AD-HOC: Appearance Driven Human tracking with Occlusion Handling

Roberto Vezzani, Rita Cucchiara
University of Modena and Reggio Emilia
{roberto.vezzani, rita.cucchiara}@unimore.it

Abstract

AD-HOC copes with the problem of multiple people tracking in video surveillance in presence of large occlusions. The main novelty is the adoption of an appearance-based approach in a formal Bayesian framework: the status of each object is defined at pixel level, where each pixel is characterized by the appearance, i.e. the color (integrated along the time) and the likelihood to belong to the object. With these data at pixel-level and a probability of non-occlusion at object-level, the problem of occlusions is addressed. The method does not aim at detecting the presence of an occlusion only, but classifies the type of occlusion at a sub-region level and evolve the status of the object in a selective way. The AD-HOC tracking has been tested in many application for indoor and outdoor surveillance. Results on PETS2006 test set are reported where many people and abandoned objects are detected and tracked.

1 Introduction

Appearance-based tracking is a well established deterministic paradigm to ensure temporal coherence of detected deformable objects in video streams. The research activity in tracking has a long history; a very good survey is [9]. Many kernel-based approaches provide tracking at level of a region of interest (e.g. an elliptical kernel) only, and do not need segmentation before tracking and are exploited in many scenario with fixed and mobile cameras. Instead, when segmentation is available the exploitation of appearance models or templates is straightforward, especially for deformable objects such as people. Templates enable the knowledge not only of the location and speed of visible people but also their visual aspect, their silhouette or the body shape at each frame. It is mandatory in people surveillance, action analysis and behavior monitoring, in order to have a precise information about the visible and non visible body aspect at each instant. The use of appearance-based tracking is now very widespread, since the pioneer works of Haritaoğlu et al. [5], and Bobick and Davis [1]. Senior et al. [7] defined appearance models and probabilistic maps to track people and vehicles even in presence of partial overlaps. Short term occlusions are implicitly taken into account in the adaptive model, by smoothing the appearance model according with a adaptive coefficient. Tracking fails if the occlusion duration is too high. In [3], in order to cope with dynamic occlusions of other people, the segmented blobs have been grouped into macro-objects with potential occlusions and then appearance-based tracking is solved with models against the points of the macro objects. Other works explore the problem of occlusion detection: for instance [6] detects

the presence of occlusion and freezes the model without updating any parameter if an occlusion occurs. In a recent work [4] an appearance model embeds both color appearance and ground occupancy map to deal with multiple occlusions in a setup with multiple cameras Occlusions are handled by computing the probability of occupancy of the ground plane at each possible location. In this work, we address tracking with occlusions using color appearance and likelihood at pixel-level (Ad Hoc - Appearance Driven tracking with Occlusion Classification). The tracking problem is formulated in a probabilistic Bayesian model, taking into account both motion and appearance status at pixel-level. The probabilistic estimation is redefined at each frame and a MAP optimization gives a single solution for each frame in a deterministic way. Then a novel occlusion detection and classification process defines a model of non visible regions not observable in the current frame e distinguishes three classes, depending on the possible cause: dynamic occlusions, scene occlusions and apparent occlusions (that are just only shape variations). The classification allows a selective updating of the status model and copes with large occlusion working in different manner in different parts of the shape. Examples on videos of PETS2006 dataset and downloaded from ViSOR[1] [8] are provided with many people, artifacts and objects. Abandoned objects are perfectly detected and people tracking is very satisfactory.

2 The Tracking Algorithm

Ad Hoc works at object-level for the status model, at pixel-level for matching and at region level for occlusions. The object state vector is $O = \{\{o_1,\ldots,o_N\},\vec{c},\vec{e},\Pi\}$, where $\{o_i\}$ is the set of the N points of the object O, $\vec{c}$ and $\vec{e}$ are respectively the position with respect to the image coordinate system and the velocity of the centroid, Π is the probability of being the foremost object, thus the *probability of non-occlusion*. Each point o_i is defined as $o_i = \{(x,y),(R,G,B),\alpha\}$, where (x,y) are the coordinates with respect to the object centroid, (R,G,B) are the color components and $\alpha \in [0,1]$ is the likelihood to belong to the object.

The scene at each frame t is described by a set of objects $\mathscr{O}^t = \{O_1,\ldots,O_M\}$ which we suppose are generating the foreground image $F^t = \{f_1,\ldots,f_L\}$, i.e. the points of the MOV_t extracted by any segmentation technique. Each point f_i of the foreground is characterized by its position (x,y) with respect to the image coordinate system and by its color (R,G,B). The tracking aims to estimate the set of objects $\mathscr{O}^{t+1}$ observed in the scene at time/frame $t+1$, based on the foregrounds extracted up to now. In a probabilistic framework, this is obtained by maximizing the probability $P(\mathscr{O}^{t+1}|F^{0:t+1})$, where the notation $F^{0:t+1} \doteq F^0,\ldots,F^{t+1}$. In order to perform this MAP (maximum a posteriori) estimation, we make the assumption of having a first order Markovian model, meaning that $P(\mathscr{O}^{t+1}|F^{0:t+1}) = P(\mathscr{O}^{t+1}|F^{t+1},\mathscr{O}^t)$. Moreover, by using the Bayes theorem, it is possible to write

$$P(\mathscr{O}^{t+1}|F^{t+1},\mathscr{O}^t) \propto P(F^{t+1}|\mathscr{O}^{t+1})P(\mathscr{O}^{t+1}|\mathscr{O}^t)P(\mathscr{O}^t). \quad (1)$$

Optimizing Eq. 1 in an analytic way is not possible, so this would require to test all the possible objects sets, by changing their positions, appearances, and probabilities of non-occlusion. This is definitely unfeasible, so we break the optimization process in two

[1] http://www.openvisor.org

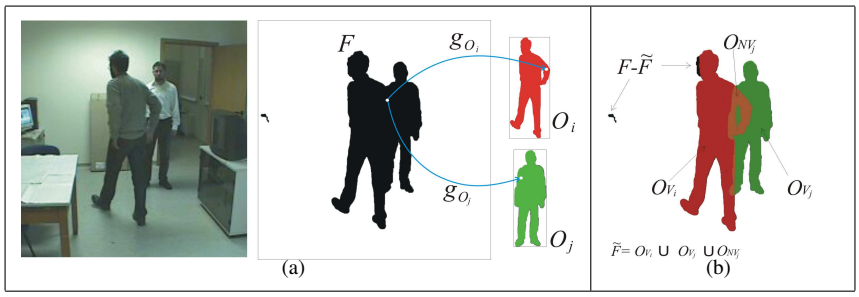


Figure 1: a) Domain and Codomain of the function g_O. g_O transforms the coordinates of a foreground pixel $x \in F$ to the correspondent object coordinates. b) Visible and non visible part of an object. $\tilde{F}$ is the foreground part not covered by an object.

steps, by locally optimizing the position, then updating the appearance and the probability of non-occlusion.

2.1 Position optimization

The optimization of the centroid position for all objects is the first task. The term $P(\mathscr{O}^t)$ in Eq. 1 is set to 1, since we just keep the best solution from the previous frame. The term $P(\mathscr{O}^{t+1}|\mathscr{O}^t)$ that is the motion model, is provided by a circular search area of radius r around the estimated position $\hat{c}$ of every object. In order to measure the likelihood of the foreground to be generated by an object, we define a relation among the corresponding points of F and O with a function $g_O : F \rightarrow O$, and its domain $\tilde{F}_O$ that is the set of foreground points matching object's points. Then $\tilde{F} = \bigcup_{O \in \mathscr{O}} \tilde{F}_O$ is the set of foreground's points which match at least one object. In the same way we call $\tilde{O}$ the codomain of the function g_O, that includes the points of O which have a correspondence in $\tilde{F}$. (See Fig. 1(a)). Since the objects can be overlapped, a point f can be in correspondence with a set of objects defined as $\mathscr{O}(f) = \left\{O \in \mathscr{O} : f \in \tilde{F}_O\right\}$. The term $P(F^{t+1}|\mathscr{O}^{t+1})$ is given by the likelihood of observing the foreground image given the objects positioning, that can be written as:

$$P(F^{t+1}|\mathscr{O}^{t+1}) = \prod_{f \in \tilde{F}} \left[\sum_{O \in \tilde{O}(f)} P(f|g_O(f)) \cdot \Pi_O \right] \tag{2}$$

obtained by adding for each foreground pixel f the probability of being generated by the corresponding point $o = g_O(f)$ of every matching object $O \in \mathscr{O}(f)$, multiplied by its non-occlusion probability Π_O. The conditional probability of a foreground pixel f, given an object point o is modeled by a Gaussian distribution, centered on the RGB value of the object point:

$$P(f|o) = \frac{1}{(2\pi)^{3/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\bar{f}-\bar{o})^{\mathrm{T}}\Sigma^{-1}(\bar{f}-\bar{o})} \cdot \alpha(o) \tag{3}$$

where $\bar{(\cdot)}$ and $\alpha(\cdot)$ give the RGB color vector and the α component of the point respectively, and $\Sigma = \sigma^2 I_3$ is the covariance matrix in which the three color channels are assumed to be uncorrelated and with fixed variance σ^2. The choice of sigma is related to the amount of noise in the camera. For our experiments we choose $\sigma = 20$.

It is reasonable to assume that the contribution of the foremost objects in Eq. 2 would be predominant, so we locally optimize the function, by considering only the foremost object for every point. The algorithm proceeds as follows: a list with the objects sorted by their probability of non-occlusion (assuming that this is inversely proportional to the depth ordering) is created; then the first object O is iteratively extracted from the list and its position $\vec{c}$ is estimated by maximizing the probability:

$$P(\tilde{F}|O) \propto \prod_{f \in \tilde{F}_O} P(f|g_O(f)). \tag{4}$$

After finding the best $\vec{c}$, the matched foreground points are removed and the foreground set F is updated as $F = F \setminus \tilde{F}_O$. The object O is removed from the list and the process continues until the object list is empty. The described algorithm may fail for objects which are nearly totally occluded, since a few pixels could force a strong change in the object center positioning. For this reason we introduce for the center estimation a *Confidence* measure $Conf(O) = \sum_{o \in \tilde{O}} \alpha(o) / \sum_{o \in O} \alpha(o)$.

2.2 Pixel to Track Assignment

This is the second phase of the optimization of Eq. (1). Once all the tracks have been aligned, in this top-down approach we aim at adapting the remaining parts of each object state. Even in this case we adopt a sub-optimal optimization. The first assumption we made is that each foreground pixel belongs to only one object. To this aim we perform a bottom-up discriminative pixel to object assignment finding the maximum of the following probability for each point $f \in \tilde{F}$:

$$P(O \rightarrow f) \propto P(f|g_O(f)) \cdot P(g_O(f)) = P(f|g_O(f)) \cdot \alpha(g_O(f)), \tag{5}$$

where $P(f|g_O(f))$ is the same of (3) and we use the symbol $\rightarrow$ to indicate that the foreground pixel f is generated by the object O. Directly from the above assignment rule, we can divide the set of object points into visible O_V and non-visible O_{NV} points:

$$O_V = \{o \in O \,|\, \exists f = g_O^{-1}(o) \wedge \underset{O_i \in \mathscr{O}}{argmax}(P(O_i \rightarrow f)) = O\} \quad O_{NV} = O - O_V \tag{6}$$

In other words, the subset O_V is composed by all the points of O that correspond to a foreground pixel and that have won the pixel assignment. (See Fig. 1(b)). The α value of each object point is then updated using an exponential formulation:

$$\alpha(o^{t+1}) = \lambda \cdot \alpha(o^t) + (1-\lambda) \cdot \delta(o, O_V) \tag{7}$$

where $\delta(\cdot,\cdot)$ is the membership function.

$$\delta(o, O_V) = \begin{cases} 1 & o \in O_V \\ 0 & o \notin O_V \end{cases} \tag{8}$$

The equation (7) includes two terms: one proportional to a parameter $\lambda \in [0,1]$ that corresponds to $P(O^{t+1}|O^t)$ and reduces the α value at each time step, and one proportional to $1-\lambda$ that increases the α value for the matching visible points $P(F|O)$. Similarly we update the RGB color of each object point:

$$\vec{o}^{t+1} = \lambda \cdot \vec{o}^t + (1-\lambda) \cdot f \cdot \delta(o, O_V) \tag{9}$$

The last step of the object state updating concern the non occlusion probability Π. To this aim we first define the probability Po^{t+1} that on object O_i occludes another object O_j.

$$Po(O_i,O_j)^{t+1} = \begin{cases} 0 & \beta_{ij} < \theta_{occl} \\ (1-\beta_{ij})Po_{ij}^t & a_{ij} = 0 \\ (1-\beta_{ij})Po_{ij}^t + \beta_{ij}e^{\frac{a_{ji}}{a_{ij}}} & a_{ij} \neq 0 \end{cases}, \tag{10}$$

where

$$a_{ij} = \left\| O_{V,i} \underset{g}{\cap} O_{NV,j} \right\| = \left\| g_{O_i}^{-1}(O_{V,i}) \cap g_{O_j}^{-1}(O_{NV,j}) \right\| \quad \beta_{ij} = \frac{a_{ij}+a_{ji}}{\left\| O_i \underset{g}{\cap} O_j \right\|} \tag{11}$$

a_{ij} is the number of points shared between O_i and O_j and assigned to O_i; β_{ik} is the percentage of the area shared between O_i and O_j assigned to O_i or O_j, that is less or equal to 1 since some points can be shared among more than two objects.

The value β is used as update coefficient, allowing a faster update when the number of overlapping pixels is high. Vice versa, when the number of those pixel is too low (under a threshold θ_{occl}), we reset the probability value to zero. The probability of non occlusion for each object can be computed as:

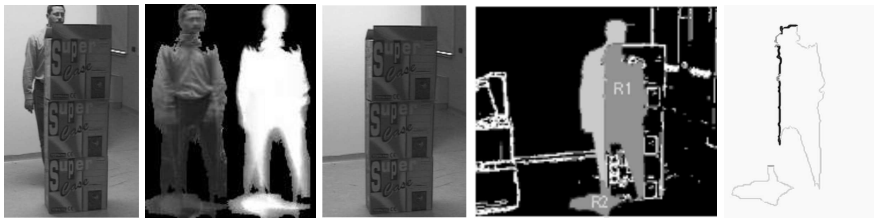
$$\Pi(O_i)^{t+1} = 1 - \max_{O_j \in \bar{\mathcal{O}}} Po(O_i,O_j)^{t+1}. \tag{12}$$

2.3 Track initialization

With the probabilistic framework previously described we can "assign and track" all the foreground pixels belonging to at least one object. Instead, the foreground image contains points $f(\in F - \tilde{F})$ without any corresponding object, due to shape changes or the entrance into the scene of new objects as well. We suppose that a blob of unmatched foreground points is due to a shape change if it is connected (or close to) an object, and in such a situation the considered points are added to the nearest object; otherwise a new object is created. In both cases the α value of each new point is initialized to a predefined constant value (e.g., 0.4). Obviously in this manner we cannot distinguish a new object entering the scene occluded by or connected to a visible object. In such a situation the entire group of connected object will be tracked as a single entity.

3 Occlusion Classification

Due to occlusions or shape changes, some points of an object could have no correspondence with the foreground F. Unfortunately, these two reasons require two different and conflicting solutions. To keep a memory of the object shape even during an occlusion the object model needs to be slowly updated; at the same time a fast updating can better face shape changes. To this aim, the adaptive update function has been enriched by the knowledge of occlusion regions. In particular, if a point is detected as occluded we freeze the color and α value of the point instead of using Eq. (7) and Eq. (9). The introduction of a higher level reasoning is necessary in order to discriminate between occlusions and shape changes. The set of non visible points O_{NV} are the candidate points for occluded



(a) (b) (c) (d) (e)

Figure 2: Example of a R_{SO} region: (a) The input frame. (b) The color and the α representation of the object points. (c) The current background model (d) The visible part of the track and the candidate occlusion regions (R1 and R2). (e) The borders of the Non Visible Regions. Points that have a good match with edge pixels of the background are marked in black. Thus R_1 is classified as R_{SO} while R_2 as R_{AO}

regions. After a labeling step over O_{NV}, a set of not visible regions (of connected points) is created; sparse points or too small regions are pruned and a final set of *Non Visible Regions* $\{NVR_j\}$ is created. Each of them can be classified as one of these three classes:

1. *dynamic occlusions* R_{DO}: occlusions due to overlap of another objects, closer to the camera; therefore the pixels of these regions were assigned to the other object;

2. *scene occlusions* R_{SO}: due to (still) objects, included in the scene and therefore into the background model and thus not extracted by the foreground segmentation algorithm, but actually positioned closer to the camera;

3. *apparent occlusions* R_{AO}: regions not visible because of shape changes, silhouette's motion, shadows, or self-occlusions.

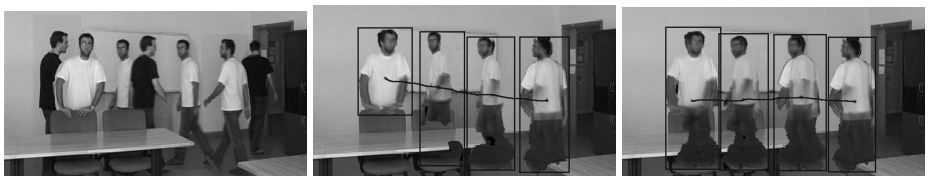
In the first and second case (R_{DO} and R_{SO}) the occluded parts of the objects should not be updated since we do not want to lose the memory of it. Instead, in the third case (apparent occlusion), not updating the object state would create an error. The solution is a *selective update* according to the region classification.

The detection of the first type of occlusion is straightforward, because we always know the position of the objects, and we can easily detect when two or more of them overlap. R_{DO} regions are composed by the points shared between object O_k and other object O_i but not assigned to O_k. We can mathematically formulate this check as:

$$R_{DO} = \left\{ o \in O_{NV,i} \,|\, \exists o' \in O_j \wedge g_{O_i}^{-1}(o) = g_{O_j}^{-1}(o') \right\} \tag{13}$$

To distinguish between R_{SO} and R_{AO} the position and the shape of the objects in the background can be helpful, but not provided with our segmentation algorithm. To discriminate between R_{SO} and R_{AO} we exploit the background edges set. This subset of points of the background model contains all points of high color variation, among which the edges of the objects are usually detected. In case of a R_{SO} we would expect to find edge points in correspondence of the boundary between this R_{SO} and the visible part of the track. In other words, if the separation of the visible and the non visible part of an object correspond to an edge of the background, then plausibly we are facing an occlusion between a still object of the scene and the observed moving object. Otherwise, the shape

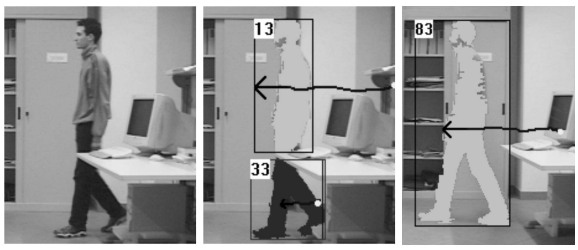
change is more reasonable cause of the no more visible points. In Fig. 2 an example is shown: a person is occluded for a large part by a stack of boxes that are included in the background image. Two parts of its body are not segmented and two candidate occlusion regions are generated (Fig. 2(c)): one of them is a shadow included in the object model but now disappeared. In Fig. 2(d) the borders of the NVRs are shown, with pixels that have a good match with the edges marked in black. In real occlusion due to a background object the percentage of the points that have a match with respect to the set of bounding points is high; thus the region is classified as R_{SO}. On the contrary, for the apparent occlusion (the shadow) we have no matching pixels, and consequently this region is classified as R_{AO}. An example of R_{SO} occlusion correctly solved is reported in Fig. 3.



(a) (b) (c)

Figure 3: Example of background object occlusion. (a) A composition of the input frames. (b) Evolution through time of the appearance model of one of the tracks in case the background object occlusions are not handled. (c) The same appearance model with the handling of R_{SO}s.

4 Refinements



(a) (b) (c)

Figure 4: Merge. (a) input frame; (b) two objects created; (c) the objects are merged

The described model works well in real situations if the initial conditions are ideal, that is if we assume that a single person is entering in the scene at a time, and not occluded by other objects. However, in order to correctly manage all the other conditions, the tracking system has to cope with the well-known problems of **merge** and **split** of objects. The same object due to occlusion could initially appear as two different objects: in the example of Fig. 4, two of them are associated to the single person entering the scene, due to the occlusion of the table. In our work, we exploit the motion vectors of the object to

(a) (b) (c)

Figure 5: Split. (a) a single object for two people separating; (b) the probability mask where two different components are visible; (c) the object is split in two objects

detect if a merge is needed. In case the objects are near and have similar motion vectors they are merged together (see Fig. 4(c)).

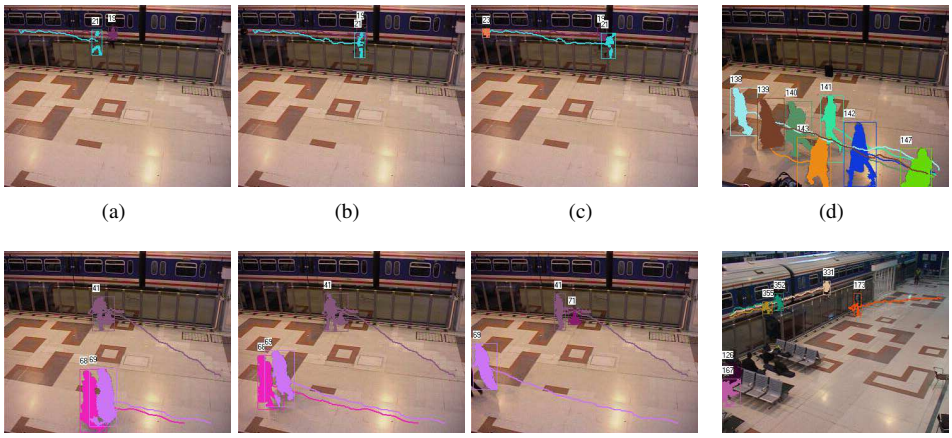
The opposite problem arises when a group of people enters the scene together (see Fig. 5). Since they are represented by a single blob, only one object is created. While some authors proposed a splitting technique based on head detection [5, 10], we decided to split the objects only in case of group separation. To this aim, the probability mask is periodically analyzed to check the presence of two or more well-separated connected components; in such case, the object is split (Fig. 5(c)) and one or more new objects are created. This method is potentially less reactive, since the presence of two or more person is detected only when they are visually separated, but it does not rely on head detection, which is not straightforward in some cases (e.g. a person with an arm higher than the head).

5 Applications and Experiments

The algorithm has been implemented in our library in C++ and used in many different applications. The proposed tracking system keeps for each object both appearance ($\{o_1, \ldots, o_N\}$) and motion ($\vec{c}$, $\vec{e}$) information. By means of these features it is possible to apply reasoning algorithm in order to (i) classify the tracked object and distinguish among real object, person, group of people, vehicle; (ii) estimate the motion activity (moving, stopped or fixed object); (iii) detect and classify object interactions.

In our experiments we exploited a background suppression algorithm based on a simple selective median as background model and with a shadows suppression [2]. Then the foreground points are matched against the object's points. This approach also allows us to segment objects remaining still in the foreground.

As described in section 3, the algorithm deal with scene occlusion detection. In the example in Fig. 3 a man (with a white t-shirt) is walking behind a table (included in the background model) and stopping there. Moreover, he also crosses another person walking in the opposite direction. In the cases where the model update is adaptive, but not selective with occlusion classification, after a given of time, depending on the update coefficient, the memory of his occluded legs is lost, and therefore the centroid and the bounding box are not correctly evaluated (Fig. 3(b)). This is a problem in applications such as posture detection, since the silhouettes of the segmented objects, and all the other features that can be computed, hardly match a standing-up posture model. The problem is solved with the detection of background object occlusions where the occluded part of the object is



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 6: Sample output of the tracking system on the PETS2006 dataset

frozen, leading to a better evaluation of object's position (Fig. 3(c)). Note that because we don't handle scaling in object alignment, the "frozen" legs appear smaller than their real dimension.

In order to give some quantitative results we tested the tracking system on some videos from ViSOR [8] and from the PETS2006 dataset. In particular we used the 7 videos of the third camera, since its point of view has a slightly changing background. In the used video 191 people appeared in the scene leaving 7 luggages. In addition to identify all the abandoned luggages, our tracking algorithm correctly manages 33 dynamic occlusions, 15 groups of people and correctly tracks 165 people. It fails in some cases with 4 identity change, 5 split of head and feet and 4 groups of people are not correctly split. The 86% of people and 100% of objects are correctly tracked also in extreme cases. For instance in Fig. 6 (S1V3 Video) two people are walking behind the plexiglass, crossing each other. The labels are correctly assigned and kept. Other example frames showing different kind of interactions between people in the scene are depicted in the right part of Fig. 6(d,h). The aim of the PETS2006 workshop is to use existing systems for the detection of left (i.e. abandoned) luggage in a real-world environment. The luggage items are at first tracked together with their owners; after having been abandoned, though, the split algorithm must be invoked in order to assign two different identities to the person and the luggage. The second row of Fig. 6 shows the efficacy of our abandoned object detection algorithm over the PETS2006 sequences.

6 Conclusion

In this work we defined, developed, and tested the *AD-HOC* tracking, a novel approach for multiple people tracking in video surveillance applications. In particular, our effort was focused on overcoming large and long-lasting occlusions by using an appearance driven tracking model. Working at a pixel level, it has been possible to define and manage *non-visible regions*, i.e. the parts of the objects that are not detected in the current frame,

allowing the detection of occlusions. A first effective aspect is the two step algorithm that gives a fast solution to a probabilistic model. The main novelty of the system is a classification of non-visible regions into three classes, which aims at distinguishing between actual occlusions (*dynamic occlusions*), occlusions with an object belonging to the background (*scene occlusions*), and shape changes (*apparent occlusions*). Therefore, based on classification results, a different behavior can be adopted to keep memory of the occluded parts of each object and to recover them once they appear again. The proposed tracking is very robust and fast; it has been adopted in several projects of indoor and outdoor people surveillance, with many people and real operating conditions.

References

[1] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[2] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1337–1342, October 2003.

[3] R. Cucchiara, C. Grana, G. Tardini, and R. Vezzani. Probabilistic people tracking for occlusion handling. In *Proceedings of Int'l Conference on Pattern Recognition*, volume 01, pages 132–135, August 2004.

[4] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multicamera people tracking with a probabilistic occupancy map. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):267–282, Feb. 2008.

[5] I. Haritaoglu, D. Harwood, and L.S. Davis. W4: real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, August 2000.

[6] Hieu Tat Nguyen and Arnold W. M. Smeulders. Fast occluded object tracking by a robust appearance filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1099–1104, 2004.

[7] Andrew Senior, Arun Hampapur, Ying-Li Tian, Lisa Brown, Sharath Pankanti, and Ruud Bolle. Appearance models for occlusion handling. *Image and Vision Computing*, 24(11):1233–1243, November 2006.

[8] Roberto Vezzani and Rita Cucchiara. ViSOR: Video surveillance on-line repository for annotation retrieval. In *"Proceedings of IEEE International Conference on Multimedia & Expo (IEEE ICME 2008)"*, June 2008.

[9] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.

[10] Tao Zhao and Ram Nevatia. Tracking multiple humans in complex situations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1208–1221, 2004.