Multimed Tools Appl (2016) 75:761–786
DOI 10.1007/s11042-014-2324-4

Extraction of stable foreground image regions for unattended luggage detection

Grzegorz Szwoch

Received: 4 March 2014 / Revised: 19 September 2014 / Accepted: 13 October 2014 /
Published online: 30 October 2014

Abstract A novel approach to detection of stationary objects in the video stream is presented. Stationary objects are these separated from the static background, but remaining motionless for a prolonged time. Extraction of stationary objects from images is useful in automatic detection of unattended luggage. The proposed algorithm is based on detection of image regions containing foreground image pixels having stable values in time and checking their correspondence with the detected moving objects. In the first stage of the algorithm, stability of individual pixels belonging to moving objects is tested using a model constructed from vectors. Next, clusters of pixels with stable color and brightness are extracted from the image and related to contours of the detected moving objects. This way, stationary (previously moving) objects are detected. False contours of objects removed from the background are also found and discarded from the analysis. The results of the algorithm may be analyzed further by the classifier, separating luggage from other objects, and the decision system for unattended luggage detection. The main focus of the paper is on the algorithm for extraction of stable image regions. However, a complete framework for unattended luggage detection is also presented in order to show that the proposed approach provides data for successful event detection. The results of experiments in which the proposed algorithm was validated using both standard datasets and video recordings from a real airport security system are presented and discussed.

Keywords Image analysis · Object detection · Video surveillance · Automatic event detection

1 Introduction

Automatic detection of important security threats by means of an unsupervised analysis of video streams from the surveillance cameras is the current thread in modern monitoring systems and in scientific research [20, 25]. One of the most common events representing a potential security threat is leaving unattended luggage in public spaces, e.g. in airport halls. Security staff at the airports indicate a great need for automatic detection of unattended luggage in real conditions. Although this problem has been a subject of scientific research for many

G. Szwoch (✉)
Faculty of Electronics, Telecommunication and Informatics, Multimedia Systems Department, Gdansk University of Technology, 80-233 GdańskNarutowicza 11/12, Poland
e-mail: greg@sound.eti.pg.gda.pl

years, it still has not reached a state in which most of the state-of-the-art commercial systems would have a reliable implementation of such a detector available for the users. The majority of the existing solutions are limited to simple scenes and do not cope with detection in areas with intensive object movement, such as airports.

A successful detection of unattended luggage requires a multi-stage video analysis. Spengler and Schiele [31] defined an abandoned object as a "non-human foreground which keeps still over a certain period of time and without humans being close by". Therefore, foreground objects that do not move (stationary objects) have to be detected, then these objects have to be identified as luggage and finally, presence of humans (preferably, the luggage owner) in their neighborhood has to be tested [37].

Part of the INDECT scientific project [16], in which the author participated, was related to development of a multi-stage framework for video content analysis and automatic threat detection. [7]. This complex, modular system performs video analysis from the low-level pixel-based image analysis to the interpretation of video content and decision making. Incorporation of the unattended luggage detection into this framework required that: (a) the detector fits into the analysis scheme, (b) it utilizes results of the low-level analysis (e.g. background subtraction) used also by other modules, in order to avoid redundancy in processing, (c) it is able to perform an online video stream analysis without omitting frames from the processing and without introducing significant delays in the processed stream, and finally (d) it performs detection of unattended luggage in real environments with a number of false positive and false negative decisions kept within reasonable limits. It was found that most of the state-of-the-art algorithms (reviewed in the next Section) provide good accuracy of detection, but they do not fulfill one or more of the abovementioned conditions, usually either not being able to fit easily into the processing scheme or performing too intensive processing due to their complexity (thus not achieving a performance level required for online systems). Therefore, the task of the work described here was to engineer an alternative solution to the discussed problem, which provides a satisfactory balance between the requirements for the algorithm accuracy and efficiency. From the analysis of existing approaches it was found that the main problem in unattended luggage detection is how to extract image regions representing stationary luggage in the video stream. With such data available, it is possible to construct a decision system which performs the actual event detection. However, most of the existing algorithms do not provide an easy and computationally efficient method of extracting this information from the video camera images.

This paper makes two main contributions. First, it presents a novel algorithm for detection of stable regions, representing stationary objects such as luggage, in a stream of video camera images. The proposed approach is conceptually easy, it is able to perform online processing, it handles short-term occlusions and it is separated from the background subtraction procedure. Second, the paper shows that the proposed algorithm provides data that may be used for the task of unattended luggage detection in a modular, multi-stage video analysis system [7]. The main focus of the paper is to present the algorithm for detection of stationary objects, but in order to demonstrate that this algorithm provides data useful for efficient unattended luggage detection, a working system, in which the proposed algorithm is supplemented with the classification and decision modules (implemented in a simplified way for evaluation purposes) is presented and discussed. It should be noted that this paper does not attempt to solve the complex problem of decision making in the unattended luggage detector, which deserves a separate research and its complete solution is out of the scope of this paper. However, the proposed algorithm provides the input data for any, potentially more elaborated, decision module, and the evaluation system presented here is able to detect unattended luggage with satisfactory accuracy, using simplified decision criteria.

The remaining of the paper is organized as follows. Section 2 reviews the related approaches to unattended luggage detection and states their limitations. Section 3 presents details of the proposed algorithm, divided into functional parts. Results of experiments performed using three datasets, including recordings from a real airport monitoring systems, are presented in Section 4 and discussed in Section 5, and the paper ends with Conclusions.

2 Related works

This paper focuses on the algorithm which detects stationary objects in camera images and provides data needed for a single-camera unattended luggage detection. Initial works on the problem relied on separating foreground pixels from the static background, tracking movement of each object, detecting motionless trackers, verifying if these objects represent luggage and testing if the luggage is unattended. The background subtraction (BS) procedure is usually performed with the Gaussian Mixture Model (GMM) approach [32, 42]. Tracking of each object is achieved employing algorithms such as nearest neighbor tracking [1], Kalman filters [22, 24, 34, 35], tracking with Bayesian models [19], trained algorithms such as Markov chain models [30], and others. Detection of left luggage is achieved by track analysis: splitting tracks are detected and if one of the split objects remains in place (its track velocity is near zero), the object is declared as stationary [1, 18]. If a motionless luggage is detected, most of the described methods test for the distance between the left luggage and the tracked owner object and raise an alarm if the distance between objects exceeds the threshold for a defined period [11]. In a more recent publication, Elhamod and Levine [9] used blob tracking with spatial and color features matching, and enhanced the procedure with behavior modeling for detection of selected threats, including unattended luggage. Hettiarachchi et al. [13] used the results of blob tracking in a textual form for evaluation of blob movement using acyclic graphs. Tripathi et al. [39] detected stationary object by analyzing blob movement and classified them using an edge based object recognition algorithm. A review of other similar approaches may be found in [23, 38].

The main drawback of the tracker-based approaches to the unattended luggage detection is that their performance under the conditions of intensive object movement, which is the usual case at the airports, is severely limited. In the presence of a large number of occlusions and constantly merging and splitting objects, tracking errors result in losing the tracks of the luggage and its owner, making the event detection impossible. Additionally, luggage occluded by other objects during the moment of splitting cannot be detected. These shortcomings were reported e.g. by Porikli [28] and they were also observed in the earlier experiments performed by the author of this paper and his co-workers [34, 35]. In order to avoid such problems, a new trend in luggage detection removed the object tracking stage in favor of detection of stable foreground regions. Various algorithms proposed in the literature differ in the method of testing pixel stability and extracting stable image regions. Porikli used two separate background models: the short-term and the long-term one, updated with different rates. The long-term model allows for extraction of unattended luggage from the image. A similar algorithm by Sing et al. [29] added selective background updating. The problem with this approach is that the image data between the update moments is not utilized, hence short-term image changes (e.g. luggage occlusions) are not taken into account. Maintaining two separate background models at the same time is also problematic. A simpler approach was proposed by Tian et al. who reused the data contained in the GMM background model for this task [38]. This approach introduces another problem: stationary objects become a part of the background model and they are no longer detected as a foreground. In order to prevent this effect, special

background maintenance is needed, which in turn may result in damaging the background model in case of detection errors [10].

In a recent work by Maddalena and Petrosino [23], a more complex approach based on three pixel-level models (background, foreground and stopped) was introduced. A framework for moving pixels between these models and for detecting unattended luggage in the stopped model with a 3D self-organizing neural model was described. In order to handle occlusions, the stopped model has to include several layers. While the accuracy of this method is good, long computation time makes this approach unsuitable for online analysis. A review of other approaches to stable region detection, with indication of their shortcomings, was given by Pan et al. [26]. They proposed a method for selective background model updating based on pixel scores representing their stability. Another approach by Guler et al. [12] is based on the concept of peripheral tracking. However, these methods are also too complex to be used in an online analysis system. It should also be noted that many of the described algorithms detect stable regions but they do not relate them to actual moving objects, so some objects (e.g. partially occluded) are detected incorrectly. There are also alternative approaches that try to directly recognize luggage in images, e.g. Zheng et al. [41] use a histogram of oriented gradients (HoG) trained with examples of luggage, but such an approach is too computationally intensive to be implemented in an online system.

Approaches based on detection of stable regions do not perform object tracking, therefore another method of locating the luggage owner is needed. For example, an algorithm by Bhargava et al. [2] searches for the luggage owner using the normalized cross-correlation, Lu et al. [21] propose a human activity model based on Hidden Markov Model, Tian et al. [38] added a separate module for tracking unattended luggage candidates and their owners. Chang et al. [4] proposed an online boosting learning algorithm for tracking objects classified as humans.

3 Description of the algorithm

Unattended luggage detection may be presented as a multi-stage process (Fig. 1). The system receives a stream of video images from a single camera at its input, performs an online analysis of each video frame and outputs the detection results in a form of unattended luggage alerts with relevant data, such as luggage position in the image. The processing is divided into several stages; each of them utilizes the results produced by the previous one and limits the area of interest for the next stage. The main processing stages are described below.

1. *Background subtraction*: all image pixels are analyzed, foreground pixels belonging to moving objects are found.
2. *Detection of stable pixels*: foreground pixels are analyzed, stability of their values in time is evaluated, pixels with values within the defined range during the observation period are marked as the stable ones.
3. *Detection of stable regions*: connected components are formed from the stable pixels; if a component has a sufficient area, a stable image region is detected.
4. *Detection of stationary objects*: the detected stable regions are compared with the contours of moving objects extracted from the BS results; contours sufficiently covered by stable regions are denoted as stationary (previously moving) objects.
5. *Classification*: stationary objects are examined by a classifier in order to test whether they represent luggage.

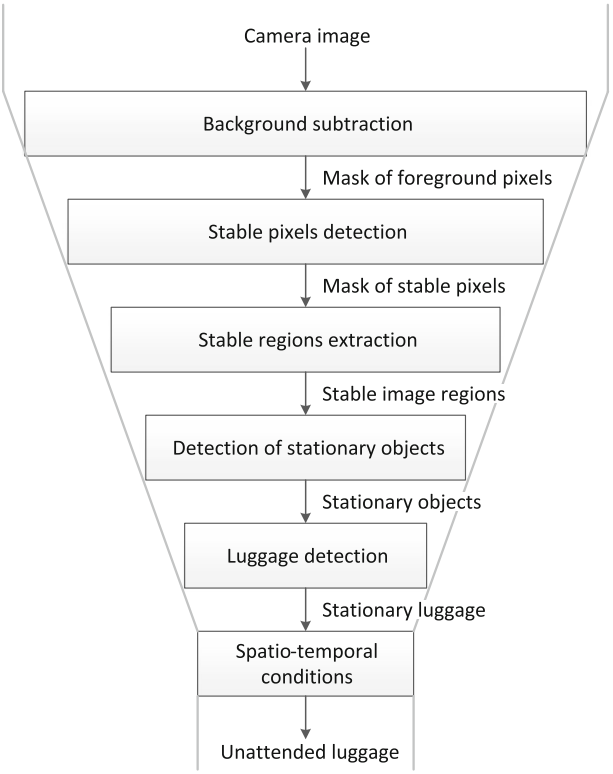

Fig. 1 Conceptual framework for a multi-stage video analysis for unattended luggage detection. Each processing stage limits the area of interest for the next stage

6. *Interpretation and decision*: spatio-temporal conditions are tested for the detected luggage objects in order to decide whether they are unattended.

With a well-known BS procedure, used in different analysis scenarios [20], contours of moving objects are separated from the background. The main problem is how to identify objects that remain stationary for a defined time (e.g. luggage). Such objects are initially detected as a foreground, but after remaining motionless for a prolonged time, they become a part of the background. Therefore, such objects have to be separated both from the static background and from moving objects (e.g. humans). As it was shown in the review of related publications, identification of such objects in images is a non-trivial problem which has not been solved in a satisfactory way. Therefore, a novel approach to this problem is proposed.

The aim of the algorithm is to provide an efficient implementation of the processing Steps 2 to 4, by performing analysis of images and BS results, and outputting contours of stationary objects extracted from the video stream. The BS operation (Step 1) is needed in order to provide the input data to this algorithm and it is realized using a standard GMM approach, therefore only a short description is presented in the paper. Although the proposed algorithm for stable regions detection is the main point of focus, it is also important to show that it provides information which may be used for successful detection of unattended luggage. In order to prove that, the proposed algorithm was supplemented with implementation of Steps 5 and 6, which will be described briefly. A design of a complete decision module for realization

of Step 6 which would cope with different conditions is a separate, complex problem, solution of which is out of scope of this paper. For the purpose of experiments and validation, a simplified decision module was developed, allowing for evaluation of the proposed algorithm and for unattended luggage detection with simplified conditions. The complete processing system is presented in Fig. 2, the blocks that are parts of the proposed algorithm for stable regions detection are enclosed in a frame. Details of each processing stage are presented in the following subsections.

3.1 Image preprocessing

The proposed algorithm for detection of stable image regions (starting from Step 2 in the processing structure from Fig. 1) receives a stream of video frames from the camera, together with foreground masks, i.e. binary images that denote foreground and background pixels with non-zero and zero values, respectively. The exact method of computing such a mask is not relevant here. In the framework used for implementation of the proposed algorithm, masks are obtained using the BS procedure, realized with the standard GMM approach [7, 32, 42] performed as follows. Each image pixel is represented by its own weighted sum of five Gaussians. Each Gaussian is described in terms of means and standard deviations of each color component of the pixel, and a weight. In the first stage of processing, a first Gaussian matching the current pixel value is found and its parameters are updated. A match is found if an absolute difference between the mean of the Gaussian and the pixel value is below a threshold (usually equal to 2.5 times the standard deviation of the Gaussian) for all color components. If no matching Gaussian was found, the one with the lowest weight is replaced by a new Gaussian initialized with the current pixel value. After that, weights of all Gaussians are updated and normalized so that they sum up to one, and the Gaussians are reordered by a decreasing ratio of weight to variance. Finally, if a matched Gaussian was found and its weight is sufficiently high, the pixel is assigned to the background, otherwise it represents a foreground (a moving object). The details of the GMM procedure may be found in the literature [7, 32]. The BS accuracy influences the outcomes of the following stages of processing, therefore all artifacts in the BS results should be removed. Pixel noise and small gaps are handled with morphological processing and parasite object shadows are removed using a dedicated procedure [7].

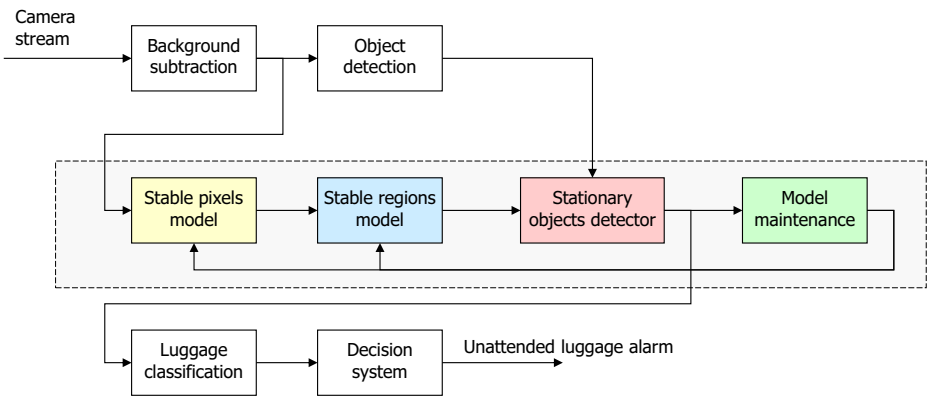

Fig. 2 Block diagram of the system for unattended luggage detection. Modules inside the gray box constitute the proposed algorithm for stationary objects detection while the remaining modules are added to form a complete detection system

3.2 Identification of stable foreground pixels

Contours of all objects moving in the current camera image may be extracted from the BS result, but for the purpose of stable regions detection it is necessary to find which of these objects remain stationary in the camera view for a defined time. In order to accomplish this, information on repeatability (stability) of pixel values within each contour is needed. The proposed algorithm utilizes a two-stage approach: the analysis is performed first on the pixel level and then on the region level. The task of the former stage is to find foreground pixels which, observed over a defined period, have values kept within small limits. Such pixels are declared as stable ones. In practical situations, values of pixels belonging to visible stationary objects may fluctuate due to light changes and camera noise, and this aspect has to be taken into account. Information on stability of foreground pixels cannot be obtained from the GMM model in which foreground pixels are represented with short-term components, without information about their stability, so using this model for foreground pixel stability analysis, as proposed by Tian [38], is problematic. Instead of implementing complex modifications in the BS model, which would result in increased computation time, a separate model is constructed for this task in the proposed algorithm. It may be considered as an additional layer put on top of the original background model. An approach based on representing pixel values using vectors (code words) was chosen for construction of such a model. A separate set of vectors is used to model each individual foreground pixel. A similar method was used e.g. for background modeling in the Codebook algorithm by Kim et al. [17], where the model is first trained using series of images, then the collected code words are used for background subtraction. Here, vectors are used only to store statistics on the reoccurring foreground pixel values and the training phase is not needed.

Each image pixel is represented in the model of stable values with a set of vectors:

$$\mathbf{v}_i = \lfloor r_i, g_i, b_i, l_{\min,i}, l_{\max,i}, t_{c,i}, t_{m,i} \rfloor \tag{1}$$

where i is the current video frame index, (r, g, b) are the averaged color values, ($l_{\min}$, $l_{\max}$) define the range of brightness changes covered by the vector, t_c stores the time of vector creation and t_m is the last time the vector matched the pixel value. For each foreground image pixel with values (r_p, g_p, b_p), vectors modeling this pixel are tested in order to find if the difference between the pixel and vector values is within the limits. First, a brightness difference is tested using the formula:

$$\alpha l_{\max,i} \le \sqrt{r_p^2 + g_p^2 + b_p^2} \le \frac{1}{\alpha} l_{\min,i} \tag{2}$$

where α <1 is a factor limiting the allowed brightness variations (e.g. due to changes in scene lighting and camera exposure). If the brightness condition is fulfilled, a color difference is calculated as a distance in the 3D color space [17] and if the difference is below a threshold $\delta_{\max}$, the pixel matches the color of the vector:

$$\sqrt{\left(r_p^2 + g_p^2 + b_p^2\right) - \frac{\left(r_p r_i + g_p g_i + b_p b_i\right)^2}{r_i^2 + g_i^2 + b_i^2}} \le \delta_{\max} \tag{3}$$

The first vector that fulfills both conditions given by Eqs. 2 and 3 is updated as follows:

$$\begin{aligned} r_i &= \rho r_{i-1} + (1-\rho) r_p \\ g_i &= \rho g_{i-1} + (1-\rho) g_p \\ b_i &= \rho b_{i-1} + (1-\rho) b_p \\ l_{\min,i} &= \min\left(l_{\min,i-1}, \sqrt{r_p^2 + g_p^2 + b_p^2}\right) \\ l_{\max,i} &= \max\left(l_{\max,i-1}, \sqrt{r_p^2 + g_p^2 + b_p^2}\right) \\ t_m &= t_i \end{aligned} \tag{4}$$

where ρ is the update factor, t_i is the timestamp of the current frame. If none of the vectors matched the current pixel value, the model is extended by adding a new vector, initialized as follows:

$$\mathbf{v}_i = \left\lfloor r_p, g_p, b_p, \sqrt{r_p^2 + g_p^2 + b_p^2}, \sqrt{r_p^2 + g_p^2 + b_p^2}, t_i, t_i \right\rfloor \tag{5}$$

Using the constructed model, stable pixels may be found by testing whether any vector in the model that was matched in the current frame has a sufficiently long 'time of life'. Therefore, the stability condition is:

$$\exists v_{i,k} : (t_m = t) \wedge (t - t_c > T_s) \tag{6}$$

where k is the index of the vector representing the pixel, T_s is the stability threshold (minimum period needed to consider the pixel stable).

Generally, only foreground pixels are tested by this algorithm and if a pixel transitions from the foreground to the background (e.g. a person passes by and uncovers the background), its vectors are removed from the model. However, such an approach would result in losing objects that remain motionless for a prolonged time. Due to learning capabilities of the GMM algorithm, stationary objects blend into the background after some time and their pixels are no longer marked as the foreground. In order to allow detection of such objects in the proposed algorithm, if a pixel is assigned to background, its vectors indicating stable value, i.e., these fulfilling condition (6), are retained in the model. Therefore, background pixels that were earlier classified as a stable foreground are also included in the analysis. One important difference in this case is that if no matching vector is found, no vector is added to the model.

The complete algorithm of the pixel-level stability analysis is presented in Algorithm 1 using a pseudocode. The final result of this step is identification of stable pixels that potentially belong to stationary objects.

3.3 Extraction of stable image regions

The next stage of analysis finds image regions composed of stable pixels and establishes their relationship with the detected moving objects. Let **I** denote the currently analyzed image frame, **M** – the BS mask (obtained by processing **I**) and **R** – the result of pixel-level stability analysis, in a form of a binary image in which non-zero values mark stable pixels (fulfilling Eq. 6). Stable image regions are extracted by performing a connected component analysis in **R**. First, eight-connected clusters of stable pixels are found using the border following algorithm [33]. At the same time, connected components are also extracted from the mask **M** in order to find the contours of moving objects. In the next step, the algorithm establishes correspondence

between the contours of moving objects and the detected stable regions, based on their position and size. Let S_i denote the i-th stable region as a set of pixel coordinates:

Algorithm 1. Pixel evaluation in the stable pixels model

```
INPUT: Pixel value p
INPUT: Background subtraction result m (1 or 0)
INPUT: Model V composed of N vectors (Eq. 1)
INPUT: Current image timestamp t
OUTPUT: Pixel state: stable=true or false
BEGIN:
      stable=false
      if m=0 then
            // for background pixels, remove unstable vectors (Eq. 6)
            RemoveVectors(t - t_c<T_s)
      end if
      // look for a vector matching the pixel
      matched=0
      for n=1 to N do
            // test for brightness match (Eq. 2), then color match (Eq. 3)
            if BrightnessMatch(p, V_n) then
                  if (ColorMatch(p, V_n) then
                        matched=n
                        break for
                  end if
            end if
      end for
      if matched>0 then
            // update the matched vector (Eq. 4)
            UpdateVector(V_matched)
            // test for stability (Eq. 6)
            if (t - V_matched.t_c>T_s) then
                  stable=true
            end if
      else if m>0 then
            // for foreground pixels, create a new vector (Eq. 5)
            AddVector(p)
      end if
      return stable
END
```

$$S_i = \{s_{i,k}\},\ k = 1, \ldots, n(S_i) \tag{7}$$

Similarly, O_j denotes the j-th closed contour detected from the mask $\mathbf{M}$. For each S_i, a search for related object O_j is performed by testing the condition:

$$S_i \sim O_j \text{ if} \exists k : s_{i,k} \in O_j \tag{8}$$

From the nature of the proposed algorithm, each stable region may correspond to one or more (in case of fragmentation) contours of detected objects, or it may have

no related object contours if the object has already been incorporated into the background model. A section of the related object contour covered by the stable region is:

$$C_{i,j} = S_i \cap O_j \tag{9}$$

Now, two situations are possible. If a related object is found ($C_{i,j}$ is not empty), the condition to declare this object stationary is that a sufficient fraction β of the object contour area is covered by the stable region which is larger than the minimum area $a_{\min}$:

$$\frac{n(C_{i,j})}{n(O_j)} \geq \beta \wedge n(S_i) \geq a_{\min},\ \beta \in (0,\ 1) \tag{10}$$

Adjusting the β value allows for proper detection in case of distortions such as background subtraction errors, noise, etc. If the object has already become a part of the background (because of remaining motionless for a prolonged time), the related object contour cannot be found. In this case, only the stable region size is tested.

In order to ensure efficient detection of stationary objects, a model of stable regions is constructed. Each new stable region is added to the model. If a stable region detected in the current frame matches the position in image of any region stored in the model, the matched region is updated. The procedure of finding stationary objects is presented in Algorithm 2. Fig. 3 shows an example of detection of a region representing stationary luggage. The 'age' of matched vectors is visualized with a grayscale map, brighter shades indicate longer time of stability, black pixels are the background. Color codes of pixels belonging to the bag which was left on the floor progress towards brighter values (Fig. 3b-e), while the pixels belonging to persons passing by are short-term (darker values). White color marks stable pixels fulfilling the condition (Eq. 6) and if the number of pixels in the stable region fulfills Eq. 10, the object is marked as a stationary one (Fig. 3e). The luggage contour is detected even in case of short-term occlusions by passing persons because the relevant vectors may still be found in the model (Fig. 3f). It should also be noted that a person remaining motionless for a prolonged time may also be detected as a stationary object (Fig. 3g). Identification of stable regions representing actual luggage is the task of the classification module.

Algorithm 2. Finding valid stable regions in the image

```
INPUT: N regions S of stable pixels detected in the current image
INPUT: M moving object contours O detected in the current image
OUTPUT: A set of valid stable image regions (stable)
BEGIN:
    stable={}
    for n=1 to N do
        // find matching object contour C (Eq. 9)
        matched=0
        C=∅
        for m=1 to M do
```

```
                if Coverage(O_m, S_n) >0 then
                        C=O_m, ∩ S_n
                        matched=m
                        break for
                end if
        end for
        // test region validity
        if C !=∅ then
                // object contour matches the stable region
                // test size and coverage (Eq. 10)
                if Size(S_n)>a_min and Size(C) / Size(O_matched)>  then
                        // test the false object condition (Eq. 14)
                        if EdgeMeasure(O_matched)>  _min then
                                stable=stable+O_matched
                        end if
                end if
        else if Size(S_n)>a_min and EdgeMeasure(S_n)>  _min then
                // object not matched to any contour
                stable=stable+S_n
        end if
    end for
    return stable
END
```

3.4 False objects removal

Stationary objects detected in the previous step may belong to one of three groups: (a) luggage, (b) other objects, e.g. motionless persons, and (c) false objects. Moving objects that remain stationary for a prolonged time are incorporated into the learning background model. If such an object is later removed, its contour (a 'hole' in the background) is detected as a foreground region because the actual, uncovered background does not match the model content. Since the contour has the same shape as the removed object, it may be incorrectly classified by the algorithm as luggage, providing incorrect data to the decision module. The proposed procedure for removal of false objects is based on the observation that in case of such objects, pixel brightness differences alongside the contour border are usually small, while in case of an actual object, much larger differences are observed. A similar 'edge-energy' method was proposed e.g. by Connel [5].

The algorithm for detection of false contours works as follows. Pixels belonging to the border of a stable object contour O_j are taken into account. Each border point corresponds to pixels $\mathbf{I}_{(x,y)}$ in the input image and $\mathbf{M}_{(x,y)}$ in the mask $\mathbf{M}$, where (x,y) denote the pixel coordinates. An eight-connected neighborhood of this point is divided into sets of inside (P_{in}) and outside (P_{out}) pixels and their brightness values are stored in respective sets:

$$\begin{aligned} P_{in(x,y)} &= \left\{ l_{(m,n)} : \quad \mathbf{M}_{(m,n)} > 0 \wedge m \in \langle x-1, x+1 \rangle \wedge n \in \langle y-1, y+1 \rangle \right\} \\ P_{out(x,y)} &= \left\{ l_{(m,n)} : \quad \mathbf{M}_{(m,n)} = 0 \wedge m \in \langle x-1, x+1 \rangle \wedge n \in \langle y-1, y+1 \rangle \right\} \end{aligned} \tag{11}$$

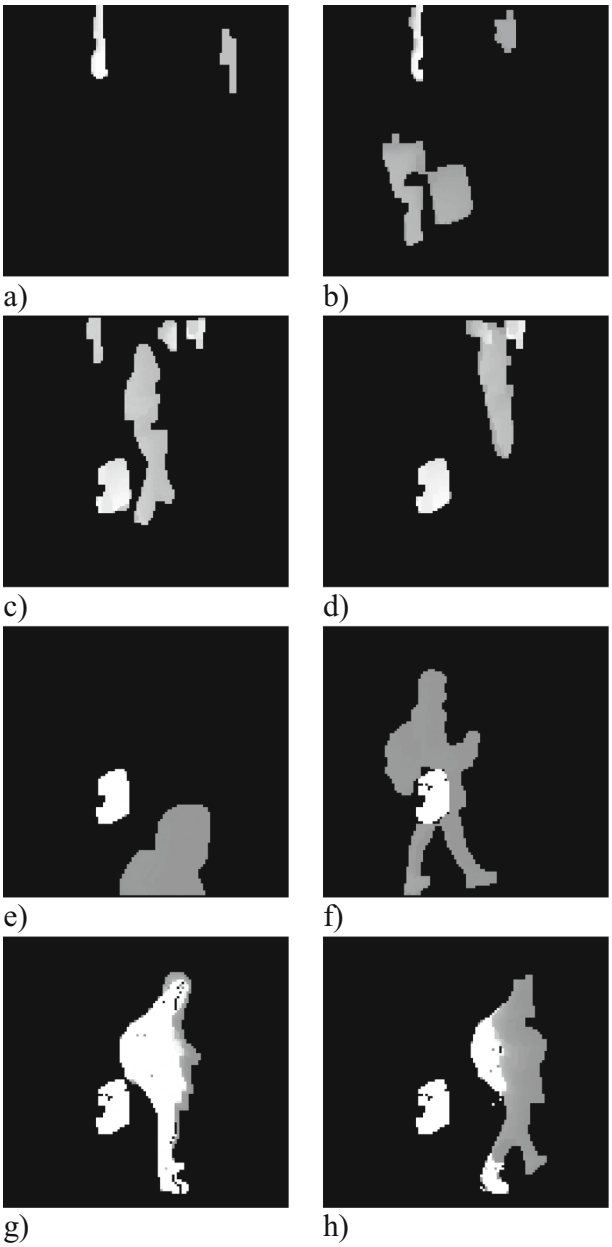

Fig. 3 Illustration of the procedure for detection of stable image regions. The time during which a pixel remains stable is visualized with shades of gray, brighter values mark pixels stable for a longer time. Clusters of pixels marked with white color may be detected as stationary objects

where l is the brightness of the pixel calculated from its RGB values:

$$l_{(x,y)} = \sqrt{r^2_{(x,y)} + g^2_{(x,y)} + b^2_{(x,y)}}, \quad \mathbf{I}_{(x,y)} = \left[r_{(x,y)}, g_{(x,y)}, b_{(x,y)}\right] \tag{12}$$

Next, pairs of adjacent inside-outside points are formed as shown in Fig. 4. The maximum absolute brightness difference between all such pairs within the neighborhood of the border point is found:

$$d_{(x,y)} = \max \left| p_{in(m1,n1)} - p_{out(m2,n2)} \right|, \quad p_{in(m1,n1)} \in P_{in(x,y)},\ p_{out(m2,n2)} \in P_{out(x,y)}, \quad |m1-m2| \le 1 \ \wedge \ |n1-n2| \le 1 \tag{13}$$

The final measure used for false objects removal is calculated as a percentage of contour border points for which the maximum brightness difference d is above a threshold $d_{\min}$:

$$\zeta_j = \frac{1}{N_j} \sum_{k=1}^{N_j} \begin{cases} 1, & d_{j,k} > d_{\min} \\ 0, & d_{j,k} \le d_{\min} \end{cases} \tag{14}$$

where N_j is the number of border pixels for which at least one pair of adjacent points was analyzed (border pixels situated on the image edge have to be excluded from this computation). Objects for which $\zeta_j \le \zeta_{\min}$, $\zeta_{\min} \in (0, 1)$, are detected as false ones and they may be removed from further analysis, as shown in Algorithm 2.

A practical illustration of the proposed method is presented in Fig. 5. It can be observed that both the left bag (Fig. 5a) and the same bag removed from the scene after it was incorporated into the background model (Fig. 5b) result in almost identical detected contours. Without the proposed procedure, the second case would result in a false positive decision of the decision algorithm. It can be seen from the analysis of brightness differences alongside both contours that edges of the actual object exhibit much higher brightness difference at the edges of the bag than in the second case, where the brightness transition through the contour borders is smooth. This is reflected by a significantly higher value of the measure ζ computed with the procedure described above. As a result, the false contour may be detected and removed from further analysis.

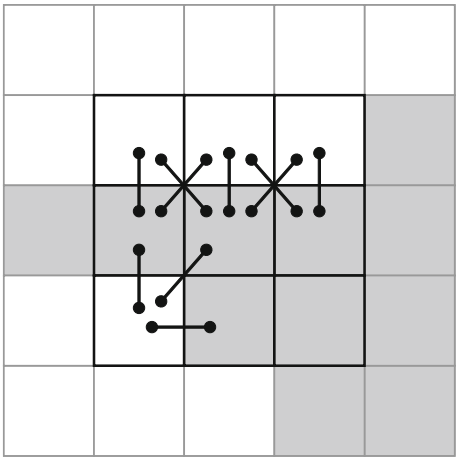

Fig. 4 Illustration of calculating the measure for false objects removal. Foreground and background pixels are denoted with gray and white colors, respectively. The black border indicates a neighborhood of the currently analyzed border pixel. Black lines with dots show pairs of pixels that are analyzed for brightness difference

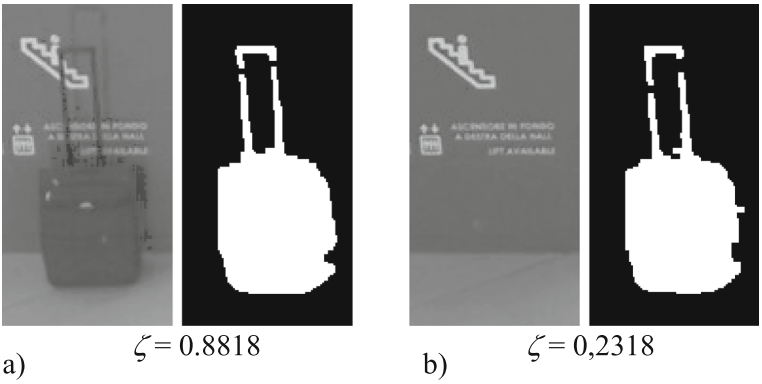

Fig. 5 An example of the problem of left and taken objects detection: a) luggage left in the scene, b) luggage taken after it was present in the scene sufficiently long to be incorporated into the background model. Both cases produce almost identical object contours in the background subtraction stage. Brightness difference alongside the contour is significantly larger in case a, therefore the indicated edge measure ζ is higher than for the false object

3.5 Maintenance of the models of stable regions and pixels

Over the time of the video stream processing, both the stable pixels model and the stable regions model accumulate all the detected units, which results in constant increase of their size and the analysis time. In order to prevent this undesired effect and to avoid detecting objects that are no longer present in the scene, a maintenance procedure was developed for removal of obsolete items from both models. First, the model of the stable regions will be considered. For the maintenance purpose, each detected stable region has to be described with counters that, together with the contour, form a vector representation of the region:

$$\mathbf{r}_i = [S_i, q_i, u_i, z_i] \tag{15}$$

where S_i is the contour of the region, q_i is the counter of frames during which the region was marked as invalid (a candidate for deletion), u_i is the index of the time the region was last observed (created or updated) and z_i is the counter of time during which the region represented an unattended luggage.

The maintenance procedure is shown in Algorithm 3. If a stable region S_j detected in the current frame overlaps S_i stored in any $\mathbf{r}_i$ in the model, this region is updated, otherwise a new region is added to the model (q_i and z_i set to 0 and u_i set to the frame index). If a region is invalid, i.e. it does not correspond to any moving object and is too small, or it was detected as a false object, its q_i counter is increased. Additionally, the results of analysis within the classification and decision modules may be used for the model maintenance. For example, if a stationary object was not classified as luggage, the q counter may be increased. Similarly, if the decision module finds that the object is currently unattended, the q counter may be reset and the z value increased (the z counter may be used for testing temporal conditions during the decision making). Therefore, although the classification and decision

modules do not belong to the main algorithm presented in this paper, they may provide important information for maintenance of the model, allowing for discarding incorrect stable regions that may potentially cause false alarms.

Algorithm 3. Maintenance of the stable regions model

```
INPUT: N stable regions S detected in the current image
INPUT: M previously detected stable regions r
INPUT: Current image timestamp t
OUTPUT: Updated model of stable regions
BEGIN:
    for n =1 to N do
        // find matching region in stable regions model
        matched =0
        for m =1 to M do
            if MatchRegion(r_m, S_n) then
                matched=m
                r_m.s=S_n
                r_m.u=t
                break for
            end if
        end for
        if matched =0 then
            AddRegion(S_n)
        end if
        // coverage C of an object by the region is calculated by Alg. 2
        if C=∅ and EdgeMeasure(S_n)< _min and Size(S_n)<a_min
        then
            S_n.q=S_n.q +1
        end if
        // classification module
        if not IsLuggage(S_n) then
            S_n.q=S_n.q +1
        end if
        // decision module
        if IsUnattended(S_n) then
            S_n.q =0
            S_n.z=S_n.z +1
        end if
    end for
    // remove inactive regions (Eq. 14)
    for m =1 to M do
        if r_m.q> T_max q or t - r_m.u> T_max u then
            RemoveRegion(r_m)
        end if
    end for
END
```

After the region analysis is completed for the current video frame, some of the regions stored in the model may not have been updated for some time (they are no longer present in the scene) and some other may be continuously flagged as incorrect ones. Therefore, the model is cleaned by removing vectors fulfilling the condition:

$$\mathbf{r}_i \,|q_i > T_{\max q} \vee (t - u_i) > T_{\max u} \tag{16}$$

where t is the current frame index, $T_{\max q}$ is the maximum time during which the region is marked as invalid and $T_{\max u}$ is the maximum period in which the region is not updated. In the experiments, threshold values of $T_{\max q} = 5T_s$ and $T_{\max u} = 3T_s$, were selected empirically (T_s is the threshold used for the pixel stability detection).

The second part of the maintenance procedure clears obsolete vectors from the stable pixels model. Two operations are performed. First, vectors that were not matched within a defined period $T_{\max\ v}$ are removed:

$$\mathbf{v}_i \,|t - t_m > T_{\max v} \tag{17}$$

This threshold value should be sufficiently large in order to cope with short-term object occlusions. In the experiments, the following values were used: $T_{\max\ v} = 5T_s$ if the pixel currently belongs to the foreground and $T_{\max\ v} = 2T_s$ for the background pixels that have stable vectors. Additionally, if the pixel belonged to a region which was removed as incorrect (according to Eq. 16) during analysis of the previous video frame, vectors that were matched in the previous step ($t - 1$) are removed before the pixel is tested for stability. In order to achieve this, the region-based analysis algorithm sends back a binary mask of incorrect regions to the pixel-level analysis algorithm.

3.6 Classification and decision stages

The algorithm for detection of stationary objects was presented in the previous subsections. In order to show that this algorithm provides data that may be used for successful detection of unattended luggage in practical situations, the algorithm was supplemented with modules for object classification and decision making. The classification module is necessary in order to separate objects representing luggage from other stationary objects. Various approaches may be utilized for this task, ranging from simple classifiers based only on object size and proportions [18], to complex models trained with image features [2]. For the purpose of the experiments described in this paper, a classifier based on shape descriptors of the object contour was implemented. For each stationary object O_j, a vector of seven Hu moments, invariant to scale, translation and rotation (as far as a projection of the object contour to the image plane is concerned) are calculated [14], forming the final object shape descriptor. The actual classifier may be realized using various machine learning algorithms. During the experiments, two classifiers were tested: the first one used the Support Vector Machine method [6], the second one utilized the Random Forests approach [3]. The classifier has to be pre-trained using a set of shape descriptor vectors calculated from the collected examples belonging to two classes: luggage and other objects. During the detection, stationary objects O_j that are assigned to the non-luggage class are excluded from further consideration and, as it was mentioned previously, information about these regions may be used for removing related stationary objects from the stable regions model.

The decision module for checking whether the luggage is unattended was simplified for the purpose of the algorithm validation. The decision rule follows the event definition mentioned in the Introduction [31]: a stationary luggage is declared to be unattended if there are no

moving or stationary non-luggage objects at a distance larger than the threshold, within the period T_A. In practical applications, a much more complex decision system, which also tracks the real luggage owner, would have to be implemented. However, solution of this problem deserves a separate research, therefore a purposely simplified decision system will be used for evaluation of the algorithm described in the paper. However, with the approach presented here, more complex decision modules may replace the test module in the analysis framework, operating on the same input data provided by the proposed algorithm. The simplified decision process is realized as follows. Once a stationary luggage is detected, its distance from all other objects is calculated between the middle points of the bottom bounding box border of the contours. These distances should ideally be expressed in physical dimensions, using the spatial camera calibration procedure [40]. Only real moving objects (no other stationary luggage or false objects) are taken into account. A time counter z stored in the stable regions model (Eq. 15) is started for each newly detected stationary luggage and it is increased for each video frame in which there are no moving objects within the defined radius. If z exceeds the alarm threshold T_A, the algorithm declares the unattended luggage event.

In a real system implementation, tracking the luggage owner is necessary. Although the proposed algorithm does not perform such analysis, it is able to provide the necessary data. An approach similar to the one proposed by Bhargava et al. [2] may be used by going back in time to the moment the luggage was introduced into the scene, finding the object representing the luggage owner and tracking its movement. Since the creation time is stored in the vectors inside the stable pixels model, it is possible to determine the time the left luggage was first observed and to track the owner using a separate module, or alternatively, test if the object near the luggage matches the owner template.

4 Experimental results

The purpose of the experiments was to test performance of the proposed algorithm in detection of stable image regions and also to verify that the results obtained with this algorithm are useful for unattended luggage detection. The algorithm was implemented in C++, using the OpenCV library only for low-level input and output operations. The experiments were run on a PC equipped with a quad-core CPU at 2.80 GHz and 6 GB RAM. The BS operation was performed using the GMM algorithm [32], tuned for indoor conditions (model update rate =0.0003, background detection threshold =0.7, Gaussians initialized with weight 0.05 and variance 320). Shadows were removed with an additional procedure [8]. Additionally, Gaussian blur with the radius of 5 was applied to the input images in order to reduce the influence of camera noise and compression artifacts on the detection accuracy. The optimal parameter values for detection of stable pixels were found experimentally: α =0.5 (Eq. 2) and $\delta_{\max}$ =40 (Eq. 3), there was no need to adjust them to different conditions. Minimum object coverage for stable region detection (Eq. 10) was β =0.9 and the minimum luggage size was adjusted according to the video resolution. The time thresholds T_s (for pixel stability, Eq. 6), and T_A for the stationary luggage detection (the alarm time) were both set to 10 s (so that they are irrespective of the camera frame rate). Additionally, in order to limit the number of false positive decisions, a maximum area of the analyzed object contour (20 000 pixels) was imposed in order to avoid false detections resulting from large, temporary false objects occurring in case of severe lighting changes during the adaptation of the BS model. Additionally, some areas in which it was not possible to place luggage (ceilings, high parts of walls,

railway tracks, etc.) were excluded from the analysis in order to avoid false detections in these areas (mainly due to object reflections). False objects were removed using the proposed procedure, with $d_{\min}$ =16 and $\zeta_{\min}$ =0.3 (Eq. 14).

Next, a classifier was constructed for selection of stationary objects representing luggage. The training set was created by reviewing a total of c.a. 8 h long recordings from seven airport cameras and manually selecting contours (detected with BS) of representative examples of both stationary luggage and other stationary objects. A set of 200 positive and 200 negative examples was created and shape descriptor vectors (Hu moments [14]) were calculated for each example. It was confirmed with the statistical Mann–Whitney test that the values of all individual Hu moments are significantly different between the two groups (p <0.05). Two classifiers were tested: the first one used SVM [6] and the second one utilized the OpenCV library implementation of the original Random Forests algorithm [3] (named RTree) in a form of an ensemble of 100 binary trees with depth level 10. Both classifiers yielded identical results in the validation stage (94.75 % accuracy). The RTree classifier was chosen because of its useful feature: for each example, a percentage of trees voting for the positive class may be returned. It was utilized in the classifier by setting a minimum value of this measure to 0.6, thus rejecting less certain decisions and potentially decreasing the number of false positives.

In the first experiment, the proposed algorithm for detection of stationary objects was tested, without any decision module for unattended luggage detection. The datasets from PETS 2006 [27] and AVSS 2007 (part of the iLids dataset) [15] conferences have become a de facto standard benchmark used in practically all related publications. The results of experiments are presented in Table 1 and compared with several recent published works on this topic. It can be observed that the proposed algorithm matches the accuracy of other approaches. It detected correctly all stationary luggage objects in the i-Lids dataset and it missed one object in the PETS 2006 dataset. Fig. 6 presents some examples of successful detections in i-Lids and PETS 2006 videos. In these examples, object contours obtained from the BS module were sufficiently accurate and it was possible to verify that each stage of the proposed algorithm worked as expected, leading to successful detections. A single false negative result was caused by problems with the BS procedure, because a dark object was placed on a similarly dark background so it was partially camouflaged (Bhargava [2] had the same problem). However, other cases were detected correctly and there were no false positive results.

The two standard datasets do not allow for a complete evaluation of the proposed algorithm. The number of events is small, the video resolution and quality do not match the current state

Table 1 Performance measures (true positives, false positives, precision and recall) of the proposed algorithm for detection of stationary objects, compared with other approaches using i-Lids and PETS 2006 datasets

Algorithm	i-Lids (AVSS 2007)				PETS 2006			
	TP	FP	Prec.	Rec.	TP	FP	Prec.	Rec.
Proposed	3/3	0	100 %	100 %	6/7	0	100 %	85.7 %
Porikli et al. 2008 [28]	3/3	2	60 %	100 %	1/1	0	100 %	100 %
Bhargava et al. 2009 [2]	3/3	0	100 %	100 %	4/5	0	100 %	80 %
Tian et al. 2011 [38]	3/3	1	75 %	100 %	7/7	0	100 %	100 %
Pan et al. 2011 [26]	3/3	0	100 %	100 %	–	–	–	–
Maddalena et al. 2013 [23]	3/3	0	100 %	100 %	–	–	–	–
Chang et al. 2013 [4]	3/3	0	100 %	100 %	6/6	0	100 %	100 %

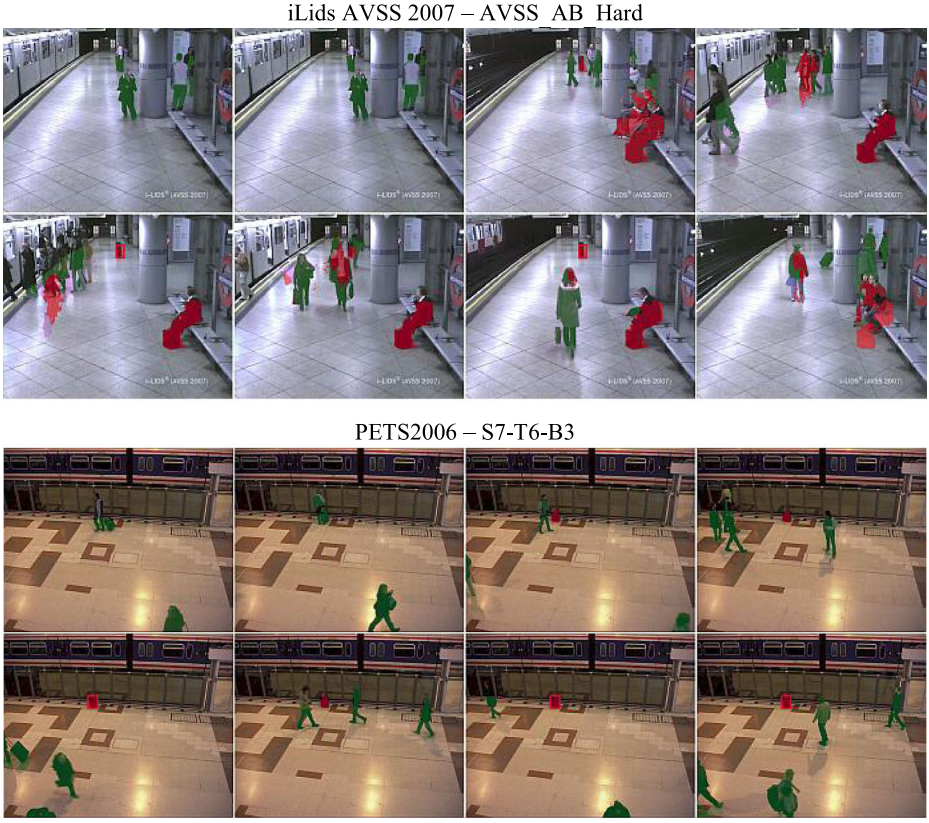

Fig. 6 Examples of successful detection of stationary luggage using the proposed algorithm. Green color marks unstable moving objects, red color – detected stable regions. Red rectangles mark the position of the detected stable regions

of technology and the intensity of movement is much lower than in real airport scenario (especially in the PETS dataset). Therefore, in order to test the algorithm in a real video surveillance system using high resolution cameras, mounted in the operating airport building, an additional dataset was created, containing video recordings from such a system. Video streams from 5 cameras were recorded during four sessions, two of them were analog cameras with 704×576 pixels resolution (the streams were digitized) and the other three were high-resolution (1,280×960 px) digital cameras. Typical abandoned luggage scenarios were played by the actors during the recordings. This additional dataset comprised of a total of over 15 h of recordings. Video recordings were downscaled by a factor of 2 for the low resolution videos and 4 for the high resolution ones in order to reduce the computational load and to smooth small variations of pixel values.

In the subsequent experiment, the dataset described above and the two standard sets were used in the experiments in order to evaluate the proposed algorithm regarding its usefulness in the unattended luggage detection. Therefore, a simple decision module, described earlier in the paper, was added to the framework. The final decision of the system was detection of stationary luggage which was unattended according to the criteria given in the algorithm description. Table 2 presents the results obtained for each of the tested datasets and the overall scores. The proposed algorithm was able to detect the abandoned luggage event in all scenes

Table 2 Results of unattended luggage detection using data obtained with the proposed algorithm

Dataset	AVSS07	PETS06	Airport	Overall
Total time (h:m:s)	0:09:42	0:14:04	15:48:32	16:12:18
Number of events (N)	3	5	23	32
Detected events (TP)	3	3	22	29
Undetected events (FN)	0	2	1	3
False alarms (FP)	0	0	4	4
Precision: TP / (TP+FP)	100 %	100 %	84.6 %	87.5 %
Recall: TP / (TP+FN)	100 %	60 %	95.7 %	90.3 %
Accuracy: TP / (N+FP)	100 %	60 %	81.5 %	80.0 %

from the iLids dataset. In case of the PETS dataset, two events were undetected. In Scene 5, the ski bag was rejected by the classifier as a non-luggage object and in Scene 6, a camouflage effect (the same as in the first experiment) prevented the detection of a stationary object, resulting in a missed event.

Implementations of algorithms presented in Table 1 are not publicly available, so it was not possible to test their performance with the airport dataset. Therefore, in order to compare the performance of the proposed algorithm to the state-of-the-art, the experiments were repeated using a tracker-based algorithm, which may be treated as a reference approach based on object tracking [18, 24]. In this algorithm, moving objects are tracked with Kalman filters [7, 35]. The predicted states of Kalman trackers are used for matching the objects detected with BS to the tracked objects and to solve conflict situations (occlusions, splitting, etc.). If an object splits, one object (person) moves away at a defined distance and the other object (luggage) remains motionless then, after a defined time, an abandoned luggage event is declared. The results for both the proposed and the reference algorithm are presented in Table 3. The tracker-based algorithm was able to detect all events in the PETS dataset, with four false positive alarms. In case of the iLids dataset where the number of object occlusions was much higher, the performance suffered: the event in the Hard scenario was not detected and a significant number of false alarms was raised. The observation that this algorithm does not cope well with frequent object merging and splitting was confirmed in tests with the reduced airport dataset in which the reference algorithm performed poorly. Three of the five events were not detected because of tracking errors while the person was leaving the luggage (the algorithm was not able to assign objects to correct

Table 3 Comparison of the proposed algorithm (Prop.) with the reference one (Ref.) based on object tracking with Kalman filters

Dataset	AVSS07		PETS06		Airport (reduced)		Overall	
Total time	0:09:42		0:14:04		0:59:11		1:22:57	
Events	3		5		5		13	
Algorithm	Ref.	Prop.	Ref.	Prop.	Ref.	Prop.	Ref.	Prop.
TP	2	3	5	3	2	5	9	11
FN	1	0	0	2	3	0	4	2
FP	20	0	4	0	56	1	80	1
Precision	9.1 %	100 %	55.6 %	100 %	3.4 %	83.3 %	10.1 %	91.7 %
Recall	66.7 %	100 %	100 %	60 %	40.0 %	100 %	69.2 %	84.6 %
Accuracy	8.7 %	100 %	55.6 %	60 %	3.3 %	83.3 %	9.7 %	78.6 %

trackers). At the same time, the number of false alarms was extremely high, causing very low precision and accuracy scores and making this algorithm unsuitable for a practical applications. The proposed algorithm significantly outperformed the reference one in the airport recordings, with a perfect recall in the limited dataset and only one false positive.

For an additional assessment, the proposed system was tested by running it on four live video streams from the airport security cameras, continuously for 12 days, and recording the numbers of true and false alarms. During the working period, 69 alarms were raised by the system, 8 of which were actual abandoned luggage events and 61 were false alarms. Ten of these false alarms were abandoned objects such as empty luggage carts so they might not be regarded as detection errors. No ground truth was available for this test because of the airport security policy, therefore the number of missed alarms is not known. However, the rate of false positive alarms may be assessed. Although as much as 88 % of the alarms were false, the average rate of such alarms was 1.28 alarm per one camera per 24 h, which may be considered acceptable in practical systems. The reduced number of false positive decisions compared to the earlier experiments may be partially contributed to an improved quality of video streams, mainly to an increased bit rate which reduced the compression noise and, as a consequence, decreased the number of background subtraction errors resulting in fragmented objects.

5 Discussion

Performance of the proposed algorithm was compared with other approaches using two methods. First, the main algorithm for detection of stationary objects was compared with several recent works, using two standard datasets. As shown in Table 1, almost all published algorithms achieve a perfect 100 % recall and some researchers report one or two false positives. The proposed algorithm matches the state-of-the-art methods in terms of accuracy. A single object was not detected due to BS problems which resulted in its lower recall value. In order to prove that the results obtained with the proposed algorithm may be used for unattended luggage detection, a decision module was implemented for testing whether the luggage is unattended using simplified criteria, without owner tracking. In the PETS2006 datasets, two errors were observed. The problem with incorrect object classification may be avoided by improving the classification module (e.g. extending the training set with examples of imperfectly detected objects). The camouflaged object problem is more difficult to solve, as it is related to BS errors in difficult conditions (similar color and brightness of luggage and the background). Although both types of errors limit the accuracy of the complete event detection system, they are not related to the main algorithm for stable regions detection which proved to work correctly in these conditions, provided that it received sufficiently accurate data at its input. The algorithm is robust against moderate imperfections in the shape of detected moving objects, but in the case of a false negative result, the contour was not detected at all so the algorithm had no chance to detect the event. Because the experiments performed on the two standard benchmarks do not provide enough information on the algorithm performance in a real world scenario, e.g. in the airport building where the conditions (the number of moving objects, image resolution, video quality) is much different from these two datasets, an additional evaluation of the proposed algorithm in such conditions was performed using the video material recorded from a real airport surveillance system. The obtained results indicate that performance of the proposed algorithm under the described conditions is satisfactory, despite the simplified decision criteria. Comparison with a reference algorithm based on Kalman trackers proved that the proposed method provides significantly higher accuracy. A single event was undetected because of BS errors (camouflaging). Four false positives were observed, most of them were resulting from errors in the object

detection stage, mainly because of object fragmentation (e.g. legs of the standing person separated from the rest of the silhouette were detected and classified as luggage), and one case was a classification error. The obtained scores may not seem very high, but it should be stressed that a relatively low number of the actual events took place during a long observation time (on average, one event approximately every 40 min). Implementing the proposed method of false contours removal resulted in a substantially reduced number of false positives compared to the preliminary tests. However, a certain amount of false positive results in such a detection system is inevitable. In the experiments, an average number of these errors was one false positive every c.a. four hours, which may be considered acceptable. Taking all of this into account, the obtained 95 % recall and 80 % accuracy are satisfactory and these values may be improved by extending the classifier with a larger set of examples and optimizing the object detection procedure by supplementing BS with a post-processing algorithm in order to reduce object fragmentation.

The combined results of all experiments prove that the proposed algorithm provides data on stationary objects in the video stream which, after using the classification and decision module, leads to successful detection of unattended luggage. It should also be noted that the results obtained from the classification and decision modules were used for maintenance of the stable regions model (i.e. removal of regions not representing luggage from the model), probably reducing the number of false positive decisions, therefore it was justified to implement these modules in the test system.

Compared with other recent approaches to stable regions detection, e.g. [4, 12, 21, 23], the algorithm presented in this paper is relatively simple, both conceptually and computationally. It does not require sophisticated algorithms such as online learning, probabilistic models, behavior modeling, etc. The algorithm uses basic mathematical operations for modeling the pixel and region stability and the required number of computations depends on video resolution and the variability of image content. Therefore, this algorithm is easy to implement in a working system, for example in an embedded system within the surveillance camera. In terms of computational complexity, the most time-consuming stages are the BS and (to a smaller degree) testing the pixel stability. The idea of the presented framework is to process live video streams, not the recordings. As indicated in Table 4, it was possible to achieve the expected processing speed (i.e. exceeding the source 15 fps) on a test machine using multi-threaded processing and by downscaling the image to 320×240 pixels, which did not decrease the accuracy of stable regions detection in a significant way. The problem with a detailed analysis of computational complexity of the proposed algorithm is that, similarly to the Codebook algorithm, computation time per pixel and per image frame is not constant, but it strongly depends on image content and its variability [36]. Each pixel is modeled with zero, one or more vectors and the number of vectors per pixel changes depending on image content. The most important factors that influence the computation time are as follows.

1. Number of moving objects in the scene. For low traffic conditions, most of the pixels will have zero vectors so that computations will be faster. If there are numerous moving objects, computation time for the same video resolution will be significantly longer, because more pixels will have a non-empty set of vectors.

Table 4 Number of frames analyzed per second in the proposed algorithm, for the original and downscaled resolutions of a video recording from the airport test set, for single- and multi-threaded processing

Video resolution	Single thread fps	Four threads fps
1,280×960	5.38	11.73
640×480	17.87	31.52
320×240	49.06	62.11
160×120	88.10	110.17

2. On the pixel level, computation time depends on how often the image content changes. For low image variability, pixels will be represented with a low number of vectors and it is more probable that the first vector will match. If the image content changes rapidly, there will be multiple vectors per pixel and there is a possibility that all of them will be analyzed and none will match, therefore the processing time will be longer.
3. Also, for each vector, two conditions are tested: for brightness and color. If the image does not change rapidly, the first vector may be matched after testing both conditions. For rapid image changes, in the worst case, both conditions will have to be checked for all vectors.

As a result, computation time differs significantly between scenarios such as 'rush hours' and late night at the airport. It is not possible to establish "standard" conditions for evaluation of complexity. Averaging the results over a long period (e.g. 24 h) is problematic and also does not provide a valid measure of the algorithm complexity. The results presented in Table 4 were obtained for one example from the airport set, with moderate intensity of object movement, representing averaged conditions occurring during constant monitoring. It should also be noted that both the BS and stable pixels detection procedures are parallel in nature, so it is possible to achieve a much higher processing rate on higher resolution video streams using massively parallel processors such as GPUs or modern processors in embedded systems.

6 Conclusions

Extraction of image regions representing stationary objects is an important step in the process of unattended luggage detection. Such objects have to be identified in the video stream with high accuracy in order to provide necessary data to the decision module. The proposed algorithm solves the problem of stationary objects detection by testing the stability of pixel values and extracting stable regions from the image. This algorithm was designed in a way that makes it easy to implement it in the framework for online event detection, operating on live video streams from the surveillance cameras. In such a system, objects representing luggage may be selected from all detected stable ones with the classifier, and detection of unattended luggage is performed by testing defined conditions. Compared to other approaches that aim to solve the problem of unattended luggage, this system is highly flexible. Each module may be tuned and even replaced by another algorithm performing the same function, for example: a complex decision system (which implements advanced criteria omitted here, e.g. tracking the owner) may replace the simple one used in the tests, utilizing the same data on stationary objects. Compared to other algorithms for stationary objects detection, the proposed one has a relatively low computational complexity so it is suitable for implementation in practical systems working in online mode. It also utilizes the results of BS, which is a standard procedure (which may already be implemented in the system, e.g. on DSP of the camera), instead of relying on time-consuming procedures such as feature matching in the whole image. The algorithm is also robust against short-term occlusions because the data on temporarily obstructed objects is preserved in the proposed model. Also, there is no need to track movement of each individual object, which is very problematic in the real airport with a high number of simultaneously moving persons.

The experiments performed using the real airport recordings confirmed that the presented algorithm works with satisfactory accuracy and is suitable for implementation in a working unattended luggage detection system. The most important drawback of the proposed approach is that if the input data coming from the BS procedure is inaccurate, false results may be obtained. The decision module was deliberately simplified for the purpose of the experiments, some important aspects (such as tracking the luggage owner) were omitted in order to focus the

experiments on detection of stationary objects. However, other researchers may use the proposed algorithm to find stationary luggage in the video stream and use this data as an input to their own, advanced reasoning algorithms. Finally, it should be stressed that in contrast to the existing commercial systems for automatic event detection in video which have a closed form, the presented video analysis framework is highly flexible and allows for tuning and reconfiguration by the end user, allowing them to adjust the algorithms to specific conditions.

Table of symbols

$a_{\min}$	Minimum area of a stable region (Eq. 10)
b	Blue color component of an image pixel
C	Part of the object contour covered by a stable region (Eq. 9)
d	Maximum absolute brightness difference between the inside and outside pixels in a neighborhood of a contour border pixel (Eq. 13)
$d_{\min}$	Minimum value of d that indicates important brightness difference
g	Green color component of an image pixel
$\mathbf{I}$	Currently analyzed color image
l	Brightness of the pixel
$\mathbf{M}$	Background subtraction mask obtained from $\mathbf{I}$
O	Contour of a detected moving object, obtained from $\mathbf{M}$
P_{in}, P_{out}	Sets of pixels inside and outside a contour in a neighborhood of a border pixel (Eq. 11)
p	Image pixel
q	Number of times in which region $\mathbf{r}$ was marked as invalid
$\mathbf{R}$	Mask of stable pixels obtained from $\mathbf{I}$ and $\mathbf{M}$
$\mathbf{r}$	Vector describing a region in the stable regions model (Eq. 15)
r	Red color component of an image pixel
S	Connected region of stable pixels obtained from $\mathbf{R}$ (Eq. 7)
s	Stable pixel belonging to a stable region S
t	Timestamp of the currently analyzed image $\mathbf{I}$
T_A	Alarm time – minimum period during which an unattended luggage is detected before an alarm is raised
t_c	Creation time of vectors in the stable pixels model
t_m	Last match time of a vector in the stable pixels model
$T_{\max\ q}$	Maximum number of frames in which a stable region $\mathbf{r}$ is invalid
$T_{\max\ u}$	Maximum period during which a stable region $\mathbf{r}$ is not updated
$T_{\max\ v}$	Maximum period during which a vector $\mathbf{v}$ is not updated
T_s	Pixel stability threshold (Eq. 6)
u	Timestamp of the last update of a stable region $\mathbf{r}$
$\mathbf{v}$	Vector in the pixel stability model (Eq. 1)
z	Number of frames in which region $\mathbf{r}$ represented unattended luggage
α	Brightness difference threshold for pixel stability testing (Eq. 2)
β	Minimum area of object contour covered by a stable region (Eq. 10)
$\delta_{\max}$	Color difference threshold for pixel stability testing (Eq. 3)
ρ	Update rate of vectors in the stable pixels model (Eq. 4)
ζ	Edge brightness difference measure of a contour (Eq. 14)
$\zeta_{\min}$	Minimum edge difference measure for valid (not false) contours

Acknowledgments The research is subsidized by the European Commission within FP7 project "INDECT" (Grant Agreement No. 218086).

References

1. Auvinet E, Grossmann E, Rougier C, Dahmane M, Meunier J (2006) "Left-luggage detection using homographies and simple heuristics," in Proc PETS 2006, New York, , pp 51–58
2. Bhargava M, Chen CC, Ryoo MS, Aggarwal JK (2009) Detection of object abandonment using temporal logic. Mach Vis Appl 20:271–281
3. Breiman L (2001) Random forests. Mach Learn 45:5–32
4. Chang L, Zhao H, Zhai S, Ma Liu YH (2013) Robust abandoned object detection and analysis based on online learning. IEEE Int Conf Robot Biomim (ROBIO), pp 940–945
5. Connell J, Senior A, Hampapur A, Tian Y, Brown L, Pankanti S (2004) Detection and tracking in the IBM people vision system. IEEE ICME 2:1403–1406
6. Cotres C, Vapnik V (1995) Support-vector networks. Mach Learn 20:273–297
7. Czyżewski A, Szwoch G, Dalka P et al (2011) Multi-stage video analysis framework. In: Lin W (ed) Video Surveillance. Intech, Rijeka, Croatia, pp 147–172
8. Dalka P (2006) Detection and segmentation of moving vehicles and trains using gaussian mixtures, shadow detection and morphological processing. Machine Graphics and Vision 15:339–348
9. Elhamod M, Levine M (2013) Automated real-time detection of potentially suspicious behavior in public transport areas. IEEE Trans Intell Transp Syst 14(2):688–699
10. Evangelio R, Patzold M, Sikora (2011) "A system for automatic and interactive detection of Static objects," Proc IEEE Workshop Pers. Oriented Ver, pp. 27–32
11. Guler S, Farrow MK (2006) "Abandoned object detection in crowded places," in Proc. PETS 2006, New York, 2006, pp. 99–106
12. Guler S, Silverstein JA, Pushee IH (2007) "Stationary objects in multiple object tracking," Proc. AVSS 2007, pp 248–253
13. Hettiarachchi CA, Nanayakkara A, Dissanayaka C, Wijenayake C, De Silva (2014) "Abandoned object detection with logical reasoning," IEEE Int Adv Comput Conf (IACC), pp 1137–1141
14. Hu MK (1962) "Visual pattern recognition by moment invariants", information theory. IRE Transactions 8: 179–187
15. i-Lids dataset for AVSS (2007) [Online] Available: http://www.eecs.qmul.ac.uk/~andrea/avss2007_d.html
16. INDECT Project website [Online] http://www.indect-project.eu/
17. Kim K, Chalidabhongse TH, Harwood D, Davis L (2005) Real-time foreground-background segmentation using codebook model. Real-time Imaging 11:167–256
18. Krahnstoever N, Tu P, Sebastian TA, Perera, Collins R (2006) "Multi-view detection and tracking of travelers and luggage in mass transit environments," in Proc. PETS 2006, New York, 2006, pp 67–74
19. Li L, Luo R, Ma R, Huang W, Leman K (2006) "Evaluation of an IVS system for abandoned object detection on PETS 2006 Datasets," in Proc PETS 2006, New York, 2006, pp 91–98
20. Lin W (ed) (2011) Video Surveillance. Intech, Rijeka, Croatia
21. Lu S, Zhang J, Feng DD (2007) Detecting unattended packages through human activity recognition and object association. Pattern Recogn 40:2173–2184
22. Lv F, Song X, Wu B, Singh VK, Nevatia R (2006) "Left-luggage detection using bayesian inference," in Proc PETS 2006, New York, 2006, pp 83–90
23. Maddalena L, Petrosino A (2013) Stopped object detection by learning foreground model in videos. IEEE Trans Neural Networks and Learning Systems 24(5):723–735
24. Martínez-del-Rincón J, Herrero-Jaraba JE, Gómez JR, Orrite-Uruñuela C (2006) "Automatic left luggage detection and tracking using multi-camera UKF," in Proc. PETS 2006, New York, 006, pp 59–66
25. McPartlin M (2011) "SUBITO (Surveillance of Unattended Baggage and the Identification and Tracking of the Owner)," SUBITO Deliverable D100.2: Final Report. [Online] Available: http://cordis.europa.eu/documents/documentlibrary/134654311EN8.zip
26. Pan J, Fan Q, Pankanti S (2011) Proc. Int. Conf. Image Process, Robust abandoned object detection using region-level analysis., pp 3597–3600

27. PETS2006 Benchmark Data. [Online] Available: http://www.cvg.rdg.ac.uk/PETS2006/data.html
28. Porikli F, Ivanov Y, Haga T (2008) Robust abandoned object detection using dual foregrounds. EURASIP J Adv Signal Process vol 2008, article ID 197875, doi 10.1155/2008/197875
29. Singh A, Sawan S, Hanmandlu M, Madasu V, Lovell B (2009) "An abandoned object detection system based on dual background segmentation," Proc. 6th IEEE Int Conf Adv Video Signal Based Surveill, pp 352–357
30. Smith K, Quelhas P, Gatica-Perez D (2006) "Detecting abandoned luggage items in a public space," in Proc PETS 2006, New York, 2006, pp 75–82
31. Spengler M, Schiele B (2003) "Automatic detection and tracking of abandoned objects," in Proc. VS-PETS 2003, Nice, France
32. Stauffer C, Grimson WE (1999) "Adaptive background mixture models for real-time tracking," in Proc IEEE Conf on Computer Vision and Pattern Recognition, pp 246–252
33. Suzuki S, Abe K (1985) Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing 30:32–46
34. Szwoch G, Dalka P (2010) "Automatic detection of abandoned luggage employing a dual camera system," in Proc MCSS 2010: IEEE International Conference on Multimedia Communications, Services and Security, Krakow, Poland, pp 56–61
35. Szwoch G, Dalka P, Czyzewski A (2010) A framework for automatic detection of abandoned luggage in airport terminal. Intelligent Interactive Multimedia Systems and Services, Smart Innovation, Systems and Technologies 6:13–22
36. Szwoch G, Ellwart D, Czyżewski A (2012) Parallel implementation of background subtraction algorithms for real-time video processing on a supercomputer platform. J Real-Time Image Process. doi:10.1007/s11554-012-0310-5
37. Thirde D, Li L, Ferryma J (2006) "Overview of the PETS2006 Challenge," in Proc PETS 2006, New York, pp 47–50
38. Tian Y, Feris RS, Haowei L, Hampapur A, Ming-Ting S (2011) Robust detection of abandoned and removed objects in complex surveillance videos. IEEE Trans Syst Man Cybern Part C Appl Rev 41(5):565–576
39. Tripathi RK, Jalal AS, Bhatnagar C (2013) "A framework for abandoned object detection from video surveillance," 4th National Conf Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), pp 1–4
40. Tsai RY (1986) "An efficient and accurate camera calibration technique for 3D machine vision," Proc IEEE Conference on Computer Vision and Pattern Recognition, pp 364–372
41. Zheng W, Gong S, Xiang T (2012) Quantifying and transferring contextual information in object detection. IEEE Trans Pattern Anal Mach Intell 34(4):762–777
42. Zivkovic Z, Van der Heijden F (2006) Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recogn Lett 27(7):773–780

Grzegorz Szwoch was born in 1972 in Gdansk. In 1996 he graduated as a student from the Sound Engineering Department. His thesis was related to physical modeling of musical instruments. Since that time he has been a member of the research staff at the Multimedia Systems Department. In 2004 he received his Ph.D degree. His Ph.D thesis was related to acoustic modeling of hearing aid waveguides. Currently he participates in research projects concerning intelligent video monitoring algorithms. His work in this area focuses on object detection and tracking, and on automatic detection of important security threats in video.