Multi-Camera Person Tracking And Left Luggage Detection Applying Homographic Transformation

Dejan Arsić, Martin Hofmann, Björn Schuller, Gerhard Rigoll
Technische Universität München
Institute for Human Machine Communication
80333 Munich, Germany
arsic | schuller | rigoll @tum.de

Abstract

Today video surveillance systems are widely used in public spaces, such as train stations or airports, to enhance security. In order to observe large and complex facilities a huge amount of cameras is required. These create a massive amount of data to be analyzed. It is therefore crucial to support human security staff with automatic surveillance applications, which will create an alert if security relevant events are detected. This way video surveillance could be used to prevent potentially dangerous situations, instead of just being used as forensic instrument, to analyze an event after it happened. In this treatise we present a surveillance system which supports human operators, by automatically detecting abandoned objects and loitering people. Two major parts have been implemented: a multi-camera tracking algorithm based on homographic transformation and the subsequent analysis of the observed object trajectories. An alarm event is triggered if an object is abandoned for 25 *seconds or a person is staying in the view for more than* 60s.

1. Introduction

The increased interest in public safety after recent terroristic activities has caused a rapid growth in the number of surveillance cameras. Traditional operator-based surveillance of public spaces is very labor intensive, due to the number of people involved in analyzing the recorder video material on- and off line. Up to now video surveillance has been used as forensic tool, to obtain knowledge about what has happened. Therefore a fully automated surveillance system seems desirable in order to provide continuous analysis of people's behaviors, discover potential threats in time and eventually prevent dangerous situations. This is not an easy task, as the system has to deal with large crowds resulting in severe occlusion, difficult and fast changing lighting situations, and views that are very narrow or too wide.

The PETS 2007 benchmark data set presents four typical security relevant problems at a a busy airport terminal. First the detection of luggage left unattended for more than 25s. This seems a relevant task for law enforcement, as luggage containing explosives or chemical threats could be left behind. The difficulty in this task is to reliable detect luggage items and additionally determine the owner of the luggage if they have left the item unattended for at least 25. For this problem several approaches have been already described in the PETS 2006 challenge [1]. The second task is the detection of loitering people. A person is considered loitering if she enters a view and stays there at least 60s. As third scenario luggage theft is considered. Theft is defined as an item of luggage moved further than 3 meters away from the original owner. A variation would be two individuals swapping a luggage item as fourth scenario. Whether the initial owner notices this procedure or not should not be an issue, as both scenarios are realistic. For each of the for scenarios two data sets, recorded from 4 camera views, are provided. In this work we will concentrate on the left luggage task and loitering.

Our approach to these problems is basically split into two modules. In the first stage we apply a multi-camera object tracking framework. Object detection is performed in each camera view separately, applying an adaptive foreground segmentation based on Gaussian mixtures. Tracking is then performed centrally in the ground plane, by homographic transformation of the single camera detection results and their fusion. The second stage deals with the analysis of the object's trajectories obtained by the tracking module. It takes the time spend in the scene into account, detects stationary objects and splits and merges in the trajectory. All this information is used to make a decision about a person's behavior.

The paper is structured as followed: The multi-camera object detection and tracking module is described in

sec. 2. The event detection is described in sec. 3. Finally results will be presented in sec. 4 followed by a short discussion sec. 5.

2. Multi-Camera Object Detection And Tracking

This section will describe the process of object tracking, which is first performed in each single camera perspective, followed by a homographic transform into the ground plane. In the end the tracking is performed with the fusioned data from each camera view.

2.1. Object Detection

For object detection in each single camera view we apply a common adaptive foreground segmentation method, based on works presented by Stauffer and Grimson[2]. Each RGB-pixel of the image is modeled by K Gaussian mixtures. This seems reasonable, as each pixel's variance due to noise can be modeled. With $K = 3$ we compute a model for background, foreground and shadow separately. The probability density function for each pixel is given by:

$$f_{X|K}(X|k,\theta_k) = \frac{1}{(2\pi)^{\frac{\pi}{2}}|\sum_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu_k)\sum_k^{-1}(X-\mu_k)}$$

Where X is a vector containing the pixel's RGB values. Each mixture is defined by $\theta_k = \mu_k, \theta_k$. Taking the probability ω_k into account we get the set of parameters: $\Phi = \omega_1, ...\omega_k, \theta_1, ..., \theta_k$. This way each image pixel is represented by:

$$f_X(X,\Phi) = \sum_{k=1}^{K} P(k) f_{X|k}(X|k,\theta_k)$$

In order to assign the observed pixel to the correct kind of surface the term $P(k,|X,\Phi)$ is maximized by applying the bayes rule and the maximum a posteriori criteria:

$$P(k|X,\Phi) = \frac{P(k) f_{X|k}(X|k,\theta_k)}{f_X(X|\Phi)}$$

$$\hat{k} = \arg\max_k P(k|X,\Phi)$$

Up to now each pixel has been modeled independently of its neighborhood and some false positives have been produced due to image noise. By applying morphologic operations such as opening and closing, noise is eliminated and holes within foreground areas are filled. After this step so called Blobs are detected within the image by using connected component analysis [3]. In order to reduce memory and computation effort only the object's shape is stored and processed in future steps.

Some additional robustness and a gain in computational effort have been achieved by masking the input images, so only in relevant image areas object detection has been performed.

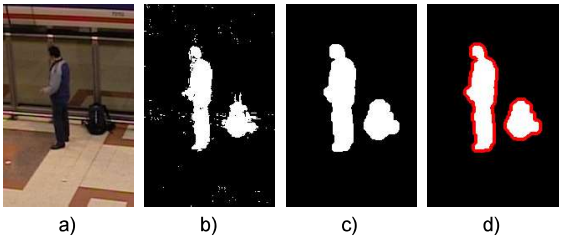

Figure 1: Foreground segmentation, morphologic operations and connected component analysis. Only the shape is required for tracking

The parameter f_α denotes the updatetime in frames and has to be set carefully. A compromise has to be made, as a fast update would model a stationary object as background after a short while, whereas too slow updates result in difficulties with changing lighting situations or if an object from the initial background is removed, the new visible area would be modeled as new object. We chose fast updates in the beginning, as the sequences are rarely empty, and to get slower after a few initialization frames.

2.2. Homographic Transformation in multiple Layers

To compute the exact position of a person in 3D, just the lowest point of a detected blob from a single camera perspective could be used, followed by simple epipolar geometry, to find the line intersecting the ground plane. A major drawback in object detection with one single camera is occlusion handling. Two objects, which are occluding each other only partially, will be recognized as one. Even methods, such as KL Tracking [4] or color histogram based mean shift tracking [5] cannot solve the occlusion problem.

Therefore Khan presented an approach using homographic transformation in [6], which performs a transformation from one image plane into another one. A minimum of seven corresponding points in image and world coordinates have to be known to locate any blob position on the image plane, assuming, that the blob's lowest point is in contact with the ground. Therefore the homographic transformation matrix has to be computed once for every single camera:

$$\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad \text{with } \det H \neq 0.$$

Figure 2: Homographic Transformation of object shapes for all four camera views

Therefore the parameters of the transformation matrix h_{ij} are determined using the Tsai camera calibration method [7]. For experimental results both the given calibration data and a set of new calibration points have been used, as the provided calibration data seemed to contain some errors.
The shape of the previously detected blobs is then transformed for each camera view. Image 2 shows the results of the homographic transformation in all four camera views. As it can be seen the transformations can be interpreted as shadows created by light sources located at the camera positions. The area within the outline of a polygon is considered as candidate for an object, as we cannot state any information about the object's depth and height at the moment. Depth information subsequently is gathered by the fusion of all four given camera perspectives. In the optimum case a boolean operation can be performed on the resulting polygon transforms on the ground plane. In areas with intersections of at least three polygons it can be assumed that a object is located. Obviously there are some errors due to shadows or changing lighting situations. These are not a major issue, as these usually do not appear in all camera views at the same time. Consequently these will be eliminated during the following fusion process. Another strength of the homographic transform and the subsequent fusion is the robustness to occlusion. Each transformed blob denotes the region an object could be located in, even if it is not visible from the actual camera view. Assuming that objects are not occluded completely in every camera view, all present objects can be detected.

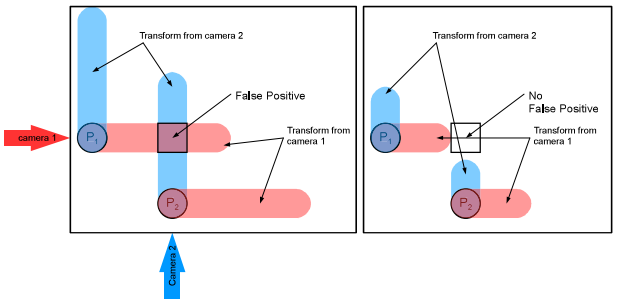

Figure 3: left: a false positive during tracking, right the corrected result

In real world tracking scenarios there are some minor drawbacks, for instance if a walking person lifts one foot it does not touch the ground and an error will be performed during the tracking process. Image 3 visualizes a second error source, here just with two cameras. There are two objects located directly in front of the system's two given cameras. With the fusion of the polygon outlines it could be assumed that there is a third object. So in fact one more camera would be needed to solve this problem. Another possibility is to perform the homographic transform not only in the ground plane, but also in additional layers. For example the transformation to a plane with $h = 1m$ would result in an intersection of a person's hips, just in the same position p as the feet touch the ground. As the blob is virtually moved a bit in the camera's direc-

tion the false intersection will be extinguished if it is a person with an average height of about $1.6m$. This approach is performed for the heights of $0m$, $0.3m$, $0.8m$ and $1.2m$. These values have been experimentally determined, and proved to provide reliable results. An unsolved problem remains if for instance a piece of luggage is located just behind the persons in view. It would be totally occluded and not detected if we apply the layer based approach. Therefore a decision has to be made if false positives or misses are to be preferred. In a practical environment false positives would probably result in a low acceptance of the implemented system.

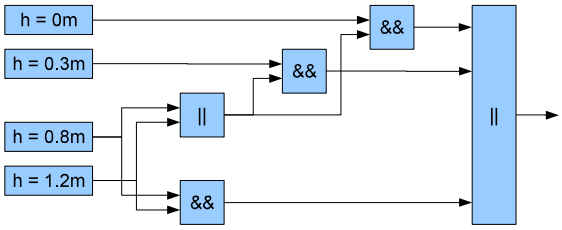

Figure 4: Rule-based fusion of the four single layers

In all four layers the previously described fusion by computation of the polygon intersections can be performed. In a following step a fusion of all four layers is performed. As we do not know the exact height of the detected objects and both small, e.g. luggage, and large objects, e.g. persons, have to be detected, a rule based decision has been applied. So at least within two layers intersections have to be found. These would be either the lower two, for small objects or the upper two for huge objects. Figure 4 illustrates this process. As a result we get the regions $R_i(t)$, in which objects can possibly be found for each frame. These regions are described by their position $p_i(t)$ and the radius $r_i(t)$, where the radius is computed with: $r = \sqrt{area/\pi}$. Figure 5 illustrates the detected region, indicated with a black circle, for a person standing near the point $(1, 0)$ which is the only person within the defined tracking area. All errors due to lighting changes and shadows have been eliminated, as only the one visible person has been detected.

2.3. Object Tracking And Detection Of Tracking Events

Up to now only possible object regions $R_i(t)$ are detected in the ground plane. There is not yet an assignment of a region $R_i(t)$ to an object $O_j(t)$, and no temporal trajectory has been yet determined. Furthermore there is no information about temporal and spatial relationships between objects. Two objects could be created by a split of one for instance. Or two objects become one after a merge. Therefore a tracking algorithm has been developed which assigns detected regions $R(t)$ to objects $O(t)$, performs a temporal, tracking and recognizes splits and merges of objects:

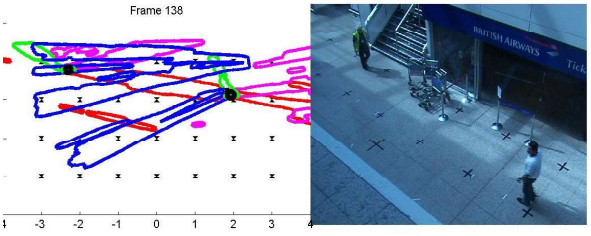

Figure 5: Fusion result for frame 138 of scene s01

$$R(t) \rightarrow O(t)$$

where $O(t)$ is a vector containing all observed objects $o_j(t)$, and R containing all regions r_i. Common tracking approaches, such as condensation, presented by Isard and Blake [8], or unscented Kalman filtering [9] have been dismissed due to high computational effort. We decided to use a heuristic approach to remain real time capable in all computation stages. Each object $o_j(t)$ is described by its position $p(t)$, the radius $r(t)$, speed $v(t)$, the time in the camera view $LifeTime(o_j(t))$ and an ID $id(t)$. In the first step an estimation of the new object position is made by taking speed and old object position into account:

$$\vec{o_i} = \vec{o_i} + \vec{v_i}$$

In the next step we determine the new possible object regions with the homografic transform in all four camera views. For each predicted object o_j we determine the probability k_{ij} that a detected region $r_i(t)$ belongs to the actual object. This is done by modeling each object with a Gaussian distribution. Regions located nearer to an object's center are assigned a larger probability than regions located further away:

$$\vec{k_i} = [k_{i1} k_{i2} \dots k_{ij}] \text{ where } k_{ij} = \exp\left(-\frac{\|o_i - p_j\|)^2}{\sigma^2}\right)$$

Where σ denotes the variance, which is approximated by the average radius of a human, here $\sigma = [0.3\ 0.5]$. Now the new objects are aligned to the detected regions. Areas with a large probability therefore are weighted higher. Additionally the size of a region's surface and the estimated object position are taken into

account:

$$o_j = \frac{0.1 \cdot o_j + \sum_i 2\pi(r_i)^2 \cdot k_{ij} \cdot p_i}{0.1 + \sum_i 2\pi(r_i)^2 \cdot k_{ij}}$$

It seems reasonable to assign more than one region to an object, as for instance feet are often detected as two independent regions, although they belong to one person.
By the simple subtraction of two subsequent object positions we can compute the object's speed:

$$v_j(t) = o_j(t) - o_j(t-1)$$

Now it is checked if all regions r_i are assigned to an object o_j. If this has not happened, as the region's center is located too far away from each object's center, a new object is created. This procedure is required to detect persons entering the observed area and detect a so called split, where one object is split into two new objects. Finally, this is the point when the object IDs are introduced. A new object appearing near the borders of one of the camera views is considered as unknown object and gets a new free ID assigned. If an object pops up in the middle of the camera views it is probably created by a split, and therefore gets the same ID as an overlapping object from the last frame. It rarely happens that an object just pops up in the middle of the tracking area and is too far away from any existing object. This usually is a false detection, caused by lighting changes or shadows. In most cases these spontaneous appearances vanish after a few frames. A merge is detected if the distance of two object centers is smaller than the medium radius of the two objects. The new object is assigned the ID of the older blob. In a last step objects, which are not in the field of view anymore, are removed from the list, if no regions are assigned to them for at least 2 frames. Figure 6 illustrates the detection of a split and merge.

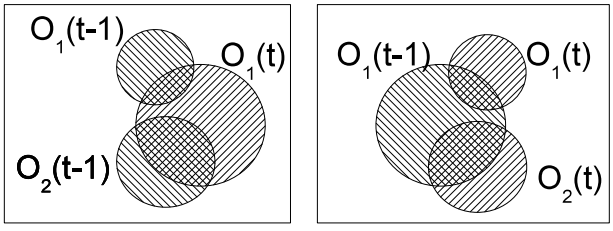

Figure 6: Right: merging objects, Left: splitting objects

Figure 7 shows the results of the tracking algorithm for two frames from the first sequence. Especially the lower image shows the capability of assigning multiple regions to one object. Without this in the lower image six individual objects would have been detected instead of just the two visible ones.

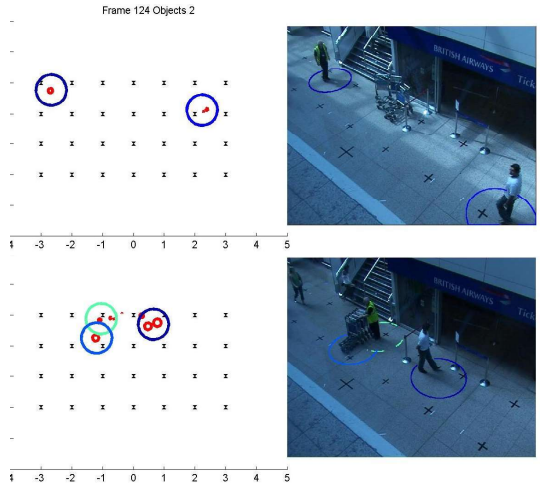

Figure 7: Two samples from S01

2.4 Stationary Object Detection

Our event recognition module mainly considers the trajectories of each tracked object, which are obtained as described in 2.3. By analyzing the trajectories' splits, merges and stationary objects are detected.
In order to be able to recognize left luggage, we decided to recognize stationary objects. This seems rather reasonable, as only non moving luggage has to be recognized. Therefore the variance directed in x and y of the object positions $o_j(t)$ is computed over the last M frames and subsequently normalized.

$$\overline{o}_i = \frac{1}{M} \sum_{t'=t-M}^{t} o_i(t')$$

$$\mathrm{var}_i(t) = \left\| \frac{1}{M} \sum_{t'=t-M}^{t} \left(o_i(t') - \overline{o}_i\right)^2 \right\|_2$$

If the variance var_i is below a threshold, which has been experimentally determined the object o_j is considered stationary, else moving.

$$\{\mathrm{var}_i(t) < \mathrm{var\_threshold}\} \Rightarrow (O_i \ \ \mathrm{stationary})$$

Figure 8 shows the result of the stationary object detection on a sample from Scene $S08$. A yellow circle with $r = 2m$ and a red one with $r = 3m$ indicate the

minimum distances required for scenario recognition. As it can easily be seen, both the human and the backpack are detected as a stationary object, as their both variances are below a predefined threshold. Therefore it could be wise to implement a human body or luggage item detector, in order to discriminate the type of stationary object. This is of major importance as the event detection has no information about the object type, but just the status. A potential system could be based on a boosted cascade of Haar-basis-like features as presented by Viola and Jones [10], or Support Vector Machines [11] [12].

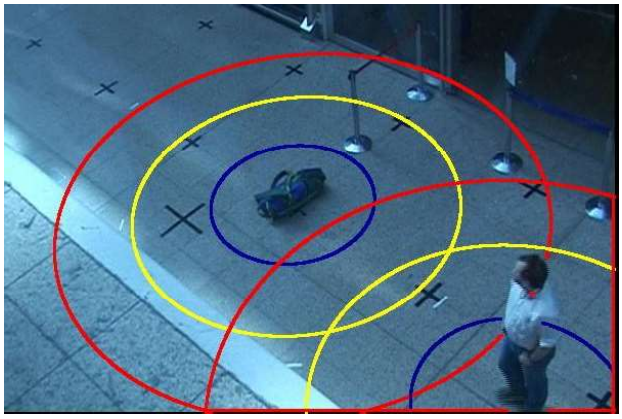

Figure 8: Stationary object detection with one miss classification.

3 Event Detection

In recent works [13], [14] on human behavior analysis Bayesian Networks or Hidden Markov Models have shown promising results. Unfortunately, most common probabilistic classifiers, both dynamic and static, require some kind of training material or at least expert knowledge. Due to the lack of a complex database we decided to work with an expert knowledge based approach. Instead of creating a probabilistic model a set of rules has been defined to detect events. This rather old fashioned approach seems to be very reliable both on the PETS 2006 and 2007 data set.
According to the definition provided for the PETS 2007 challenge a person is loitering if she stays in the field of view for at least 60 sec. This task can be solved rather easily, by adding a time stamp $LifeTime(o_j(t))$ to every tracked object if it appears for the first time. If the difference is larger than the required lifetime within the camera view, an alert can be created.
Various approaches have been presented for the left luggage detection task in the past [1]. Having a close look at the provided data sets, it becomes clear, that leaving luggage is following a quite similar scheme. In the first place an object will split into two. One of these parts will remain stationary in the following frames. The other one will be leaving the stationary one after a while. A warning is displayed if the object distance is larger than $3m$. If the distance is larger than $3m$ for at least $30s$ an alert signal is displayed. Figure 9 illustrates this process, which has been implemented as a simple decision tree. A second case is that a split is detected and after a short while the moving object vanishes within the $3m$ region. This commonly happens near the borders of the field of view. Therefore the sudden vanishing of the moving split object has to be also modeled.

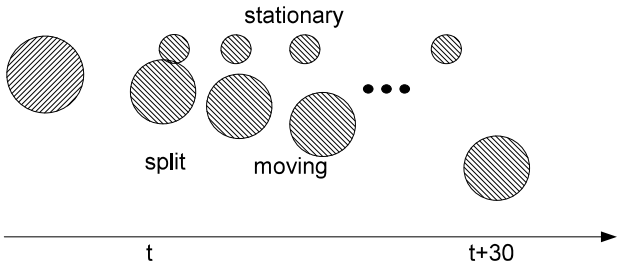

Figure 9: Scheme of leaving a piece of luggage

4. Results

In this section we will discuss the results on the datasets $1, 2, 7$ and 8 of the PETS 2007 tracking challange, as luggage theft has not been implemented yet. For the sequences $3, 4, 5$ and 6 just the timestamps for leaving luggage unattended and the position have been computed. Two sets of results will be given. First an overview on the implemented tracking algorithm and second the results of the alarm detection.
Assuming an exact annotation of the data within the provided ground truth and provided an accurate camera calibration it is possible to compare the tracking results in the spatial domain. A possible object to compare tracking accuracy could be the unattended luggage left at the patch $x = 0, , y = 0, z = 0$. Table 2 shows the detected x and y position within the event detection. Note, that part of the error should result from different calibration, as the calibration data has been changed, because we experienced some errors with the provided calibration data. Furthermore the bags were probably not dropped of exactly at $(0, 0)$. Anyways the error should be tolerable for the given application scenario.
Loitering persons were recognized without any misses.

Scene:	t_d	t_e	δt	x	y
S01	1648	148	23	-1.130388	0.845811
S02	1718	218	89	-0.161693	-0.172595
S07(obj)	1833	333	-1	-0.047781	-0.153542

Table 1: Timestamps and positions for loitering

Scene:	t_d	x	y	x_{own}	y_{own}
S07 unatt.	1491	-0.01	-0.20	NA	NA
S08 unatt.	1147	-0.14	0.04	-2.38	-1.05
S08 left	1773	-0.12	0.04	NA	NA

Table 2: Timestamps and positions for unattended and left luggage in S07 and S08

Table 1 shows the timestamps for detection of loitering L_d, the resulting time of scene entering E and the position of the loitering person when the alarm is set x,y. Image 10 shows a sample from an alert created in scene $S01$ at frame 1648. Without applying a pedestrian detection left luggage will also be recognized as loitering object, as it happened in sequence $S07$ at frame 1833, see fig. 12. It should not be a major issue, as luggage and loitering are both security issues and are meant to cause alerts.

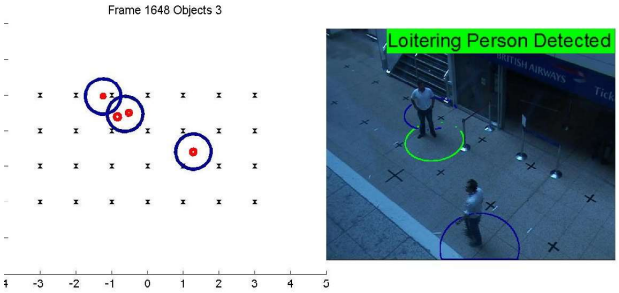

Figure 10: A loitering person has been detected

The recognition of luggage events is by far more complicated than the detection of loitering people. Especially the split and merge detection tends to be rather difficult. The maximum distance between two objects has to be set carefully and be fitted to each application scenario. Here the maximimum distance for splits was set to $0.5r$ and resulted in no false positive and no miss for left luggage detection. Figure 11 shows the resulting warning and subsequent alert for the implemented system. Table 2 shows both the time stamps for the detection of unattended luggage and abandoned luggage. Additionally the position of the original owner (x_{own}, y_{own}) is indicated, if available.

Figure 11: A sample from S08. First the person leaves the 3m radius. After 25s an alert will be set

5. Conclusion

The proposed algorithms have the advantage of being very simple and being of low computational effort. Especially the tracking based on the homographic transform has shown reliable results, although the required foreground segmentation often showed errors, due to lighting changes and shadow, which have not been compensated during preprocessing. The event detection performed surprisingly robust despite the fact, that a simply rule-based system has been applied.

As a consequence of the systems simplicity there are some limitations both for tracking and event detection. Tracking based on heuristics shows to be reliable for small spatial movements, compared to the object radius, as present in both the PETS 2006 and PETS 2007 benchmark data. As soon as 'jumps' occur and and objects moves a distance larger than its radius the tracker unfortunately fails. Therefore a more complex description of motion is required for other scenarios. Another weakness is the temporal supervision of the tracked blobs. There is no identification of the tracked blobs. For instance if two objects merge and after a while split again we are not able to surely assign the object IDs before the merge.

The rule based event detection works surprisingly well with the given data set. Due to its simplicity the computational effort is kept low and no training steps are required. Unfortunately not every potential threat has been modeled with this approach. For instance an unattended luggage item might merge with a passing person. The following split will be recognized, but not if the luggage had eventually been swapped in the meantime. An accidental shift of a stationary object would also distract the implemented algorithm. Therefore a blob correspondence could be implemented based on the individual color histograms and silhouettes, which would allow modeling more com-

Figure 12: Left: As soon as the person is not detected in the 3m radius a warning is set. Right: Objects can be recognized as loitering people. Due to lighting changes false detections can occur

plex scenes.

Acknowledgments

This work has partially been funded by the European Union within the FP6 IST SAFEE Project. www.safee.info.

References

[1] *Proceedings 9th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS), New York, USA*, 2006.

[2] Chris Stauffer, "Adaptive background mixture models for real-time tracking," in *Proc. of IEEE conference on Computer Vision and Pattern Recoginitions,Fort Collins, USA*, 1999, pp. 246–252.

[3] C. A. Bouman, "Connected component analysis," in *Digital Image Processing, Course Notes*, 2007.

[4] Jianbo Shi and Carlo Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, June 1994.

[5] Z. Zivkovic and B. Krose, "An em-like algorithm for color-histogram-based object tracking," in *Proceedings IEEE Conference CVPR 2004, Washington, USA*, June 2004.

[6] Saad M. Khan and Mubarak Shah, "A multiview approach to tracking people in crowded scenes using planar homography constraint," in *In Proceedings IEEE Conference ECCV 2006*, 2006, pp. 133–146.

[7] Roger Y. Tsai, "An efficient and accurate camera calibration technique for 3d machine vision," in *Proc. of IEEE conference on Computer Vision and Pattern Recoginitions*, 1986, pp. 364–374.

[8] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29(1), pp. 5–28, 1998.

[9] E. Wan and R. van der Merwe, "The unscented kalman filter," Wiley Publishing, 2001.

[10] P. Viola and M. Jones, "Robust real-time object detection," in *Second International Workshop On Statistical and Computional Theories of Vision - Modeling, Learning, Computing, and Sampling*, Vancouver, July 2001.

[11] B. Schoelkopf, "Support vector learning," *Neural Information Processing Systems*, 2001.

[12] Constantine Papageorgiou and Tomaso Poggio, "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[13] D. Arsić, J. Schenk, B. Schuller, and G. Rigoll, "Submotions for hidden markov model based dynamic facial action recognition," in *Proceedings IEEE International Conference on Image Processing (ICIP) 2006, Atlanta, Georgia, USA*, Oct. 2006.

[14] D. Arsić, B. Schuller, and G. Rigoll, "Suspicious behavior detection in public transport by fusion of low-level," in *Proceedings 8th International Conference on Multimedia and Expo ICME 2007, Beijing, China*, July 2007.