Event Detection Using an Attention-Based Tracker

G. Dalley, X. Wang, and W.E.L. Grimson

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
{dalleyg,xgwang,welg}@csail.mit.edu

Abstract

In the last decade, significant progress has been made on developing automatic farfield tracking systems. The PETS 2007 dataset provides some interesting event detection challenges. In this paper, we describe a method of bootstrapping a background model in the presence of significant lighting changes. We then use a blob tracker as an attention mechanism for finding tracks of interest, which we temporally extend using the meanshift algorithm. Using only weak human and luggage models (based purely on size), our system performs well at detecting loitering events, and several events involving interactions between actors and their luggage.

1. Introduction

Police and security officials are increasingly faced with potential threats that can be at least partially mitigated through surveillance of key physical assets such as seaports, airports, transportation hubs, and government buildings. By automating portions of the surveillance network, officials may broaden the range of events they can track and detect in an effort to increase security.

Beginning in the late 1990s, some key advances were made by researchers such as Stauffer and Grimson [19] and Haritaoglu et al. [10] that allowed the construction of real-time farfield visual tracking systems on commodity hardware. Work has continued to progress in handling more challenging low-level situations such as variable lighting and dynamic backgrounds. With those improvements, researchers have been able to also make progress on higher-level tasks such as directly detecting events of interest.

For the PETS 2006 workshop [1], a standardized dataset was created to evaluate various automatic visual event detection systems. Seven videos were taken from each of four calibrated cameras overlooking a train station platform. Each video set recorded a left-luggage event that the automatic visual surveillance systems were expected to detect. Specularities, shadows, a partially-reflective glass surface, and mutual occlusions from multiple actors provided varying levels of difficulty depending on the camera views used and the individual staged scenarios.

Figure 1: *Actor entering the abandoned-luggage warning zone:* In this image (frame 1120, camera 3, clip *S08*), we mark the location of a dropped piece of luggage with a green dot. The owner has just left the $2m$-radius area about the bag defining a warning zone indicated by the yellow circle. After remaining outside the red circle ($3m$) for more than 15 seconds, an abandoned luggage alarm should be generated.

Nearly all of the accepted papers used background subtraction and/or motion detection to first identify foreground blobs. del-Rincón et al. [7] used multiple time scales and feedback loops to improve robustness. Lv et al. [15] and Grabner et al. [9] used the background subtraction results for static object detection and then used and/or learned a classifier for humans.

Auvinet et al. [2] took the foreground blobs and projected them onto the groundplane, looking for multiple silhouette intersections from the four cameras. Li et al. [14] used a layered model to track objects across time. Lv et al. used Kalman filters on the human classifier output, falling back to meanshift [6] when necessary. del-Rincón et al. used an unscented Kalman filter to track the owners of discovered static objects while Smith et al. [18] used a full MCMC sampler. Krahnstoever et al. [13] and Auvinet et al. showed results using nearest neighbors for their data asso-

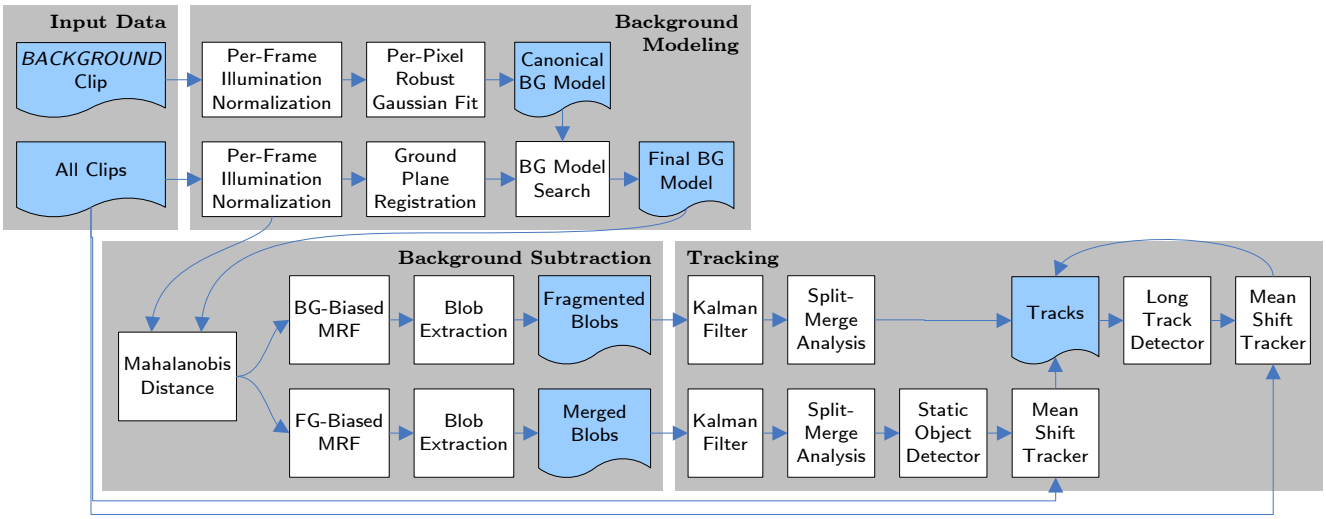

Figure 2: This diagram outlines the primary data processing steps in our tracking pipeline. The motivation behind and implementation of each block is described in §2–§4.

ciation.

This PETS 2007 workshop is staged similarly to the 2006 workshop, adding several interesting real-world challenges: (a) more inter- and intra-clip lighting changes, (b) a changing mixture of harsh and soft shadows, (c) camera movement between clips, (d) denser pedestrian traffic, (e) lower effective resolution in a key camera view,[1] and (f) a broader set of events to detect.

The dataset consists of 10 video clips from each of four calibrated PAL cameras (25fps). The *BACKGROUND* clip is a 1000-frame clip with sparse pedestrian traffic, provided to allow for algorithm training, as needed. The remaining clips have durations of 2750 to 4500 frames. *S00* is a 4500-frame control sequence in which none of the defined events occur. *S01* and *S02* were designed to contain one staged loitering event each under easy and hard conditions, respectively. For this dataset, loitering is defined as remaining in the scene for more than 60 consecutive seconds. *S03* and *S04* contain easy and hard staged luggage retrieval events where a group of two people enter and a bag is placed on the ground. Both members of the couple stay near the bag and then later the second person picks up the bag and the couple leaves together. *S05* and S06 each contain an example of theft, where someone other than the owner picks up a bag that was placed on the ground by the original owner. In *S07* and *S08*, a person drops a bag and "abandons" it by moving more than 3m away for more than 15s (see Fig. 1)[2].

[1] In the PETS 2007 data, the most overhead view (camera 3) recorded interlaced video, whereas in PETS 2006, the camera for the best view was progressive scan.

[2] Originally, the time period was 25s, but it was later changed to 15s because the actors did not stay away from their luggage for strictly more than 25s

Before proceeding to describe our method, we note a few details. Our approach is purely monocular and we use camera 3 (see Fig. 1) exclusively because it offers the viewpoint with the fewest inter-human occlusions. Because this camera's video was interlaced, we subsample the input video without smoothing down to 360×288. Our algorithm is based on generic blob tracking techniques that are readily implemented and require little training and tuning relative to ones that use strong human appearance models. For all tunable parameters, we used the same settings for all clips.

We have implemented a tracking system to detect this workshop's events. Similar in spirit to the work of del-Rincón et al. in PETS 2006, our system is attention-based. Using background subtraction (our attention mechanism), we identify (a) likely dropped luggage and (b) long spatially isolated human tracks. When dropped luggage appears, we perform a local spatio-temporal search for the human owner. Humans identified by long tracks or by association with luggage then have their tracks temporally extended via meanshift. Our system is able to accurately detect nearly all of the events that occur in the dataset with no false positives, including some actual loitering events that were omitted from the official ground truth.

Our processing pipeline is illustrated in Fig. 2. In §2, we discuss our background modeling approach and why the traditional approach is insufficient for this dataset. We then detail our foreground/background segmentation algorithm in §3 and tracking in §4. Our event detection rules are described in §5 and its results are given in §6. We summarize our system in §7.

2. Background Modeling

The first step in our processing pipeline is background modeling. Our goal in this stage is to build, for each pixel in each clip, a model of the appearance of the static elements in the scene. The most common approaches to solve this type of problem are to adaptively model the background as a mixture of Gaussians (c.f. Stauffer and Grimson [19]) or using a kernel density estimate, as done by Mittal and Paragios [17]. Unfortunately these approaches cannot be used directly for the PETS 2007 datasets because they make the fundamental assumption that the most common color modes for each pixel correspond to the background: the clips recorded for PETS are short and many have loitering events where one or more people remain in nearly the same location for almost the entire clip. In fact, in *S02*, some pixels view the background less than 10% of the time.

A natural approach would be to use the *BACKGROUND* clip for training a standard background model to be used directly and without adaptation in the test clips. *BACKGROUND* is indeed mostly background for all important pixels, but its lighting is significantly different from all of the other clips, as we will see when we discuss Fig. 3. For most of the clips, the overall illumination is lower, and the locations of bright highlights on the walls and floor move. Additionally, camera 3 was moved slightly for clips *S05* through *S08*.

An alternative to starting with a foreground/background segmentation is to simultaneously learn the appearance and extent of all objects and the background. Unfortunately, full layered model alternatives either make modeling assumptions that are too strong for this dataset and are very computationally expensive [11, 20] or they bootstrap off background subtraction [14, 21]. Also, most existing implementations are inappropriate for very busy scenes with very frequent occlusions and changes to layer interleaving over time.

2.1. Implementation

We now give the details of our background modeling implementation as outlined in Fig. 2.

As already mentioned, one of the challenges of this dataset is non-constant illumination, as we now illustrate in Fig. 3. Even when just observing the mean intensity of the scene, we can see large differences over time within many clips as well as large differences between clips. The lighting effects we observe are consistent with natural lighting during a partially-overcast day with slowly passing cloud-cover. To reduce these effects, we illumination-normalize each video frame, by using the color $\tilde{c}_{i,t}$,

$$\tilde{c}_{i,t} = \Sigma_t^{-\frac{1}{2}}(c_{i,t} - \bar{c}_t), \quad (1)$$

instead of the original RGB value, $c_{i,t}$, where i is the pixel location, t is the current frame number, $\bar{c}_t$ is the mean RGB color vector for the frame, and Σ_t is the frame's diagonal RGB covariance matrix. The remainder of the background modeling is done independently for each pixel.

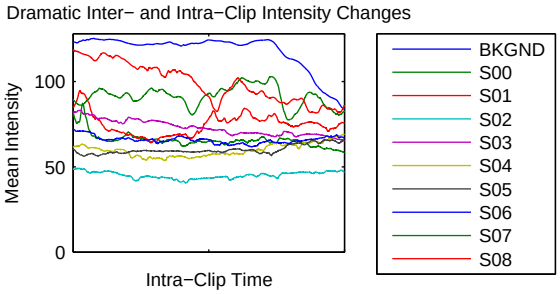

Figure 3: Here we plot the average intensity (on a scale of 0 to 255) of each frame over time, for each video clip. We note that there are significant inter-clip changes (e.g. *BACKGROUND* vs. S02) and intra-clip changes (e.g. *BACKGROUND* and *S08*).

Since all pixels of interest in the *BACKGROUND* clip are sampled from the background distribution nearly all of the time, we fit a single joint Gaussian distribution to each pixel's illumination-normalized RGB value. To add a small measure of robustness to foreground pixels that are present, we discard any pixels with a Mahalanobis distance greater than 5 from the Gaussian and then refit the distribution to the inliers. To avoid overfitting, if any eigenvalue in the covariance matrix is less than 0.002, we round it up to that value. Minimal tuning was used to select these parameters. The output from this step is a canonical background model for each pixel location.

If the lighting changes were more mild, we could directly use our canonical model in all clips; however, our illumination normalization is insufficient to be used directly in these clips. To illustrate the issue, refer to Fig. 4, where we examine the observed colors when a given pixel views foreground objects nearly 90% of the time. For that pixel in clip *S02*, we have manually labeled all frames as foreground or background, and observed the normalized color distributions. The black set of axes represent the canonical background model. We can see that that model poorly represents the hand-labeled background distribution (red X's), but it is much closer to the true background mode than to any modes from the foreground distribution (blue dots).

In essence we wish to track the background's color distribution between clips. We assume (a) that the background is well-modeled by a Gaussian, (b) that the Gaussian distribution's mean and covariance shift slowly over time, and (c) that the background distribution is distinctive from the modes in the foreground distribution. Given these assumptions, we wish to find the Gaussian mode in the new clip's

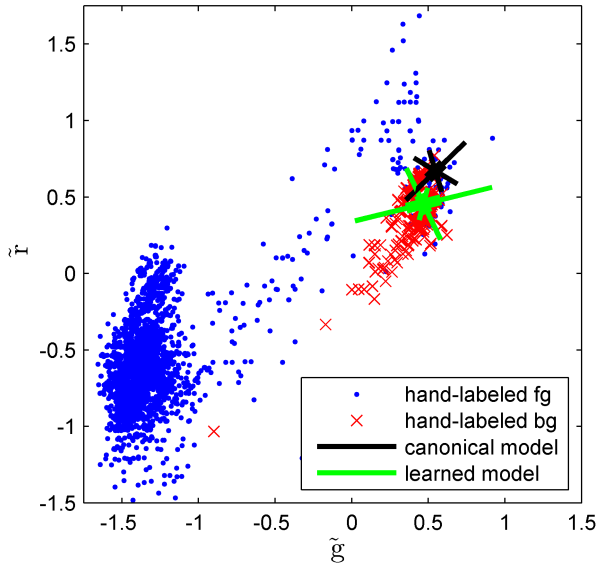

Figure 4: *Background model adaptation for a challenging case:* For pixel $(253, 319)$ in the *S02* clip, we labeled all frames in which the pixel was viewing the background versus any foreground object. Color values when the pixel was viewing background are shown as red X's in a scatter-plot (projected onto the red and green axes). Blue dots represent observed background color samples. For this pixel, the Gaussian distribution learned for the *BACKGROUND* clip is shown as a black-colored set of principle axes. The green axes represent the learned model after it was adapted to this clip. If we construct a Gaussian classifier from the learned model for this pixel, the area under the ROC curve is 0.9926 (not shown). The distribution shown here is a particularly challenging case as the pixel is viewing foreground objects nearly 90% of the time.

distribution that is closest to the canonical model, as suggested in the previous paragraph. In doing so, we wish to be agnostic to the observed foreground distribution and only concern ourselves with finding the closest color mode.

We can simplify our search by warping our illumination-normalized pixel values so the canonical model is represented as an origin-centered unit normal distribution:

$$\tilde{\tilde{c}}_{i,t} = \Sigma_{BG,i}^{-\frac{1}{2}}(\tilde{c}_{i,t} - \bar{c}_{BG,i}) \quad (2)$$

where we search for a mode in the $\tilde{\tilde{c}}_{i,t}$ space, given the canonical model's center $\bar{c}_{BG,i}$ and covariance $\Sigma_{BG,i}$.

We then iteratively slide a spherical template of radius 5 to the mean location of the sample points that fall within the template in a meanshift-like loop, until convergence[3].

[3]The template radius was tuned by selecting a single predominantly-foreground pixel location in each of two test videos and performing a quick ROC analysis.

We then fit a Gaussian distribution to the samples that fall within the final template boundaries and limit the covariance eigenvalues as we did for the canonical model. The output from this step is a single-Gaussian background model, $\left(\hat{c}_i, \hat{\Sigma}_i\right)$, for each pixel, tuned for each video clip, and defined in the illumination-normalized RGB space.

For the PETS 2007 dataset, we have used the method just described to track the background distribution from the *BACKGROUND* clip to other clips. We have found this novel implementation to be effective and to require minimal tuning.

3. Background Subtraction

Given a statistical model of the background, we can perform likelihood tests to classify sampled pixels as foreground or background, as is standard practice. Markov Random Fields (MRFs) are an effective mechanism for applying spatial smoothing priors to a label field [8] instead of relying on a purely independent thresholding at each pixel. Our MRF optimizes the following objective function:

$$E(l) = \sum_{\{i,j\}\in\mathcal{N}} V_{i,j}(l_i, l_j) + \sum_{i\in\mathcal{P}} T_i(l_i) + \sum_{i\in\mathcal{P}} D_i(l_i) \quad (3)$$

where l is the field of foreground-background labels, $\mathcal{P}$ is the set of pixel sites, $\mathcal{N}$ is the 8-neighborhood graph, $V_{i,j}(l_i, l_j)$ encourages spatially neighboring pixels to have the same label, $T_i(l_i)$ encourages pixels to have the same foreground/background label they had in the previous frame, and $D_i(l_i)$ encourages pixels to be labeled as foreground when they do not match the background model well. These energy terms are defined as

$$V_{i,j}(l_i, l_j) = t_n \delta\left(l_i, l_j\right) \quad (4)$$

$$T_i(l_i) = t_t \delta\left(l_i, l_i'\right) \quad (5)$$

$$D_i(l_i) = \begin{cases} t_f, & \text{if } l_i = \text{fg;} \\ \sqrt{\left(\tilde{c}_{i,t} - \hat{c}_i\right)^T \hat{\Sigma}_i^{-1} \left(\tilde{c}_{i,t} - \hat{c}_i\right)} & \text{otherwise,} \end{cases} \quad (6)$$

where $\delta(\cdot,\cdot)$ is the Kronecker delta function, t_n is the spatial mismatch potential, t_t is the temporal mismatch potential, l_i' is the label assigned to pixel i in the previous frame, and t_f is the foreground label potential. The background potential (the "otherwise" case) in $D_i(l_i)$ is the Mahalanobis distance from the learned background model for the given pixel in the given clip.

We have used an existing MRF implementation that uses the fast two-label *st*-cut implementation of Boykov, Veksler, and Zabih [4, 5, 12] with the the temporal smoothness terms suggested by Migdal [16]. For other background subtraction work, we have found $t_t = t_n = 5$ and $t_f = \log_2(256^3) \approx 16.6$ to be good default values. We

Figure 5: *An Extreme FG/BG Segmentation Example:* Here we show segmentation results for the foreground- (left) and background-biased (right) MRFs for frame 500 of clip *S08*. The frame shown here is a somewhat extreme example of the different solutions found by the two MRFs. Both MRFs use the Mahalanobis distance values visualized in the middle image as the background label potential. For display purposes, the intensity is saturated at a distance of 64, twice the foreground potential for the background-biased MRF (so a 50% gray pixel sits on the classification threshold, absent any neighborhood effects).

have used those t_t and t_n values without any tuning. We next discuss how we chose t_f.

Instead of choosing a single value for t_f for the PETS dataset, we construct pair of MRFs to address two different detection tasks. One is biased to produce more foreground labels ($t_f = 8$) and other is biased to produce more background labels ($t_f = 32$). These values were chosen empirically by starting with the baseline value of $t_f \approx 16.6$ and observing foreground/background classification results on a handful of frames with a few different settings.

As demonstrated in Fig. 5, the foreground-biased MRF (left) tends to capture all of the foreground into coherent blobs at the cost of mislabeling shadows as foreground and joining independent objects. On the right, we see the same frame segmented with the background-biased MRF. Its silhouettes are cleaner and distinct, but camouflaging effects have caused some objects to be split into multiple blobs.

When attempting to detect dropped luggage, we wish to obtain robust detections of relatively small isolated objects. The foreground-biased MRF resists camouflaging effects of humans that result in fragmented blobs. Thus when a small foreground blob is present in its solution, there is a high likelihood that it was actually the result of a small object, not a blob fragment from a human.

For the foreground-biased MRF segmentations, we extract blobs using a standard 4-connected neighborhood connected components extractor. Any blobs that have fewer than 75 pixels are discarded. Before evaluating the MRF, we mask out regions of the image corresponding to the ledge on the bottom left of camera and corresponding to the "British Airways" sign and above on the wall. No humans can move here and these areas are especially susceptible to lighting changes in some clips. We also reject any blobs that are very near the edges of the image because they are unreliable for tracking purposes. The 75-pixel threshold was chosen with minimal tuning and could be made more robust by scaling it by the square of the estimated distance to the camera using its calibration. For this dataset, we did not find it necessary to take this extra step.

When attempting to detect individual humans who are present in the scene for a long time, tracking at a blob-level only works well when the blobs are disjoint. With the density of human traffic in the PETS 2007 dataset, our foreground-biased MRF tends to group multiple people into a single blob very frequently and it also mislabels shadows as foreground. By biasing a second MRF to prefer background labels, we obtain clean silhouettes which can be tracked more easily, at the cost of needing to be robust to some blob fragmentation.

For the background-biased segmentations, we extract blobs by grouping any 4-connected blobs that are within 10 pixels of another blob. The 10 pixel dilation diameter was chosen by observing differences in the MRF segmentations in a handful of frames from two clips.

The output of the background subtraction is a collection of segmented foreground blobs from each MRF. We keep both sets of blobs separately since they are redundant and optimized for different tracking tasks (dropped luggage versus human tracking).

4. Tracking

After extracting the foreground blobs, we need to do enough tracking to detect the desired events: loitering, luggage abandonment, and theft. These events rely primarily on (a) tracking and maintaining the identity of humans who remain in the scene for a long time, (b) detecting luggage placed on the ground, (c) identifying who owns a piece of dropped luggage, and (d) identifying those who pick up luggage. Because the clips all have many actors who occlude each other, simple blob tracking without appearance information cannot succeed. One approach is to explicitly build strong models and attempt to track all humans, as was done by Grabner et al. and Lv et al. for PETS 2006. We have chosen instead to build an attentional system that identifies (a) proactive opportunities to build robust models and (b) times when more extensive tracking are desired.

Our first tracking submodule takes the background-biased foreground blobs and performs standard Kalman tracking on them, using a constant velocity model on the blob's centroid in the image plane. During data association, we independently associate each track with the blobs in the current frame. If multiple tracks are associated with a single blob, we initialize a new track to follow the merged group as long as it remains coherent. This track is tagged as containing a group of actors. We similarly detect tracks that split into multiple blobs. Although we did not find it nec-

essary for this dataset, it is possible to use trajectory and/or appearance information to disambiguate mild to moderate split-merge graphs using approaches such as that of Bose et al. [3]. At greater computational cost, an MCMC tracker such as the one used by Smith et al. [18] could be used as well. For this dataset, we used a simplified version of the implementation of Bose et al.

Our second tracking submodule identifies loitering candidates. When we observe a non-group track that lasts more than 16 seconds and whose a mean blob area is between 1500 and 3000 pixels (in a 360×288 frame), we consider this track as a good candidate. We use the pixels in that person's tracked blob to learn a color histogram appearance model for that person. We then use meanshift tracking [6] to temporally extend the track both temporally forward and backward through occlusions, dropouts, and merge-split events. We stop temporally extending meanshift tracker when the Bhattacharya distance,

$$BC(p,q) = \sqrt{1 - \sum_{c \in [0,15]^3} \sqrt{p(c)q(c)}} \qquad (7)$$

exceeds some threshold, $\tau_{meanshift} = 0.14$, where c is an RGB color value, p is the the color histogram model, and q is the color histogram of the pixels in the putative object location. Our color histograms have 16^3 bins. We have used an existing meanshift tracker implementation.

If a tracked object disappears at a scene boundary, we terminate the tracking, but we retain the meanshift model. This allows us to reacquire targets that reenter the scene soon after leaving.

The third tracking submodule performs Kalman tracking on the foreground-biased blobs. Humans walking near each other are often grouped in the same blob with this tracker, so it is less effective at identifying individuals. It is however robust to long-term camouflage effects with humans. This means that a given person rarely produces a track fragment that represents only a small portion of their body for more than a few frames at a time. When people drop luggage and leave it on the ground for an extended period of time, they often are segmented separately from the dropped luggage for a second or more at a time. This presents us with an opportunity to detect these static objects. When we see a track that with 200 to 1000 pixels, we hypothesize that it is a piece of dropped luggage. As was done with the long isolated human tracks, we initialize a meanshift tracker using appearance information from the initial track. We then extend the track temporally in both directions until significant movement ($\tau_{ds} = 50$ pixels) is observed. This gives us the drop-off and pickup times.

Given a drop-off and pickup time and location for a piece of luggage, we would like to identify the individual who has initiated each event. If a human-sized blob overlaps the bag's meanshift template area when it moves by more than τ_{ds} pixels from its starting position, we assume that that this person is moving the bag. If we were searching back in time with meanshift, this blob is the original owner (or victim in the case of theft), and when searching forward in time, the blob is the new owner (or thief).

To disambiguate theft, luggage swaps, luggage that always stays near the owner, and dropped luggage, we need to track the original and new owners. If either of these people has already been associated with a loitering track, no extra work is needed. Otherwise, we use a new meanshift tracker for each person to discover their long-term trajectory.

Any time we have two temporally-separated full tracks (those which we extended with meanshift trackers), we compare the color histograms that were used for the meanshift tracking. If the Bhattacharya distance is less than $\tau_{reacquire} = 0.15$, we consider the two tracks to belong to the same person.

5. Event Detection

As described in the previous section, the attentional tracker is configured to output the primary pieces of information required for event detection. We are able to translate the events from image coordinates to real-world coordinates by assuming the middle of the bottom of the blob or meanshift tracker's bounding box is touching the ground. Using the supplied camera calibration, we are able to infer the world coordinates of this point.

Any individual tracks that have a human-like size and aspect ratio and exist for more than 60 seconds trigger loitering events. If the owner of a piece of luggage travels more than 2 meters away from their luggage, a warning is triggered, and if they stay more than 3 meters away for more than 15 or 25 seconds (depending on the scenario, as indicated in the ground truth), an abandoned luggage alarm is triggered. If a new owner removes a piece of luggage beyond the 3 meter radius, a theft alarm is triggered after 15 or 25 seconds, as appropriate for that clip. If the new and original owners both exit the scene together, we have chosen to output a reattended alarm.[4]

As the events of interest are precisely and deterministically defined for this dataset, our system attempts to directly detect the required conditions for alarms.

6. Results

In this section, we briefly summarize our results on the PETS 2007 dataset. We first note that we had no false positive events for any of the clips.

For *S00*, there were no events that took place (and our system raised no alarms).

For *S01*, the first loitering clip, there was a single loitering event. Using our meanshift tracker, we were able to

[4] In our XML results, we encode this alarm as a LeftLuggage/Theft alarm pair where the original owner and the thief share the same track ID.

Clip	TP	FN	Temporal Error	Subjective Difficulty
S00	0/0/0	0/0/0	- / - / -	**
S01	1/0/0	0/0/0	5.08s / - / -	**
S02	1/0/0	0/0/0	1.44s / - / -	***
S03	2/0/0	1/0/0	8.22s / - / -	**
S04	1/0/0	2/0/0	0s / - / -	****
S05	1/1/1	1/0/0	19.2s / 0.08s / 0.08s	**
S06	0/0/0	0/0/1	- / - / ∞	****
S07	0/1/1	0/0/0	- / 0.12s / 0.08s	**
S08	0/1/1	0/0/0	- / 0.2s / 0.12s	****

Table 1: *Results:* TP are true positive and FN are false positive event alarms. In the PETS 2007 dataset, events of interest have a well-defined start time (e.g. a loitering alarm should sound the moment a person has been in the scene for 60 consecutive seconds). The temporal errors reported here are the time we raised an alarm minus the time we should have raised it, according to the supplied ground truth. For the TP, FN, and Temporal Error columns, we give three values. The first is for loitering events, the second is for left luggage events, and the third for theft or reattended luggage events. We had no false positives. The subjective difficulty was defined by the PETS 2007 organizers.

track the loiterer back to 5 seconds after the actual scene entry. For *S02*, the second loitering clip, we achieved excellent results.

In each of *S03* and *S04*, a couple swapped a piece of luggage. Since the luggage was never unattended, according to the PETS definition, only loitering events occurred. For *S04*, we detected the time correctly, to the exact frame.

In each of *S05* and *S06*, a theft occurs. Because these are very busy scenes, especially while the event in question is occurring, we fail to track the second member of the victim couple in *S05* and we are not able to track the couple all the way back to their entrance time (yielding a high temporal error in the detected loitering event). For *S06* the scene is too busy for us to obtain a successful event detection.

In *S07* and *S08*, a person loiters in the scene, drops a bag, exits the scene, then later the same person reenters and picks up the luggage. Using the model trained for meanshift tracking, we are able to not only trigger the left-luggage alarms with high temporal accuracy, but we are also able to detect that it is the same person picking up the luggage.

The loitering events we report in *S03*, *S04*, and *S05* were not in the official ground truth data, but our own manual verification indicates that they did occur.

7. Conclusions

In this paper, we have described a event detection system based on a bootstrapped background subtraction system. We use blob tracking as an attention mechanism and when we identify tracks of interest, we employ meanshift trackers to temporally extend tracks, find related tracks, and associate tracks that are temporally separated. Our system performs well on the PETS 2007 dataset.

Acknowledgments

This work was funding in part by the Singapore national government.

References

[1] *International Workshop on Performance Evaluation of Tracking Systems (at CVPR 2006)*, number 9. IEEE, June 2006.

[2] Edouard Auvinet, Etienne Grossmann, Caroline Rougier, Mohamed Dahmane, and Jean Meunier. Left-luggage detection using homographies and simple heuristics. In *PETS Workshop*, pages 51–58. IEEE, 2006.

[3] Biswajit Bose, Xiaogang Wang, and Eric Grimson. Multiclass object tracking algorithm that handles fragmentation and grouping. In *CVPR*. IEEE, 2007 2007.

[4] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9), September 2004.

[5] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. In *Pattern Analysis and Machine Intelligence*, volume 23, pages 1222–1239. IEEE, November 2001.

[6] D. Comaniciu, V. Ramesh, and Peter Meer. The variable bandwidth mean shift and data-driven scale selection. In *International Conference on Computer Vision*, volume 1, pages 438–445. IEEE, 2001.

[7] J. Martínez del Rincón, J. E. Herrero-Jaraba, J. R. Gómez, and C. Orrite-Uru nuela. Automatic left luggage detection and tracking using multi-camera ukf. In *PETS Workshop*, pages 59–66. IEEE, 2006.

[8] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 564–584, 1987.

[9] H. Grabner, P. M. Roth, M. Grabner, and H. Bischof. Autonomous learning of a robust background model for change detection. In *PETS Workshop*, pages 39–46. IEEE, 2006.

[10] Ismail Haritaoglu, David Harwood, and Larry S. Davis. W4: Real-time surveillance of people and their activitis. *Pattern Analysis and Machine Intelligence*, 22(8):809–803, August 2000.

[11] Nebojsa Jojic and Brendan Frey. Learning flexible sprites in video layers. In *CVPR*. IEEE, 2001.

[12] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *PAMI*, May 2002.

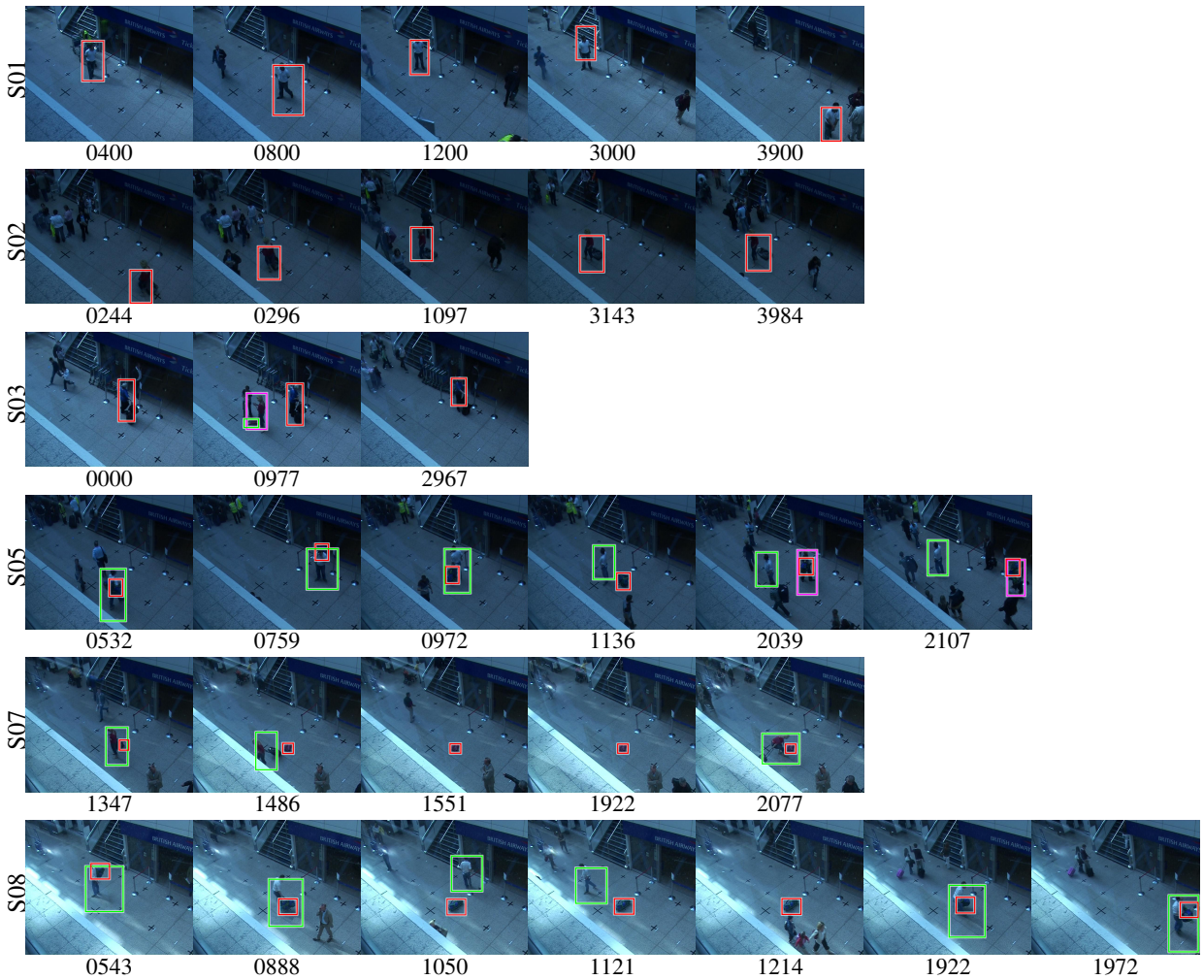

Figure 6: *Selected tracking results.* Here we show meanshift tracker bounding boxes for objects involved in some of the detected events. In *S01* and *S02*, we track loiterers (red boxes in each). In *S03*, we track a loiterer (red), a piece of luggage (green), and the bag's owner (magenta). In *S05*, we track a bag (red), its original owner (green), and the thief (magenta). In *S07* and *S08*, we track an abandoned bag (red), and its owner (green) who retrieves it later.

[13] N. Krahnstoever, P. Tu, T. Sebastian, A. Perera, and R. Collins. Multi-view detection and tracking of travelers and luggage in mass transit environments. In *PETS Workshop*, pages 67–74. IEEE, 2006.

[14] L. Li, R. Luo, R. Ma, W. Huang, and K. Leman. Evaluation of an ivs system for abandoned object detection on pets 2006 datasets. In *PETS Workshop*, pages 91–98. IEEE, 2006.

[15] F. Lv, X. Song, B. Wu, V. K. Singh, and R. Nevatia. Left luggage detection using bayesian inference. In *PETS Workshop*, pages 83–90. IEEE, 2006.

[16] Joshua Migdal. Robust motion segmentation using Markov thresholds. Master's thesis, MIT, Sept. 2003.

[17] Anurag Mittal and Nikos Paragios. Motion-based background subtraction using adaptive kernel density estimation. In *CVPR*. IEEE, July 2004.

[18] K. Smith, P. Quelhas, and D. Gatica-Perez. Detecting abandoned luggage items in a public space. In *PETS Workshop*, pages 75–82. IEEE, 2006.

[19] Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, pages 246–252. IEEE, 1999.

[20] Christopher K. I. Williams and Michalis K. Titsias. Greedy learning of multiple ojbects in images using robust statistics and factorial learning. In *Neural Computation*, volume 16, pages 1039–1062. MIT Press, May 2004.

[21] Yue Zhou and Hai Tao. A background layer model for object tracking through occlusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1079–1085. IEEE, 2003.