

# Left-Luggage Detection using Bayesian Inference

Fengjun Lv    Xuefeng Song    Bo Wu    Vivek Kumar Singh    Ramakant Nevatia

Institute for Robotics and Intelligent Systems  
University of Southern California  
Los Angeles, CA 90089-0273  
{flv|xsong|bowu|viveksin|nevatia}@usc.edu

## Abstract

*This paper presents a system that incorporates low-level object tracking and high-level event inference to solve the video event recognition problem. First, the tracking module combines the results of block tracking and human tracking to provide the trajectory as well as the basic type (human or non-human) of each detected object. The trajectories are then mapped to 3D world coordinates, given the camera model. Useful features such as speed, direction and distance between objects are computed and used as evidence. Events are represented as hypotheses and recognized in a Bayesian inference framework. The proposed system has been successfully applied to many event recognition tasks in the real world environment. In particular, we show results of detecting the left-luggage event on the PETS 2006 dataset.*

## 1. Introduction

Video event recognition has been an active topic in computer vision and artificial intelligence community recently. It is a difficult task because it consists of not only low-level vision processing tasks such as tracking and object detection, which by themselves are difficult problems, but also high-level event inference, which has to handle the ambiguity in event definition, the variations in execution style and duration and the fusion of multiple source of information.

Another major difficulty is the lack of event data and evaluation criteria. This is because the acquisition of event data is hard either because events of interest (especially unusual events) rarely happen in real life or because recording of such events is prohibited in many situations. So many researchers have to rely on staged data, which requires a lot of effort to make it look realistic. This difficulty impedes the evaluation and comparison of different event recognition algorithms.

The PETS 2006 workshop serves the need of event data by providing a publicly available dataset that contains event scenarios in a real-world environment. The workshop fo-

cuses on detecting the left-luggage event in a railway station. The event has a clear definition consisting of the following three rules: **(1) Contextual rule:** A luggage is owned and attended by a person who enters the scene with the luggage until such point that the luggage is not in physical contact with the person. **(2) Spatial rule:** A luggage is unattended when the owner is further than 3 meters from the luggage. **(3) Temporal rule:** If a luggage has been left unattended by the owner for a period of 30 consecutive seconds in which time the owner has not re-attended to the luggage, nor has the luggage been attended to by a second party, the alarm event is triggered.

Although detection of left-luggage is the only considered task of PETS 2006, the event contains large variations in terms of the types of luggage and the complexity of scenarios. The dataset includes five types of luggage: {briefcase, suitcase, 25 liter rucksack, 70 liter backpack, ski gear carrier}. The size and the shape of these different types are different, which increases the difficulty in luggage detection and tracking. The dataset considers the following seven scenarios with increasing complexity. **(1)** Person 1 with a rucksack loiters before leaving the rucksack unattended. **(2)** Person 1 and 2 enter the scene from opposite directions. Person 1 places a suitcase on the ground, before person 1 and 2 leave together without the suitcase. **(3)** Person 1 temporarily places his briefcase on the ground before picking it up again and leaving. **(4)** Person 1 places a suitcase on the ground. Person 2 arrives and meets with person 1. Person 1 leaves the scene without his suitcase. **(5)** Person 1 with a ski gear carrier loiters before leaving the luggage unattended. **(6)** Person 1 and 2 enter the scene together. Person 1 places a rucksack on the ground, before person 1 and 2 leave together without the rucksack. **(7)** Person 1 with a suitcase loiters before leaving the suitcase unattended. During this event five other persons move in close proximity to the suitcase. Accordingly, the event recognition system should have the flexibility to accommodate these variations.

We present an event recognition system that tackles the challenges in tracking and event recognition problems. For

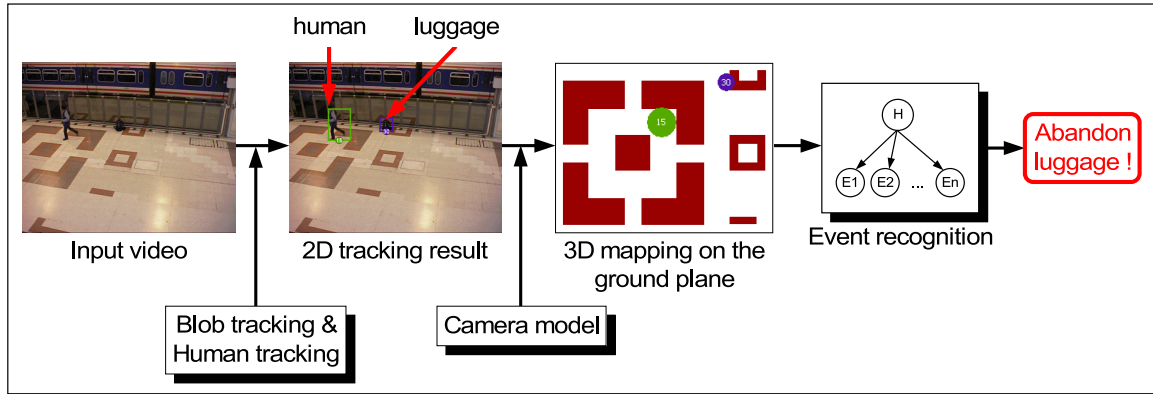


Figure 1: The overview diagram of our system

tracking, a combination of a blob tracker (see Section 2.1) and a human tracker (see Section 2.2) is used to provide the trajectory of each human and carried object. For event recognition, each event is represented and recognized in a Bayesian inference framework (see Section 3.1). In particular, the algorithm is applied to detect the left-luggage event on the PETS 2006 dataset (see Section 3.2). The overview diagram of our system is shown in Figure 1.

Note that in PETS 2006, each scenario is filmed from multiple cameras. We only use the third camera because it provides the best view point and the results based on this single view point is satisfactory.

Many efforts have been made to solve the human tracking problem. As it is hard to provide an adequate description of the whole literature, we only list some that are closely related to this work. The observations of human hypotheses may come from various cues. Some use the result of background subtraction as observations and try to fit multiple human hypotheses to explain the foreground blobs [12]. As the hypotheses space is usually of high dimension, sampling algorithms such as MCMC [12], or dynamic programming algorithms such as EM [7] are applied. Other methods, *e.g.* [2], build deformable silhouette models based on edge features to detect pedestrians and associate the detection responses to form tracks. These two types of approaches are complementary.

Proposed methods for video event recognition can be divided into three sub-categories based on the underlying features they use: **(1)** those based on 2-D image features such as optical flow [3], body shape/contour [11], space-time interest point [4], **(2)** those based on 2-D trajectories of tracked objects such as hands, legs or the whole body [8] and **(3)** those based on 3-D body pose acquired from motion capture or 3-D pose tracking system [5]. None of above methods, however, involves the manipulation of objects such as luggage.

## 2. The Tracking Module

We use two components for the tracking task. A Kalman filter based blob tracker, which provides trajectories of all foreground objects including humans and carried objects, is described in Section 2.1. A human body detection based human tracker is described in Section 2.2. We describe the combination of both trackers in Section 2.3.

### 2.1 Kalman Filter based Blob Tracking

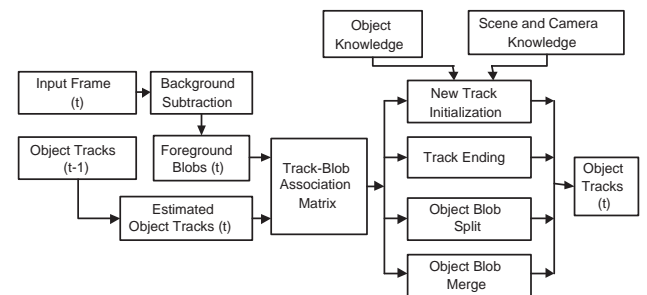


Figure 2: An overview of the blob tracker

Figure 2 shows an overview of our blob tracking algorithm. We learn the background model from the first five hundred frames and apply it to detect the foreground pixels at each frame. Connected foreground pixels form foreground blobs. At each frame  $t$ , a blob  $B_t^i$  contains the following information:

- ◇ Size and location: for simplicity, we use a rectangular bounding box  $(x_t^i, y_t^i, width_t^i, height_t^i)$
- ◇ Appearance: we use the color histogram of the blob

Ideally, one blob corresponds to one real object and vice versa. But we have to consider the situations such as one object splits into several blobs or multiple objects merge into

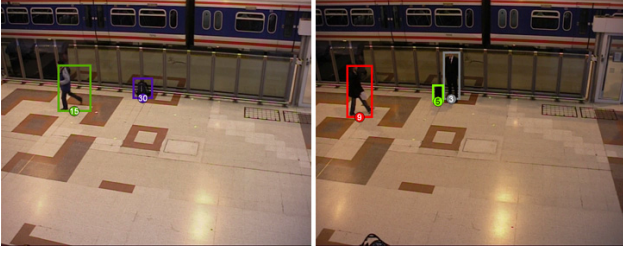


Figure 3: Blob tracking results

one blob. For a blob  $B_i^t$  and an object  $O_j^t$ , we assign an association value based on the overlap between the bounding box of  $B_i^t$  and  $O_j^t$ :

$$M(B_i^t, O_j^t) = \begin{cases} 1, & \text{if } \frac{\text{Area}(B_i^t \cap \hat{O}_j^t)}{\min(\text{Area}(B_i^t), \text{Area}(\hat{O}_j^t))} > \tau \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where  $\hat{O}_j^t$  is the predicted bounding box of  $O_j^t$  from the previous frame using the Kalman filter and  $B_i^t \cap \hat{O}_j^t$  is the intersection of the two bounding boxes and  $\tau$  is a threshold between 0 and 1. If  $M(B_i^t, O_j^t)$  is one, we call  $B_i^t$  and  $O_j^t$  are matched.

At time  $t$ , assume we have  $m$  blobs and  $n$  predicted objects. For all pairs of blob and object, we construct an  $m \times n$  association matrix. The following are possible situations that we need to consider: **(1)** If the blob-object match is one-to-one, we simply update the object with the blob. **(2)** If a blob has no matched object, a new object is created. **(3)** If an object has no match blobs for several frames, the object is removed. **(4)** If an object has multiple matched blobs, the blobs are merged into a bounding box of the object. **(5)** If a blob has multiple matched objects, the blob is segmented based on the appearance (color histogram) of each matched object.

A luggage is detected based on its mobility: It seldom moves after the start of its track. Some detected humans and luggage are shown in Figure 3.

## 2.2 Detection based Human Tracking

For shape based human tracking, we take the single frame human detection responses as the observations of human hypotheses. Tracking is done in 2D with a data association style method. The detector is a generalization of the body part detector proposed by Wu et al. in [9]. We do not rely on skin color, or face, hence our method can deal with rear views. The algorithm of [9] has been extended in [10] to track multiple humans based on the detectors. Our tracking algorithm is a simplified version of [10].



Figure 4: Human tracking results

### 2.2.1 Multi-View Human Body Detection

In [9], four part detectors are learned for detecting full-body, head-shoulder, torso, and legs. We only use the full-body detector here. Two detectors are learned: one for the left profile view, and one for the frontal/rear view. The third detector is for right profile view, which is generated by flipping the left profile view horizontally. Nested cascade detectors are learned by boosting edgelet feature based weak classifiers, as in [9]. The training set contains 1,700 positive samples for frontal/rear view, 1,120 for left profile view, and 7,000 negative images. The positive samples are collected from the Internet and the MIT pedestrian set [6]. The negative images are all from the Internet. The training set is fully independent of the test sequences, and the detectors learned are for generic situations. For detection, the input image is scanned by all three detectors and the union of their responses is taken as the multi-view detection result.

### 2.2.2 Multiple Human Tracking

Humans are tracked in 2D by associating the frame detection responses. This 2D tracking method is a simplified version of [10]. In [10], the detection responses come from four part detectors and a combined detector. To start a trajectory, an initialization confidence  $InitConf$  is calculated from  $T$  consecutive responses, which correspond to one human hypothesis, based on the cues from color, shape, and position. If  $InitConf$  is larger than a threshold  $\theta_{init}$ , a trajectory is started. To track the human, first data association with the combined detection responses is attempted; if this fails, data association with the part detection responses is attempted; if this fails again, a color based meanshift tracker [1] is used to follow the person. The strategy of trajectory termination is similar to that of initialization. A termination confidence  $EndConf$  is calculated when an existing trajectory has been lost by the detector for  $T$  time steps. If  $EndConf$  is larger than a threshold  $\theta_{end}$ , the trajectory is terminated.

In this work, we do not use the combined detection method described in [9]. Since the occlusions are not strong in this data set, a local feature based full-body detector gives

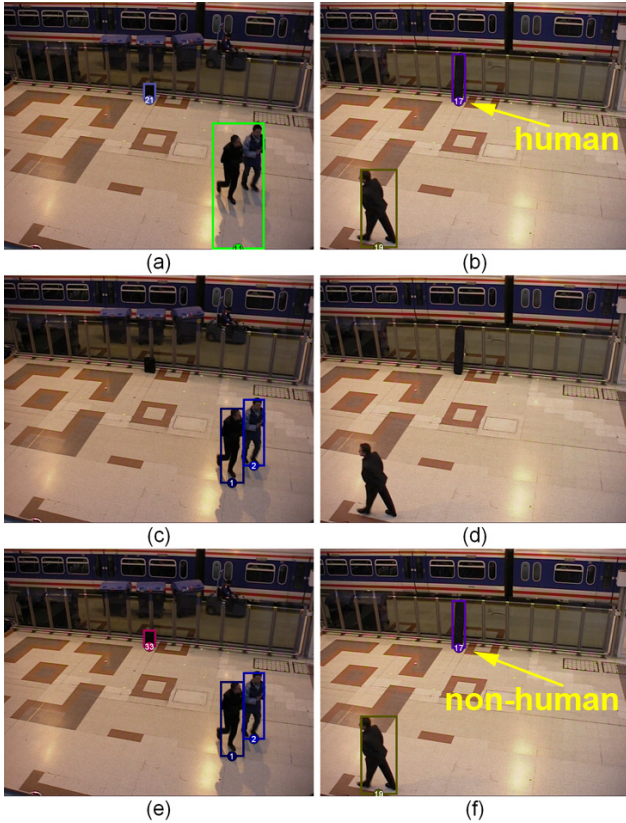


Figure 5: Tracking results of individual trackers and combined tracker. (a),(c),(e): The tracking results on the same frame using the blob tracker alone, the human tracker alone and the combined tracker. The combined tracker can separate two persons from a merge blob and locate the luggage as well. (b),(d),(f): Another example. The combined tracker confirms that the object in the middle is not human. It also tracks the person that is missed by the human tracker due to the large tilt angle.

satisfactory results. Fig.4 shows some sample frames of the tracking results.

### 2.3 The Combination of the Blob Tracker and the Human Tracker

In most cases, the blob tracker does a good job of tracking objects and locating luggage. However, when multiple humans appear merged from the beginning, the tracker can not separate them well and sometimes the tracker mis-identifies the type of the tracked objects based only on the mobility of the objects. See Figure 5(a) and (b) for examples of these issues.

On the other hand, the human tracker does not track objects other than humans. Also, although the learned detectors work with a large range of tilt angles (within about

$[0^\circ, 45^\circ]$ ), when the tile angle is too large, the detectors can not find the human reliably. These issues can be clearly seen in Figure 5(c) and (d).

We can overcome the above limitations of the blob tracker and the human tracker by combining their results so that if one trajectory from the human tracker largely overlaps one trajectory from the blob tracker, the first trajectory supersedes the second one. This is because the human tracker is based on human shape so it is more reliable. The blob tracker, on the other hand, enhances the results by including objects that are missed by the human tracker. The combined results of the above examples are shown in Figure 5(e) and (f).

## 3. The Event Recognition Module

We first describe our event recognition algorithm, followed by the details of how this algorithm is applied to detect the left-luggage event on the PETS 2006 dataset.

### 3.1 Event Recognition using Bayesian Inference

Our event recognition algorithm takes object trajectory (provided by the underlying detection/tracking modules) as the only observation. The algorithm assumes that the type of each object is known. The basic object types include human, carried object, vehicle and scene object (*e.g.* door, stairs). This information is either automatically provided by the detection/tracking modules or specified by system users.

The trajectory is first smoothed using median filter to remove abrupt changes. Useful physical properties such as the position, speed and direction of a moving object and the distance between two objects are inferred from object trajectories. Since these physical properties are measured in terms of world coordinates, 2D object trajectories need to be mapped to 3D world coordinates first. For an object on the ground plane, we assume that the vertical coordinate of its lowest point is zero. Given the camera model, we use the center of the bottom of the image bounding box as the object's lowest point and map this point to 3D position on the ground plane.

System users have to provide an event model for each event in advance. A successful event model has to handle the ambiguity in the event definition (which implies that we should define an event in a probabilistic domain instead of a logical one) and incorporate multiple cues into a computational framework.

Due to these considerations, we use Bayesian inference as the event modeling and recognition framework: events and related cues are considered as hypotheses and evidence, respectively; Competing events are treated as competing hypotheses. For example, regarding the relationship between

a moving object  $A$  and a zone  $B$  in terms of position of  $A$  and  $B$ , there are four possibilities (hypotheses), corresponding to four competing events:  $\{H_1:A$  is outside of  $B$ ,  $H_2:A$  is entering  $B$ ,  $H_3:A$  is inside  $B$ ,  $H_4:A$  is exiting  $B\}$ . We consider the following two pieces of evidence:  $\{E_1$ :the distance between  $A$  and  $B$  sometime (e.g. 0.5 second) ago,  $E_2$ :the distance between  $A$  and  $B$  now $\}$ . The complete model also includes the prior probabilities of each hypothesis and the conditional probabilities of each evidence given the hypotheses. In the above example, we can set the prior probabilities of all hypotheses to be equal ( $P(H_i) = 1/4$ ,  $i = 1, 2, 3, 4$ ) and use the following conditional probabilities.

$$\begin{array}{c} \frac{P(E_1 = 0|H_i):\{0,0,1,1\}}{P(E_1 > 0|H_i):\{1,1,0,0\}} \\ \frac{P(E_2 = 0|H_i):\{0,1,1,0\}}{P(E_2 > 0|H_i):\{1,0,0,1\}} \end{array}$$

The meaning of this table is clear. Take the hypothesis  $H_2$  “ $A$  is entering  $B$ ” for example,  $E_1 > 0$  and  $E_2 = 0$  means  $A$  was outside of  $B$  sometime ago and  $A$  is inside  $B$  now. Given the evidence, the probability of  $H_2$  is computed using the Bayesian rule:

$$P(H_2|E_1, E_2) = \frac{P(E_1, E_2|H_2)P(H_2)}{\sum_{i=1}^4 P(E_1, E_2|H_i)P(H_i)} \quad (2)$$

If we assume each evidence is conditionally independent given the hypothesis, Eq.2 can be rewritten as:

$$P(H_2|E_1, E_2) = \frac{P(E_1|H_2)P(E_2|H_2)P(H_2)}{\sum_{i=1}^4 P(E_1|H_i)P(E_2|H_i)P(H_i)} \quad (3)$$

Note that the above example has the simplest form of a probability distribution function: a threshold function (*i.e.* the distance is quantified to only two values “=0” and “>0”) and the probability is either 0 or 1 (a deterministic case). For more complex cases, the following distribution functions are provided in our system: {Histogram, Threshold, Uniform, Gaussian, Rayleigh, Maxwell}. The parameters of the distribution functions are learned from training data. If such data are not available, the parameters are specified based on user’s knowledge of the event.

If a hypothesis  $H$  has no competing counterparts, we consider  $\neg H$ , the opposite of the hypothesis, and use the following equation to compute the posterior probability of hypothesis given evidence.

$$P(H|E_1, \dots, E_n) = \frac{\prod_{i=1}^n P(E_i|H)P(H)}{\prod_{i=1}^n P(E_i|H)P(H) + \prod_{i=1}^n P(E_i|\neg H)P(\neg H)} \quad (4)$$

When  $P(E_i|\neg H)$  is unknown or hard to estimate, we use a default value 0.5.

### 3.2 Left-Luggage Detection

For the specific task of detecting left-luggage, as described in the introduction, the following events are modeled.

◇ Drop off luggage (denoted as  $D$ ): Three pieces of evidence are used.

- $E_1$ : The luggage did not appear  $-0.1$  (or other small negative value) second ago.
- $E_2$ : The luggage shows up now.
- $E_3$ : The distance between the person and the luggage now is less than some threshold  $d_{drop-off}$  (e.g. 1.5 meters).

$E_1$  and  $E_2$  together impose a temporal constraint: the luggage was carried by the human before the *drop-off* so the separate track of the luggage has not started yet. Once it appears, we know the *drop-off* has just happened.

$E_3$  is a spatial constraint: The owner has to be close to the luggage when *drop-off* just happened. This constraint eliminates irrelevant persons who are just passing by when *drop-off* takes place. In case that multiple persons are close to the luggage when *drop-off* takes place, the closest person is considered as the owner.

The prior and the conditional probabilities are listed as follows:

- $P(D) = P(\neg D) = 0.5$
- $P(E_i|D) = 0.99, i = 1, 2, 3$
- $P(E_i|\neg D) = 0.5, i = 1, 2, 3$

◇ Abandon luggage (denoted as  $A$ ): Two pieces of evidence are used.

- $E_1$ : The distance between  $H$  and  $L$   $-0.1$  (or other small negative value) second ago is less than the alarm distance  $d_{alarm}$  ( $=3$  meters).
- $E_2$ : The distance now is larger than  $d_{alarm}$ .

$E_1$  and  $E_2$  together incorporate the **spatial rule** of PETS 2006 that is described in the introduction.

The prior and the conditional probabilities are listed as follows:

- $P(A) = P(\neg A) = 0.5$
- $P(E_i|A) = 0.99, i = 1, 2$
- $P(E_i|\neg A) = 0.5, i = 1, 2$

Sequence	1	2	3	4	5	6	7
# of persons	1	2	1	2	1	2	6
# of persons	1	2	1	2	1	2	5
# of luggage items	1	1	1	1	1	1	1
# of luggage items	1	1	1	1	1	1	1
Luggage location	(0.22,-0.44)	(0.34,-0.52)	(0.86,-0.54)	(0.24,-0.27)	(0.34,-0.56)	(0.80,-0.78)	(0.35,-0.58)
Luggage location	(0.11,-0.22)	(0.02,-0.44)	(1.10,-0.31)	(0.40,-0.10)	(0.27,-0.30)	(0.45,-0.53)	(0.2,-0.3)
Warning triggered time	2825 (113.0)	2280 (91.2)	none	2585 (103.4)	2749 (110.0)	2403 (96.1)	2317 (92.7)
Warning triggered time	2834 (113.4)	2278 (91.1)	none	2577 (103.1)	2743 (109.7)	2407 (96.3)	2310 (92.4)
Alarm triggered time	2843 (113.7)	2296 (91.8)	none	2602 (104.1)	2764 (110.6)	2422 (96.9)	2349 (94.0)
Alarm triggered time	2848 (113.9)	2298 (92.0)	none	2599 (104.0)	2757 (110.3)	2423 (96.9)	2350 (94.0)

Table 1: The ground-truth (with dark background) and our result (with white background). Locations are measured in meter. The last four rows show the frame number and time (in second) when an even is triggered.

We first detect the *drop-off* event on all combination of the human-luggage pair and only those pairs with high probability are considered further by the *abandon* event. This is how the **contextual rule** is applied.

For the **temporal rule**, we require that if current frame  $i$  gets a high probability of the *abandon* event and if any previous frame  $i - t$  ( $t$  within the range of 30 seconds) also gets high probability, the high probability at frame  $i - t$  will be discarded. Finally, alarm events are triggered at the frames with high probability. PETS 2006 also defines a warning event, which is treated exactly the same here except for a smaller distance threshold (=2 meters).

## 4. Results and Evaluation

The following information is provided in the ground-truth data of PETS 2006 dataset.

- ◊ The number of persons and luggage involved in the alarm event
- ◊ The location of the involved luggage
- ◊ The frame (time) when the warning event is triggered
- ◊ The frame (time) when the alarm event is triggered

We compare our results with the ground-truth in Table 1. The rows with dark background are the ground-truth data.

Figure 6 shows the key frames (with tracking results) in which the *drop off luggage* and the *abandon luggage* event are detected. The 3D mapping on the ground plane is shown at the right side of each frame. The floor pattern is also provided as reference. Note that the frame number shown in the *abandon luggage* event is the frame number of the *alarm* event minus 750 (equivalent to 30 seconds in a video with 25 fps frame rate).

We can see in Figure 6, some errors on the ground plane are visible because the size and position of bounding boxes

are not perfect and a small offset of a point in 2D can result in a significant error in 3D when the object is far away from the camera. But nonetheless, our tracking algorithms work very well. Note that in Figure 6(e), two persons on the left are not separated because they are merged in a single moving blob and the human detector fails to respond due to a large tilt angle.

The event recognition algorithm detects all (warning and alarm) events within an error of 9 frames (0.36 second). The involved persons include anyone who is close enough to the involved luggage when and after the *abandon luggage* event takes place. The correct number of involved persons in all sequences are detected, except for the last sequence. In that sequence, three persons are walking closely as a group. One of them is severely occluded by the other two and thus missed by the tracker.

The clear definition of the *alarm* event helps contribute to the good performance of our event recognition system. The contextual rule is crucial because detecting the *drop-off* event provides a filtering mechanism to disregard the irrelevant persons and luggage before being considered for the *abandon* event. Furthermore, sometimes our tracker misidentifies the type of an object based only on the mobility of the object, which can result in many false alarms. Due to the contextual rule, the possibility of such false alarms is significantly reduced. To illustrate the importance of the contextual rule, we list in Table 2 the number of triggered events without applying the contextual rule. This demonstrates that the high-level reasoning can eliminates the errors of low-level processing.

## 5. Conclusions and Future Work

We have presented an object tracking and event recognition system that has successfully tackled the challenge proposed by the PETS 2006 workshop. For tracking, we combine the results of a blob tracker and a human tracker, which provides not only the trajectory but also the type of each



Figure 6: The key frames (with tracking results) in which the *drop off luggage* and the *abandon luggage* event are detected. Figures shown here are, from left to right in each row, the key frame of *drop off luggage* in 2D and 3D, the key frame of *abandon luggage* in 2D and 3D, respectively.

Sequence	1	2	3	4	5	6	7
Total tracked persons	35	30	18	21	23	28	45
Total tracked luggage	1	8	1	1	4	1	1
Total drop off events	1	2	1	1	1	1	1
Total warning events	1	2	0	2	3	1	10
Total alarm events	2	4	0	1	2	1	10

Table 2: Without the contextual rule, the results contain many false alarms.

tracked object. For event inference, events are modeled and recognized in a Bayesian inference framework, which gives perfect event recognition result even though the tracking is not perfect.

Note that our system is designed as a framework for general event recognition task and is capable (and has already been applied) of recognizing a much broader range of events. For this specific (*left-luggage*) task, we just used our existing system and did not make any additional effort except that we specified the event models (as described in Section 3.2). This was done immediately. In conclusion, our system provides quick solutions to event recognition tasks such as the *left-luggage* detection.

Real-world scenarios usually are more complicated than the one presented here, in terms of the number of involved persons and objects and the variation in execution style and duration. For future work, we need to consider more sophisticated algorithms to handle such complexities.

## References

- [1] D. Comaniciu, V. Ramesh and P. Meer. The Variable Bandwidth Mean Shift and Data-Driven Scale Selection. In ICCV 2001, Vol I: 438-445.
- [2] L. Davis, V. Philomin and R. Duraiswami. Tracking humans from a moving platform. In ICPR 2000, Vol IV: 171-178.
- [3] A. A. Efros, A. C. Berg, G. Mori and J. Malik. Recognizing Action at a Distance. In ICCV 2003, 726-733.
- [4] I. Laptev and T. Lindeberg. Space-time interest points. In ICCV 2003, 432-439.
- [5] F. Lv and R. Nevatia. Recognition and Segmentation of 3-D Human Action using HMM and Multi-Class AdaBoost. In ECCV 2006, 359-372.
- [6] C. Papageorgiou, T. Evgeniou and T. Poggio. A Trainable Pedestrian Detection System. In Proc. of Intelligent Vehicles, 1998, 241-246.
- [7] J. R. Peter, H. Tu and N. Krahnstoeber. Simultaneous Estimation of Segmentation and Shape. In CVPR 2005, Vol II: 486-493.
- [8] C. Rao, A. Yilmaz and M. Shah. View-Invariant Representation and Recognition of Actions. In IJCV 50(2), Nov. 2002, 203-226.
- [9] B. Wu and R. Nevatia. Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors. In ICCV 2005, Vol I:90-97.
- [10] B. Wu and R. Nevatia. Tracking of Multiple, Partially Occluded Humans based on Static Body Part Detection. To appear in CVPR 2006.
- [11] A. Yilmaz and M. Shah. Actions Sketch: A Novel Action Representation. In CVPR 2005, 984-989.
- [12] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. In CVPR 2004, Vol II: 406-413.