt framework.

Third party plugins can extend event detection capabilities of the software by utilizing commonly provided services. These services carry information of pedestrian tracks and optical flows which plugins use in their event detection algorithms. As this information is calculated only once per video frame and commonly shared with any plugin attached, the overall speed of execution improves. Moreover, this enables simultaneous event detection.

All plugins implement an interface class defined by the host application with necessary variables and functions. Unique tokens are added to each of these classes to qualify as interfaces. Once compiled, all plugin classes are exported as Dynamic-link libraries (DLLs) and placed in a directory specified by the host application. DLLs that implement the interface are recognized by the host application as its extensions and rendered common services. Detection results of each plugin are then displayed on the GUI of the host application.

B. Multiple Pedestrian Tracking

The objective of MPT is to determine the location of each pedestrian in the scene along with the tracks they have followed up to that point within a realtime execution constraint. Results from pedestrian detection, background subtraction and object tracking are combined obtain pedestrian trajectories. Heavy resource consuming tasks are offloaded to a Graphics Processing Unit (GPU) to improve performance.

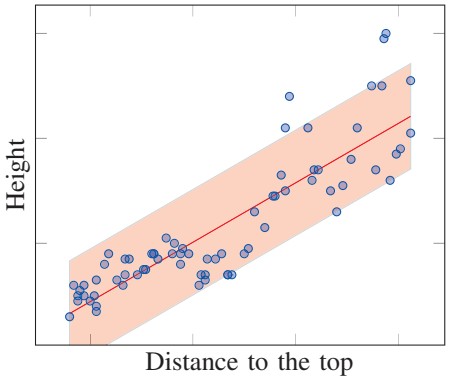


Fig. 2: Height variation of true detections against their y coordinate.

1) Pedestrian Detection: The HOG detector is used for detecting pedestrians. Results from the detector are filtered prior to further processing as erroneous detections can significantly degrade overall performance of the system. Two types of noisy detections are mainly produced. The first type comprises of detections with unusual dimensions. The second type is the false positive. Separate techniques are employed to handle these scenarios.

Erroneous Detections - Type I : For a given scene, the height of a pedestrian increases linearly as he walks towards the camera. Figure 2 shows a plot of the height variation of true positive detections against the y coordinate of the bottom of the detector output (which is a rectangle) for a short segment of video. (On a given frame the $(0, 0)$ coordinate is at the top left corner and the y-axis points downwards.) A curve is fitted to the observed data and the parameters are determined using maximum likelihood estimation. Equation 1 shows the fitted model.

$$h = w_0 + w_1 * y + \varepsilon \tag{1}$$

Here, ε is assumed to be distributed normally with mean zero and constant variance (β^{-1}). For a given camera angle, w_0, w_1 and β have to be estimated first. We use a training video clip that contains pedestrians with significant height variations for estimating these parameters. Once determined, they are used to check whether a pedestrian in a test clip has a false size compared to true positives detected at the same y coordinate.

Erroneous Detections - Type II : Certain locations in a scene have a tendency to generate false positives. To filter them out, the results of the tracker itself are used. Trajectories generated by these detections remain relatively stationary. Unlike detections corresponding to a normal pedestrian who moves about, false positives have random vertical and horizontal movements around a fix point. The total motion estimated of a false positive follows a mean zero Gaussian distribution with a relatively small standard deviation, for instance in the order of 5-10 pixels. This information is used to determine locations generating false positives.

2) Background Subtraction: We implement the Gaussian Mixture Model proposed in [13] for background subtraction. Subsequently a set of morphological operations are performed to suppress noise and fill in missing areas inside foreground objects. Next, foreground objects within the size range of pedestrians are filtered out similar to the method used to minimize false pedestrian detections. The extracted foreground regions supplement the tracking paradigm. However, track initialization is only performed using the results of the HOG detector and not foreground blobs due to their unreliability. The results from background subtraction should only be used to support the determination of pedestrian tracks.

3) Object Tracking: We use the KLT tracker [14] for object tracking. Up to 10 interest points are calculated for each pedestrian within a frame which are then tracked to the next frame. The average movement of the points successfully tracked are estimated and the position of the pedestrian from the previous frame is mapped into the current one.

4) Determining Pedestrian Tracks: At each frame pedestrians from the previous frame need to be tracked to the current one and new detections must also be handled. We follow a five step procedure to estimate tracks accurately. The first three correspond to tracking pedestrians from the previous frame to the current one. The fourth handles the introduction of new pedestrians while the final one processes the complete list of trajectories to eliminate duplicates.

Step 1 : Pedestrians tracked from the immediate previous frame to the current one using the KLT tracker are mapped to the output of the HoG detector. A cost matrix W is calculated.

$$w_{(i,j)} = \frac{T_i \cap D_j}{T_i} \tag{2}$$

Here, T_i is the area of the rectangle enclosing the pedestrian tracked to the current frame while $T_i \cap D_j$ is the area of the intersection between a tracked pedestrian and a pedestrian detected using the HOG detector in the current frame. W is an $m \times n$ matrix where m is the number of pedestrians tracked from the previous frame into the current one and n is the number of pedestrians detected in the current frame. Once W is computed, the best match for each pedestrian tracked is selected from among the detections. If a match exists, the track will be updated using the new detection.

Step 2 : If no suitable match exists for a tracked pedestrian, we seek to ensure if the pedestrian still appears in the scene. If a foreground region significantly overlaps with that tracked pedestrian we assume it is safe to continue the track with the tracked pedestrian even if the HOG detector does not indicate that a pedestrian exists there. However, if the HOG detector does not yield a detection for 3 consecutive frames, we assume the pedestrian has been lost and suspend further tracking.

Step 3 : If no suitable mapping to any HOG detector output exists and we have no evidence to support the existence of the pedestrian from foreground extraction, further tracking is suspended.

Step 4 : Subsequent to assigning HOG detector outputs to all tracked pedestrians, if there still remain detections without

Fig. 3: Anomaly Detection using LSA (Top Left: UMN, Top Right: PETS and Below: UCSD Peds1)

any assignment to a pedestrian they are declared as new pedestrians. They will be given a new identifier and a trajectory will be initialized.

Step 5 : It is possible to obtain multiple feasible tracks for a single pedestrian. This typically occurs when the HOG detector output fails to get mapped to a correct track and instead gets classified as a new entrant. Although initialized separately, the tracks eventually converge due to the identical individual being tracked twice. At this point the two tracks are combined to a single track.

C. Crowd Anomaly Detection

1) Local Statistical Aggregates (LSA) Algorithm: We implement the methodology in [15] for anomaly detection in dense crowds. The LSA algorithm is based on the extraction of appearance and optical flow based features from spatio-temporal cuboids (atoms) of video. Given a set of training and testing data, the video is first partitioned into snippets (i.e. short clips having, for instance, lengths of 5, 45 frames). The problem is to identify if the test snippets contain an anomaly. A distance metric is computed for each of these snippets based on how much the descriptors of each atom deviate from those at the corresponding locations of the training data. The test snippets are then ranked according to this distance and an anomaly is declared if the rank of a particular snippet falls below a certain false alarm threshold. Anomalies are localized by identifying the atom that gave rise to corresponding value in the distance metric.

We use the initial set of video frames as training data and thereafter test each snippet for the presence of an anomaly as it is received in realtime.

Some of the results we obtained with the UMN Crowd Anomaly dataset, UCSD Peds1 dataset [16] and the PETS Event Recognition dataset are shown in Figure 3.

2) Non-trajectory based Dominant Motion Pattern Extraction: The LSA utilizes a feature vector comprising of a histogram of optical flows corresponding to 8 different directions for each video atom. Yang *et al.* [17] propose a method

TABLE I: Quantitative results of our system on three datasets

Dataset	MOTA	MOTP	Precision	Recall	fps
PETS	67.7%	75.5%	90.6%	77.7%	10
T.C.	66.9%	77.6%	95.1%	70.3%	7
UOM	71.7%	77.8%	86.7%	85.7%	12

TABLE II: Comparison of multiple pedestrian tracker results on PETS dataset.

Method	MOTA	MOTP	fps
Anton *et al.* [6]	90.3%	74.3%	0.5 - 1
Breitenstein *et al.* [7]	66.9%	77.6%	0.4 - 2
Berclaz *et al.* [8]	71.7%	77.8%	-
Our method	67.7%	75.5%	10

where such optical flow direction histograms can be utilized to improve dense crowd anomaly detection. They propose learning motion patterns at each of the atom locations and clustering them to extract dominant flows in the scene. LSA treats each atom independently and does not consider coherent global dominant flows in anomaly detection. We propose using the method in [17] to improve LSA by reducing false positive detections.

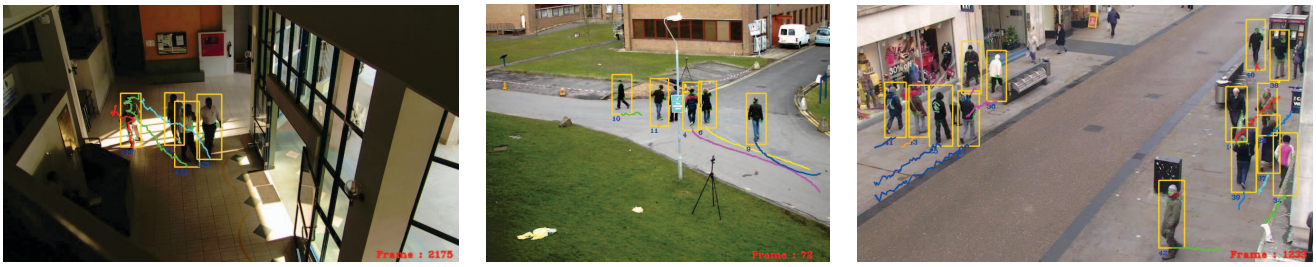
The video is first partitioned into small clips. Then optical flows between adjacent frames are calculated for all the frames in all the clips. Thereafter, the clips are further divided in to spatio-temporal cuboids. We utilize the same cuboids as used for LSA. For each cuboid an 8-bin histogram will be created using the motion vectors calculated using optical flow representing the 8 directions. Each bin in a histogram is considered as a video word. For a single video clip, the number of histograms generated will be equal to the number of cuboids. Then the number of video words will be 8 times the number of cuboids. Each clip produces this *bag of video words* representation of the scene.

An entropy value is calculated next for each video word. Two threshold values are defined (upper and lower) and words having entropies within this range are considered as useful words. Once the useful words are obtained, they are used to extract dominant motion patterns. First a diffusion map embedding of the useful words is done to project the data in to a Euclidean space and then K-means clustering is used to obtain clusters which represent the dominant motion patterns.

Instead of merely flagging anomalies once a particular feature for a test video atom differs significantly from its corresponding training values according to LSA, we suggest examining adjacent atoms in the same dominant cluster for similar deviations as an improvement. If anomalous features are observed within a dominant cluster the probability that a suspicious activity is genuinely occurring should be higher.

IV. Evaluation

The system is separately evaluated on low and highly dense crowded scenes. Both processing speed and accuracy are evaluated and compared with competing algorithms that have been evaluated on standard datasets.



(a) (b) (c)

Fig. 4: Results obtained from the multiple pedestrian tracking algorithm. Three sample frames are displayed from the three datasets : (a) UOM dataset, (b) PETS dataset and (c) Town Centre dataset

In the case of low density crowds, the accuracy of the pedestrian tracker determines the accuracy of the overall system. Thus the tracker is evaluated on two publically available video sequences (PETS-S2-L1-view001 and Town Centre) and one of our own (UOM dataset). To quantitatively evaluate the pedestrian tracker CLEAR MOT metrics proposed by Bernardin *et al.* [18] are used. The two metric values being calculated are the Multi-Object Tracking Accuracy (MOTA) and the Multi-Object Tracking Precision (MOTP). A match between the tracker output and the ground truth is defined as more than 50% intersection-over-union of their bounding boxes. For evaluation, more than 1000 target locations are compared with their ground truths for each dataset. The pedestrians that appear too small to be detected are not used during the evaluation.

The results of the tracker on the three video sequences are shown in Table I. The HOG detector fails to detect certain pedestrians over consecutive frames. This leads to missed targets by the tracker resulting in a lower MOTA value. However, the tracker is capable of tracking targets quite successfully. A better pedestrian detector would significantly improve the accuracy of MPT.

With three plugins installed into the software for low density crowd analysis (line crossing detection plugin, directional verification plugin, and crowd counting plugin), the system is capable of processing a 768×576 resolution video stream at a rate of 10 frames per second (fps) on a desktop computer with an Intel Core i5 processor and a NVIDIA Gefore GTX 480 GPU. The PETS dataset has been captured at 7 fps while the UOM dataset is captured at 8 fps. Therefore achieving 10 fps is sufficient to achieve realtime processing required by a surveillance system. A comparison of our system with the work by other authors is shown in Table II.

We also tested LSA and the proposed improvement using motion pattern clusters. However, the datasets on which we conducted the tests do not have the type of structured motion to fully realize dominant motion cluster extraction. Hence we are unable prove the validity of the proposed improvement to LSA. LSA is able to detect all global crowd anomalies in the UMN and PETS datasets though it does not perform well with the UCSD Peds1 dataset as we do not utilize overlapping atoms as do the authors in [15] for only this particular dataset. With the crowd anomaly plugin enabled the software achieves a execution speed of 30fps processing on a 320×240 video stream.

It should be noted that when low density crowd analysis plugins are enabled we have complete knowledge about all pedestrians in the scene and hence do not activate the high density crowd analysis plugin. Similarly, when feeding in footage of densely crowded scenes, we only activate the crowd anomaly detection plugin as MPT does not perform well due to heavy occlusion and fragmented tracks.

V. Conclusion and Future Work

We presented a video surveillance software based on a plugin architecture. The plugin architecture permits third party users to develop additional features extending the functionality of the software. It also enables simultaneous event detection capability. Along with this, a novel multiple pedestrian tracking algorithm is proposed to track pedestrians in realtime and is incorporated into the host application enabling event detections in low density crowds. Services provided by the host application are utilized in the crowd anomaly detection plugin to identify anomalies in densely crowded scenes. In future work we plan to incorporate more plugins such as removed item detection, abandoned item detection and loitering detection by utilizing already existing services of the host application and also adding any extra services that might be useful for the operation of plugins.

References

[1] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. 886–893, 2005.

[2] P. Dollar, S. Belongie, and P. Perona, "The fastest pedestrian detector in the west," in *Proceedings of the British Machine Vision Conference*, pp. 68.1–68.11, 2010.

[3] D. Comaniciu, "Bayesian kernel tracking," in *Pattern Recognition*, pp. 438–445, Springer Berlin Heidelberg, 2002.

[4] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pp. 593–600, IEEE, 1994.

[5] B. Benfold and I. Reid, "Stable multi-target tracking in real-time surveillance video," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3457–3464, 2011.

[6] A. Milan, K. Schindler, and S. Roth, "Detection- and trajectory-level exclusion in multiple object tracking," in *Computer Vision and Pattern Recognition, IEEE Conference on*, pp. 3682–3689, June 2013.

[7] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, pp. 1820–1833, Sept 2011.

[8] J. Berclaz, F. Fleuret, and P. Fua, "Multiple object tracking using flow linear programming," in *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pp. 1–8, Dec 2009.

[9] A. Butt and R. Collins, "Multi-target tracking by lagrangian relaxation to min-cost network flow," in *Computer Vision and Pattern Recognition, IEEE Conference on*, pp. 1846–1853, June 2013.

[10] R. Mehran, B. E. Moore, and M. Shah, "A streakline representation of flow in crowded scenes," in *Computer Vision ECCV*, pp. 439–452, Springer Berlin Heidelberg, 2010.

[11] M. Hu, S. Ali, and M. Shah, "Learning motion patterns in crowded scenes using motion flow field," in *Proceedings of the International Conference on Pattern Recognition*, pp. 1–5, 2008.

[12] S. Wu, B. E. Moore, and M. Shah, "Chaotic invariants of lagrangian particle trajectories for anomaly detection in crowded scenes," in *Computer Vision and Pattern Recognition, IEEE Conference on*, pp. 2054–2060, IEEE, 2010.

[13] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 773–780, 2006.

[14] C. Tomasi and T. Kanade, *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon University, 1991.

[15] V. Saligrama and Z. Chen, "Video anomaly detection based on local statistical aggregates," in *Computer Vision and Pattern Recognition, IEEE Conference on*, pp. 2112–2119, IEEE, 2012.

[16] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *Computer Vision and Pattern Recognition, IEEE Conference on*, pp. 1975–1981, 2010.

[17] Y. Yang, J. Liu, and M. Shah, "Video scene understanding using multi-scale analysis," in *Computer Vision, IEEE 12th International Conference on*, pp. 1669–1676, Sept 2009.

[18] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *J. Image Video Process.*, vol. 2008, pp. 1:1–1:10, Jan 2008.