IDIAP RESEARCH REPORT

Detecting Abandoned Luggage Items in a Public Space

Kevin Smith [a] Pedro Quelhas [a]
Daniel Gatica-Perez [a]

IDIAP–RR 06-39

June 2006

paru dans
IEEE Performance Evaluation of Tracking and Surveillance Workshop (PETS)

[a] IDIAP Research Institute, Switzerland

IDIAP Research Institute **www.idiap.ch**
Rue du Simplon 4 P.O. Box 592 1920 Martigny – Switzerland
Tel: +41 27 721 77 11 Fax: +41 27 721 77 12 Email: info@idiap.ch

Rapport de recherche de l'IDIAP 06-39

Detecting Abandoned Luggage Items in a Public Space

Kevin Smith Pedro Quelhas Daniel Gatica-Perez

June 2006

paru dans

IEEE Performance Evaluation of Tracking and Surveillance Workshop (PETS)

Résumé. Visual surveillance is an important computer vision research problem. As more and more surveillance cameras appear around us, the demand for automatic methods for video analysis is increasing. Such methods have broad applications including surveillance for safety in public transportation, public areas, and in schools and hospitals. Automatic surveillance is also essential in the fight against terrorism. In this light, the PETS 2006 data corpus contains seven left-luggage scenarios with increasing scene complexity. The challenge is to automatically determine when pieces of luggage have been abandoned by their owners using video data, and set an alarm. In this paper, we present a solution to this problem using a two-tiered approach. The first step is to track objects in the scene using a trans-dimensional Markov Chain Monte Carlo tracking model suited for use in generic blob tracking tasks. The tracker uses a single camera view, and it does not differentiate between people and luggage. The problem of determining if a luggage item is left unattended is solved by analyzing the output of the tracking system in a detection process. Our model was evaluated over the entire data set, and successfully detected the left-luggage in all but one of the seven scenarios.

Fig. 1 – *Experimental Setup.* An example from the *PETS 2006* data set, sequence S1 camera 3. A man sets his bag on the ground and leaves it unattended.

1 Introduction

In recent years the number of video surveillance cameras has increased dramatically. Typically, the purpose of these cameras is to aid in keeping public areas such as subway systems, town centers, schools, hospitals, financial institutions and sporting arenas safe. With the increase in cameras comes an increased demand for automatic methods for interpreting the video data.

The PETS 2006 data set presents a typical security problem : detecting items of luggage left unattended at a busy train station in the UK. In this scenario, if an item of luggage is left unattended for more than 30s, an alarm should be raised. This is a challenging problem for automatic systems, as it requires two key elements : the ability to reliably detect luggage items, and the ability to reliably determine the owner of the luggage and if they have left the item unattended.

Our approach to this problem is two-tiered. In the first stage, we apply a probabilistic tracking model to one of the camera views (though four views were provided, we restrict ourselves to camera 3). Our tracking model uses a mixed-state Dynamic Bayesian Network to jointly represent the number of people in the scene and their locations and size. It automatically infers the number of objects in the scene and their positions by estimating the mean configuration of a trans-dimensional Markov Chain Monte Carlo (MCMC) sample chain.

In the second stage, the results of the tracking model are passed to a bag detection process, which uses the object identities and locations from the tracker to attempt to solve the left-luggage problem. The process first searches for potential bag objects, evaluating the likelihood that they are indeed a bag. It then verifies the candidate bags, and searches the sequences for the owners of the bags. Finally, once the bags and owners have been identified, it checks to see if the alarm criteria has been met.

The remainder of the paper is organized as follows. We discuss the data in Section 2. The tracking model is presented in Section 3. The process for detecting bags is described in Section 4. We present results in Section 5 and finish with some concluding remarks in Section 6.

2 The Left Luggage Problem

In public places such as mass transit stations, the detection of abandoned or left-luggage items has very strong safety implications. The aim of the PETS 2006 workshop is to evaluate existing systems performing this task in a real-world environment. Previous work in detecting baggage includes e.g. [3], where still bags are detected in public transport vehicles, and [2], where motion cues were used to detect suspicious background changes. Other work has focused on attempting to detect people

TAB. 1 – Challenges in the PETS 2006 data corpus.

Seq.	length (s)	luggage items	num people nearby	abandoned ?	difficulty (rated by PETS)
S1	121	1 backpack	1	yes	1/5
S2	102	1 suitcase	2	yes	3/5
S3	94	1 briefcase	1	no	1/5
S4	122	1 suitcase	2	yes	4/5
S5	136	1 ski equipment	1	yes	2/5
S6	112	1 backpack	2	yes	3/5
S7	136	1 suitcase	6	yes	5/5

carrying objects using silhouettes, e.g. [4]. Additionally, there has been previous work done on other real-world tracking and behavior recognition tasks (including work done for PETS), such as detecting people passing by a shop window [8, 5].

The PETS data corpus contains seven sequences (labeled S1 to S7) of varying difficulty in which actors (sometimes) abandon their piece of luggage within the view of a set of four cameras. An example from sequence S1 can be seen in Figure 1. A brief qualitative description of the sequences appears in Table 1.

An item of luggage is owned by the person who enters the scene with that piece of luggage. It is *attended* to as long as it is in physical contact with the person, or within two meters of the person (as measured on the floor plane). The item becomes *unattended* once the owner is further than two meters from the bag. The item becomes *abandoned* if the owner moves more than three meters from the bag (see Figure 2). The PETS task is to recognize these events, to trigger a warning 30s after the item is unattended, and to trigger an alarm 30s after it is abandoned.

The data set contains several challenges. The bags vary in size ; they are typically small (suitcases and backpacks) but also include large items like ski equipment. The activities of the actors also create challenges for detecting left-luggage items by attempting to confuse ownership of the item of luggage. In sequence S4, the luggage owner sets down his suitcase, is joined by another actor, and leaves (with the second actor still in close proximity to the suitcase). In sequence S7, the luggage owner leaves his suitcase and walks away, after which five other people move in close proximity to the the suitcase.

A shortcoming of the PETS 2006 data corpus is that no training data is provided, only the test sequences. We refrained from learning on the test set as much as possible, but a small amount of tuning was unavoidable. Any parameters of our model learned directly from the data corpus are mentioned in the following sections.

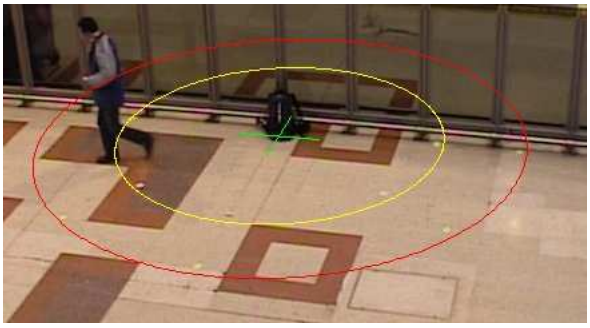

FIG. 2 – *Alarm Conditions.* The green cross indicates the position of the bag on the floor plane. Owners inside the area of the yellow ring (2 meters) are considered to be *attending* to their luggage. Owners between the yellow ring and red ring (3 meters) left their luggage *unattended*. Owners outside the red ring have *abandoned* their luggage. A warning should be triggered if a bag is unattended for 30s or more, and an alarm should be triggered if a bag is abandoned for 30s or more.

3 Trans-Dimensional MCMC Tracking

The first stage of left-luggage detection is tracking. Our approach jointly models the number of objects in the scene, their locations, and their size in a mixed-state Dynamic Bayesian Network. With this model and foreground segmentation features, we infer a solution to the tracking problem using trans-dimensional MCMC sampling.

Solving the multi-object tracking problem with particle filters (PF) is a well studied topic, and many previous efforts have adopted a rigorous joint state-space formulation to the problem [6, 7, 9]. However, sampling on a joint state-space quickly becomes inefficient as the dimensionality increases when objects are added. Recently, work has concentrated on using MCMC sampling to track multiple objects more efficiently [7, 9, 11]. The model in [7] tracked a fixed number of interacting objects using MCMC sampling while [9] extended this model to handle varying number of objects via reversible-jump MCMC sampling.

In a Bayesian approach, tracking can be seen as the estimation of the filtering distribution of a state $\mathbf{X}_t$ given a sequence of observations $\mathbf{Z}_{1:t} = (\mathbf{Z}_1, ..., \mathbf{Z}_t)$, $p(\mathbf{X}_t|\mathbf{Z}_{1:t})$. In our model, the state is a joint multi-object configuration and the observations consist of information extracted from the image sequence. The filtering distribution is recursively computed by

$$\begin{aligned} p(\mathbf{X}_t|\mathbf{Z}_{1:t}) &= C^{-1} p(\mathbf{Z}_t|\mathbf{X}_t) \times \\ & \int_{\mathbf{X}_{t-1}} p(\mathbf{X}_t|\mathbf{X}_{t-1}) p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1}) d\mathbf{X}_{t-1}, \end{aligned} \tag{1}$$

where $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ is a dynamic model governing the predictive temporal evolution of the state, $p(\mathbf{Z}_t|\mathbf{X}_t)$ is the observation likelihood (measuring how the predictions fit the observations), and C is a normalization constant.

Under the assumption that the distribution $p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1})$ can be approximated by a set of unweighted particles $\{\mathbf{X}_t^{(n)}|n = 1, ..., N\}$, where $\mathbf{X}_t^{(n)}$ denotes the n-th sample, the Monte Carlo approximation of Eq. 1 becomes

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) \approx C^{-1} p(\mathbf{Z}_t|\mathbf{X}_t) \sum_n p(\mathbf{X}_t|\mathbf{X}_{t-1}^{(n)}). \tag{2}$$

The filtering distribution in Eq. 2 can be inferred using MCMC sampling as outlined in Section 3.4.

3.1 State Model for Varying Numbers of Objects

The dimension of the state vector must be able to vary along with the number of objects in the scene in order to model them correctly. The state at time t contains multiple objects, and is defined by $\mathbf{X}_t = \{X_{i,t}|i \in \mathcal{I}_t\}$, where $\mathcal{I}_t$ is the set of object indexes, $m_t = |\mathcal{I}_t|$ denotes the number of objects and $|\cdot|$ indicates set cardinality. The special case of zero objects in the scene is denoted by $\mathbf{X}_t = \emptyset$.

The state of a single object is defined as a bounding box (see Figure 3) and denoted by $\mathbf{X}_{i,t} = (x_{i,t}, y_{i,t}, sy_{i,t}, e_{i,t})$ where $x_{i,t}, y_{i,t}$ is the location in the image, $sy_{i,t}$ is the height scale factor, and $e_{i,t}$ is the eccentricity defined by the ratio of the width over the height.

3.2 Dynamics and Interaction

Our dynamic model for a variable number of objects is

$$\begin{aligned} p(\mathbf{X}_t|\mathbf{X}_{t-1}) &\propto \prod_{i \in \mathcal{I}_t} p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1}) p_0(\mathbf{X}_t) && (3) \\ &\overset{def}{=} p_V(\mathbf{X}_t|\mathbf{X}_{t-1}) p_0(\mathbf{X}_t), && (4) \end{aligned}$$

where p_V is the predictive distribution. Following [9], we define p_V as $p_V(\mathbf{X}_t|\mathbf{X}_{t-1}) = \prod_{i \in \mathcal{I}_t} p(\mathbf{X}_{i,t}|\mathbf{X}_{t-1})$ if $\mathbf{X}_t \neq \emptyset$, and $p_V(\mathbf{X}_t|\mathbf{X}_{t-1}) = C$ otherwise. Additionally, we define $p(\mathbf{X}_{i,t}|\mathbf{X}_{t-1})$ either as the object dynamics $p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1})$ if object i existed in the previous frame, or as a distribution $p_{init}(\mathbf{X}_{i,t})$ over

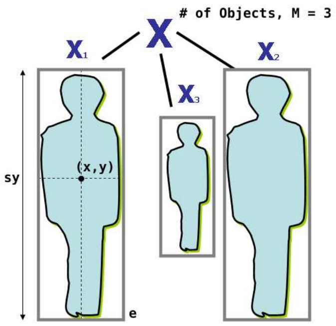

FIG. 3 – The state model for multiple objects.

potential initial object birth positions otherwise. The single object dynamics is given by $p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1})$, where the dynamics of the body state $\mathbf{X}_{i,t}$ is modeled as a 2nd order auto-regressive (AR) process.

As in [7, 9], the interaction model $p_0(\mathbf{X}_t)$ prevents two trackers from fitting the same object. This is achieved by exploiting a pairwise Markov Random Field (MRF) whose graph nodes are defined at each time step by the objects and the links by the set $\mathcal{C}$ of pairs of proximate objects. By defining an appropriate potential function $\phi(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})$, the interaction model, $p_0(\mathbf{X}_t) = \prod_{ij\in\mathcal{C}} \phi(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})$, enforces constraints in the dynamic model of objects based on the locations of the object's neighbors.

With these terms defined, the Monte Carlo approximation of the filtering distribution in Eq. 2 becomes

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) \approx C^{-1} p(\mathbf{Z}_t|\mathbf{X}_t) \prod_{ij\in\mathcal{C}} \phi(\mathbf{X}_{i,t}, \mathbf{X}_{j,t}) \times \sum_n p_V(\mathbf{X}_t|\mathbf{X}_{t-1}^{(n)}). \tag{5}$$

3.3 Observation Model

The observation model makes use of a single *foreground segmentation* observation source, $\mathbf{Z}_t$, as described in [10], from which two features are constructed. These features form the *zero-object likelihood* $p(\mathbf{Z}_t^{zero}|\mathbf{X}_{i,t})$ and the *multi-object likelihood* $p(\mathbf{Z}_{i,t}^{multi}|\mathbf{X}_{i,t})$. The *multi-object likelihood* is responsible for fitting the bounding boxes to foreground blobs and is defined for each object i present in the scene. The *zero-object likelihood* does not depend on the current number of objects, and is responsible for detecting new objects appearing in the scene. These terms are combined to form the overall likelihood,

$$p(\mathbf{Z}_t|\mathbf{X}_t) = \left[\prod_{i\in\mathcal{I}_t} p(\mathbf{Z}_{i,t}^{multi}|\mathbf{X}_{i,t})\right]^{\frac{1}{m_t}} p(\mathbf{Z}_t^{zero}|\mathbf{X}_t). \tag{6}$$

The *multi-object likelihood* for a given object i is defined by the response of a 2-D Gaussian centered at a learned position in precision-recall space (ν_l, ρ_l) to the values given by that objects current state (ν_t^i, ρ_t^i) (where ν and ρ are precision and recall, respectively) [9]. The *multi-object likelihood* terms in Eq. 6 are normalized by m_t to be invariant to changing numbers of objects. The precision for object i is defined as the area given by the intersection of the spatial support of object i and the foreground F, over the spatial support of object i. The recall of object i is defined as the area given by the intersection of the spatial support of object i and the dominant foreground blob it covers F_p, over the size of the foreground blob it is covering. A special case for recall occurs when multiple objects overlap the same foreground blob. In such situations, the upper term of the recall for each of the affected objects is computed as the intersection of the combined spatial supports with the foreground blob.

The *zero-object likelihood* gives low likelihoods to large areas of uncovered pixels, thus encouraging the model to place a tracker over all large-enough foreground patches. This is done by computing a

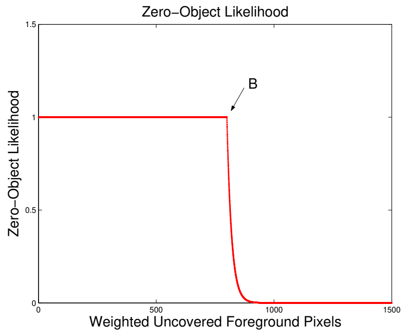

FIG. 4 – The Zero-Object likelihood defined over the number of weighted uncovered pixels.

weighted *uncovered pixel count* between the foreground segmentation and the current state $\mathbf{X}_t^i$. Pixels from blobs unassociated with any tracker i receive b times the weight of normal pixels. If U is the number of weighted uncovered foreground pixels, the zero-object likelihood is computed as

$$p(\mathbf{Z}_t^{zero}|\mathbf{X}_{i,t}) \propto exp(-\lambda \;\; max(0, U - B)) \tag{7}$$

where λ is a hyper-parameter and B is the amount of weighted uncovered foreground pixels to ignore before penalization begins (see Figure 4).

3.4 Inference with Trans-Dimensional MCMC

To solve the inference issue in large dimensional state-spaces, we have adopted the Reversible-Jump MCMC (RJMCMC) sampling scheme proposed by several authors [11, 9] to efficiently sample over the posterior distribution, which has been shown to be superior to a Sequential Importance Resampling (SIR) PF for joint distributions over multiple objects.

In RJMCMC, a Markov Chain is defined such that its stationary distribution is equal to the target distribution, Eq. 5 in our case. The Markov Chain must be defined over a variable-dimensional space to accommodate the varying number of objects, and is sampled using the Metropolis-Hastings (MH) algorithm. Starting from an arbitrary configuration, the algorithm proceeds by repetitively selecting a *move type*, υ from a set of moves Υ with prior probability p_υ and sampling a new configuration $\mathbf{X}^*$ from a proposal distribution $q(\mathbf{X}^*|\mathbf{X})$. The move can either change the dimensionality of the state (as in birth or death) or keep it fixed. The proposed configuration is then added to the Markov Chain with probability

$$\alpha = \min\left(1, \frac{p(\mathbf{X}^*)q(\mathbf{X}|\mathbf{X}^*)}{p(\mathbf{X})q(\mathbf{X}^*|\mathbf{X})}\right) \tag{8}$$

or the current configuration otherwise. The acceptance ratio α can be re-expressed through *dimension-matching* as

$$\alpha = \min\left(1, \frac{p(\mathbf{X}^*)p_\upsilon q_\upsilon(\mathbf{X})}{p(\mathbf{X})p_{\upsilon^*} q_{\upsilon^*}(\mathbf{X}^*)}\right) \tag{9}$$

where q_υ is a move-specific distribution and p_υ is the prior probability of choosing a particular move type.

We define three different move types in our model : birth, death, and update :

- **Birth** of a new object, implying a dimension increase, from m_t to $m_t + 1$.
- **Death** of an existing object, implying a dimension decrease, from m_t to $m_t - 1$.
- **Update** of the state parameters of an existing object according to the dynamic process described in Section 3.2.

The tracking solution at time t is determined by computing the mean estimate of the MCMC chain at time t.

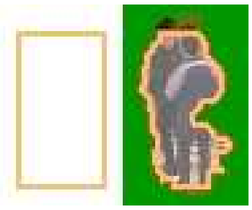

FIG. 5 – Object blobs (yellow contour on the right) are constructed from bounding boxes (left) and foreground segmentation (right).

4 Left-Luggage Detection Process

The second stage of our model is the *left-luggage detection process*. It is necessary to search for bags separately because the tracking model does not differentiate between people and bags. Also, the left-luggage detection process is necessary to overcome failures of the tracking model to retain consistent identities of people and bags over time.

The left-luggage detection process uses the output of the tracking model and the foreground segmentation, F, for each frame as input, identifies the luggage items, and determines if/when they are abandoned. The output of the tracking model contains the number of objects, their identities and locations, and parameters of the bounding boxes.

The left-luggage detection process relies on three critical assumptions about the properties of a left-luggage item :

1. Left-luggage items probably don't move.
2. Left-luggage items probably appear smaller than people.
3. Left-luggage items must have an owner.

The first assumption is made with the understanding that we are only searching for unattended (stationary) luggage. For this case it is a valid assumption. The more difficult task of detecting luggage moving with its owner requires more information than is provided by the tracker.

To tackle the left-luggage problem, the detection process breaks it into the following steps :

- **Step 1** : Identify the luggage item(s).
- **Step 2** : Identify the owners(s).
- **Step 3** : Test for alarm conditions.

Step 1. To identify the bags, we start by processing the tracker bounding boxes ($\overline{\mathbf{X}}_t$) and the foreground segmentation F to form object blobs by taking the intersection of the areas in each frame, $\overline{\mathbf{X}}_t^i \cap F$, as seen in Figure 5. The mean x and y positions of the blobs are computed, and the size of the blobs are recorded. A 5-frame sliding window is then used to calculate the blob velocities at each instant. Examples of blob size and velocity for sequence S1 can be seen in Figure 6. Following the intuition of our assumptions, likelihoods are defined such that small and slow moving blobs are more likely to be items of luggage :

$$p_s(B^i = 1|\overline{\mathbf{X}}_{1:t}^i) \propto \mathcal{N}(s_t^i, \mu_s, \sigma_s) \quad (10)$$

$$p_v(B^i = 1|\overline{\mathbf{X}}_{1:t}^i) \propto exp(-\lambda v_t^i) \quad (11)$$

where p_s is the size likelihood, p_v is the velocity likelihood, $B^i = 1$ indicates that blob i is a bag, s_t^i is the size of blob i and time t, μ_s is the mean bag blob size, σ_s is the bag blob variance, v_t^i is the blob velocity and λ is a hyper-parameter. These parameters were hand picked with knowledge of the data, but not tuned (see Section 5). The velocity and size likelihood terms can be seen in Figure 7. Because long-living blobs are more likely to be pieces of left-luggage, we sum the frame-wise likelihoods without normalizing by blob lifetime. The overall likelihood that a blob is a left-luggage item combines p_v and p_s,

$$p(B^i = 1|\overline{\mathbf{X}}_{1:t}^i) \propto \sum_{t=1:T} \mathcal{N}(s_t^i, \mu_s, \sigma_s) exp(-\lambda v_t^i). \quad (12)$$

An example of the overall likelihoods for each blob can be seen in the top panel of Figure 8.

The bag likelihood term $p(B^i = 1|\overline{\mathbf{X}}^i_{1:t})$ gives preference to long-lasting, slow, small objects as seen in the top of Figure 8. Bag candidates are selected by thresholding the likelihood, $p(B^i = 1|\overline{\mathbf{X}}^i_{1:t}) > T_b$. In the example case of S1, this means blobs 41 (which tracked the actual bag) and 18 (which tracked the owner of the bag) will be selected as bag candidates. Because of errors in tracking, there could be several unreliable bag candidates. Thus, in order to be identified as a bag, the candidates must pass the following additional criteria : (1) they must not lie at the borders of the image (preventing border artifacts), and (2) their stationary position must not lie on top of other bag candidates (this eliminates the problem of repeated trackers following the same bag).

The next part of the detection process is to determine the lifespan of the bag. The identities of the tracker are too unreliable to perform this alone, as they are prone to swapping and dying. But as we have assumed that items of left luggage do not move, we can use the segmented foreground image to reconstruct the lifespan of the bag. A shape template $\mathcal{T}^i$ is constructed from the longest segment of frames below a low velocity threshold, T_v, to model what the bag looks like when it is stationary. The template is a normalized sum of binary image patches taken from the segmented foreground image around the boundaries of the bag candidate blob with the background pixels values changed from 0 to -1.

A *bag existence* likelihood in a given frame is defined for the blob candidate by extracting image patches from the binary image at the stationary bag location $\mathcal{I}_t$ and performing an element-wise multiplication

$$p(E_t = 1|B^i) \propto \sum_u \sum_v \mathcal{T}^i(u,v) \times \mathcal{I}_t(u,v) \tag{13}$$

where $E_t = 1$ indicates that a bag exists at time t, and u and v are pixel indices. The bag existence likelihood is computed over the entire sequence for each bag candidate. An example of the existence likelihood for the bag found from blob 41 in the example can be seen in Figure 8 (bottom). A threshold, T_e, is defined as 80% of of the maximal likelihood value. The bag is defined as existing in frames with existence likelihoods above the threshold.

Step 2. In step 1, the existence and location of bags in the sequence were determined ; in step 2 we must identify the owner of the bag. Unlike left-luggage items, we cannot assume the owner will remain stationary, so we must rely on the results of the tracker to identify the owner.

Typically, when a piece of luggage is set down, the tracking results in a single bounding box contain both the owner and the bag. Separate bounding boxes only result when the owner moves away from the bag, in which case, one of two cases can occur : (1) the original bounding box follows the owner and a new box is born to track the bag, or (2) the original bounding box stays with the bag, and a new bounding box is born and follows the owner. Thus, to identify the owner of the bag, we inspect the history of the tracker present when the bag first appeared as determined by the bag existence likelihood. If that tracker moves away and dies while the bag remains stationary, it must be the one identifying the owner. In this case, we designate the blob results computed from the estimate of the tracker and the foreground segmentation as the owner. If the tracker remains with the bag and dies, we begin a search for nearby births of new trackers within radius r pixels. The first nearby birth is deemed the owner. If no nearby births are found, the bag has no owner, and violates assumption 3, so it is thrown out.

Step 3. With the bag and owner identified, and knowledge of their location in a given frame, the last task is straightforward : determining if/when the bag is left unattended and sounding the alarm (with one slight complication). Thus far, we have been working within the camera image plane. To transform our image coordinates to the world coordinate system, we computed the 2D homography between the image plane and the floor of the train station using calibration information from the floor pattern provided by PETS [1]. Using the homography matrix H, a set of coordinates in the image γ_1 is transformed to the floor plane of the train station γ_2 by the discrete linear transform (DLT) $\gamma_2 = H\gamma_1$ This can even be done for the image itself (see Figure 9).

However, the blob centroids of objects in the image do not lay on the floor plane, so using the DLT on these coordinates will yield incorrect locations in the world coordinate system. Thus, we estimate

Tab. 2 – Luggage Detection Results (for bags set on the floor, *even if never left unattended*). Mean results are computed over 5 runs for each sequence.

seq		# luggage items	x location	y location
S1	ground truth	1	.22	-.44
	mean result	1.0	.22	-.29
	error	0%	0.16 meters	
S2	ground truth	1	.34	-.52
	mean result	1.2	.23	-.31
	error	20%	0.22 meters	
S3	ground truth	1	.86	-.54
	mean result	0.0	-	-
	error	100%	N/A	
S4	ground truth	1	.24	-.27
	mean result	1.0	.13	.03
	error	0%	0.32 meters	
S5	ground truth	1	.34	-.56
	mean result	1.0	.24	-.49
	error	0%	0.13 meters	
S6	ground truth	1	.80	-.78
	mean result	1.0	.65	-.41
	error	0%	0.40 meters	
S7	ground truth	1	.35	-.57
	mean result	1.0	.32	-.39
	error	0%	0.19 meters	

the foot position of each blob by taking its bottommost y value and the mean x value, and estimate its world location by passing this point to the DLT. Now triggering warnings and alarms for unattended luggage can be performed by computing the distance between the bag and owner and counting frames.

It should be noted that the left-luggage detection process can be performed *online* using the 30s alarm window to search for bag items and their owners.

5 Results

To evaluate the performance of our model, a series of experiments was performed over the entire data corpus. Because the MCMC tracker is a stochastic process, five experimental runs were performed over each sequence, and the mean values computed over these runs. To speed up computation time, the size of the images was reduced to half resolution (360 $\times$ 288).

As previously mentioned, because no training set was provided, some amount of training was done on the test set. Specifically, the foreground precision parameters of the foreground model for the tracker were learned (by annotating 41 bounding boxes from sequences S1 and S3, computing the foreground precision, and simulating more data points by perturbing these annotations). Several other parameters were hand-selected including the e and sy limits of the bounding boxes and the parameters of the size and velocity models, but these values were not extensively tuned, and remained constant for all seven sequences. Specific parameter values used in our experiments were : $\mu_s = 380$, $\sigma_s = 10000$, $\lambda = 10$, $T_v = 1.5$, $T_b = 5000$, $r = 100$, $b = 3$, $B = 800$.

We separated the evaluation into two tasks : luggage detection (Table 2) and alarm detection (Table 3). Luggage detection refers to finding the correct number of pieces of luggage set on the floor and their locations, *even if they are never left unattended* (as is the case in S3). Alarm detection refers to the ability of the model to trigger an alarm or warning event when the conditions are met.

The error values reported for *# luggage items*, *# alarms*, and *# warnings* are computed similarly to the word error rate, often used in speech recognition :

$$\text{error rate} = \frac{\text{deletions} + \text{insertions}}{\text{events to detect}} \times 100 \tag{14}$$

As shown in Table 2, our model consistently detected each item of luggage in sequences S1, S2,

Tab. 3 – Alarm Detection Results (Mean results are computed over 5 runs for each sequence).

seq		# Alarms	# Warnings	Alarm time	Warning time
S1	ground truth	1	1	113.7s	113.0s
	mean result	1.0	1.0	112.9s	112.8s
	error	0%	0%	0.78s	0.18s
S2	ground truth	1	1	91.8s	91.2s
	mean result	1.0	1.2	90.8s	90.2s
	error	0%	20%	1.08s	1.05s
S3	ground truth	0	0	-	-
	mean result	0.0	0.0	-	-
	error	0%	0%	-	-
S4	ground truth	1	1	104.1s	103.4s
	mean result	0.0	0.0	-	-
	error	100%	100%	-	-
S5	ground truth	1	1	110.6s	110.4s
	mean result	1.0	1.0	110.6s	110.5s
	error	0%	0%	0.04s	0.45s
S6	ground truth	1	1	96.9s	96.3s
	mean result	1.0	1.0	96.9s	96.1s
	error	0%	0%	0.08s	0.18s
S7	ground truth	1	1	94.0s	92.7s
	mean result	1.0	1.0	90.4s	90.3s
	error	0%	0%	3.56s	2.38s

S4, S5, S6, and S7. A false positive (FP) bag was detected in one run of sequence S2 as a result of trash bins being moved and disrupting the foreground segmentation (the tracker mistook a trash bin for a piece of luggage). We report 100% error for detecting luggage items in S3, which is due to the fact that the owner never moves away from the bag, and takes the bag with him as he leaves the scene (never generating a very bag-like blob). However, it should be noted that for this sequence, our system correctly predicted 0 alarms and 0 warnings.

The spatial errors were typically small (ranging from 0.13 meters to 0.40 meters), though they could be improved by using multiple camera views to localize the objects, or by using the full resolution images. The standard deviation in x ranged from 0.006 to 0.04 meters, and in y from 0.009 to .09 meters (not shown in Table 2).

As seen in Table 3, our model successfully predicted alarm events in all sequences but S4, with the exception of a FP warning in S2. Of these sequences, the alarms and warnings were generated within 1.1s of the ground truth with the exception of S7 (the most difficult sequence). Standard deviation in alarm events was typically less than 1s, but approximately 2s for S2 and S7 (not shown in Table 3).

Our model reported a 100% error rate for detecting warnings and alarms in S4. In this sequence, the bag owner sets down his bag, another actor joins him, and the owner leaves. The second actor stays in close proximity to the bag for the duration of the sequence. In this case, our model repeatedly mistook the second actor as the bag owner, and erroneously did not trigger any alarms. This situation could have been avoided with better identity recognition in the tracker (perhaps by modeling object color).

In Figure 10, we present the tracker outputs for one of the runs on sequence S5. Colored contours are drawn around detected objects, and the item of luggage is highlighted after it is detected. Videos showing typical results for each of the sequences are available at `http://www.idiap.ch/~smith`.

6 Conclusion and Future Work

In this paper, we have presented a two-tiered solution to the left-luggage problem, wherein a detection process uses the output of an RJMCMC tracker to find abandoned pieces of luggage. We evaluated the model on the PETS 2006 data corpus which consisted of seven scenarios, and correctly predicted the alarm events in six of the seven scenarios with good accuracy. Despite less-than-perfect tracking results for a single camera view, the bag detection process was able to perform well for highlevel tasks. Possible avenues for future work include using multiple camera views and investigating methods for maintaining object identities in the tracker better.

Acknowledgements : This work was partly supported by the Swiss National Center of Competence in Research on Interactive Multimodal Information Management (IM2), the European Union 6th FWP IST Integrated Project AMI (Augmented Multi-party Interaction, FP6-506811, AMI-181), and the IST project CARETAKER.

Références

[1] R. Hartley and A. Zisserman, *Multiple View Geometry.* Cambridge University Press, 2000. 8

[2] D. Gibbons *et al*, Detecting Suspicious Background Changes in Video Surveillance of Busy Scenes. In *IEEE Proc. WACV-96*, Sarasota FL, USA, 1996. 2

[3] Milcent, G. and Cai, Y, Location Based Baggage Detection for Transit Vehicles. *Technical Report, CMU-CyLab-05-008, Carnegie Mellon University.* 2

[4] Haritaolu, I. *et al*, Backpack : Detection of People Carrying Objects Using Silhouettes. In *IEEE Proc. ICCV*, 1999. 3

[5] I. Haritaoglu and M. Flickner, Detection and tracking of shopping groups in stores. In *IEEE Proc. CVPR*, 2001. 3

[6] M. Isard and J. MacCormick, BRAMBLE : A Bayesian multi-blob tracker. In *IEEE Proc. ICCV*, Vancouver, July 2001. 4

[7] Z. Khan, T. Balch, and F. Dellaert, An MCMC-based particle filter for tracking multiple interacting targets. In *IEEE Proc. ECCV*, Prague, May 2004. 4, 5

[8] A. E. C. Pece, From cluster tracking to people counting. In *3rd PETS Workshop*, June 2002. 3

[9] K. Smith, D. Gatica-Perez, J.-M. Odobez, Using particles to track varying numbers of objects. In *IEEE Proc. CVPR*, June 2005. 4, 5, 6

[10] C. Stauffer and E. Grimson, Adaptive background mixture models for real-time tracking. In *IEEE Proc. CVPR*, June 1999. 5

[11] T. Zhao and R. Nevatia, Tracking multiple humans in crowded environment. In *IEEE Proc. CVPR*, Washington DC, June 2004. 4, 6

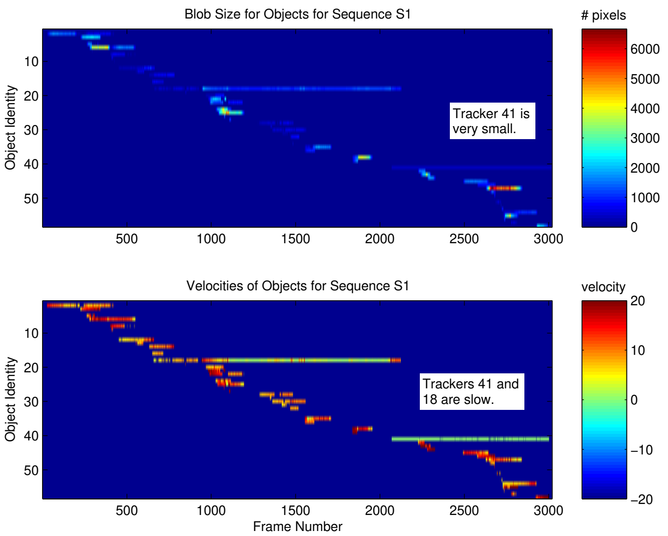

FIG. 6 – *(top)* Size and *(bottom)* velocity for each object computed over the course of sequence S1. In this example, blob 41 was tracking the bag

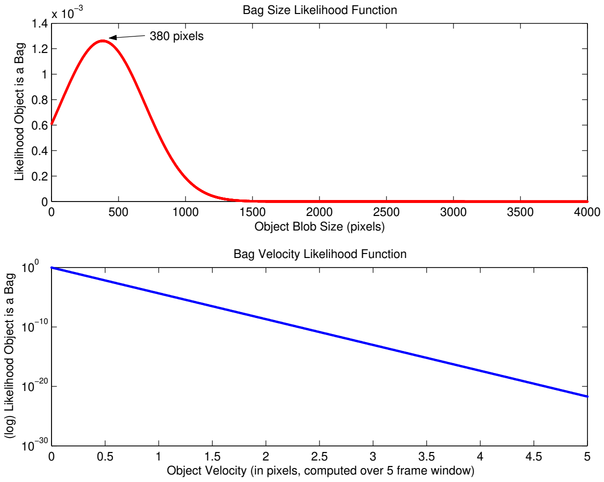

FIG. 7 – *(top)* Likelihood that a blob is a bag based on size (p_s). *(bottom)* Likelihood that a blob is a bag based on velocity (p_v).

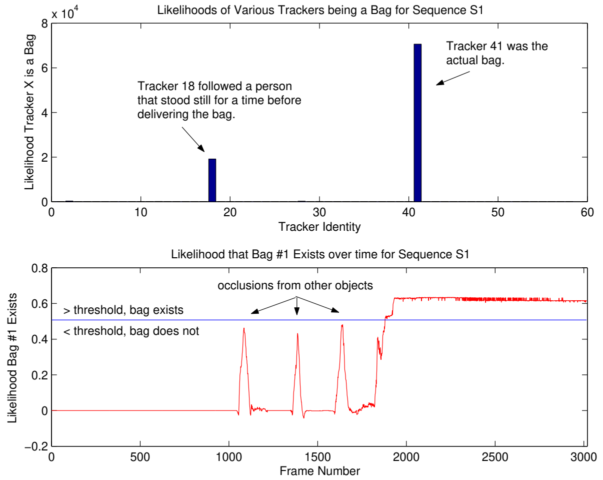

FIG. 8 – *(top)* The overall likelihoods of various tracked blobs in sequence S1. *(bottom)* The likelihood that bag candidate 1 exists for the length of sequence S1.

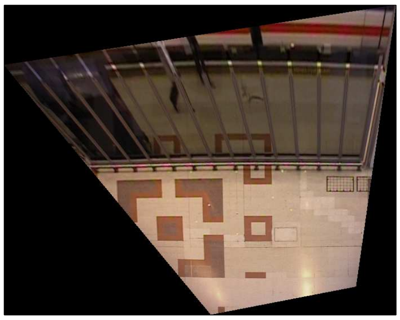

Fig. 9 – Image from S1 (camera 3) transformed by the DLT. In this plane, floor distance can be directly calculated between the bag and owner.

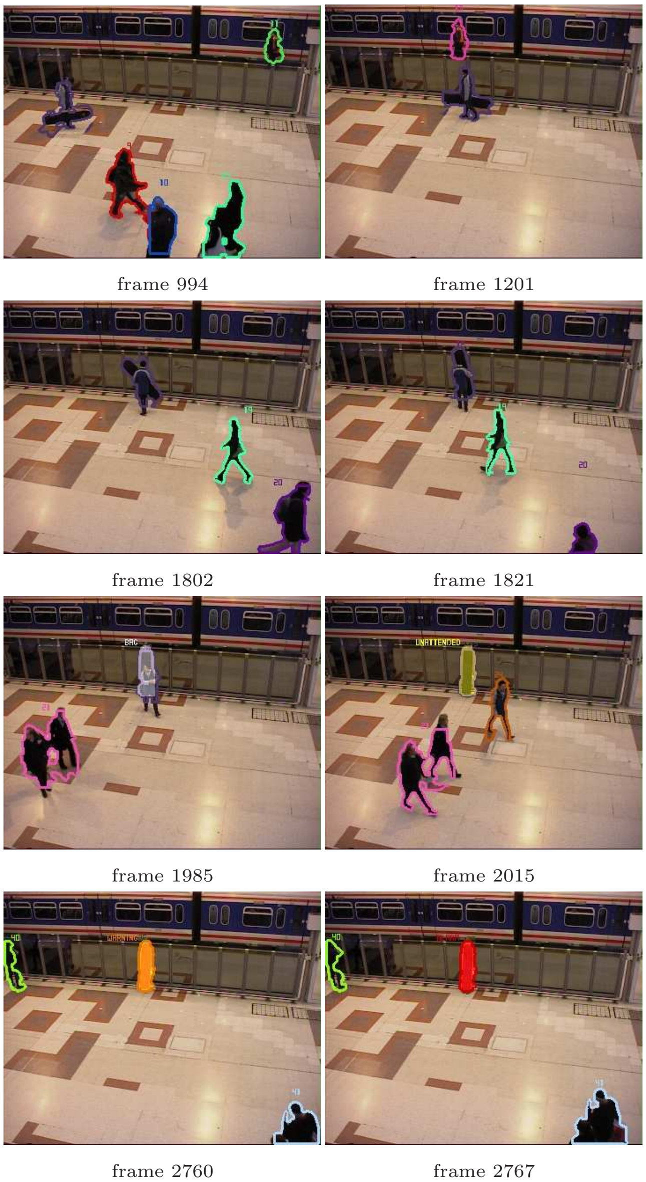

FIG. 10 – *Results : detecting left luggage items for sequence S5 from the PETS 2006 data corpus.* The luggage owner is seen arriving in frame 994, loiters for a bit in frame 1201, and places his ski equipment against the wall in frames 1802 and 1821. In frame 1985 the bag is detected, but the owner is still within the *attending* limits. In frame 2015, the model has determined he is more than 3 meters from the bag, and marks the bag as *unattended*. In frame 2760, 30s have passed since the owner left the 2 meter limit, and a warning is triggered (ground truth warning occurs in frame 2749). In frame 2767, 30s have passed since the owner left the 3 meter limit, and an alarm is triggered (ground truth alarm occurs in frame 2764).