10

Parsing Object Events in Heavy Urban Traffic

Yun Zhai, Rogerio Feris, Arun Hampapur, Stephen Russo, Sharath Pankanti
Physical Security, IBM Corp.
United States

1. Introduction

Due to rapid development of video capturing and management technologies and decreasing cost of the hardware infrastructure, digital cameras are nowadays widely deployed in cities, industry campuses, public facilities, etc. In particular, a vast amount of cameras are installed in urban environments to monitor traffic flows. One important video analytics task in traffic monitoring is to capture and track moving objects, and archive their trajectories in the database for future forensic activities, such as event search. Different from conventional event capture technologies, which are mostly based on explicit object tracking, we propose a new technique for capturing moving object using a specialized virtual tripwire.

Virtual boundaries (many times referred as tripwires) are defined as continuous lines or curves drawn in the image space to capture directional crossings of target objects. Due to their intuitive concept and reliable performance under controlled conditions, they are widely used in many digital surveillance applications. In border security, tripwires are set-up along the border line to detect illegal entries from outer perimeter. In urban surveillance, tripwires are often used to achieve many goals, including vehicle/pedestrain counting, wrong-way traffic detection, alerting with pre-defined object attributes, etc. In retail environments, store managements usually obtain the display-effectiveness of a certain merchandize by analyzing the triggering frequencies of the tripwire placed around that item. Conventional tripwire techniques consist of three main steps:

- Object detection: during this process, the target object is spatially segmented in each input image.
- Object tracking: once objects are localized in the frames, they are further correlated to each other within the temporal domain to achieve the trajectories representing the spatial paths of the objects.
- Tripwire crossing detecting: this is the process of determining if the object trajectory intersects with the tripwire in the defined crossing direction(s).

Since reliable object detection and tracking are the corner stones of conventional tripwires, its performance is extremely dependent upon the success of the first two steps. When there is a dense traffic pattern in the scene, object detection and tracking generally fail, and consequently, the tripwire crossing detection no longer provides accurate and meaningful results.

1.1 Related work

Object detection and tracking are long studied problems in the field of computer vision. In this section, we briefly review some of the well known techniques in these areas. One popular type of object detection techniques is the background subtraction (BGS), where active objects (foreground) are detected as the difference regions between the current image and the background reference. Color and gradient distributions are often applied to model the pixels. Some representative work in this area are Stauffer and Grimson (8), Tian *et.al.* (9) and Cucchiara *et.al.* (2). Another common trend in object detection is to localize the target objects in single images by applying pre-learned templates. The object templates are convolved with the input image with scale and orientation variations. One of the most cited work in this category is by Viola and Jones (12) in real-time face detection.
Once objects are localized in images, their correspondences in the temporal domain are established by object tracking techniques. Tomasi and Kanada (10) proposed the KLT tracker for point object tracking. Another popular interest-point tracking method, Scale-Invariant Feature Transform (SIFT), is proposed by Lowe (6). Comaniciu *et.al.* (1) have proposed the Mean-Shift tracking by utilizing the mode-seeking and kernel density estimation techniques. A geometric shape based matching method is developed by Fieguth and Terzopoulos (3). A survey on object tracking methods can be found at Yilmaz *et.al.* (13). Utilizing different object tracking techniques, tripwire is widely applied in many commercial digital video surveillance systems (5; 7; 11).

1.2 Proposed approach

As aforementioned, object detection and tracking usually fail in the dense traffic scenarios. To address the problem of degraded tripwire performance in these situations, we propose a new mechanism to approach the tripwire crossing detection. The proposed new type of tripwire is referred as the "trackerless tripwire", i.e., it does not rely on explicit object tracking results. Different from conventional tripwires which have zero width, our proposed tripwire contains an "envelope" that covers a spatial extend on both sides of the tripwire, and crossing detection is carried out by analyzing the spatiotemporal and appearance properties of the tripwire envelope. Two sets of *ground patches*, called *inner patches* and *outer patches*, are automatically calibrated and generated along the envelope on both sides of the defined tripwire. Each of the patches maintains a history of appearance and motion models. Patch models are cross-checked to find matching candidates. The incorrect candidates are further filtered out by applying a motion consistency test. A set of spatiotemporal rules are also applied to eliminate potential false positive detections. The new proposed trakerless tripwire has been deployed in scenarios with different traffic loads, and promising results have been obtained for both low and high activity scenes.
The remainder of this paper is organized as follows: Section 2 presents the proposed new tripwire, including tripwire construction, feature extraction and crossing detection; Section 3 demonstrates the performance of the proposed method and its comparison against conventional tracking-based tripwire method; finally, Section 4 concludes our work.

2. Trackerless tripwire crossing detection

Numerous object tracking techniques have been developed over the past decades. The common goal of these techniques is to achieve accurate spatiotemporal segmentation of the foreground at the object-level. In crowded videos, such segmentation is extremely challenging and very difficult to accomplish. In the context of tripwire crossing detection, full-scale object tracking across the entire image field-of-view (FOV) many times is an overkill for the problem.

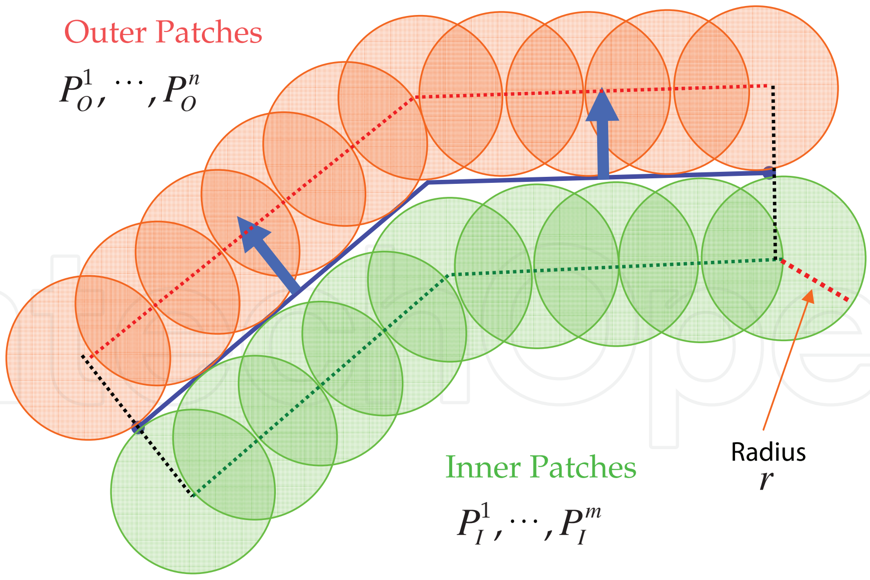


Fig. 1. The proposed trakerless tripwire with inner and outer patches. An envelope is generated along the defined tripwire, and the ground patches are equally sampled on the envelope. The thick arrows represent the defined tripwire crossing direction.

Rather, localized analysis on the video content around the tripwire area is often more reliable and efficient. In our method, we take this as our guiding direction and propose a new type of tripwire detection method that does not reply on explicit object tracking output.

2.1 Ground patches

The proposed tripwire is defined as a set of connected line segments. To model the regions around the tripwire, an envelope is created with a width of $2r$, where r is a scale factor representing the spatial extend of the tripwire region. To capture the local contents of the tripwire envelope, two sets of circular ground patches are equally sampled on both sides of the tripwire. They are denoted as the *Inner Patches* $\mathbf{P}^I$ and *Outer Patches* $\mathbf{P}^O$, where $\mathbf{P}^I = \{P_1^I, \cdots, P_m^I\}$ and $\mathbf{P}^O = \{P_1^O, \cdots, P_n^O\}$. The directions "inner" and "outer" are defined with respect to the order of the tripwire vertices. A graphical illustration is shown in Figure 1. Each ground patch P_i has a center $c_i = (x_i, y_i)$ and a unified radius r.

The centers of the ground patches lie on the edges of the tripwire envelope such that they have the same distance to the tripwire line. In addition to the center and radius, each patch P_i also has a directional vector w_i pointing to the tripwire from the patch center. The purpose of this vector is to assist in the crossing direction determination. Overall, a ground patch P_i is represented as a triplet,

$$P_i = (c_i, r, w_i). \tag{1}$$

An object crosses the tripwire in the *positive* direction if it moves from one of the inner patches to one of the outer patches. Same concept applies to the *negative* crossing. In this paper, the proposed tripwire crossing detection method is demonstrated in the context of *positive* crossings.

2.2 Feature extraction

Both appearance and motion features are extracted for each ground patch. We use the color histogram in the RGB space for the patch appearance model and the motion vector to represent the moving direction of the object inside the patch. A ground patch P_i maintains a history H_i of its appearances,

$$H_i = \{h_i^{T_0}, \cdots, h_i^{T_t}\}, \tag{2}$$

where h^T is the color histogram of the target patch at time T and these histograms are recorded in the chronological order, i.e., $T_x < T_y$ if $x < y$. To avoid including the background information (such as the road surface in a street scene) in the patch model, the model history is updated only if when the patch is occupied by one or more foreground objects. This is determined by analyzing the percentage of the foreground area inside the target patch based on common motion detection techniques such as frame differencing or Multi-Gaussian background subtraction (MG-BGS). If the foreground occupies a sufficient proportion of the patch, the patch is declared to have a "foreground" status, and its current color histogram is extracted for potential model update. In our implementation, we utilize the MG-BGS output of (9) for the motion detection.

The patch history model is an on-line process. When a new color histogram h^{T_+} is extracted at time T_+, it is compared to the latest model h^{T_t} in the history, if there is any. The distance measure used in the model comparison is defined as the Bhattacharyya distance, where $D(h_1, h_2) = \sum_{\text{all bins i}} \sqrt{h_1^i \times h_2^i}$. If the distance between the new model and the last model in the history is significant, then the new model is included in the model history. One exception to this rule is when there is a period that the patch is classified as "background" between the current time and the last time the model history was updated. In this situation, even if the new model has small distance to the last model in the history, they may be caused by two different objects due to the discontinuity in the patch status. Thus, the new patch model should also be included in the history.

Once the color histogram feature is extracted at time T_+ and successfully added to the model history, the motion vector of the target patch is also estimated to determine the moving direction of the object inside the patch. Simple approaches like the Sum of Squared Differences (SSD) could be used to estimate the motion vectors. Let this vector be v^{T_+} and coupled with the color histogram h^{T_+}. The overall feature representation of a patch P_i is,

$$\mathbf{\Gamma}_i = \{\Gamma_i^{T_0}, \cdots, \Gamma_i^{T_t}\}, \tag{3}$$

where $\Gamma_i^\tau = (h_i^\tau, v_i^\tau)$ is the feature vector at time τ.

2.3 Tripwire crossing detection

To determine if the tripwire crossing occurs in the positive direction, patch feature matching is performed by computing the similarity between the target outer patch P_o at the current time T_+ and the inner patches P_i on the other side of the tripwire. In order to comply with the crossing direction, the object in patch P_o should possess a motion opposite to the direction pointing to the tripwire from the patch center,

$$\text{Condition } C_d := v_o^{T_+} \cdot w_o < 0. \tag{4}$$

If condition C_d is not satisfied, that means the object is moving in the opposite direction as the tripwire crossing direction definition, and no match is performed for this patch.

Fig. 2. Keyframes of example videos. (a) Sparse traffic scene; (b) Crowded activity scene. Red overlaid regions in the image represent the areas occupied by foreground moving objects.

If condition C_d is satisfied, the current feature $\Gamma_o^{T_+}$ is compared against with the history features $\mathbf{\Gamma}_i$. The feature matching is composed of two parts: (1) color similarity $S_{\mathbf{c}}$ and (2) motion direction consistency $S_{\mathbf{m}}$, where

$$S_{\mathbf{m}}(\tau) = \frac{v_o^{T_+} \cdot v_i^{\tau}}{\parallel v_o^{T_+} \parallel \times \parallel v_i^{\tau} \parallel}, \tag{5}$$

and,

$$S_{\mathbf{c}}(\tau) = \begin{cases} 0, & \text{if } S_{\mathbf{m}}(\tau) \leq 0, \\ 1 - D(h_o^{T_+}, h_i^{\tau}), & \text{otherwise.} \end{cases} \tag{6}$$

The overall patch-to-patch similarity $\mathbf{S}(P_o, P_i)$ is then computed as a weighted sum of both the similarity measures,

$$\mathbf{S}(P_o, P_i) = max\left(\alpha_{\mathbf{c}} S_{\mathbf{c}}(\tau) + \alpha_{\mathbf{m}} S_{\mathbf{m}}(\tau)\right), \forall \tau < T_+ - T_{th}. \tag{7}$$

where $\alpha_{\mathbf{c}}$ and $\alpha_{\mathbf{m}}$ are the fusion weights of the appearance similarity and motion consistency, respectively. Threshold T_{th} indicates the minimum possible temporal interval for an object to travel from a patch on one side of the tripwire to another patch on the other side. Based on the empirical analysis, we have found that the patch appearance is more distinctive than the motion consistency cue. Thus, a higher weight is specified for the color similarity $S_{\mathbf{c}}$. If the patch similarity $\mathbf{S}$ is above the desired threshold, a tripwire crossing in the positive direction is detected with τ contributing the maximum matching score.

3. Performance evaluation

To demonstrate the effectiveness of the proposed tracker-less tripwire, we have compared the performances of the proposed method and a conventional tracker-based tripwire in two cases: (1) sparse traffic scenes and (2) dense traffic scenes. For the conventional tripwire, we utilize the tripwire feature provided by the IBM Smart Surveillance System (4). The core components in this feature are (1) a background subtraction based object detection process, (2) an appearance based object tracking method and (3) a conventional tripwire based on the object tracking outputs. For the proposed trakerless tripwire, the same foreground modelling technique is used to classify the status of the patches. In this section, the tripwire of the IBM SSS solution is referred as "CFT", and the proposed tripwire is called "NTT".

The activity level of the scene is qualitatively defined. In sparse traffic scenes, the moving objects are apart from each other, i.e., there is a minimal amount of occlusions in the scene. In this case, object detection process is able to achieve clean object-level segments in the video, and consequently, the object tracking result is clean and reliable. A keyframe of a sparse traffic

Frame Rate (FPS)	CFT		NTT	
	Detections	True Positives	Detections	True Positives
30	28	28	31	26
15	21	21	31	26
10	29	29	30	25
5	23	23	30	25
3	21	21	29	25

Fig. 3. **Sparse Traffic**: Accuracy summary of the trackerless tripwire (NTT) and the conventional tripwire (CFT).

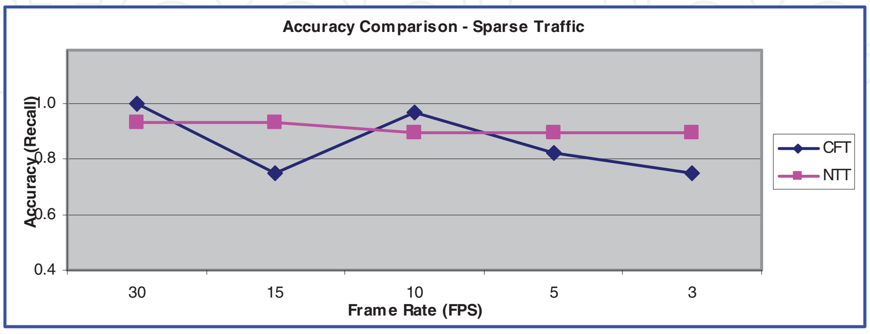


Fig. 4. **Sparse Traffic**: Accuracy comparison between the trackerless tripwire (NTT) and the conventional tripwire (CFT).

video is shown in Figure 2.a. On the other hand, high activity scenes are those that contain a significant amount of occlusions and/or the objects are close to each other, such that object detection can only produce a group-level segmentation. Therefore, it is very challenging to the object tracking module to generate meaningful trajectories that could be used by the tripwire detection module. A keyframe of one crowded scene is shown in Figure 2.b.

3.1 Performance in sparse traffic situations

The purpose of this test is to determine whether the proposed trackerless tripwire detection method could maintain the same performance as the conventional methods, as the object tracking in low activity scenes are reliable. The goal of the test is to accurately count the objects that cross the tripwire in the defined direction. Our testing video is obtained from a street scene in an urban environment. In this video[1], there are totally 28 vehicles in the ground truth, including sedans, SUVs, buses, etc. They are all moving in the same direction. Furthermore, since there is no traffic light, the vehicles are continuously moving forward without any stop. Two tripwires are deployed at the same location to capture the number of passing vehicles. One is the CFT and the other one is NTT. We used the standard recall measure for the accuracy analysis.

$$\text{Accuracy} = \frac{\text{Num. of True Positives}}{\text{Num. of Ground Truth}}. \tag{8}$$

We have also generated a set of videos with different frame rates using the same testing video. The purpose of this is to understand the robustness of the approaches against frame-dropping defects in real situations. The testing frame-rates are 30 frames-per-second (FPS), 15, 10, 5 and 3 FPS.
Detailed tripwire detection results are presented in Figure 3. This table lists the total detections and the true positive samples produced by CFT and NTT. Accuracy comparison between these

[1] Due to privacy issue, the images of the testing data used in this paper cannot be disclosed for public use.

Frame Rate (FPS)	CFT		NTT	
	Detections	True Positives	Detections	True Positives
30	35	26	80	53
15	27	24	80	54
10	24	20	73	51
5	11	10	71	49
3	2	2	46	33

Fig. 5. **High Activity**: Accuracy summary of the trackerless tripwire (NTT) and the conventional tripwire (CFT).

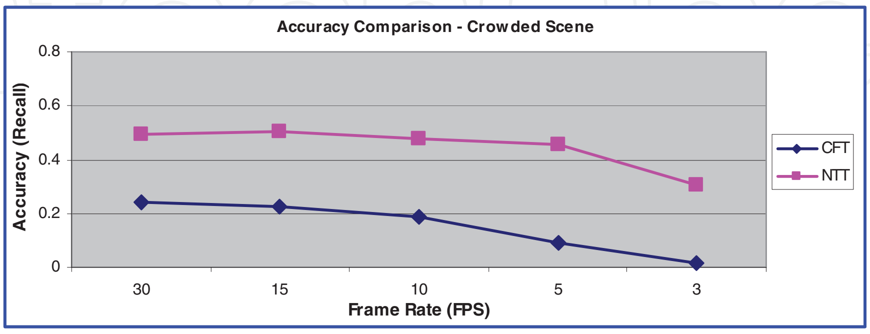


Fig. 6. **High Activity**: Accuracy comparison between the trackerless tripwire (NTT) and the conventional tripwire (CFT).

two methods is illustrated in Figure 4. Based on the results, the mean accuracies of the CFT and NTT methods are 0.85 and 0.91, respectively. From the frame-rate comparison graph, NTT performs in a more stable manner. This is because that as the frame-rate drops, object tracking quality drops as well.

3.2 Performance in crowded scenes

The second test is to compare the performances of CFT and NTT in a high activity environment. In high activity scenarios, object detection and tracking often do not provide meaningful object-level spatiotemporal segmentation. Therefore, conventional tracker-based tripwire crossings cannot be detected in this case. In this test, we apply both CFT and NTT on a video recorded at a heavy traffic street. There are totally 107 vehicles in the video, and all traffic are going in the same direction. In addition to the high density of the traffic, there is a traffic light in the scene so that cars are periodically stopped and piled up. That is extremely challenging for the object tracking module. On the other hand, our proposed tripwire does not rely on the tracking results. It analyzes the appearance and motion patterns just around the target tripwire region. That way, many of the irrelevant information is filtered out automatically, and cleaner detection results can be obtained. Figure 5 shows the detailed tripwire detection results of CFT and NTT, and their accuracy measure comparison is shown in Figure 6.

The performance of the conventional tripwire CFT is merely acceptable to the end users (0.15 in average). The superior performance of the proposed trackerless tripwire is apparent from the performance summary (average 0.45 for NTT). For all frame-rates, NTT over-performed CFT by detecting 25% more tripwire crossing events.

3.3 Failing case analysis

From the performance results presented in Figure 5, a high percentage of false positives is observed for the trackerless tripwire detection method. There are several reasons associate with this. The main reason is the unified patch size. This parameter is fixed in the detection

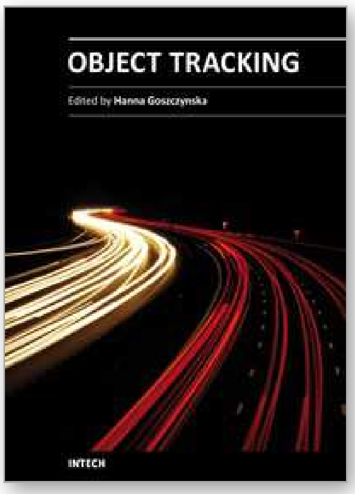
process. If a single object has a size that is much greater than this parameter (e.g., a big bus), what the patch captures is only a small portion of the object. Thus, multiple events are detected for different parts of the same object based on the current feature extraction and matching formulation. One way to improve this is to have a grouping method to intelligently cluster the events that correspond to the same object into a single trigger. Patch motions can be applied to discover the relative movement of the detected objects. Events could be clustered if they possess steady spatial layout.
Another cause for the false positives is the background modelling error. In our testing video, there is a traffic light, and cars are often stopped and be static for awhile. Due to background updates, the object pixels actually became the background model. Therefore, after the vehicles leave the scene, the true background, road surface, is now classified as foreground by the BGS process. Since road surface has strong similarity in color, they are often mis-classified as objects that cross the tripwire. This could be improved by incorporating a more sophisticated BGS process.

4. Conclusions

In this paper, we have proposed a new tripwire detection method. The proposed framework is targeting the challenging problem of reliably detecting tripwire crossings in situations where explicit object tracking fails (e.g., in crowded scenes). Two sets of ground patches are generated along the tripwire to represent the content of the tripwire's proximity. Both appearance and motion information are incorporated in the feature models of the patches. By analyzing the similarities between the patches on the opposite sides of the tripwire, a strong matching candidate can be found to indicate the crossing event. The proposed new tripwire has been deployed in both low and high activity scenes, and very promising results have been obtained to demonstrate its effectiveness.

5. References

[1] D. Comaniciu, V. Ramesh and P. Meer, "Real-Time Tracking of Non-Rigid Objects Using Mean-Shift", *IEEE Conf. on Computer Vision and Pattern Recognition*, 2000.
[2] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting Moving Objects, Ghosts, and Shadows in Video Streams", *IEEE T-PAMI*, 25:(10), 2003.
[3] P. Fieguth and D. Terzopoulos, "Color-Based Tracking of Heads and Other Mobile Objects at Video Frame Rates". *IEEE Int'l Conf. on CVPR*, 1997.
[4] A. Hampapur *et.al.*,, "Multi-Scale Tracking for Smart Video Surveillance", *IEEE T-SP*, Vol.22, No.2, 2005.
[5] IntelliVision, http://www.intelli-vision.com/.
[6] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", *IJCV*, 2004.
[7] ObjectVideo VEW, http://www.objectvideo.com/products/ vew/capabilities/.
[8] C. Stauffer and W. Grimson, "Learning Patterns of Activity Using Real-Time Tracking", *IEEE T-PAMI*, 1999.
[9] Y-L. Tian, M. Lu and A. Hampapur, "Robust and Efficient Foreground Analysis for Real-time Video Surveillance", *IEEE Int'l Conf. on CVPR*, 2005.
[10] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features". CMU Tech Report CMU-CS-91-132, 1991.
[11] Verint, http://www.verint.com/video_solutions/.
[12] P. Viola and M. Jones, "Robust Real-time Object Detection", *IJCV*, 2001.
[13] A. Yilmaz, O. Javed and M. Shah, "Object Tracking: A Survey", *Computing Surveys of ACM*, Vol.38, No.4, 2006.



Object Tracking

Edited by Dr. Hanna Goszczynska

ISBN 978-953-307-360-6
Hard cover, 284 pages
Publisher InTech
Published online 28, February, 2011
Published in print edition February, 2011

Object tracking consists in estimation of trajectory of moving objects in the sequence of images. Automation of the computer object tracking is a difficult task. Dynamics of multiple parameters changes representing features and motion of the objects, and temporary partial or full occlusion of the tracked objects have to be considered. This monograph presents the development of object tracking algorithms, methods and systems. Both, state of the art of object tracking methods and also the new trends in research are described in this book. Fourteen chapters are split into two sections. Section 1 presents new theoretical ideas whereas Section 2 presents real-life applications. Despite the variety of topics contained in this monograph it constitutes a consisted knowledge in the field of computer object tracking. The intention of editor was to follow up the very quick progress in the developing of methods as well as extension of the application.