Exploiting Contextual Data for Event Retrieval in Surveillance Video

Philip Kelly
CLARITY: Centre for Sensor Web Technologies
Dublin City University
Dublin 9, IRELAND
kellyp@eeng.dcu.ie

Ciarán Ó Conaire
CLARITY: Centre for Sensor Web Technologies
Dublin City University
Dublin 9, IRELAND
oconaire@eeng.dcu.ie

Noel E. O'Connor
CLARITY: Centre for Sensor Web Technologies
Dublin City University
Dublin 9, IRELAND
oconnorn@eeng.dcu.ie

ABSTRACT

Contextual information is vital for the robust extraction of semantic information in automated surveillance systems. We have developed a scene independent framework for the detection of events in which we provide 2D and 3D contextual data for the scene under surveillance via a novel fast and convenient interface tool. In addition, the proposed framework illustrates the use of integral images, not only for detection, as with the classic Viola-Jones object detector, but also for efficient tracking. Finally, we provide a quantitative assessment of the performance of the proposed system in a number of physical locations via groundtruthed datasets.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*indexing methods, abstracting methods*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*query formulation*; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding—*video analysis*; I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*tracking*

Keywords

Surveillance, retrieval, event detection

1. INTRODUCTION

Automated surveillance has received much attention in the research community in recent literature [11, 6, 4, 14]. Its goal is to reduce the burden on operators by assisting them in retrieving relevant events and gathering statistical information automatically, instead of requiring hours of video to be viewed.

As specific goals of various applications differ, so too does the type of event required to be detected. However, for many flexible applications the exact event detector algorithm *cannot* be hard-coded into the system framework as either; (1) the event definition is dependent on an undefined scene; or (2) the event is itself undefined [14]. The first scenario is typical of many surveillance applications as although the event required to be detected is known (for example, in an automated pedestrian traffic light system the event may be to detect static pedestrians in a designated area waiting to cross the road), contextual information about the scene (e.g. the *exact* area where pedestrians tend to wait to cross) is unknown and will tend to vary depending upon the camera positioning and other scene specific properties. To this end, we have developed a flexible system for the detection of events that is independent of the scene under consideration. As such, the algorithmic techniques employed within the system are designed to be independent from the camera location and underlying scene structure.

We present two key contributions in the area of event retrieval in surveillance video. Firstly, we have developed an interface tool that allows a variety of 2D and 3D scene-specific contextual data to be easily provided by the operator to the system. This contextual data facilitates the improvement of the semantic inference of a variety of user-defined events by individually tailoring them to a variety of differing scene specific scenarios. Secondly, due to the large volume of data that must be managed in surveillance scenarios, we focus on real-time processing and exploit the power of integral images to perform fast detection and tracking of people with a given scene.

This paper is organised as follows: In section 2, we review related work in this area. We describe our surveillance event-detection system in section 3, this includes our tool for supplying contextual data in section 3.1. Section 4 provides quantitative experimental evaluation of the proposed system framework in a number of differing areas. Finally, we detail our conclusions and outline our future work in section 5.

2. RELATED WORK

With the large number of surveillance cameras now in operation, both in public spaces and in commercial centres, significant research efforts have been invested in attempts to automate surveillance video analysis. Hu et al. [11] provide a thorough survey on the visual surveillance of object motion and behaviours. In a similar vein, Cucchiara [6] gives an overview of surveillance-related research into combining multiple media streams such as audio, video and other sensors. Both Hu et al. and Cucchiara argue that the use of multiple sensors and additional data streams "will constitute the fundamental infrastructure for new generations of mul-

CIVR '09, July 8-10, 2009 Santorini, GR

timedia surveillance systems". The types of sensors and additional data streams they refer to include thermal infrared [12, 19, 16], use of depth information from stereo cameras [10, 14] and multiple cooperating surveillance cameras [4].

While future surveillance systems may rely on multiple streams of data to perform accurate surveillance event detection, such hardware infrastructure changes will not be widespread in the immediate future. In this paper, we focus on maximising the potential of automated single-camera surveillance, and on providing the means to supply strong contextual information to the surveillance system using a 2D and 3D annotation tool, described in section 3.1.

Secondly, in this paper we leverage the power of integral images to perform robust person detection and tracking in surveillance video. Integral images allow constant time computation of sums of pixels in rectangular areas. They were used by Viola and Jones to compute Harr-features for rapid object detection [22] and by Bay et al. for interest point detection [3] as a fast approximation to the difference of Gaussians operation to find scale-space extrema. In this paper, we describe how integral images, combined with background modelling, can provide very fast and occlusion-tolerant person-tracking.

3. EVENT DETECTION SYSTEM

Figure 1 provides an overview of our proposed event detection surveillance system decomposed into basic sub-systems. The first component (A) consists of a number of low-level detectors, including foreground region and person detectors. In the second sub-system (B), false-positives obtained from the low-level person detector are subsequently filtered using scene specific contextual data to remove false-positives. The third layer within the framework (C) applies the filtered low-level information and contextual data to track pedestrians temporally through a scene. In the final module (D) of our system, events are detected from the movements and interactions between people and/or their interactions with specific areas that have been manually annotated by the user. Each stage within the system framework is tailored to the particular scene under surveillance via user-supplied contextual data.

3.1 Contextual Data Annotation Tool

Within the system framework a variety of contextual data is provided to the sub-systems – some is provided to improve low-level detection and pedestrian tracking performance, other contextual data is provided to tailor event detection algorithms. The annotation tool allows a variety of 2D annotations via traditional point-and-click techniques. These 2D annotations are used in a variety of ways in this work. This includes the generation of training data for a Haar classifier cascade (which is discussed in section 3.2.2) and a linear person height model (outlined in section 3.3) as well as the inclusion of 2D annotated areas into event detection algorithms (see section 4.2). However, in this section we focus on the generation of 3D regions of interest, known as *hotspots* [14].

In order to create 3D hotspot regions, an image-to-groundplane homography must first be created as this can be used to describe both the relationship between the real-world groundplane and a camera's image plane. To obtain this homography, four corresponding points between an input image (such as the four illustrated points in figure 2(a) or the points from a calibration shape) and the corresponding real-world coordinates of these points should be obtained. The software then employs the technique outlined in [9] to obtain the homography.

Once estimated, the image-to-groundplane homography can be used to generate a *plan-view*, or *birdseye-view*, of the scene – see figure 2(b). It is from this viewpoint that 3D hotspot regions can be created via the annotation tool interface. This is achieved by simply circling a region of interest within the plan-view image, for example see the red area in figure 2(c). The resultant hotspot can be seen as a 3D area of interest which can be incorporated into the algorithmic definitions within the proposed system. The 3D properties of the hotspot can be clearly seen when overlaid on a 3D rendering of the scene – see figures 2(d)-(f). It should be noted that in this work the 3D model is employed solely as a visualisation tool – it allows users to examine hotspot annotations from a variety of "more intuitive" angles. In addition, the 3D model facilitates the visualisation of the state of the pedestrian tracking system so a user can examine the perceived positioning and tracking of pedestrians in an intuitive manner. In future work, we hope to extend the use of the 3D model into the system sub-system algorithms, one such example would be to use it for occlusion reasoning between tracked people and static background objects.

These annotated hotspot regions have a variety of uses within the proposed framework. One such example is within the pedestrian tracking sub-system, whereby the 3D position of tracked pedestrians can be constrained to within the area defined by the hotspot. This technique reduces the possibility of persons being tracked erroneously – for example, physically impossible tracks such as persons walking through walls. In fact, the hotspot illustrated in figure 2(c) is employed for this purpose in one set of our experiments in section 4. In addition, our system facilitates the easy application of hotspots to constrain event detectors to filter out specific types of pedestrian tracks – for example, tracks that start-on, start-off, or pass-through a hotspot, or pedestrians that pass through the hotspot within a narrow range of directions, such as only those travelling in a northerly direction. Finally, in our system framework any number of contextual hotspots can be created and incorporated into a specific event detector. If more than one is created, then logical operators can then be applied between them – thus, for example, it becomes possible to obtain all those pedestrians who pass-through multiple hotspots – or conversely, obtain all the tracks that do not.

A sample of possible 2D and 3D contextual annotations are outlined in sections 3.5 and 4.2, all of which can be created quickly and easily within the annotation tool. Before discussing those specific examples, we shall describe in detail each of the four sub-systems outlined in figure 1.

3.2 Low-level Data Extraction

The first sub-system in our framework consists of a number of low-level detectors, including background subtraction and person detectors.

3.2.1 Background Modelling

In typical surveillance scenes, the camera is on a fixed platform, so background modelling can be used to eliminate the stationary pixels and find pixels belonging to moving objects such as people. Our model is similar to the semi-

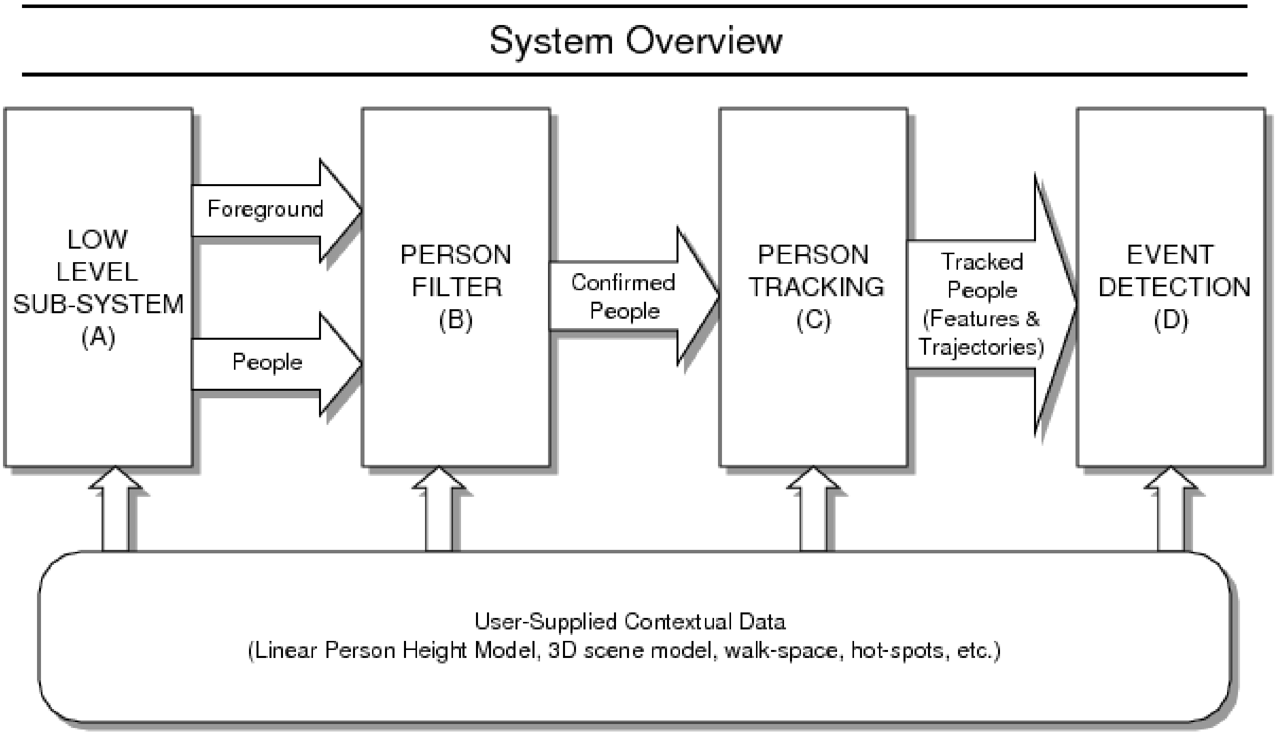

System Overview
LOW LEVEL SUB-SYSTEM (A)
Foreground
People
PERSON FILTER (B)
Confirmed People
PERSON TRACKING (C)
Tracked People (Features & Trajectories)
EVENT DETECTION (D)
User-Supplied Contextual Data
(Linear Person Height Model, 3D scene model, walk-space, hot-spots, etc.)

Figure 1: System Overview

nal work of Stauffer and Grimson [21], except that we use a fixed variance to simplify the model and avoid problems of degeneracy. In our framework, we employ an N-layer background model to obtain moving foreground regions within each video frame. Within this model, each pixel is described by N layers (or colours). In our experiments, we set $N = 4$. In each layer the pixel has an associated colour, a *weight* and the frame number of when the colour was last observed. When a new colour is observed, the model is updated as follows:

- If the colour exists in one of the layers already (with a distance threshold), then the weight of that colour is incremented by one. The position of this layer is swapped with the layer directly above it if its incremented weight is greater, or if the higher colour has not been observed for C frames. In our experiments, we used $C = 1500$.
- If the colour *does not* exist in any of the layers, then the lowest layer is reset (i.e. the weight is set to 1) and initialised with the detected colour.

Using this technique, a pixel's background colour is modelled by each of the colours that appear within the top layers that make up at least 75% of the total weight sum within the layered background model. A pixel is detected as foreground if its colour is not found in these top colours. In our system, the layered background models were initialised using a manually generated background image of the particular scene in question.

Shadow removal.

Shadows and other lighting-changes are frequent in surveillance scenarios [18, 7]. Using the proposed technique of background subtraction, shadows tend to be erroneously detected as foreground pixels. Shadows can result in increased false detections of pedestrians, or lost tracks. Thus, the foreground data obtained was post-processed using a shadow suppression technique. For all foreground pixels, we remove shadow pixels by computing the change in luminance and chrominance. We first convert from (R,G,B) to (L,g,b) using $L = R + G + B, g = G/L, b = B/L$. We then compute the change in luminance and the change in chrominance as follows:

$$L_{dif} = |log(\frac{I_L}{B_L})| \qquad (1)$$

$$C_{dif} = sqrt((I_g - B_g)^2 + (I_b - B_b)^2) \qquad (2)$$

Where I is the current image and B is the background image (lowest layer in the background model). If the following inequality holds, then the pixel is set as shadow (i.e. removed as a foreground pixel):

$$L_{dif} + \alpha \times C_{dif} < \beta \qquad (3)$$

These parameters were learned using manually labelled foreground and shadow pixels (In our data: $\alpha = 18$, $\beta = 0.5$). Adaptive approaches are also possible, such as using a general shadow detector and exploiting unlabelled pixels to improve the model by co-training [13].

3.2.2 Person Detection

To detect people we use the OpenCV object detector framework [2]. It exploits integral images to compute Haar features and quickly detect rectangular regions that appear to

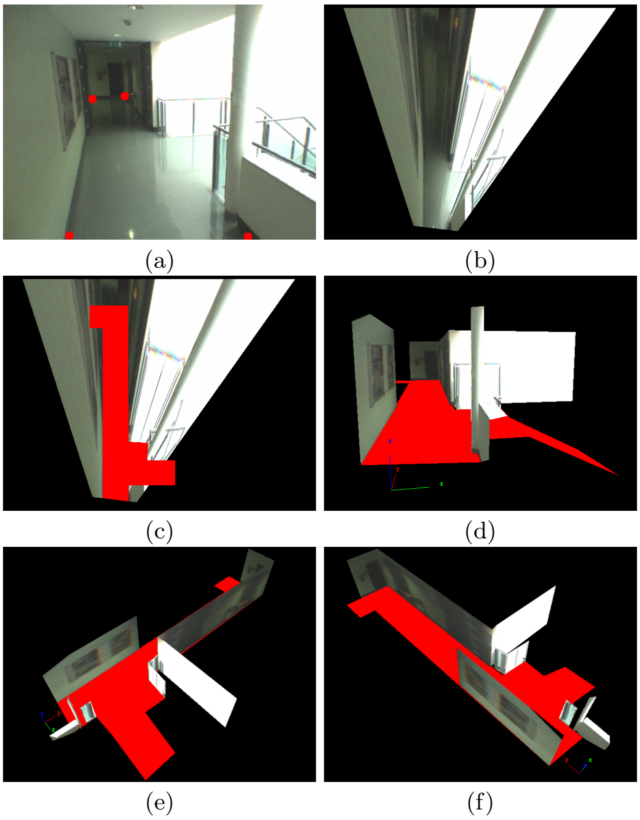


Figure 2: Contextual Data; (a) Annotated Input Image; (b) 2D plan-view image; (c) Hotspot; (d)/(e)/(f) 3D Visualisation of Hotspot.

contain people. Training the detector involves supplying manually segmented positive examples (regions containing a single person) and negative examples (images with no people) from training data via the contextual data annotation tool. Negative sample images can also be provided automatically by sampling random areas of the scene image with no foreground activity. Positive samples can also be obtained from public datasets (such as the MIT pedestrian dataset [17]) or using online learning [8]. In our approach, the training consisted of two iterations; (1) the person detector was initially trained using solely positive examples; (2) the second stage consisted of one iteration of active-learning. This second stage worked as follows. After the first version of the detectors were trained, they were then run on a large batch of test images consisting of both positive and negative examples. All detections registered by the detectors were then manually classified as true-positives or false-positives via the annotation tool. The detectors were then retrained using these new samples.

3.3 Filtering Person Detections

The second sub-system layer within our framework, represented by B in figure 1, filters out false-positives obtained from the low-level person detector using low-level information, a variety of scene-specific contextual data and previously tracked pedestrians within the scene. This is achieved using the following filtering techniques; (a) *Foreground filter*: detections that contain less than 25% of foreground are removed – in this process we exploit the use of integral images to compute the amount of foreground in a detected person in constant time; (b) *Height filter*: detections that are too small or too large to be human are discarded – this is achieved by the creation of a simple linear model of an average person's 2D image height as related to their 2D image foot location. This contextual information is created by manually marking the foot and head positions of pedestrians in test data via the annotation tool. The software then fits a line to the data using a least squares error technique; (c) *Foot filter*: detections that are determined not to be standing in the the scenes *walking area* hotspot (such as that illustrated in figure 2) are removed. In addition to these filters, all person detections which overlap significantly with a currently tracked pedestrians are discarded.

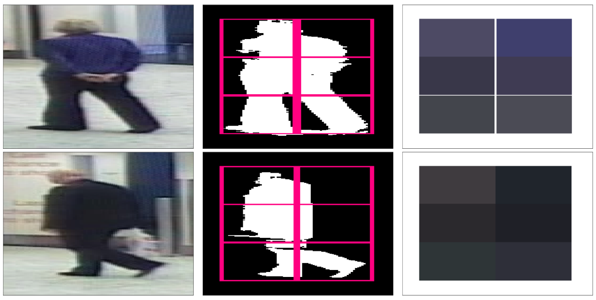


Figure 3: Model of tracked person: original image, foreground with grid overlay and computed foreground colour model.

3.4 Person Tracking

The third layer within our framework, applies the data extracted in the previous sub-systems to track pedestrians temporally through a scene. Tracking is performed in a depth-ordered way, so that people closer to the camera are tracked first. This allows us to infer occlusion for the people further back in the scene. As we do in the detection stage, we exploit integral images to perform very fast person tracking that is invariant to person size.

Each person is represented by a 2×3 colour grid model, illustrated in figure 3. Each part is represented by the average colour of the foreground pixels within it. We use this model for speed and efficiency, as it is very fast to compute using integral images. Tracking is done by exhaustive search in a 21×11 pixel window centred on the last known location of the person. If people nearer the camera occluded people further back, we down-weight parts of the model that are determined to be occluded. Since the lower parts of the body are more likely to be in motion (feet and arms), we also weight the sections of the model differently. We express the similarity between a candidate region T and a person model M as:

$$\phi(T,M) = S_{FG} \times \frac{\sum_{x=1}^{2}\sum_{y=1}^{3}\sum_{c=1}^{3} w_{x,y} exp(-0.5(\frac{D}{\sigma})^2)}{\sum_{x=1}^{2}\sum_{y=1}^{3}\sum_{c=1}^{3} w_{x,y}} \tag{4}$$

where x and y are the column and rows of the model sections, c is the colour channel (in RGB), D is the colour distance in RGB-space, and $w_{x,y}$ is the weight for section (x,y). We set the weight as: $w_{x,y} = u_{x,y}/sqrt(y)$, where

$u_{x,y}$ is the *unoccluded* portion of the section. We set the foreground weighting $S_{FG} = F^2/A$, where F is the number of foreground pixels and A is the area in pixels. We update the model in each frame to account for pose and lighting changes.

In addition to a tracked persons location in the 2D image, we infer their 3D location using their 2D image foot location and the image-to-groundplane homography data described in section 3.1. This information is valuable input to some of the event detectors detailed in sections 3.5 and 4.2, where 3D position and velocity information are required. Additionally, the 3D location is used to constrain the tracking search by ensuring that a person cannot be tracked out of the contextually annotated *walking area* hotspot. This technique not only improves tracking robustness, but decreases computational complexity by eliminating the need to search specific areas of the scene for the continuation of a temporal track. A second improvement to computational complexity lies in the ability to apply the contextual linear height model (outlined in section 3.3) and the 2D location of a person's tracked position to dynamically infer the height in pixels of a tracked pedestrian anywhere within the scene, thereby avoiding a search through different scales.

Finally, each tracked person has an associated confidence value, that tends to zero over time if it's existence is not supported in the data. This confidence can be increased by significant foreground pixel data within a persons bounding box, or a significant overlap with a detection from the low-level person detector, with a greater overlap causing a greater increase in confidence. However, if the confidence becomes too low then the tracked person is removed from the system.

3.5 Event Detectors

As specific goals of various applications differ, so too does the type of event required to be detected. To this end, the final layer in our system provides the framework from which a variety of user-defined events can be declared. In general, events are inferred by the system using information provided by the 2D/3D movements of the tracked people, and/or their interactions with other people and 2D/3D contextual data which has been manually annotated by the user. The 2D contextual data may include regions of interest within an image, and areas of the image where, according to annotated training data, events tend to occur. The 3D contextual data manifest themselves as hotspot regions, which as outlined in section 3.1 can be then used to filter pedestrians based on their 3D statistics such as velocity, location, direction of movement, etc.

To illustrate the use of hotspots within this event detection framework, we will provide an example using our publicly available *Corridor* sequence [15] (see figures 2 and 4 for example images), which comes with ground-truth annotation of people in the scene. The dataset is challenging due to the number of people, frequent occlusions, illumination conditions, and the ability of persons to ascend/descend a staircase on the right hand side of the image. Within this sequence it is known that on one occasion a number of people stand and read a notice board on the left of the scene for an extended period of time. In order to detect this event a hotspot is created in front of the notice board – see figures 4(a) and (b). In addition, the hotspot is set to ignore all pedestrians *except* those that have; (a) been standing on it for 30 consecutive frames; and (b) had a velocity of less than 10cm/sec during that time. In addition, it is set to ignore all times where there is only one person on the hotspot. Using this definition, the event is triggered for a number of frames during the sequence (all within 20 frames of each other). An example event result is illustrated in figure 4(c), with the corresponding 3D model visualisation of the state of the person tracker depicted in figure 4(d).

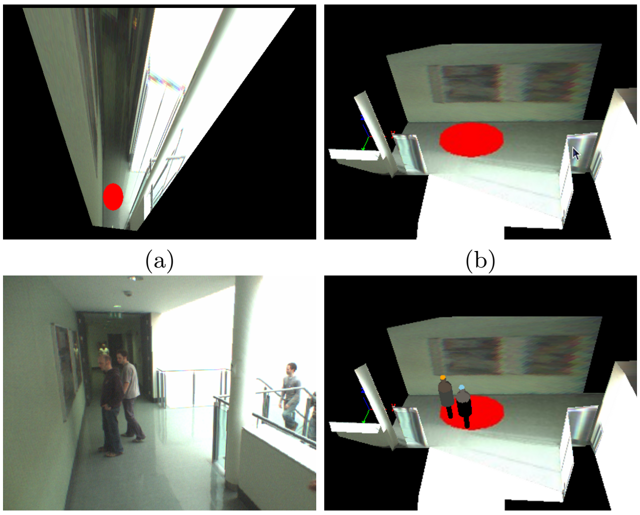

(a)
(b)
(c)
(d)

Figure 4: Waiting Event; (a) Hotspot; (b) 3D Visualisation of Hotspot; (c) Waiting Detected; (d) 3D Visualisation of System State.

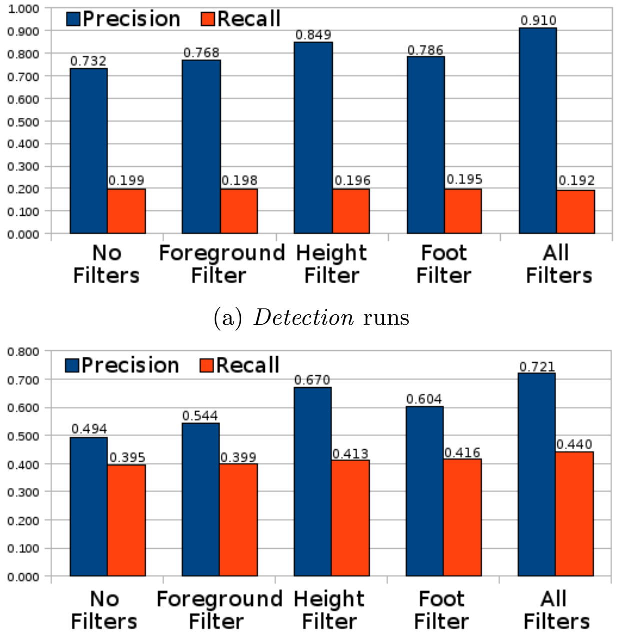
4. EXPERIMENTAL RESULTS

In the following section, we quantitatively evaluate the improvement provided by the contextual person detection filters of section 3.3 with regards to precision and recall using a groundtruthed dataset. In addition, we provide a quantitative indication to the improvement in performance due to the use of integral images. Finally, in section 4.2 it is demonstrated how the framework can be tailored to detect a variety of surveillance video events and we quantitatively evaluate these events via a second manually annotated dataset.

4.1 Quantitative Performance Evaluation

To investigate the performance of our contextual data filters on the person detection and tracking sub-systems, we apply the *Corridor* dataset introduced in section 3.5. In order to train the pedestrian classifier we used samples from the background images of the sequence as negative examples, and obtained positive examples from the MIT pedestrian dataset [17] and from annotated people in the TRECVID Gatwick sequences [20]. In a single frame, a person is determined to be correctly detected if the overlap between the ground-truth is at least 50% of the larger rectangle. Using this dataset we made 10 runs of the system, the first five did not include tracking results (this set is called *Detection*), the second five runs included persons tracked by sub-system D (this set is called *Detection and Tracking*). In each set of 5 runs, the filters outlined in section 3.5 were either: (a) all turned off; (b) individually turned on; or (c) all turned on.

Figure 5 shows the results this experiment. Since we are



(a) *Detection* runs

(b) *Detection and Tracking* runs

Figure 5: Person detection results on *Corridor* sequence

using *filters*, the recall values within the *Detection* runs cannot be increased, and do decrease very slightly, dropping by 0.007. However, the overall effect of the filters was to substantially increase precision by 0.178. The *Detection and Tracking* run, on the other hand, results in a jump of 0.227 and 0.045 in precision and recall figures respectively. These results clearly demonstrate the benefits of using the contextual data to improve semantic inference.

In addition, throughout the final *Detection and Tracking* run (with all filters turned on) we quantitatively evaluated the decrease in computational complexity during the tracking stage due to the use of integral images. It was found that the exhaustive search stage of the proposed tracking technique required over 750 times less operations when using integral images than it would have required to perform when accessing individual pixels via traditional techniques.

4.2 Quantitative Event Detection Evaluation

To evaluate the proposed system's performance to detect relevant surveillance events, we used 50 hours of video (10 hours from 5 cameras – see figures 6 and 8 for some sample views) from the Gatwick airport dataset, made available by the UK Home Office Scientific Development Branch and released as part of the iLids project [1]. Surveillance events within the data were annotated as part of the TRECVID event-detection task of 2008 [20]. For each of the 5 camera views, a person-detector was trained using manually segmented positive and negative example data from the development video datasets. Figures 7 and 8 shows some training examples and results of the trained detector on sample data respectively.

In total, 6 events detectors were created for 5 TRECVID groundtruth annotated events (*ElevatorNoEntry*, *OpposingFlow*, *PeopleMeet*, *Embrace* and *PersonRuns*), as well one event-type that was not annotated, *DoorOpenClose*. All events detectors are outlined below.

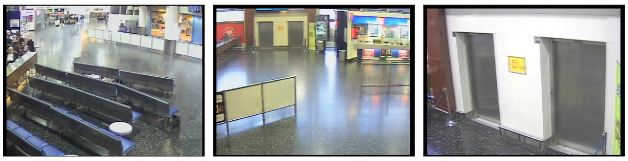


Figure 6: Background images from 3 of the 5 cameras.

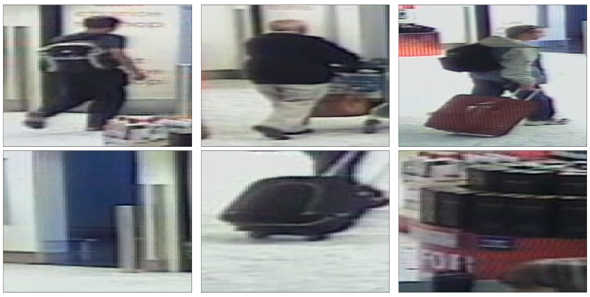


Figure 7: Examples of positive (top row) and negative (bottom row) examples used for training our person detector.

DoorOpenClose.

In order to detect the state of a door (i.e. opened or closed) in a given camera view, 2D contextual data was provided to the detector to indicate regions of the door that significantly changed colour depending upon the door state. The colour differential (the difference of two rectangular region colours) was calculated using integral images to efficiently compute the area sums and simply applied a threshold to determine the state of the door.

ElevatorNoEntry.

To detect when an elevator door opened, but a waiting person did not enter the elevator, two event detectors similar to those outlined in both section 3.5 (see figure 4) and the *DoorOpenClose* event were combined.

OpposingFlow.

To detect people travelling the wrong way through a one way area (such as the doors in figure 9(a)), a hotspot was created (see figure 9(c)/(d)). This hotspot was used to filter out all people in the scene who were on the hotspot for a minimum of 3 frames and whose direction of motion was opposed to the normal flow through that given hotspot area.

PeopleMeet.

This event called for the detection of times when two or more people walk up to one another, stop and communicate. We deemed that communication was too subtle to detect with our system, and as such we created an event detector that triggered when two people, who were far apart (i.e. the Euclidean distance between the 3D locations of

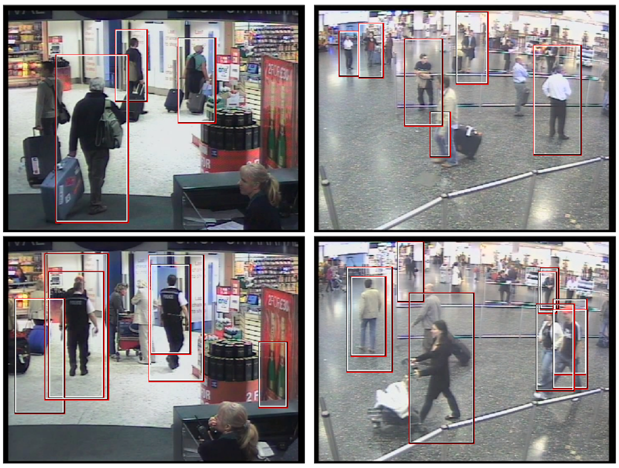


Figure 8: People detected by our person detector in TRECVID data.

two people was greater than a threshold), then come into close proximity (i.e. their distance dropped below a second threshold).

Embrace.

Since an embrace is a difficult semantic concept to detect directly in crowded scenes, we inferred *Embrace* events by taking all detected *PeopleMeet* events and weighting their confidences using a learned prior in the form of a confidence map (an example of such a map is shown in figure 10) created using development data via the annotation tool. In this map, the brighter an area the more likely it is that an *Embrace* event will occur (scaled between 0 and 1). We compute α, the maximum average confidence map value within either person's bounding box (computed efficiently using integral images). The final confidence of the event occuring was deemed to be the product of α and the tracking confidences for the two people. If the final confidence was over a predefined threshold then an event was triggered.

PersonRuns.

This event was detected in each camera using the 3D velocity of tracked persons. If a person's velocity magnitude remains over 150cm per second for 3 consecutive frames, then the system then triggered an event when the person either stopped travelling at that velocity or stepped out of the scene. The confidence of the event was computed as the product of the person's tracking confidence and a sigmoid function of the length of time they were travelling at a high velocity.

4.2.1 *Quantitative Results*

A comparison of our performance, using Detection Cost Rate (DCR) which is a value that consists of a linear combination of missed detections and false alarms, to other TRECVID participants can be seen in table 1. These results are competitive when compared to the results of other event detection systems submitted for TRECVID evaluation.

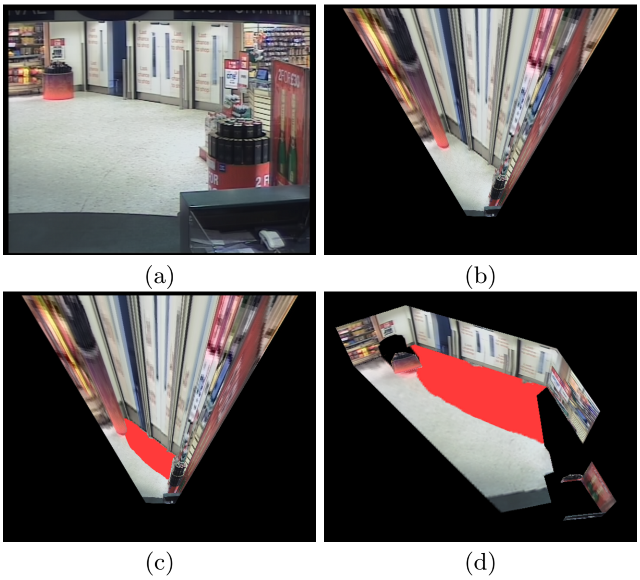


Figure 9: Opposing Flow Event; (a) Background Image; (b) 2D plan-view image; (c) Hotspot; (d) 3D Rendering of Scene with Hotspot.

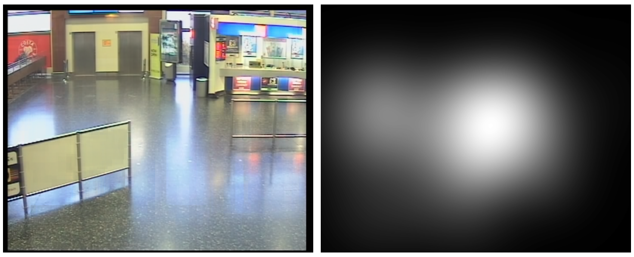


Figure 10: Background image for camera 3 and its confidence map for the *Embrace* event, indicating the likelihood for the event to occur in image space.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we illustrated how contextual information can be supplied very efficiently for single static surveillance cameras using our interface tool. This annotation tool allows the user quickly annotate 2D and 3D information. This data can be used to provide context that improves the detection and tracking of people in our surveillance system and to provide a framework for visualisation and event-based querying. The system we developed exploits the power of integral images to rapidly perform both detection, using the Viola-Jones technique, and also tracking of people, using an integrated colour-foreground model and using occlusion reasoning. The system's performance was illustrated on our own publicly available *corridor sequence* and on the challenging surveillance sequences from Gatwick airport that were part of the TRECVID 2008 dataset [20]. The system performed well on all sets of data, despite the large number of people, frequent occlusions and substantial difference between the camera views.

In future work, we will extend the use of the 3D rendered scene, which is currently being used solely a visualisation tool, to improve algorithms within the framework. One such

Event	Mean DCR	Our DCR	Rank
ElevatorNoEntry	0.702	0.415	5/16
PersonRuns	1.000	0.994	7/22
Embrace	1.014	0.990	1/10
PeopleMeet	1.004	1.000	2/7
OpposingFlow	0.787	0.782	10/23

Table 1: Event detection results on TRECVID data using optimised (min) DCR scores.

approach would be to apply the model to perform occlusion reasoning when a person is tracked behind static objects in the scene (such as the vertical pole to the right hand side of figure 2(a)). In addition, we will investigate the annotation of not only hotspots on the ground but also other areas of the 3D model (such as the walls, etc) and the automatic generation of the *walking area* hotspot from the 3D model. Additionally, another possibility for future research is to improve our person detection using online learning techniques using unlabelled data [5].

Acknowledgements

This work is supported by Science Foundation Ireland under grant 07/CE/I1147.

6. REFERENCES

[1] i-lids datasets. http://www.ilids.co.uk.

[2] Opencv. http://sourceforge.net/projects/opencvlibrary.

[3] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision*, 2006.

[4] S. Calderara, R. Cucchiara, and A. Prati. Multimedia surveillance: content-based retrieval with multicamera people tracking. In *International workshop on Video surveillance and sensor networks*, pages 95–100, 2006.

[5] H. Celik, A. Hanjalic, E. Hendriks, and S. Boughorbel. Online training of object detectors from unlabeled surveillance video. In *Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–7, 2008.

[6] R. Cucchiara. Multimedia surveillance systems. In *ACM international workshop on Video surveillance & sensor networks*, pages 3–10, 2005.

[7] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew. On the removal of shadows from images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):59–68, 2006.

[8] H. Grabner and H. Bischof. On-line boosting and vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 260–267, 2006.

[9] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision Second Edition*. Cambridge University Press, 2003.

[10] M. Harville. Stereo person tracking with adaptive plan-view templates of height and occupancy statistics. *International Journal of Computer Vision*, 22:127–142, 2004.

[11] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man and Cybernetics*, 34(3):334–350, 2004.

[12] G. D. Jones, R. E. Allsop, and J. H. Gilby. Bayesian analysis for fusion of data from disparate imaging systems for surveillance. *Image and Vision Computing*, 21(10):843–849, 2003.

[13] A. J. Joshi and N. Papanikolopoulos. Learning of moving cast shadows for dynamic environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 987–992, 2008.

[14] P. Kelly and N. O'Connor. Vision-based analysis of pedestrian traffic data. In *Workshop on Content-Based Multimedia Indexing*, pages 133–140, 2008.

[15] P. Kelly, N. O'Connor, and A. F. Smeaton. A framework for evaluating stereo-based pedestrian detection techniques. *Transactions on Circuits and Systems for Video Technology*, 18(8):1163–1167, 2008.

[16] C. Ó Conaire, N. E. O'Connor, and A. Smeaton. Thermo-visual feature fusion for object tracking using multiple spatiogram trackers. *Machine Vision and Applications*, pages 483–494, 2006.

[17] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[18] A. Prati, I. Mikic, M. Trivedi, and R. Cucchiara. Detecting moving shadows: Algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):918–923, 2003.

[19] V. Sharma and J. Davis. Feature-level fusion for object segmentation using mutual information. In *Workshop on Object Tracking and Classification in and Beyond the Visible Spectrum*, pages 139–148, 2006.

[20] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *Workshop on Multimedia Information Retrieval*, pages 321–330, 2006.

[21] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, volume 2, pages 246–252, 1999.

[22] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.