Exploiting Scene Cues for Dropped Object Detection

Adolfo López-Méndez[1], Florent Monay[1], Jean-Marc Odobez[1]
[1]*IDIAP Research Institute, Martigny, Switzerland*
{*adolfo.lopez-mendez, florent.monay, jean-marc.odobez*}*@idiap.ch*

Keywords: Dropped Object Detection, Human Detection, Background Subtraction, Geometry Constraints

Abstract: This paper presents a method for the automated detection of dropped objects in surveillance scenarios, which is a very important task for abandoned object detection. Our method works in single views and exploits prior information of the scene, such as geometry or the fact that a number of false alarms are caused by known objects, such as humans. The proposed approach builds dropped object candidates by analyzing blobs obtained with a multi-layer background subtraction approach. The created dropped object candidates are then characterized both by appearance and by temporal aspects such as the estimated drop time. Next, we incorporate prior knowledge about the possible sizes and positions of dropped objects through an efficient filtering approach. Finally, the output of a human detector is exploited over in order to filter out static objects that are likely to be humans that remain still. Experimental results on the publicly available PETS2006 datasets and on several long sequences recorded in metro stations show the effectiveness of the proposed approach. Furthermore, our approach can operate in real-time.

1 INTRODUCTION

Automated detection of dropped or abandoned objects is one of the foremost concerns within automatic video surveillance systems. Being able to robustly detect suspicions items is a key issue in the protection of public spaces such as airports or train stations.

A cornerstone of abandoned object detection is the automated detection of static objects and, more specifically, of dropped objects. Here dropped objects are understood as items such as luggage, packets, etc. that remain static in the scene for a given period of time. Being able to automatically detect such objects over time is important in order to draw the attention of security operators or to identify object owners.

Several challenges are involved in dropped object detection: lighting conditions, changing backgrounds, object occlusions and false alarms caused by non-suspicious static objects such as humans that remain still. The latter is very notorious in typical surveillance scenarios (Fan and Pankanti, Sept). This problem can be alleviated by exploiting object detectors for known elements in the scene (such as humans). Surprisingly, and despite recent advances, existing dropped and abandoned object detection methods aiming at realistic surveillance scenarios (Fan and Pankanti, Sept)(Fan and Pankanti, pt 2)(Caro Campos et al., pt 2)(Tian et al., 2011) do not attempt to use state-of-the-art object detection methods (Dalal and Triggs, June) (Felzenszwalb et al., 2010), which could help in reducing false alarms caused by such known objects.

In this paper, we propose a dropped object detection method that leverages on prior information about the scenario. This information is mainly exploited in two ways. Firstly, with an efficient implementation of geometric constraints, that allows filtering objects based on contextual cues such as feasible ground plane positions and object sizes. Secondly, with a method that reasons about how likely is a static object to be generated by a human remaining still. For the latter, we rely on the output state of the art human detector (Dubout and Fleuret, 2012). Additionally, the proposed dropped object detector characterizes objects by their appearance and a set of temporal variables, such as the *set down time* (estimated time of dropping). This information is valuable for potential abandonment analysis. Experiments conducted in different datasets show the effectiveness of the proposed dropped object detection approach and the contribution of each one of the components of the system. Furthermore, the proposed method runs in real-time and has been deployed in real operational conditions on two metro stations.

2 RELATED WORK

Substantial research effort has been recently devoted to providing robust solutions to dropped and abandoned object detection (Fan and Pankanti, Sept)(Caro Campos et al., pt 2)(Smith et al., 2006)(Tian et al., 2011). Because dropped objects can belong to different classes, foreground detection and adaptive background modeling are central elements in most of the existing approaches.

A recent approach by Fan and Pankanti (Fan and Pankanti, pt 2) models temporally static objects with a finite state machine, where the similarity between objects and foreground matches is used as a cue to update the background model. The same authors later proposed a robust foreground detection approach (Fan and Pankanti, Sept) based on classifying patches as being foreground or background. Such a classification involves training RBF-SVMs using several texture and region features. Liao et al. (Liao et al., Sept) propose a foreground-mask sampling approach consisting in the intersection of foreground masks within a period of time. Campos et al. (Caro Campos et al., pt 2) rely on active contours obtained from foreground masks in order to detect abandoned and stolen objects.

Automated tracking of people in the scene is an important component of some approaches, especially those that aim at finding owners of abandoned objects (Liao et al., Sept)(Smith et al., 2006). Inferring people location in the scene might also be used to filter false alarms caused by still people. However, real surveillance scenes might be crowded, making tracking unreliable. Human detection is an alternative to tracking, but despite recent advances in the task of detecting humans in unconstrained scenes (Dalal and Triggs, June)(Felzenszwalb et al., 2010), current dropped and abandoned object detection approaches usually do not rely on human detection. In some methods, human detection is reduced to classifying foreground blobs as humans or static objects based on generic features such as blob size or skin color (Liao et al., Sept). According to (Fan and Pankanti, Sept), the main cause of false alarms are still persons being identified as static objects. This result emphasizes the importance of using advanced methods in order to detect known classes (e.g., people) in surveillance scenes.

3 DROPPED OBJECT DETECTOR

The proposed approach comprises several phases. Firstly, we rely on a multi-layer background subtraction algorithm (Yao and Odobez, 2007) to detect blobs that represent static objects. Our method uses these blobs to create dropped object candidates by gathering appearance and temporal aspects associated to the blobs. In this process, the prior knowledge about position and size of dropped objects is used to remove spurious candidates. Finally, because some detections are caused by still people in the scene, we integrate over time the output of a human detector to filter false alarms. An overview of the method is depicted in Figure 1.

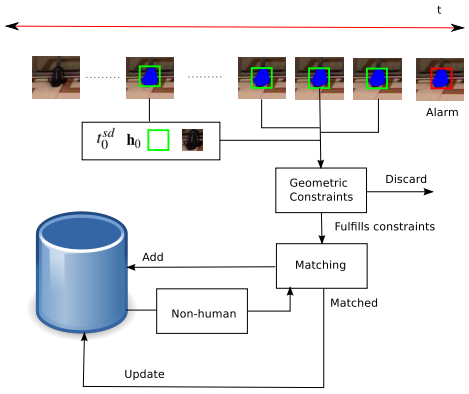


Figure 1: Overview of the proposed method (best viewed in color). A static object blob (blue pixels) is used to define a dropped candidate (represented by the dropping time or set down time t_i^{sd}, the vector $\mathbf{h}_i$, a bounding box $\mathbf{b}_i$ and a color patch $\mathbf{S}_i$, see details in Section 3.2). Incoming candidates undergo a geometric verification. If they fulfill geometric constraints, candidates are matched against those objects in the pool that are not likely to be generated by a human at time t. In the example, the stored object is repeatedly matched, indicating that it is present in the scene. After n_s seconds since the estimated set down time t_0^{sd}, the system triggers an alarm (red bounding box).

3.1 Multi-Layer Static Object Detection

Static objects are identified by relying on a multi-layer background modeling method (Yao and Odobez, 2007), which is described in this section for the sake of completeness. The motivation of the multi-layer framework is that layers can be regarded as stacked objects, where the bottom layer contains the dominant background, and successive layers contain static objects.

Multi-modal pixel representation The approach in (Yao and Odobez, 2007) is based on capturing the appearance statistics of each pixel $\mathbf{x}$ in terms of RGB color and Local Binary Patterns (LBP) (Heikkila and Pietikainen, 2006). At each time instant t, these statistics are represented by a set of modes that we denote as $\mathbf{M}^t(\mathbf{x}) = \{K_x^t, \{\mathbf{m}_k^t(\mathbf{x}), t_k^0(\mathbf{x})\}_{k=1,\cdots,K_x^t}\}$, where K_x^t is

the number of modes, $\mathbf{m}_k^t(\mathbf{x})$ represents the appearance statistics of one mode and $t_k^0(\mathbf{x})$ is the time instant when the k-th mode was observed for the first time at pixel $\mathbf{x}$. In the following, we drop $\mathbf{x}$ for readability.

Each mode is associated to two other variables: a weight w_k and a layer L_k. Each time a mode $\mathbf{m}_k$ is observed in the target pixel, weight w_k grows. On the contrary, if it is not observed, w_k decays. Hence, w_k represents how likely is $\mathbf{m}_k$ to belong to the background. Layer L_k determines whether a mode is a reliable background mode or not, i.e., if the mode has been observed for a sufficient amount of time. $L_k = 0$ indicates that $\mathbf{m}_k$ is not a reliable background mode, whereas $L_k > 0$ indicates a reliable background mode. Clearly, an important aspect of the presented method is the speed at which w_k grows and decays, since it determines the time elapsed until modes are set into reliable background layers. The rate of change is different depending on whether the weight is increased or decreased, and it is controlled by the mode weight learning rate α_w and a constant τ (Yao and Odobez, 2007). When a mode is matched its weight increases according to the following expressions:

$$\begin{aligned} w_k^t &= (1-\alpha_w^i)w_k^{t-1} + \alpha_w^i \\ \hat{w}_k^t &= \max(\hat{w}_k^{t-1}, w_k^t) \\ \alpha_w^i &= \alpha_w(1+\tau w_k^{t-1}) \end{aligned} \tag{1}$$

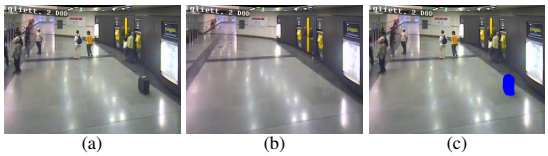
where $\hat{w}_k$ is the maximum weight achieved by the k-th mode. The rest of existing modes at a given pixel, which are not matched, get their weight decreased as follows:

$$\begin{aligned} w_k^t &= (1-\alpha_w^d)w_k^{t-1} \\ \alpha_w^d &= \frac{\alpha_w}{(1+\tau\hat{w}_k^{t-1})} \end{aligned} \tag{2}$$

To simplify the parameters of the proposed dropped object detection method, we fix $\tau = 5$. This value was found to be convenient after several experiments on background subtraction.

Layer management Layer management consists in assigning a layer to each mode based on their respective weights. The goal is to represent the scene as a set of layers containing static objects, where the *most static* object, the *true background* is represented by layer $L_k = 1$ (see Fig. 2(b)). To this end, a *layer addition* and *layer removal* mechanisms are proposed.

Layer addition takes place when a pixel mode $\mathbf{m}_{\bar{k}}$ was observed during enough time. This is equivalent to say that such a mode is promoted to a layer mode when its weight is larger than a threshold T_{bw}. Mode $\mathbf{m}_{\bar{k}}$ is assigned the first available layer.



(a) (b) (c)

Figure 2: Multi-Layer Static Object Detection (best viewed in color) (a) Original image (b) Layer 1 (*true background*) (c) Generated static object blob from layers $L_k > 1$ (blue pixels)

Analogously to layer addition, modes that have not been observed $\mathbf{m}_r$ during a sufficient amount of time must be removed from the background layers. That is, when the weight of a mode $\mathbf{m}_r$ drops below a threshold T_{br}, it indicates that the mode disappeared or that it has been occluded for a long time. The mode is then set as a non-layer mode.

The described removal procedure might be slow for static object analysis. Consider the case in which an object is left in the scene and, after some time it is promoted to layer 2 (1 is assigned to *true background* modes). If the object is removed from the scene, the described removal approach will need a long time to remove pixel modes belonging to the object from layer 2. This is clearly inefficient, since background modes in layer 1 will be observed again, indicating that the object was removed (we assume that bottom layers cannot *get in front* of top layers). Based on this observation, a second removal mechanism is defined.

When a mode $\mathbf{m}_a$ in layer L_a re-appears and it is observed for a sufficient proportion of time over a given period (its weight increases), the background layers on top of L_a are removed. The ultimate criterion to decide when such removal takes place is determined by the percentage ($T_{ba} \in (0,1)$) of total weight hold by $\mathbf{m}_a^t$ (see Algorithm 1).

Algorithm 1: Removal due to $\mathbf{m}_a$ re-appearing

```
if m_a^t re-appears and w_a^t / Σ_{k=1}^{K^t} w_k^t > T_ba then
    for k = 1...K^t do
        if L_k^t > L_a^t then
            remove the mode m_k^t
        end
    end
end
```

Layer post-processing All layers above the first, which is considered to be the true background, are regarded as candidates for static objects. We first gather the content of these layers into a single image whose non-zero pixels represent modes that are likely to belong to static objects (see Fig. 2c). To remove noise and obtain blobs from the objects modeled in this image, we apply a Gaussian smoothing (in our

experiments we use a kernel of 11x11 and σ = 3.5). We subsequently apply a morphological opening with squared structuring element of 5x5 to further remove artifacts.

3.2 Dropped Object Management

The output of the multi-layer static object detection block is a set of blobs on a per-frame basis. Blobs being detected as static objects are further processed in order to decide whether they might be dropped objects. In the following, we describe the method we propose in order to keep track of potentially dropped objects.

Let a dropped object candidate be defined using a static object blob. Specifically, a candidate i is represented by the tuple $\{t_i^{sd}, \mathbf{h}_i, \mathbf{b}_i, \mathbf{S}_i\}$ (see Fig. 1). The parameter t_i^{sd} is the set down time and represents the time instant when the object was dropped. The set down time is estimated upon candidate creation (see details on the estimation below) and it is an important feature of our approach, since it is used to trigger alarms, but also allows users to find the moment when the object was dropped, thus being important for potential abandonment analysis. The vector $\mathbf{h}_i$ accumulates evidence of the dropped object along time. This evidence will be ultimately used to trigger alarms for dropped objects. Specifically, for a given time instant t, $\mathbf{h}_i(t)$ is non-zero if the object was observed at time t. This history only contains the time instants between the object creation and its deletion. Finally, the appearance of the candidate is represented by the bounding box coordinates $\mathbf{b}_i$ that enclose the blob and the RGB patch $\mathbf{S}_i$ inside such a bounding box.

Set Down Time Estimation An important aspect of the proposed dropped object detection is the set down time. This variable is estimated using the pixel creation time described in Section 3.1. At a give time instant t, each pixel $\mathbf{x}$ in a static object blob has an associated set of K_x^t modes with their respective $t_k^0(\mathbf{x}) \in \mathbb{N}$ creation times.

A mode creation time set is defined by collecting the pixel mode creation times of the most recent reliable background modes (excluding those in layer 1):

$$\mathcal{M} = \{t_k^0(\mathbf{x}) | L_k^t > 1, L_k^t = \max\{L_{\tilde{k}}^t\}_{\tilde{k}=1\ldots K_x^t}\} \quad (3)$$

Based on the collected set of times, our goal is to estimate the set down time of the object represented by the blob. The likelihood of t_{sd}^i can be derived by non-parametric density estimation from the observed creation times:

$$l(\mathcal{M}) = \sum_{t=-\infty}^{\infty} \phi(t_\psi^0)\delta(t - t_\psi^0) \quad (4)$$

where t_ψ^0 are the unique creation times within $\mathcal{M}$ and $\phi(t_\psi^0)$ represents the counts for pixel creation time t_ψ^0. Since $\mathcal{M}$ is created from pixel-wise measurements, the distribution of creation times in $\mathcal{M}$ is, in general, multimodal.

Fortunately, we can use prior information given by the time elapsed from the drop until the object is detected by our approach (it generates a blob). This time, which we denote as t_{ud}, is common for every detected object, and it depends on the mode weight learning rate α_w and on a constant τ (see Section 3.1). Hence, we can experimentally learn a regressor $t_{ud}(\alpha_w, \tau, T_{bw}, T_{br}, T_{ba})$. In practice, after background subtraction and dropped object detection experiments in validation sets, we found convenient to set $\tau = 5, T_{bw} = 0.9, T_{br} = 0.0001, T_{ba} = 0.6$ (see Section 3.1). This simplifies the regressor to $t_{ud}(\alpha_w)$.

To generate the data for learning $t_{ud}(\alpha_w)$, we run experiments with synthetic objects, and we measure the time elapsed since the object is created until it is detected as a static blob. Specifically, 10 equispaced values of α_w in the interval $[0.01, 0.1]$ are tested. Then, a nonlinear exponential regressor of the form $t_{ud}(\alpha_w) = \frac{A}{\alpha_w}\exp(-\lambda\alpha_w) + C$ is fitted to the data.

The set down time t_i^{sd} is estimated using mean-shift (Comaniciu and Meer, 2002) on $l(\mathcal{M})$, with $t_{ud}(\alpha_w)$ as initialization point.

Matching Dropped Object Candidates Let us denote $\mathcal{D}$ the pool of potential dropped objects. Since the pool $\mathcal{D}$ is initially empty, the first candidate is directly stored in the pool.

Subsequent candidates are compared against the objects in $\mathcal{D}$. In order to determine if candidate j matches an object i contained in $\mathcal{D}$, we define two tests. The first one is the overlap between bounding boxes:

$$f_{ij} = \frac{2\ \text{area}(\mathbf{b}_i \cap \mathbf{b}_j)}{\text{area}(\mathbf{b}_i) + \text{area}(\mathbf{b}_j)} \quad (5)$$

the second is the averaged L_2 norm between the color patches of both objects in the overlapping area:

$$d_{ij} = \frac{\|\mathbf{S}_i(\mathbf{b}_i \cap \mathbf{b}_j) - \mathbf{S}_j(\mathbf{b}_i \cap \mathbf{b}_j)\|_2}{\text{area}(\mathbf{b}_i \cap \mathbf{b}_j)} \quad (6)$$

where patches have pixel values between 0 and 1.

We say that an object i is matched with candidate j if $f_{ij} > T_f$ and $d_{ij} < T_d$.

If at time t a match occurs, $\mathbf{h}_i(t)$ is set to 1; otherwise is set to 0. Similarly, a candidate j which is not

matched against any object in $\mathcal{D}$ is added to the pool as a new object.

The appearance model $\{\mathbf{S}_i, \mathbf{b}_i\}$ of a stored object being matched is updated by incorporating $\{\mathbf{S}_j, \mathbf{b}_j\}$. This is done by first updating the bounding box as:

$$\mathbf{b}_i = \alpha \mathbf{b}_i + (1-\alpha)\mathbf{b}_j \quad (7)$$

where $\alpha \in (0,1)$ is a learning rate. In practice, we set $\alpha = 0.9$. Then, we obtain $\mathbf{S}_i$ as the patch inside the updated $\mathbf{b}_i$.

Objects i in $\mathcal{D}$ that have not been matched during the last minutes are removed.

3.3 Geometric Filtering

If camera calibration is available, one can take advantage of the prior knowledge about dropped object expected size and ground plane position in the scene. Our method incorporates such prior information in a look-up-table (LUT). In test time, LUTs allow to efficiently discard dropped object candidates that do not fulfill the implemented geometric constraints, i.e., we do not consider them to be neither stored nor matched against the candidates in the pool $\mathcal{D}$.

The LUT is computed offline or during the algorithm initialization, and it requires camera calibration parameters and a binary mask denoting the possible dropped object locations (i.e., the floor) (see Fig. 3) [1]. We start by densely sampling the binary mask where pixel values are different than zero. This procedure gives a list of possible ground plane positions where objects can be dropped. In each of these positions we generate a 3D object of minimum and maximum allowed sizes. Such objects are projected onto the image plane and modeled by their enclosing bounding boxes (see Fig. 3). These bounding boxes model the minimum and maximum apparent size of dropped objects. The set of bounding boxes after dense sampling in the image plane is stored in a LUT.

Figure 3: Example of the proposed geometric constraints. Left: scene Middle: binary mask with valid ground plane positions. Right: Sampled ground plane positions (red dots) and valid objects (green bounding boxes). The picture depicts a sparse version of the actual constraints

In test time, we efficiently retrieve the minimum and maximum allowed sizes by looking at the closest pixel position with associated minimum and maximum bounding boxes. A dropped object candidate is considered valid if its associated bounding box $\mathbf{b}_j$ falls inside the area left in between the minimum and maximum retrieved bounding boxes. If the object lies on a non-valid position, there will not exist a pair of valid bounding boxes such that constraints are fulfilled. The described procedure is very efficient since bounding box retrieval is fast thanks to the LUT and checking the constraints involves comparing bounding boxes.

[1] If the mask is not provided then the whole image plane is considered as valid

(a) (b) (c)

Figure 4: Example of still human filtering (a) Detected static objects causing dropped object alarm (b) Human detections in frames where static objects triggered alarms are used to filter out false alarms (c) Dropped object alarms after filtering.

3.4 Filtering Static Humans

A particular issue in most abandoned object detection scenarios is that one typically has more information about uninteresting classes (mostly humans and/or cars) than the target class (abandoned objects can belong to different classes and their appearance might strongly differ). We leverage on this prior information to filter spurious dropped object detections that are in fact caused by people standing still. For that matter, we rely on human detection with deformable part models (Felzenszwalb et al., 2010) and more specifically, on the fast implementation proposed in (Dubout and Fleuret, 2012). These methods obtain state-of-the-art performance in detecting humans in several surveillance scenes.

As described in Section 3.2, the vector $\mathbf{h}_i$ denotes whether the i-th object has been observed in the scene during specific time instants. Our goal is to determine whether an observation of the i-th object is in fact generated by a person standing still, in order to be able to avoid alarms. This is achieved by computing the intersection over union (IOU) between $\mathbf{b}_i$ and each one of the bounding boxes generated by the human detector at time t. If the maximum IOU is above a threshold T_h, we consider that the i-th object is not matched and $\mathbf{h}_i(t)$ is set to 0. That is, even if it exists a candidate j that repeatedly matches the object i in $\mathcal{D}$, the proposed human filtering strategy avoids accumulating dropped object evidence.

3.5 Dropped Object Alarms

Dropped objects that stay in the scene for a given amount of seconds n_s must generate alarms. Consequently, if for the i-th object there is enough evidence n_s seconds after the set down time, the proposed system will trigger an alarm. By denoting t the current time instant and Δt a given time interval, we formulate the mentioned conditions as follows:

- $t - t_i^{sd} > n_s$
- $\frac{1}{|\Delta t|} \sum_{n=t-\Delta t+1}^{t} \mathbf{h}_i(n) > \theta$

  where $\theta \in (0,1)$ is the detection score.

4 EXPERIMENTAL RESULTS

We conduct several experiments in order to evaluate the effectiveness of our approach. We use different datasets: the publicly available PETS2006 dataset (PETS 2006,) and two datasets recorded in Torino and Paris metro stations. The use of the latter datasets is motivated by the fact that the proposed approach is deployed in those scenarios. Compared to PETS2006, these datasets capture more realistic conditions.

Hereinafter, we use ML-DOD to denote our baseline multi-layer dropped object detection approach. ML-DOD+Geometry denotes the use of geometry whereas ML-DOD-HD denotes the baseline plus human filtering.

In all the experiments, we run the algorithms at 1 fps. This is convenient since we do not need more redundancy and because, for all the selected datasets, our C++ implementation on a desktop PC Intel i7 with 3.2 GHz needs less than 1 second to process a frame. In our implementation we use several constant parameters: $T_{bw} = 0.9$, $T_{br} = 0.0001$, $T_{ba} = 0.6, \tau = 5, T_f = 0.3, T_d = 0.02, T_h = 0.06$. We employed several validation datasets in order to set these parameters. This yields a dropped object detector approach depending on three variable parameters: the number of seconds after which alarms should be triggered n_s, the weight learning rate α_w - which can be interpreted as the speed at which objects are incorporated to static background layers - and the detection score θ. From a user viewpoint, this reduced set of variable parameters renders a more understandable algorithm.

For the human detector (Dubout and Fleuret, 2012) we use a person model trained in the INRIA person dataset (INRIA,) and we set a detection threshold of -0.1.

Finally, for the geometry constraints, we employ scenario specific masks and we tune the object sizes according to expected values (in all the experiments we allow objects to have minimum width and height of 20 cm, while we conveniently vary the maximum).

Figure 5: Examples of detected dropped objects in PETS2006 (best viewed in color).

4.1 PETS2006

PETS2006 dataset (PETS 2006,) contains 7 sequences for evaluation of abandoned luggage detection. All 7 sequences have been recorded with 4 calibrated 720x576 cameras, at 25 fps. In our experiments we use view 3, since it is the most commonly used by state-of-the-art approaches. Overall, this dataset contains approximately 15 min of video data for evaluation.

In this work, we focus on dropped luggage detection, understood as the task of finding luggage being static in the scene for more than 30 seconds. In PETS2006 datasets, there are a total of 6 sequences with objects fulfilling the dropped object criteria, and they are the same sequences annotated with abandonment alarms.

To determine whether we correctly detect dropped objects, we determine whether spatio-temporal overlap between the alarms triggered by our approach and the ground truth exist.

We run the proposed ML-DOD with all the possible configurations (ML-DOD, ML-DOD+Geometry, ML-DOD-HD, ML-DOD-HD+Geometry). Additionally, we experiment with $\alpha_w = [0.05, 0.1], n_s = 30$ seconds and thresholds $\theta \in [0,1]$. In this dataset we allow objects of big sizes (maximum width and height of 2 m) due to the presence of big luggage (skis). Example dropped object detections are shown in Fig. 5.

In our experiments only ML-DOD-HD+Geometry constraints is able to attain 100% precision and recall in this dataset for both values of $\alpha_w = [0.05, 0.1]$. For $\alpha = 0.05$, when geometric constraints or human filtering are not used the algorithm fires false alarms on wrong objects in the background or on humans. This is summarized in Table 1, where the best performance for $\alpha = 0.05$ is shown.

Additionally, we manually annotate the dropping times, in order to evaluate the accuracy of the estimated set down time t_i^{sd} for the alarmed objects. The proposed method has a t_i^{sd} estimation error of 0.5 ± 0.7 seconds, with a position estimation error of 28.6 ± 24.2 cm. Hence, even if the proposed method

Method	Precision	Recall
(Smith et al., 2006)	0.83	1
(Tian et al., 2011)	0.88	1
ML-DOD	0.67	1
ML-DOD + Geometry	0.86	1
ML-DOD-HD	0.75	1
ML-DOD-HD +Geometry	1	1

Table 1: Comparative results for dropped object detection in PETS2006. ($\alpha = 0.05$)

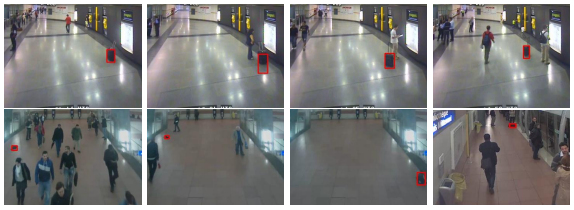


Figure 6: Examples successfully detected dropped objects in metro station sequences (best viewed in color). Top: Torino metro station Bottom: Paris Metro station.

does not target abandonment analysis, the set down time feature allows to accurately locate the time instant when the object was dropped without having to analyze past frames.

We compare the proposed method to existing approaches reporting dropped object results in PETS2006 dataset (see Table 1). The approach by (Smith et al., 2006) detects one bag that is not dropped for a sufficient amount of time and some objects in the background, while (Tian et al., 2011) detects a person as dropped luggage. ML-DOD-HD+Geometry outperforms these approaches by using prior knowledge about object positioning and the human filtering strategy.

4.2 Torino Metro Station

The Torino metro station dataset consists of 2 sequences of 45 min. each. Videos are recorded with one single camera at 5 fps. The image resolution is 704x288.

We manually annotated the ground-truth of dropped object events for evaluation. Similarly to the PETS2006 dataset, the ground-truth events correspond to objects that are dropped for more than 30 seconds. We annotate the bounding boxes and dropping time of each object. In total, we annotated 24 drops fulfilling the 30 seconds criterion. This is a challenging dataset, since the station is crowded, some dropped objects are left for slightly more than 30 seconds and these objects get often occluded by people.

In order to evaluate the accuracy of the proposed methods, we compute spatio-temporal overlaps between predicted and ground truth dropped objects.

Method	Video 1		Video 2	
	Prec.	Rec.	Prec.	Rec.
ML-DOD	0.11	1	0.15	0.83
ML-DOD + Geometry	0.23	1	0.24	0.83
ML-DOD-HD	0.14	1	0.19	0.83
ML-DOD-HD + Geometry	0.33	1	0.36	0.83

Table 2: Dropped Object detection results in Torino metro dataset for $n_s = 20$ seconds.

We run the proposed ML-DOD with all the possible configurations and different ranges of parameters. Specifically, $\alpha_w \in [0.05, 0.1], n_s = 20$ seconds and thresholds $\theta \in [0,1]$. We choose $n_s = 20$ since several dropped objects stay static for slightly more than 30 seconds. Since objects get often occluded, our approach requires an additional time gap to properly trigger dropped object alarms.

Similarly to PETS, we allow objects of rather big size (maximum width and height of 1.2 m). For a security operator, the target is to detect the majority of dropped objects. For that matter, we report the maximum precision at the best achieved recall obtained by each variation of the proposed approach (see Table 2).

Results in this dataset show that despite the challenging evaluation conditions, our approach is able to detect the majority of the dropped objects (examples are shown in Fig. 6). Precision values reveal that adding geometric constraints yields larger accuracy increase compared to human filtering alone. By varying the sizes of objects we found that the most determining aspect is the ground plane position. Without these constraints, the proposed method can trigger alarms in almost every position of the image. Most interestingly, accuracy is further increased when geometric constraints and human filtering are combined, which suggests that these cues are complementary.

For the correctly detected dropped objects, the average set down time estimation error is 4.2 ± 3.3 seconds. Considering that some objects have occlusions during the dropping instant, this is an accurate estimate.

4.3 Paris Metro Dataset

The Paris metro station dataset consists of 2 sequences recorded at the footbridge and one of the platforms of the station respectively. In total, both recordings consist of 2 hours of video data. Videos are recorded with one single camera at 25 fps. The image resolutions are 640x480 and 800x600 respectively.

Method	Footbridge		Platform	
	Prec.	Rec.	Prec.	Rec.
ML-DOD	0.43	1	0.03	1
ML-DOD + Geometry	1	1	0.2	1
ML-DOD-HD	0.43	1	0.04	1
ML-DOD-HD + Geometry	1	1	0.5	1

Table 3: Dropped Object detection results in Paris metro dataset for $n_s = 90$ seconds.

We follow the same annotation and evaluation methodology as in the Torino dataset experiments (see Section 4.2), with the difference that in this dataset we consider objects that remain static for a minimum of 2 minutes, which renders more realistic evaluation conditions (more similar to (Fan and Pankanti, pt 2)(Fan and Pankanti, Sept)). A total of 4 objects are annotated in both videos (3 objects on the first video and 1 on the second).

ML-DOD is run with all the possible configurations and $\alpha_w \in [0.02, 0.05], n_s = 90$ seconds (similarly to Torino, some objects are removed just after 2 minutes, thus requiring extra time for triggering alarms robustly). Since larger n_s are targeted, lower α_w values are chosen. Regarding the geometric constraints, we allow objects of maximum width and height of 80 cm. As in the Torino experiments, we report the best precision at maximum recall (see Table 3).

This dataset serves to evaluate the ability of the proposed approach to deal with scenes where people normally stay still, such as the metro platform (see Fig. 6). Both the geometry constraints and the filtering of humans are very important in order to successfully deal with these cases. Clearly, in an area where people is transiting, human filtering does not help in increasing the precision (see Table 3). On the contrary, it gives a performance boost on the platform.

While the proposed approach is generally very accurate in estimating the set down time, it predicts one of the dropping times almost 1 minute in advance, hence yielding an average of 11.5 ± 9.9 seconds. Such a failure is caused by mode creation times generated by the owner (who remains in the same position before dropping the object). The average set down time error for the remaining cases is 1.4 ± 1.2 seconds.

5 CONCLUSIONS

We have presented an automatic approach for detecting dropped objects in surveillance scenarios. Our method leverages on multiple elements to address this problem. First, objects are characterized by their appearance and temporal statistics that allow to accurately retrieve the dropping time. Secondly, an efficient implementation of a set of position and size constraints that allow removing a number of false alarms. Finally, the proposed approach leverages on state-of-the-art detectors to filters spurious objects produced by known classes such as still humans. Experimental results conducted on several datasets show the effectiveness of our approach that, in addition, runs in real-time.

6 ACKNOWLEDGEMENT

This work was supported by the Integrated Project VANAHEIM (248907) of the European Union under the 7th framework program.

REFERENCES

Caro Campos, L., SanMiguel, J., and Martinez, J. (30 2011-Sept. 2). Discrimination of abandoned and stolen object based on active contours. In *AVSS 2011*, pages 101–106.

Comaniciu, D. and Meer, P. (2002). Mean shift: a robust approach toward feature space analysis. *TPAMI*, 24(5):603–619.

Dalal, N. and Triggs, B. (June). Histograms of oriented gradients for human detection. In *CVPR 2005*, volume 1, pages 886–893 vol. 1.

Dubout, C. and Fleuret, F. (2012). Exact acceleration of linear object detectors. In *ECCV 2012*, pages 301–311, Berlin, Heidelberg. Springer-Verlag.

Fan, Q. and Pankanti, S. (30 2011-Sept. 2). Modeling of temporarily static objects for robust abandoned object detection in urban surveillance. In *AVSS 2011*, pages 36–41.

Fan, Q. and Pankanti, S. (Sept.). Robust foreground and abandonment analysis for large-scale abandoned object detection in complex surveillance videos. In *AVSS 2012*, pages 58–63.

Felzenszwalb, P., Girshick, R., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *TPAMI*, 32(9):1627–1645.

Heikkila, M. and Pietikainen, M. (April 2006). A texture-based method for modeling the background and detecting moving objects. *TPAMI*, 28(4):657–662.

INRIA. http://pascal.inrialpes.fr/data/human/.

Liao, H.-H., Chang, J.-Y., and Chen, L.-G. (Sept.). A localized approach to abandoned luggage detection with foreground-mask sampling. In *AVSS 2008*, pages 132–139.

PETS 2006. http://www.cvg.rdg.ac.uk/PETS2006/data.html.

Smith, K. C., Quelhas, P., and Gatica-Perez, D. (2006). Detecting abandoned luggage items in a public space. In *IEEE PETS*.

Tian, Y., Feris, R., Liu, H., Hampapur, A., and Sun, M.-T. (Sept. 2011). Robust detection of abandoned and removed objects in complex surveillance videos. *TSMC-C*, 41(5):565–576.

Yao, J. and Odobez, J.-M. (2007). Multi-layer background subtraction based on color and texture. In *CVPR 2007*, pages 1–8.