Interactive Motion Analysis for Video Surveillance and Long Term Scene Monitoring

Andrew W. Senior[1], YingLi Tian[2], and Max Lu[3]

[1]Google Research,
76 Ninth Ave, New York, NY 10011
andrewsenior@google.com

[2]Department of Electrical Engineering
The City College, City University of New York,
160 Convent Ave., New York, NY 10031
ytian@ccny.cuny.edu

[3]IBM Global Technology Services,
17 Skyline Drive, Hawthorne, NY 10532
maxlu@us.ibm.com

Abstract. In video surveillance and long term scene monitoring applications, it is a challenging problem to handle slow-moving or stopped objects for motion analysis and tracking. We present a new framework by using two feedback mechanisms which allow interactions between tracking and background subtraction (BGS) to improve tracking accuracy, particularly in the cases of slow-moving and stopped objects. A publish-subscribe modular system that provides the framework for communication between components is described. The robustness and efficiency of the proposed method is tested on our real time video surveillance system. Quantitative performance evaluation is performed on a variety of sequences, including standard datasets. With the two feedback mechanisms enabled together, significant improvement in tracking performance are demonstrated particularly in handling slow moving and stopped objects.

Keywords: video surveillance; slow-moving and stopped object tracking; motion analysis; interaction; background subtraction.

1 Introduction

Automatic video surveillance is a rapidly expanding field, driven by increases in the affordability of technology and the perceived need for security. Demand and the constrained domain make it one of the most commercially viable application areas for computer vision technology. Many applications in the field require the tracking of moving objects (usually people and vehicles), so that events (such as entering a secure zone) can be detected or those objects can be found through a search interface.

In most automatic surveillance systems, objects of interest are first detected, usually by background subtraction (BGS) which will find moving objects [2, 3, 17]. Detected objects are then tracked by a tracking module [1, 4]. Most surveillance video

analysis systems operate in a feed-forward manner to pass detections from background subtraction to the tracker and then tracks are stored or processed further, for instance by behavior analysis modules. Such a system provides an efficient mechanism for detecting moving objects, but practical implementations suffer from a number of limitations when exposed to particular conditions (lighting variations, weather, heavy occlusion, crowding, non-rigid objects). A rich literature attempts to deal with each of these problems. In this paper, we concern the problems that arise in scenes with slow moving objects and where objects stop for significant periods of time. In particular these scenes challenge the fundamental assumption of a strict differentiation between foreground and background, and the pragmatic choice of using motion, or its proxies, to distinguish between the two. A given object may change from foreground to background and vice versa. For instance, a moving car may park and for all practical purposes needs to be treated as "background" — at least until it starts moving again.

Background subtraction algorithms are generally designed to be adaptive to be able to deal with scene changes (changing lighting; backgrounds whose appearance changes, such as trees and water; static objects). However, a slow moving, or stopped, object can lead to just such repeated observations, and result in the object being adapted piecemeal into the background. This leads to errors in tracking, as the object dissolves into multiple fragments, and false "ghost" fragments appear where the background contains the object after it moves away.

The tracking process usually treats groups of pixels collectively, as unitary objects, and this higher-level information derived by the tracker can be used to inform the process of background subtraction. The tracker explicitly models the objects, whose behaviors are subject to physical constraints (such as rigid motion) in ways different to the physical constraints that control the appearance of individual pixels. Many object tracking techniques focus on handling occlusions but neglect how to track slow moving or stopped objects. Boult *et al.* [4] describe a system that performs well at detecting slow moving objects.

There have been a few systems that have investigated the possibility of feedback from the tracker to the background subtraction module. In order to improve the robustness and efficiency of background subtraction methods, some papers [3, 5, 6] introduced feedback from the frame level and some papers employed the feedback from the tracker [7–11]. Abbott *et al.* [7] proposed a method to reduce computational cost in visual tracking systems by using track state estimates to direct and constrain image segmentation via background subtraction and connected components analysis. Harville [8] used application-specific high level feedback (frame level, person detector and tracker, and non person detector) frame to locally adjust sensitivity to background variation. Senior [12] suggests recalculating the background and foreground segmentation using the model of the tracked object after the background subtraction stage. Wang *et al*. [11] proposed a unified framework to address detection and tracking simultaneously to improve the detection results. They feed the tracking results back to the detection stage.

The interaction between the tracking and background subtraction can also be used to improve the tracking of the slow moving and stopped objects. Venetianer *et al.* [13] examine a way of pushing foreground objects into the background and vice versa. Yao and Odobez [14] use a similar layered background mechanism to remember

stopped objects. Taycher *et al.* [15] proposed an approach that incorporates background modeling and object tracking to prevent stationary objects fading into the background. Our approach is most closely related to that of Pnevmatikakis and Polymenakos [9], who to overcome the problem of stationary targets fading into the background, propose a system combining a mixture of Gaussians background subtraction algorithm and a Kalman tracker in a feedback configuration. They control the learning parameters of the background adaptation on a pixel level in elliptical regions around the targets based on the tracking states from the Kalman tracker. A smaller learning parameter was used for a slow moving object. However, this mechanism will fail when the targets stay stationary for a long period. They will gradually fade into the background even with very small learning parameters.

In contrast, we create two feedback mechanisms that allow the tracker to suppress background updating for slow moving objects that are being tracked. Further, we introduce an active, tracker-driven, object-level healing process where whole objects are pushed to the background to solve the challenges in tracking caused by the stopped objects.

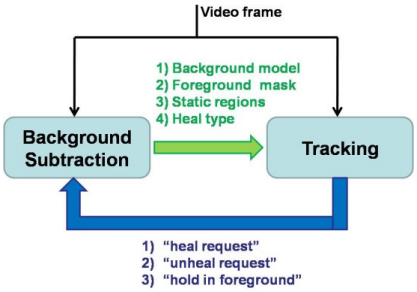


Fig. 1. Diagram of the interaction of background subtraction and tracking, showing the passing of metadata messages.

2 Interaction between BGS and Tracking

2.1 Feedback Mechanisms for Interactions between BGS and Tracking

In order to improving tracking accuracy, we create two feedback mechanisms that allow interactions between BGS and tracking. The feedback required to handle slow and stopped objects are implemented by adding information to metadata of tracking observations which are accessible by BGS processing through following three requests: 1) "heal request"—tracking requests BGS to push the region back to background model; 2) "unheal request" "—tracking requests BGS to convert the background model of a healed region back to that before the heal happened; 3) "hold in foreground"—tracking requests BGS to hold a region without updating. Figure 1 shows the diagram of the interaction between BGS and tracking with the passing of metadata messages.

2.2 BGS Adaption Suppression for Tracking Slow Moving Objects

Slow Moving Objects Tracking Problem: Slow moving objects can present a significant problem to conventional background subtraction algorithms. In multiple Gaussian mixtures based BGS algorithms [2, 5], on which many current systems are based, each pixel is modeled by a mixture of Gaussians distribution. Observations of a pixel's color are assigned to the closest mode of the mixture, or to a newly created mode (replacing the least observed previous mode). The most frequently observed mode is considered the "background" mode, and observations matching that are considered to be background. Other values are flagged as foreground. When an object moves slowly or stops, any pixel may fall on the object for many frames and, if it is of a consistent color, that pixel will eventually be considered background. If multiple pixels are affected in the same way, parts of the object will be considered to be background and the object will progressively be "lost".

Previous systems have partially addressed this problem by detecting groups of foreground pixels that are being adapted into the background, and actively push the whole group in to the background [5]. Here, however the problem is that the detection may come only after some pixels have already been adapted into the background, and may only affect part of the object. Thus, while the switch to background is no-longer independent for each pixel, it may still occur in several fragments, and results in part of an object being background and part being foreground.

BGS Adaption Suppression: To deal with this situation, we institute a feedback mechanism that allows the tracker to suppress background updating for slow moving objects that are being tracked. When the tracker detects a slow-moving object (based on conditions of centroid movement $\leq$ 3 pixels in 0.5s; number of observations > 30, and no recent splitting behavior), it flags the object observations as "slow moving" and the background subtraction algorithm suppresses the adaptation in the region where the slow moving object was observed (as indicated by a mask passed in the metadata).

Typically adaptation will already have been carried out by the background subtraction (as the video frame was received), although some algorithms may wait until the video frame processing has completed. According to the algorithm used, adaptation is suppressed in the region of a slow moving object by copying pixels from a copy of the model saved before adaptation, or by carrying out the inverse operation on those pixels (for instance decreasing the observation counts).

Suppressing adaption in this way has the effect of maintaining the tracked object in the foreground, and uses object level information from the tracker — that the pixels belong to a known object that is moving slowly and has been reliably detected and tracked for some period — to which the background subtraction module by itself does not have access.

A drawback of this mechanism is that it inhibits the process by which false alarm foreground objects are removed. For instance a shadow or a reflection which appears but is tracked for a while, would ordinarily quickly be forgotten as the background model adapts, but, if the "hold in foreground" method engages then these objects can be preserved indefinitely. However, the following mechanism can prevent this from happening

2.3 Tracking-based BGS Healing for Stopped Objects

Stopped Objects Tracking Problem: Stopped objects lead to a different problem, and a dilemma for the design of a tracking system. Background modeling needs to adapt to changes in order to ignore "irrelevant" changes such as lighting. In a simple adaptive background subtraction system, when an object stops, as with slow moving objects above, then it will become part of the background and cease to be tracked. However the object is still present in the scene, and for some purposes (for instance the query "show me all cars present at 3p.m.") the system needs to explicitly represent that presence. A further problem is that since background subtraction algorithms typically operate independently on each pixel, then different pixels of the object will be declared background at different times, resulting in a progressive fragmentation as the object is incorporated into the background.

Fig. 2. Selected frames demonstrate ghosting. The car starts in the background and moves forward, leading to multiple foreground fragments and ultimately a large "ghost" or "hole" where it had been covering up the "true" background by using mixture of Gaussians BGS method.

When a static object starts moving, the background subtraction algorithm detects difference regions around the edges of an object, and as the original background is revealed, those pixels are detected as "foreground regions" and a "ghost" of revealed background is detected as foreground along with the true moving object, as shown in Figure 2. Toyama *et al.* describe this as the "waking person" problem, and conclude that it is not solvable in a self-contained background subtraction module. This presents several challenges to a tracking algorithm: (1) the object appears as many small foreground fragments; (2) the growing object is made up of a moving component and a static region; (3) the true object eventually separates from the static "ghost" region. Some background subtraction methods explicitly tackle this problem [3].

Tracking-based BGS Healing: With the adaptation-inhibition described in Section II.B, slow moving and stationary objects are not adapted into the background at all, so healing and fragmentation are no longer a problem. However static objects will now be held indefinitely in the foreground. As a parking lot fills slowly with cars, the number of "tracked" objects increases and their interactions and mutual occlusions become progressively more complex and unmanageable.

Consequently, we introduce an active, tracker-driven, object-level healing process where whole objects are pushed to the background. In this process, the tracker tracks

whole objects and monitors their movement. When an object is stationary for a sufficient period (dependent on the scene context, for example dependent on the amount of activity in the scene and typical behaviors — whether objects stop for long or short periods) then the tracker determines that the object can be pushed to the background. The tracker sends a "heal request" message to the background subtraction algorithm, including a foreground mask indicating which pixels belong to the object.

On receiving the message, the BGS algorithm takes the selected pixels and adjusts the background model so that the currently observed pixels become categorized as background. The original contents of the region's background model are sent back to the tracker in a "heal" message. The heal message can also incorporate a categorization of the region, indicating whether it looks like a foreground object or a hole, based on integral of the edges in the object perimeter. On receiving the heal message, the tracker can optionally keep the track in a suspended state, ready to be reactivated if the object moves again. Alternatively (if the region was classified as a "hole") the entire track can be discarded as a false positive.

In this manner, stopped objects are quickly pushed to the background and cease to need active tracking. This reduces the complexity of the tracking problem since fewer tracked objects leads to fewer occlusions and difficult tracking situations, and also reduces the computational load by not "tracking" objects once they are stationary.

When the stopped object begins to move, the background subtraction will detect motion in the region and generate one or more foreground regions in or around the object. Any otherwise unexplained foreground region is compared to the stack of suspended tracks and if a matching track is found it is popped. The background subtraction module is sent an "unheal" request, with the old, stored background appearance, which is pushed into the background model, causing the entire object to again be detected as foreground in the following frame, and thus avoiding the ghosting of Figure 2.

Depending on the scene and typical behavior, the suspended "parked" tracks can be maintained indefinitely or forgotten when too old, invalid or too numerous. A grocery parking lot with rapid turnover may warrant keeping the suspended tracks until a car moves again, but an airport lot where cars are parked for days will not. Lighting changes can lead to significant changes in the background appearance while a stopped object is present, and make the stored background patches invalid. The age of a suspended track may also be of interest — for instance picking out parking violations or understanding parking behavior.

This layered approach will also fail in complex environments. Consider an oblique view, looking along a row of vehicles in a parking lot. As vehicles come and go, many different foreground layers will obscure a particular pixel, and the background exposed by an object's departure may be different from the background that was covered by its arrival. A more complex management of layers is imaginable for this scenario, but was not thought likely to be robust.

3 Interaction Mechanisms Implementation

In this section we describe a publish-subscribe architecture that supports the feedback mechanisms described above. The system processes video through a number

of self-contained modules that are linked together through a publish-subscribe framework. Each component receives and transmits metadata packets exclusively through a "first-in, first-out" queue of messages managed by the framework. A metadata packet is taken from the front of the queue and is offered in turn to each of the components for processing before being discarded. While a component is processing a piece of metadata, it may add result metadata to the end of the queue. Most metadata packets are ignored by most components, and many packets will only be relevant to one other "downstream" component, but the architecture allows for considerable flexibility for broadcasting and feedback mechanisms in addition to a simple pipeline model. Components are able to request a priority, which allows the correct ordering of processing for metadata that is processed by multiple components.

The publish-subscribe system encapsulates the functionality of each component and allows for great flexibility in customizing processing on each channel of video, independently selecting one or more detection, tracking and classification algorithms and allowing optional components, such as camera stabilization or performance analysis modules to be added.

In practice the system processes multiple channels of video on a single machine, and each channel is handled by a single "engine" operating in a separate thread but with all engines managed by a single framework. The framework thus scales automatically to multiple processors, and can also handle load-sharing onto embedded coprocessors. The architecture also makes some processing amenable to pipelining of video frames (*e.g.* running BGS on one frame while tracking is executing on the previous frame in a separate thread), though the feedback mechanisms complicate this. Network relaying of selected metadata between processors permits multi-camera operations on a distributed system such as camera hand-off and multi-camera compound alerts.

The framework initiates processing of video by sending a "grab frame" message, which is handled by the video source, which responds by putting a video frame onto the queue. Where appropriate the first component may be stabilization which compensates for motion of the camera (from vibration, wind or active control) and can suspend other processing operations when motion is too great. The background subtraction algorithm operates on the video frame and outputs a foreground mask to the back of the queue. References to the video frame are held by all the components that will subsequently need it, but most other components require further metadata to begin their processing.

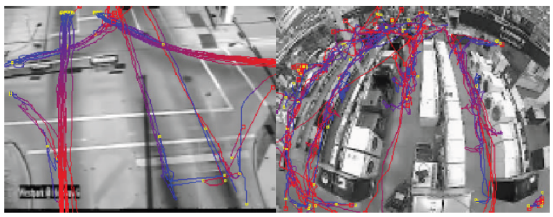
The tracker can begin processing when it receives the foreground mask, and it outputs a variety of result metadata, including "track start", "track end" and "track observation". Subsequent plug-ins such as object classifier, color classifier and alert detector all process the output of the background subtraction and tracker, and finally the index writer plug in sends information to be stored in a database.

Before issuing another "grab frame" message, the framework will issue an "end of sample" message to allow components to clean up before the next frame.

As shown in Figure 1, the feedback required to handle slow and stopped objects has been implemented by adding "heal request" and "unheal request" metadata to the original architecture. "hold in foreground" was implemented by adding a flag to the existing "track observation" metadata which were previously ignored by the background subtraction system, but are now acted upon when flagged in this way.

4 Experimental Results

The feedback mechanism of interactions between BGS and tracking is tested and evaluated on our surveillance system. The quantitative evaluation is performed on a set of six video sequences include four videos from the PETS2001 dataset [16] of cars and pedestrians crossing a university campus (about 2800 frames each) and two our own sequences: a top-down view of a four way intersection with cars stopping and waiting for a traffic light to change (Figure 3(a)) and an overhead view of a retail store taken through a fish-eye lens (Figure 3(b)).



(a) (b)

Fig. 3. Camera views of the test data with tracker output. (a) the traffic intersection (3400 frames, 64 tracks), (b) the store view (3100 frames, 23 tracks). The paths of object centroids are shown, fading from blue to red from start to finish.

The feedback mechanism was tested using simple tracking performance metrics comparing the tracker output to hand-labeled ground truth. The ground truth for each sequence consists of bounding boxes drawn around each object approximately every 30 frames, with labeling to associate a particular object's bounding boxes over time. Since the task requires tracking, evaluation is based on track-level rather than BGS level.

The performance analysis processing matches each ground truth track to the tracker's outputs by comparing the distance between the object centroids at each frame (linearly interpolating between the sparse ground truth points), with hysteresis. When at any time t, an object lies close to a ground truth track (within r, here 20, pixels) then the tracks are considered to match for the entire period around t where the tracks lie within $2r$ pixels. Trivial matches (where the match interval between an output track and ground truth track is a subset of the match for another output track, for instance when two tracks cross) are removed.

The track matching was verified to correctly match intervals of output tracks to ground truth tracks. The performance tool produces a variety of statistics, including the number of false positives (output tracks not corresponding to any ground truth track) and false negatives (ground truth tracks that have no corresponding output track); the "underrepresentation"—the proportion of ground truth track frames with no correspondence in an output track (e.g. because the object was not detected); and the fragmentation — the average number of output tracks matched to each ground truth track (because of gaps in detection, or identity confusion during occlusions).

Quantitative analysis results of performance on six sequences of video, from PETS 2001 and two proprietary datasets for particular scenarios, are shown in Table 1. The comparison between experimental results and ground truth averaged across all the six sequences shows that there is a 39% reduction in false negatives (ground truth tracks

that are not matched in the tracker output) with a 2.7% increase in the number of false positives (tracker output tracks that do not match any ground truth).

Table 1. Tracking performance results on 4 sequences from the PETS2001 dataset and two other datasets. "Under" is the percentage of ground truth frames missing and "Frag" is the average number of tracks matched to a ground truth track.

Sequences	Without feedback		With feedback	
	Under%	Frag	Under%	Frag
PETS D1 C1	21.2	1.56	8.1	1.22
PETS D1 C2	27.8	1.36	12.3	1.36
PETS D2 C1	20.3	1.93	17.6	2.27
PETS D2 C2	8.1	1.50	4.6	1.50
Intersection	33.7	1.04	24.3	1.00
Retail Store	13.1	2.38	14.5	1.90

Errors come from a variety of sources: (1) objects that are too small to be detected, particularly in the store and PETS sequences D1C2 and D2C1 which have distant objects labeled; (2) in the intersection sequence several cars are in the scene at the beginning and ghosting effects mean that their tracks are not matched. (3) Failure to resolve occlusions correctly leads to multiple matches for some ground truth tracks.

Qualitative results are shown in Figure 4. This shows how the interaction between BGS and tracking prevents adaptation and fragmentation of the slowly moving and stopped vehicles, and prevents "ghosts" when they move away.

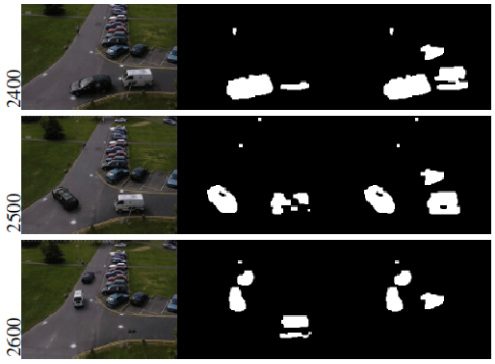


Fig. 4. Selected frames from a PETS2001 video sequence and corresponding foreground regions, demonstrating BGS adaptation without feedback from tracking (middle column) and with the feedback mechanisms (right column). Note the fragmentation (fr.2500) and ghosting (fr.2600) on the middle column. The central stopped car is lost on the middle, but maintained on the right.

5 Conclusions

The two feedback mechanisms for handling slow moving and stopped objects work together to improve the results (in terms of underrepresentation, fragmentation and

false negatives, with a small increase in false positives) on the tested sequences. The inhibition of background updates for tracked objects successfully prevents slow moving and stopped objects from being absorbed into the background. This inhibition interferes with existing healing mechanisms and requires the addition of the "active healing" controlled by the tracker. With the two mechanisms enabled together, the system shows significant improvement in tracking performance, particularly in the proportion of ground truth tracks that are detected and in reduced fragmentation

References

1. J. Connell, A.W. Senior, A. Hampapur, Y.-L. Tian, L. Brown,and S. Pankanti, "Detection and tracking in the IBM People-Vision system", in IEEE ICME, June 2004.
2. C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking", in Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999.
3. K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance", in Proc. IEEE International Conference on Computer Vision, vol. 1. 1999.
4. T. Boult, R.J. Micheals, X. Gao, and M. Eckmann, "Into the woods: Visual surveillance of non-cooperative and camouflaged targets in compex outdoor settings", Proceedings of the IEEE, vol. 89, no. 10, pp. 1382–1402, October 2001.
5. Y. Tian, M. Lu, and A. Hampapur, "Robust and Efficien foreground Analysis for Real-time Video Surveillance", Computer Vision and Pattern Recognition, 2005.
6. O. Javed, K. Shafique, and M. Shah, "A hierarchical approach to robust background subtraction using color and gradient information", in IEEE Workshop on Motion and Video Computing, 2002.
7. R. Abbott and L. Williams, "Multiple target tracking with lazy background subtraction and connected components analysis", Tech. Rep., University of New Mexico, June 2005.
8. M. Harville, "A framework for high-level feedback to adaptive, per-pixel, mixture-of-Gaussian background models", European Conference on Computer Vision. vol. 3, 2002.
9. A. Pnevmatikakis and L. Polymenakos, "Kalman tracking with target feedback on adaptive background learning", in Workshop on Multimodal Interaction and Related Machine Learning Algorithms, 2006.
10. S.-C. Cheung and C. Kamath, "Robust background subtraction with foreground validation for urban traffic video", EURASIP Journal of Applied Signal Processing, Special Issue on Advances in Intelligent Vision Systems, 2005.
11. J.X. Wang, G.N. Bebis, and R. Miller, "Robust video-based surveillance by integrating target detection with tracking", Computer Vision and Pattern Recognition, 2006.
12. A. Senior, "Tracking with probabilistic appearance models", in Third International workshop on Performance Evaluation of Tracking and Surveillance systems, June 2002.
13. P. Venetianer, Z. Zhang, W. Yin, and A. Lipton, "Stationary target detection using the objectvideo surveillance system", Advanced Video and Signal-based Surveillance, 2007.
14. J. Yao and J.-M. Odobez, "Multi-layer background subtraction based on color and texture", in Proc. IEEE Conference on Visual Surveillance, 2007.
15. L. Taycher, J.W. Fisher III, and T. Darrell, "Incorporating object tracking feedback into background maintenance framework", in IEEE Workshop on Motion and Video Computing, 2005.
16. http://www.cvg.rdg.ac.uk/PETS2001/. PETS 2001 Benchmark Data,
17. Robert Collins, Alan Lipton, Hironobu Fujiyoshi, and Takeo Kanade, "Algorithms for cooperative multisensor surveillance," Proceedings of the IEEE, Vol. 89, No. 10, 2001.