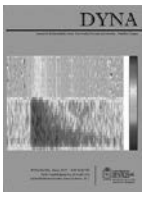
UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE MEDELLÍN
FACULTAD DE MINAS



DYNA
http://dyna.medellin.unal.edu.co/

Pedestrian tracking using probability fields and a movement feature space

Pablo Negri [a] & Damián Garayalde [b]

[a] *Universidad Argentina de la Empresa (UADE). CONICET. Buenos Aires, Argentina. pnegri@uade.edu.ar*
[b] *Instituto Tecnológico de Buenos Aires (ITBA), Buenos Aires, Argentina. dgarayal@itba.edu.ar*

Received: April 18th, 2016. Received in revised form: November 1rd, 2016. Accepted: December 2nd, 2016.

Abstract
Retrieving useful information from video sequences, such as the dynamics of pedestrians, and other moving objects on a video sequence, leads to further knowledge of what is happening on a scene. In this paper, a Target Framework associates each person with an autonomous entity, modeling its trajectory and speed by using a state machine. The particularity of our methodology is the use of a Movement Feature Space (MFS) to generate descriptors for classifiers and trackers. This approach is applied to two public sequences (PETS2009 and TownCentre). The results of this tracking outperform other algorithms reported in the literature, which have, however, a higher computational complexity.

Keywords: pedestrian tracking, movement feature space, target framework

Seguimiento de peatones utilizando campos probabilísticos y un espacio de descriptores dinámicos

Resumen
Recuperar información de secuencias de video, como la dinámica de peatones u otros objetos en movimiento en la escena, representa una herramienta indispensable para interpretar que está ocurriendo en la escena. Este artículo propone el uso de una Arquitectura basada en Targets, que asocian a cada persona una entidad autónoma y modeliza su dinámica con una máquina de estados. Nuestra metodología utiliza una familia de descriptores calculados en el Movement Feature Space (MFS) para realizar la detección y seguimiento de las personas. Esta arquitectura fue evaluada usando dos bases de datos públicas (PETS2009 y TownCentre), y comparándola con algoritmos de la literatura, arrojó mejores resultados, aun cuando estos algoritmos poseen una mayor complejidad computacional.

Palabras clave: seguimiento de peatones, espacio de descriptores dinámicos, target framework.

1. Introduction

Vision-based object tracking is an important task and a useful source of information. It analyzes video sequences to retrieve the motion of an object at each frame [1]. Recovered metrics can consist of location, orientation, speed, and acceleration, computed on the image plane (2D) or real world (3D) reference coordinates. In general, the complexity is closely related to the object tracked: its articulated nature, or abrupt motion changes. Complex scenarios with illumination changes, noise, and object-to-object and/or object-to-scene occlusions will also degrade the tracking performance, particularly on non-controlled real life video sequences. Some examples of object tracking applications include: motion-based recognition [2], automatic surveillance [3], traffic monitoring [4], and vehicle navigation [5]. In this list, pedestrians or people are one of the most interesting objects to track for researchers and developers. In addition, it is an open subject because of its high complexity given a person's changing appearance, non-rigid structure, and occasional hazardous motion.

1.1. Related work

There are different approaches to tackle human tracking. Some trackers use a bounding box at an initial frame of the

How to cite: Negri, P. and Garayalde, D., Pedestrian tracking using probability fields and a movement feature space. DYNA 84(200), pp. 217-227, 2017.

sequence. Among them, two methods from the Tracking-by-Matching group [1] are distinguished by their simplicity and good performance. The first one is the Lukas-Kanade (LKT) algorithm [6]. It seeks to locate either a moving or non-moving object from one image to the next frame in the sequence. This method iteratively minimizes a dissimilarity measure in the neighborhood of a point of the tracked object. Shi and Tomasi [7] prove that corners are the best choice to obtain optimal tracking results. The second method is Mean Shift [8], which locates the object position within the next frame by maximizing a similarity coefficient calculated with the Bhattacharyya distance. This coefficient compares the color distribution of the target object against the possible object positions on the following image.

Online discriminative classification [9] also uses initial bounding boxes. This method trains an adaptive classifier considering the first bounding box as positive, and the surrounding background as negative. Exhaustive research on the image at t+1 (the consecutive frame) will provide a new positive and negative sample which updates the classifier, and the loop repeats itself.

When pedestrians (or vehicles) are the only moving objects on the scene, background suppression methods can estimate their motion. In [4], the dynamics of collected blobs are described by a collection of key-points. They are tracked by similarity functions matching the new blobs with stored objects.

In surveillance applications, where new pedestrians continuously enter the scene, pre-trained trackers using a target model which is known *a priori* are employed. This methodology, called Tracking-by-Detection, is generally implemented starting with pedestrian location hypotheses generated by a person detector. Dalal's and Triggs's people detector [10], provided by OpenCV [11], is perhaps the most widely used in the literature. Those hypotheses are associated with previously saved tracks. This matching can be performed using the Hungarian algorithm [12]. Finally, the tracking itself can be performed by particle filters [13], Kalman-inspired Event Cones [14], or the evolution of a state machine [4]. The procedures of object detection and trajectory estimation can be combined into a coupled optimization problem [14], enhancing their individual performance. This methodology is robust to changing backgrounds, moving cameras, and the presence of other moving objects, and is the best adapted approach to be used on real-world, non-controlled, video tracking sequences.

Offline tracking systems are a variant of the Tracking-by-Detection pipeline [15,16]. They seek a global optimization of people's trajectories scanning forwards and backwards through the hypothesis locations at each frame to find the best path to explain the collected data. Ben Shitrit *et al.* [15] divide the ground plane on cells, and associate a probability occupancy map from people detector outputs. They use the K-Shortest Paths algorithm (KSP) to find the trajectories on the grid cells. The identities of the path are found by running a Linear Program procedure.

1.2. Proposed methodology

This paper aims at conducting pedestrian tracking from monocular video sequences captured by a calibrated camera with a fixed view on outdoor real scenes. Pedestrians can have different postures, and they are walking at different distances from the camera. Given that recorded sequences have cluttered and changing backgrounds, our tracking system follows a methodology based on the Tracking-by-Detection Framework.

The first contribution of this paper is an adapted tracking procedure using the Movement Feature Space (MFS). In [17,18], the MFS was successfully used to detect vehicles and pedestrians respectively. The advantages of this detector include efficient calculation time, increased robustness, and minimum loss of information. Also, in the MFS, all the operations are performed in motion, thus the presence of cluttered backgrounds does not interfere with the tracking algorithm. Since the MFS does not have a notion of pixel intensity or color to compute an image gradient [6], or color histogram [8], the tracking approach is based on tracking fields: an object detection field, and an appearance field. The former is constructed using the people detector output scores, and provides the likelihood of a person at a given location. The appearance field is computed using a corner analysis applied on the MFS, capturing a pedestrian texture which is robust enough to perform the tracking when partial information is available.

This paper also proposes an architecture where each pedestrian is considered as an autonomous entity, and their evolution on the scene is followed individually by a *Target Framework*. The framework associates one target with one pedestrian from his/her first view on the scene until he/she disappears from sight. A state machine models the dynamics of the target, which is continuously stored at a repository inside the framework. The evolution of the states of each target is employed in the data association stage, filtering false alarms, or concatenating broken trajectories.

The Target Framework combines on-line and off-line tracking methods. Firstly, the video sequence is analyzed on-line, populating with targets a repository which saves the temporal information about all the hypotheses generated during the detection and tracking. The next stage implements off-line algorithms to filter false alarms and concatenate broken trajectories. The performance of the detection and tracking system is evaluated in two public datasets, and compared against two state-of-the-art tracking systems [12,16]. The sensitivity of the procedure with different people detectors is also analyzed.

This paper is organized as follows: the section below details the detection and tracking procedure on the MFS, as well as the state machine associated with each target. Section 3 develops the procedures pruning the target framework in order to improve the results. Next, the Implementation System Setup is presented. In the Results and Discussion section the performance of the system on the tests datasets is described, followed by the Conclusions of the paper.

2. Online target framework generation

The Target Framework builds an on-line target repository based on the scene dynamics. Fig. 1 details the pipeline to generate this repository. Frame F_t at instant *t* is projected on

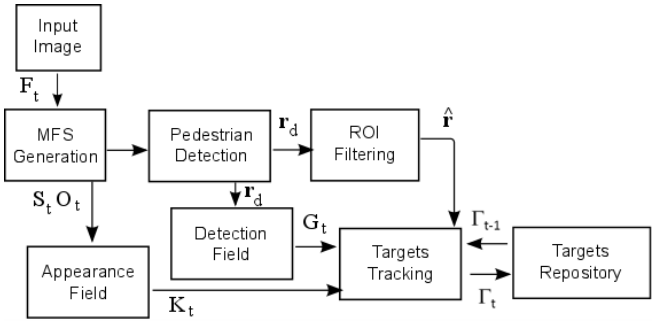


Figure 1. Block diagram of the pedestrian on-line tracking system and the Target Framework.
Source: The authors.

the MFS capturing the motion on the scene. The *Pedestrian Detection* block uses the MFS on a people detector to generate pedestrian hypotheses $\mathbf{r}_d$, also referred to as Regions of Interest (ROIs or rois). The *ROI Filtering* block filters neighbors and superposed rois in $\mathbf{r}_d$ using the Non-Maximal Suppression (NMS) algorithm [19]. The resulting set $\hat{\mathbf{r}}$ is used in the data association step. $\mathbf{r}_d$ also generates a probability map G_t on the *Detection Field* block. The MFS data are used in the *Appearance Field* step to generate a Gaussian corner map K_t. The *Target Repository* block saves all the targets produced within the sequence. It is defined as the set of active targets $\Gamma_{t-1} = \{T_{1,t-1}, \ldots, T_{n,t-1}\}$ using those present on the previous frame. The *Target Tracking* block employs Probability fields G_t and K_t, detected rois $\hat{\mathbf{r}}$, and active targets set Γ_{t-1}, to update Γ_t opening new targets, and closing others.

2.1. Pedestrian detection on the MFS

The MFS is an adaptive motion extraction model. It uses level lines and their orientations as features to generate an adaptive background model. The motion in the frame at time t, corresponds to the set of level lines which do not belong to the background model. It is encoded in two arrays: St and Ot, as shown on Fig. 2. Matrix St(p) counts the number of moving level lines passing through pixel p, and Ot(p) indicates the orientation of the level lines with a different color. The background details are not present on the St(p) and Ot(p) matrix, as can be seen on the figure. New static objects entering the scene, will integrate the background model after a temporal window.

Fig. 2 shows an example of the pedestrian detector output. The detector consists of a cascade of boosted classifiers trained using the Real Adaboost approach. The feature family encoding the information of the MFS are the histograms of oriented level lines (HO2L). They are computed by accumulating the number of pixels on Ot(p) having the same orientation (see [18] for further details). The detector output is a list of rois, as depicted on Fig. 2, with their associated confidence score: rd={ri,si}n=0,...,n-1. Each roi is defined as ri = [xic , yic , wi , hi], where (xic, yic) is its central position and (wi, hi) are its width and height, respectively. The NMS filtering is applied to rd in order to determine the estimated pedestrian positions $\hat{\mathbf{r}}$.

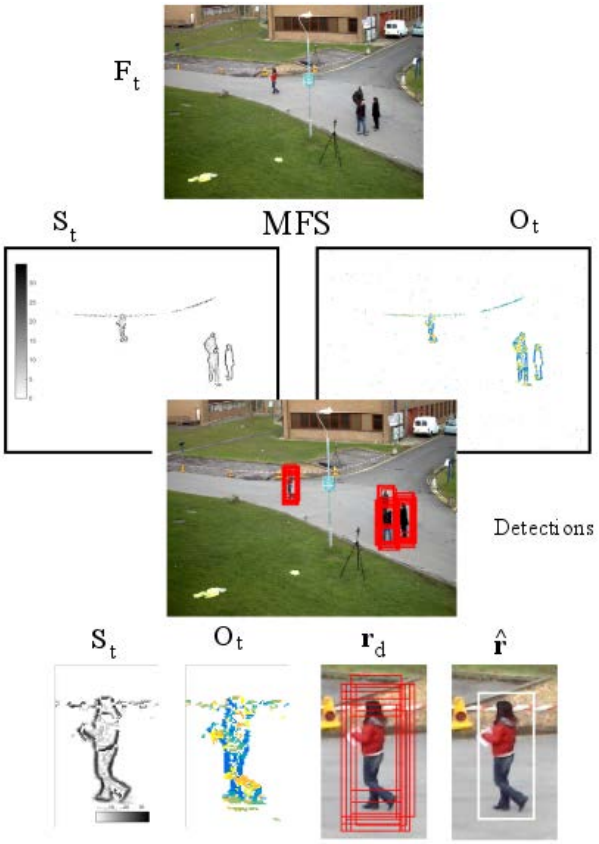


Figure 2. MFS computation in the PETS2009 dataset and the result of a pedestrian detector. The Fig. shows the MFS information of the S_t and the O_t arrays of the entire capture, and a zoom on the pedestrian position. It also shows set $\mathbf{r}_d$ of pedestrian detected rectangles, and filtered pedestrian position $\hat{\mathbf{r}}$.
Source: The authors.

2.2. Association of targets and detections

A target is an autonomous entity individually tracing a pedestrian moving on the scene. Target *i* is described with parameters: $T_{i,t-1} = \{b,id,e,\mathbf{m}\}$. $T_{i,t-1}.b=\{x,y,w,h\}$ is the bounding box containing the pedestrian, $T_{i,t-1}.id$ is a label identifying the target, $T_{i,t-1}.e$ is the state of T_i, and $T_{i,t-1}.\mathbf{m}$ is the motion history consisting of the last *z* displacement vectors: $\mathbf{m}=\{\mathbf{d}_{t-z},\ldots,\mathbf{d}_{t-1}\}$.

The association of the active targets on Γ_{t-1} and the pedestrian hypotheses $\hat{\mathbf{r}}$ at time *t* is key for the Tracking-by-Detection approach. As a result, it is possible to validate targets, filter false alarms, or use alternative tracking procedures if the detector fails.

The association task is as follows: each pair (r_j ,T_i), with r_j being a detection on $\hat{\mathbf{r}}$ and T_i one of the *n* targets on Γ_{t-1}, generates a displacement vector $\vec{v} = \{|\vec{v}|, \theta\}$ from the central point of $T_i.b$ to the central point of r_j , where $|\vec{v}|$ and θ are the modulus and the angle of $\vec{v}$. Overlap ratio a_0 between $T_i.b$ and r_j is used as a confidence criterion, and is evaluated employing the PASCAL VOC formula [19]:

$$a_0 = \frac{area(T_i.b \cap r_j)}{area(T_i.b \cup r_j)} \quad (1)$$

Assuming slow changes in the pedestrian dynamics, $|\vec{v}|$ should have low values, θ should be similar to $\angle(\bar{m})$, where $\bar{m}$ is the average of the motion vectors saved in T_i.**m**, and a_0 would be greater than zero. The angle between $\vec{v}$ and $\bar{m}$ is computed using the dot product:

$$\measuredangle(\vec{v},\bar{m}) = \frac{cos^{-1}(\vec{v}.\bar{m})}{|\vec{v}||\bar{m}|} \quad (2)$$

After all the (r_j, T_i) pairs are evaluated, target T_i is associated with the detection r_j which best matches its historical motion. The best match will consist of the pair which minimizes$|\vec{v}|$ and $\angle(\vec{v},\bar{m})$, maximizing a_0. If r_j is not associated with any T_i, a new target T is created and saved in the Γ_t set. A T_i of Γ_{t-1} is considered as `lost` if no detection r_j matches its dynamic.

2.3. Probability fields for tracking targets

To track one target, the procedure estimates its position on the current frame F_t from its position recorded in Γ_{t-1}. In this paper, tracking is conducted using two types of tracking fields: a detection field and an appearance field. The detection field is computed using output score s_i of the people detector on the detected set $\mathbf{r}_d=\{r_i,s_i\}_{i=0,\dots,n-1}$ The appearance field is based on a corner extraction on the MFS.

2.3.1. Detection field

The Detection Field $G_t(\mathbf{x})$ is a probability map generated using the rois set $\mathbf{r}_d=\{r_i,s_i\}_{i=0,\dots,n-1}$, where detected rois $r_i=\{x_i, y_i, w_i, h_i\}$. To compute the G_t field, ROIs r_i in $\mathbf{r}_d$ generate the map M(**x**) as follows: $\mathbf{M}(\mathbf{x}) = \sum_i s_i\,\delta(\mathbf{x}-\mathbf{x}_i)$, where δ(**x**) is a kroneker delta in R^2, and $\mathbf{x}_i$ is the central point of r_i. The detection field is computed by convolving the map M(**x**) with a 2D Gaussian filter:

$$G_t = \mathbf{M}(x) * \mathbf{g} \quad (3)$$

$$\mathbf{g}(\mathbf{x}') = \frac{1}{\sqrt{|\Sigma|(2\pi)^2}} e^{-\left(\frac{1}{2}(\mathbf{x}'-\mathbf{x}_c)^T\Sigma^{-1}(\mathbf{x}'-\mathbf{x}_c)\right)} \quad (4)$$

where the parameters of the Gaussian filter include the covariance matrix $\Sigma=0.1^2\,[(w_i)^2\ 0\ ;\ 0\ (h_i)^2]$, the central point in patch $\mathbf{x}_c=(w_i^k/2, h_i^k/2)$, and the position inside patch $\mathbf{x}'=(x',y')$ where $x'=0,\dots,w_i-1$ and $y'=0,\dots,h_i-1$. The highest values in $G_t(\mathbf{x})$ are associated with a high confidence output of the people detector, and could suggest the presence of a person at position **x** of the image.

2.3.2. Appearance Field

The tracking using the Appearance Field is activated when one target $T_i \in \Gamma_{t-1}$ is not associated with a detection in $\mathbf{r}_d$. As an example, if Γ_{t-1} has only one target T_0 and $\mathbf{r}_d$ is empty (the detector failed to detect the pedestrian), all the elements of Detection Field G_t will be zero. In those cases, the Appearance Field will be used to compare the pedestrian characteristics from the previous and the present frame of the sequence. Furthermore, this procedure is robust enough to track the target with partial information.

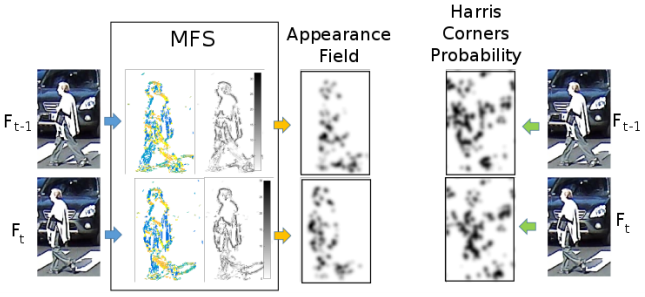


Figure 3. Appearance field generation from MFS corners of two consecutive captures from an urban video sequence.
Source: The authors.

The Appearance Field uses arrays S_t and O_t of the MFS, on a vector-based corner detector [17]. This corner detector considers that, in a neighborhood B_p of one corner point *p*, there are pixels with significant gradients with different orientations. The average of the cross products between all the pixels in B_p is, in general, greater than the value computed in the neighborhood of a pixel that is not a corner.

The average cross product in neighborhood B_p can be computed as: $K_t = I_x^2 \langle I_y^2\rangle + I_y^2 \langle I_x^2\rangle - 2\, I_x I_y \langle I_x I_y\rangle$. $\langle\rangle$ is the convolution with a 5x5 mask, where all its elements are ones, except for the center which has a zero value. Assuming that orientations O_t are given in radians, values I_x and I_y are defined as: $I_y = S_t \sin(O_t)$ and $I_x = S_t \cos(O_t)$.

Higher values of K_t suggest the presence of corners, and are shown as darker regions in Fig. 3. A Gaussian filter with standard deviation σ=3 is applied to smooth map K_t. As can be seen in O_t and S_t in Fig. 3, the rear vehicle does not generate corners, because it belongs to the background model. This is a great advantage of the methodology. Two consecutive captures of the dataset and the corresponding Appearance Field are shown on Fig. 3. Both fields are similar and the tracking system can successfully follow the person. Fig. 3 also compares the corner map obtained using the Harris corner detector which works on the gray scale image. As can be seen, the background behind the person incorporates a lot of noise for the tracking system.

2.4. Tracking procedure

This section describes the iterative tracking method. It is inspired by the LKT and Mean Shift trackers. Instead of using image intensities [13] or colors [15], the algorithm uses tracking fields: Detection and Appearance Fields.

2.4.1. Iterative Tracking Algorithm

For a given $T_{i,t-1} = \{b,id,e,\mathbf{m}\} \in \Gamma_{t-1}$, the methodology seeks their most probable position in F_t using $T_{i,t-1}.b$ as the first hypothesis. Algorithm 1 presents a pseudo-code of the

Algorithm 1. Iterative Tracking Algorithm

Require P, Q, $\mathbf{y}_0$, **g**

Ensure Flow motion vector **d**

```
it = 0
Compute target distribution q
y1 = y0 + g
ρ0 = ρ(y0)
By1 are the 8-connected neighbors of y1
While it < MAXIT
        ρmax = -inf
        Compute ρ[p(yn), q], n=1,...,8 ∈ By1
        For each point n in By1
                If ρ(yn) > ρ0 then
                        If ρ(yn) > ρmax then
                                ρmax = ρ(yn)
                                y1 = yn
        If ρmax = -inf then
                Break
        it = it + 1
        ρ0 = ρmax
Return d = y1 - y0
```

Source: The authors.

iterative tracking. It has four inputs: two tracking fields, Q and P, obtained from the previous frame and the current frame respectively, initial position $\mathbf{y}_0$, and a first displacement hypothesis **g**. If the association stage matches a detection ROI to T_i, the tracking fields correspond to Gaussian fields: $Q = G_{t-1}$ and $P=G_t$. Otherwise, the tracking fields employed correspond to appearance fields: $Q=K_{t-1}$ and $P=K_t$. $\mathbf{y}_0$ is the central point of $T_{i,t-1}$.b: $\mathbf{y}_0$ = {b.x + b.w / 2,b.y + b.h / 2}. Subsection 2.4.2 details the use of the displacement hypothesis.

Distributions Q and P are respectively computed to obtain a similarity score comparing position $\mathbf{y}_0$ at time *t-1* to a new position $\mathbf{y}_i$, at time *t*. We define the target distribution $\mathbf{q}=\{q_u\}_{u=1,\ldots,g}$, which are all the pixel values of field Q inside the ROI centered at $\mathbf{y}_0$. Distribution $\mathbf{p}(\mathbf{y}_i)$ are the pixel values of field P inside the ROI centered at $\mathbf{y}_i$. These vectors are normalized to obtain distributions, i.e. $\sum_{u=1}^{g} p_u = 1$. The similarity score is calculated based on the Bhattacharyya coefficient [8]:

$$\rho(y_i) \equiv \rho[p(y_i, q)] = \sum_{u=1}^{g} \sqrt{p_u(y_i) q_u} \qquad (5)$$

In (5), both distributions have the same length g (the ROIs have the same size). The algorithm will seek to find the new position $\mathbf{y}_1 = \mathbf{y}_0 + \mathbf{d}_t$ in P, where $\mathbf{d}_t$ is a displacement vector, which maximizes the similarity criteria.

The initial position of $\mathbf{y}_1$ is the location of the target in the previous frame, $\mathbf{y}_0$. At each iteration, $\mathbf{y}_1$ moves one pixel toward the largest Bhattacharyya coefficient in the 8-connected neighborhood B_{y1}. This algorithm converges to a local maximum due to the nature of the Gaussian fields. However, a necessary condition for an accurate convergence is that the local maximum should be in the neighborhood of $\mathbf{y}_0$. To ensure this condition, and a fast convergence, next location $\mathbf{y}_1$ of the object is searched in a pyramidal representation of tracking field P. The number of iterations of the algorithm was fixed to *MAXIT=20* in our tests, but, in general, the convergence is reached after two or three iterations.

Fig. 4 shows an example of the computation of the displacement vector $\mathbf{d}_t$ using the probability fields. In the upper box, the people detector had found the person in both frames F_{t-1} and F_t. Detection fields G_{t-1} and G_t could be created using their output ROIs $\mathbf{r_d}^{t-1}$ and $\mathbf{r_d}^{t}$. Thus, the tracking algorithm uses both probability fields to find the flow vector $\mathbf{d}_t$ associated with the dynamics of the pedestrian. The second box simulates when a pedestrian is lost by the detector, and then, the system uses appearance fields to obtain flow motion vector $\mathbf{d}_t$.

4.2.2. Pyramidal search

The pyramidal search is carried out on downsampled versions of the original probability fields P and Q. These versions are denoted as P^L and Q^L, with L={0,1,2}. The new size of P^L and Q^L corresponds to the original size of P and Q divided by factor 2^L. The tracking of target T_k starts in the third level L = 2 of the pyramidal representation. Algorithm 1 is executed using P^2, Q^2 and $\mathbf{y^2_0} = \mathbf{y_0} / 4$ as parameters. We also define a vector $\mathbf{g}^L$, which pre-translates tracking field Q^L, and represents a first displacement hypothesis. For L = 2, $\mathbf{g}^L=[0\ 0]^T$. Algorithm 1 returns flow vector $\mathbf{d}^L$, which is used to compute the pre-translation vector of the next level: $\mathbf{g}^{L-1} = 2 \cdot (\mathbf{g}^L + \mathbf{d}^L)$. The algorithm is executed a second time using the next level of the pyramidal representation. Final solution **d** of the pyramidal tracking is obtained by computing the displacement at level *0*: $\mathbf{d} = \mathbf{g}^0 + \mathbf{d}^0$. Motion vector $\mathbf{d}_t$ is stored in target k, and its new location is computed as: $T_{k,t}.b = T_{k,t-1}.b + \mathbf{d}_t$.

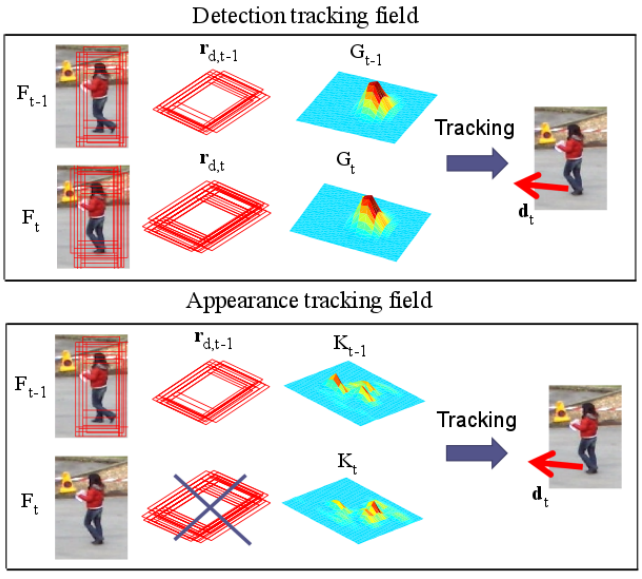


Figure 4. Flow motion vectors dt obtained from probability fields: using detection fields in the upper box, and using appearance fields in the lower box.
Source: The authors.

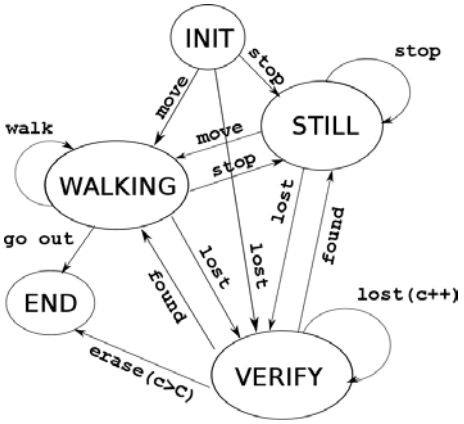


Figure 5. State machine of an individual target.
Source: The authors.

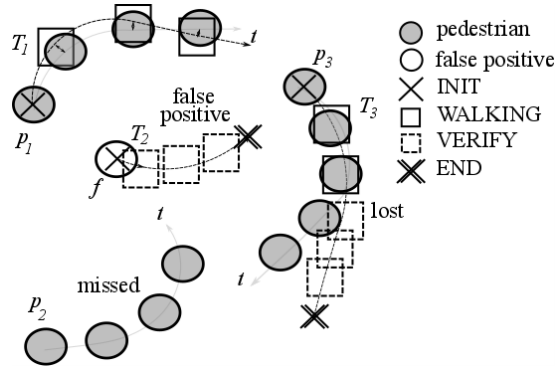


Figure 6. This figure shows different situations using the target framework. In this schema, real pedestrians p_i are represented by filled circles, false positives by empty circles, and T_i are shown by their different states.
Source: The authors.

4.3. State machine of a target

The target framework models each pedestrian as an autonomous agent with a state machine. Throughout the target lifetime, from the first time it appears in the sequence, until it exits the view, the tracking system collects information about its evolution generating events that trigger transitions between the states. Fig. 5 shows the states of a target and the events generating the transitions to the following states.

- INIT STATE:

A detection roi r_k in Υ_t not associated with an existing target in Γ_{t-1} generates a new target T_{new}={b,id,e,**m**}. The initial position of T_{new}.b copies the value from r_k. The number identifying the target, T_{new}.id, is computed incrementing by one the last id number. Motion history T_{new}.**m** is created as an empty array. The state of T_{new}.e is initialized with the INIT value. Then, target T_{new} is stored in the Γ_t set of the Target Repository. In the next frame, if the association stage does not find a corresponding detection on Υ_t, the `lost` event is generated and the new state will be VERIFY. When a detection is associated with the target, the tracking procedure generates a motion vector **d**. If the module of **d** is near zero, the `stop` event triggers the transition to the STILL state. Otherwise, the `move` event is generated and the new state of the target will be WALKING.

- STILL STATE

The target remains in this state until a `move` event, generated with a value of motion vector **d** other than zero, triggers the machine to the WALKING state. A `lost` event triggers a transition to the VERIFY state.

- WALKING STATE

In this state, the target is supposed to be continuously moving. The average of the last three motion vectors, $\mathbf{d}_{t-2}$, $\mathbf{d}_{t-1}$ and $\mathbf{d}_t$ saved on the **m** array, estimates this movement. If this value is near zero, the `stop` event is generated and the target changes to the STILL state. Otherwise, the target remains on the WALKING state. This procedure helps to filter some tracking errors and to generate smooth transitions between the states. If the target goes beyond the limits of a scene, a `go out` event is generated and there is a transition to the END state.

- VERIFY STATE

This state is triggered when the association task does not match any detection to the target, and the `lost` event is generated. The transition to this state initializes a counter *c* of the number of captures where the target is `lost`. In this state, the tracking of the target is executed using the Appearance Field. If counter *c* reaches a threshold *C*, an `erase` event is generated, and the state machine changes to the END state. However, when the association procedure finds a detection corresponding to the target position, the `found` event is generated and the target state has a transition to STILL or WALKING, depending on the modulus of motion vector $\mathbf{d}_t$.

- END STATE

This state closes the target, and will not be present at next set Γ_{t-1} .

Fig. 6 shows some examples of the tracking procedure with different detection results. Target T_1 correctly tracks pedestrian p_1. T_3 is an example of a lost pedestrian, and T_2, which was generated by a false alarm *f*, was immediately closed because *f* was the only detection.

5. Off-line target framework pruning

This section develops the off-line evaluation of the target framework through the sequence. All the targets are analyzed in order to filter false alarms and concatenate broken tracks. The procedures detailed in the next sections can be considered, however, as causal. This implies that it could be possible to adapt the false positive filtering and the concatenation algorithm to work on-line, but this is beyond the scope of this paper.

5.1. Filtering of false positives

False positives (detections that are not pedestrians) can be easily identified if the only states of the corresponding target are INIT and VERIFY, i.e. target T2 in Fig. 6. The example depicts the case where only one detection, possibly due to light conditions, shadows, etc., generates target T2 which is closed four frames later. To filter those cases, the target is eliminated if: #NV > #NS+#NW$, where #NV, #NS, #NW are the number of times the target is in the VERIFY, STILL and WALKING states, respectively.

5.2. Target concatenation with the kalman filter

The concatenation procedure aims at connecting a target in VERIFY state and a new target created before, but corresponding to the same pedestrian. The Kalman filter is used for its robust estimation of trajectories, providing metrics to match the lost target with one of the potential new ones as proposed by Deriche & Faugeras [20].

To perform the first task, a Kalman filter is used. It computes the optimal estimation of state vector X_t (positions, speeds and accelerations) from noisy measurements V_t, consisting of the ground plane positions of the target at time t [21]. Those positions are obtained using the calibration parameters of the camera and the scene. The model of our application, following the system dynamics and measurements, is shown below:

$$X_{t+1} = \boldsymbol{\Phi}_{t+1,t} X_t + \omega_t \quad (6)$$

$$V_t = H_t X_t + \upsilon_t \quad (7)$$

where ω_t and υ_t are assumed to be normally distributed with zero mean and covariances Q_t representing the model error, and R_t the covariance of the measurement error. $\boldsymbol{\Phi}_{t+1,t}$ is the evolution matrix, and $\mathbf{H}_t$, the selection matrix. A detailed description of matrices can be found in [22].

Let T_0 be the tracked target which changes to the VERIFY state at time t_v and whose position was estimated by the Kalman filter. Let $T_{i=1,\ldots,n}$ be all the other targets generated at t_v. The matching procedure consists of evaluating those targets that are near T_0 and have a similar evolution. Proximity is measured by the Mahalanobis distance between the filtered position of T_0 and the candidate position [20]:

$$d_i(T_0, T_i) = \left(X_{x,y}^{T_0} - V^{T_i}\right)^T (S)^{-1} \left(X_{x,y}^{T_0} - V^{T_i}\right) \quad (5)$$

where the covariance of difference $\left(X_{x,y}^{T_0} - V^{T_i}\right)$, S, is the sum of the Kalman covariance matrix $\mathbf{P}^{T_0}$ and R. Distance d_i has a χ^2 distribution with one degree of freedom. A threshold of $d_i < 3.84$ is proposed in order to account for a 95 percent probability of matching targets related to the same pedestrian. More distant targets are discarded. Candidate targets should also have a similar evolution to T_0. This is computed by comparing their displacement vector, and the displacement vector of the target candidate. From those targets which fulfill both conditions, the one with the minimum distance is matched with T_0 by unifying their ID. If none applies at time t_v, the targets on the next frame are evaluated. This procedure is repeated for W frames before considering the target as not concatenable.

Fig. 7 shows an example where a pedestrian has an abrupt change in direction, and the original T_0 lost the track. The search area where T_0 changes to the VERIFY state is defined by S. There are two candidates, T_i and T_k, initiating inside the area. However, the evolution of T_k is different from that of T_0, and the second criterion is not fulfilled. Then, T_i is retained and concatenated with T_0 updating its ID (T_i.id = T_0.id).

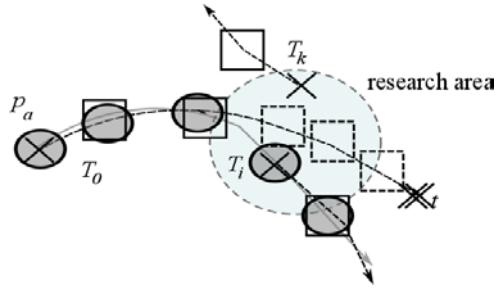


Figure 7. The figure shows the research area around the position of pedestrian p_a, target T_0 which is closed, and target candidates T_i and T_k for the matching procedure.
Source: The authors.

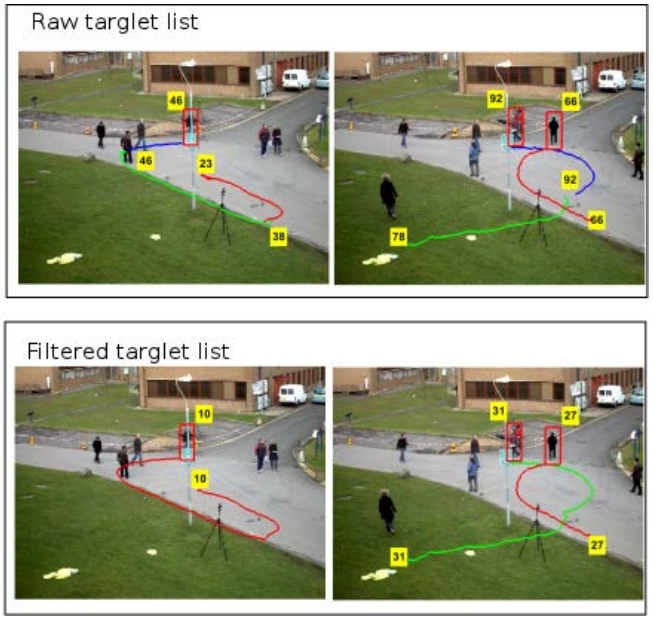


Figure 8. This figure shows the result of the concatenation procedure on a set of targets. The first row shows unconcatenated targets corresponding to a single pedestrian, and the second row shows concatenated results. The labels indicate the ID number of the targets at the beginning of their paths.
Source: The Authors.

Fig. 8 shows an example of the concatenation procedure of targets 29 and 65. The upper image of column (a) shows the trajectories of targets 29, 32, 35 and 42. Here, the track is broken due to abrupt changes in direction. A kind of *tail* on their path before closing can be noticed. This corresponds to the positions where the targets have VERIFY states, and lose the pedestrian's track. The lower image of the left column shows the tails, which were filtered on a concatenated trajectory. The right column shows a different case. The trajectories of targets 65, 66, 83, and 94 correspond to the same pedestrian, but target 66 is lost due to the occlusion by a pedestrian tagged with target 62. The lower image shows the concatenated result.

6. Experimental setup

6.1. Video sequences used for the tests

The system was evaluated in two public datasets. The first one is the PETS2009 dataset, task S2.L1 (view 1) [22]. The

sequence consists of 795 frames with 4650 annotated ROI positions corresponding to 19 pedestrians. The second is the Oxford Town Centre [23,24] dataset, which captures a pedestrian street with hundreds of people walking. The sequence was captured at 25 fps, and the total number of labeled frames is 4500. Thus, the total number of pedestrians is 230 from 71460 annotated ROIs.

6.2. Pedestrian detector training

The detection step of the target framework is performed by a cascade of boosted classifiers using the Real Adaboost algorithm. The detector was trained following the guidelines from [18], using a three-fold cross validation. The input features are the histogram of oriented level lines, HO2L, which are computed on the MFS. The resulting cascade has, on average, 23 strong classifiers, which are composed of a combination of generative and discriminative classification functions [25,26]. Training positive patches consist of pedestrian images captured from an outdoor street sequence. The total number is 6,726 positive samples [27]. The motion information of each patch is captured by the MFS, and the information employed to compute HO2L features are the S_t and the O_t matrices, as shown in Figs. 2 and 3. It is important to notice that the pedestrian detector is trained using people from different views and the appearance of the TownCentre and PETS2009 datasets (cross-database evaluation). The negative training set is PASCALVOC 2012, composed of 7,166 images without people [28]. The number of negatives is increased by rotating the images 90 degrees three times.

6.3. Tracking system benchmark

Two different kinds of software codes for tracking were evaluated in this paper to compare the performance of the Target Framework. The first is the *HierarchyEnsemble* on-line tracking algorithm proposed by Zhang [12,29]. The methodology is based on a Tracking-by-Detection algorithm combining a Mean Shift using a color appearance model, and a Kalman filter to follow the target motion dynamics. It also introduces a Tracker Hierarchy used to label each tracker as *novice* or *expert* depending on the number of templates accumulated during the tracking. The advantage of this methodology is that it does not need any calibration (it works on 2D) and the available code runs on-line. The *Continuous Energy Minimization* (CEM) is an off-line algorithm introduced by Milan [16,30]. It is a traditional forward and backward methodology which employs pedestrian detections to connect paths using linear approximation or splines. The final trajectories minimize a Total Energy computed using different metrics. It uses both 2D and 3D information (it needs calibration) and gives good output results.

The three tracking systems, *HierarchyEnsemble*, CEM, and MFS Target Framework (MFS-TF), employ pedestrian location hypotheses as input. In order to evaluate the sensitivity of the tracking algorithm with different inputs, the following files with pedestrian rois were used on the tests:

- *det_opencv*: these files are the result of the HOG detector [10,11] applied on the TownCentre and the PETS2009 datasets, and shared in Zhang's webpage [29].
- det_autre: there are two additional detection files, one for the PETS2009 provided by Milan [30], and the other for the TownCentre dataset shared at [24].
- ada_mfs_color: the pedestrian detector uses the MFS (see sec. 4.2), and a Color Texton Space (CTS) which better captures transitions between colored regions which could be lost on grayscale transformation [31].

6.4. Tracking evaluation

We have used Milan's code [30] implementing the CLEAR MOT metrics to test the performance of the target framework on the datasets. The multiple object tracking accuracy (MOTA) evaluated the tracking performance considering the false negative rate, false positive rate, and number of identity switches. The multiple object tracking precision (MOTP) measured the precision of the tracker computing an overlap ratio of the estimated position for matched true pedestrian-target pairs. For the MOTP, as in [13], a score of 50 percent was considered as significant for tracking, like the Pascal VOC Challenge [20]. False Negatives (FN) indicate the number of missed pedestrians. This score is closely related to the performance of the detector. False Positives (FP) measure the number of targets ROIs not matched to a pedestrian position. The other scores are False Alarms per Image (FPPI), the number of ground true (GT) unique pedestrians on the sequence and Mostly Tracked (MT) pedestrians.

7. Results and discussions

This section presents and discussed the results of the different detection files and the performance of the three tracking systems on the datasets.

7.1. Pedestrian detection performance

The performance of the tracking algorithms is closely related to pedestrian hypotheses provided by the detectors. Table 1 presents the results of the different detection files on the datasets. The first column indicates the name of the detection file's, and the second is the total number of pedestrian rois in the sequence. The correct detection percentage is shown on column *Det*. The other metrics, FN, FP and FPPI, were introduced in the "Tracking Evaluation" section. It is worth noticing that the PETS2009 dataset will be evaluated without using a region of interest, as suggested in many papers of the state-of-the-art. This reduces our *Det* ratio and increases the number of FP, compared with those

Table 1.
Performance of detection files on the datasets.

Detection file	Pos	Det (%)	FN	FP	FPPI
	PETS2009 S2.L1				
PETS09_S2L1_det_opencv	4650	84.6	715	235	0.29
PETS2009-S2L1-c1-det	4650	87.7	569	1353	1.70
ada_mfs_color	4650	82.9	794	90	0.11
	TownCentre				
TownCenter_det_opencv	64579	68.5	20340	5863	1.30
TownCentre_Body_DET	64579	77.1	14730	11126	2.47
ada_mfs_color	64579	74.4	16508	5843	1.29

Source: The Authors.

papers. The best performance of the detectors is found in the PETS2009 dataset. This is the noiseless sequence, with a simple static background. People walk without blocking each other, and there are no other moving objects on the scene.

The Town Centre sequence has a cluttered background, i.e. there are mannequins on the shop windows which are considered as pedestrians by the classifiers. In addition, pedestrians mask, either totally or partially, other people along their paths. This effect is especially remarkable when people are far away from the camera, and their size is small. It is then possible to conclude that the number of distracters for the detectors is more important in this sequence than in the first one. This would account for the decreased performance. As can be seen in Table 1, the MFS Adaboost detector works at an operational point which minimizes FP. On the other hand, the operational point of *PETS2009-S2L1-c1-det* and *TownCentre_Body_DET* maximizes the *Det* ratio, but increases by 15 and 1.9 times the FP number for the PETS2009 and TownCentre, respectively. The corresponding opencv classifier files have a behavior comparable to the MFS classifiers. However, with similar FP values, the MFS classifiers have lower FN values (miss rate).

7.2. Results of the tracking systems

The best performance on the MOTA metrics was obtained by MFS-TF using the *ada_mfs_color* detection file. MFS-TS also gets the highest MOTP metric ratio, which proves that the tracking algorithm generates targets that correctly overlap the pedestrian positions. When MFS-TS works with accurate detection files with a low miss rate and a low FP, such as the *ada_mfs_color* and *det_opencv* files, the behavior outperforms the other tracking systems in almost all the datasets. In the case of *ada_mfs_color*, our tracking algorithm improves the FN metric (related to the miss rate), increasing the number of the original detections by 290 for PETS2009, and 3600 for TownCentre. At the same time, the number of FP remains at very low values. The number of targets created by the algorithm is low, similarly to the SWIDs number showing that each pedestrian is well tracked. When MFS-TF works with detectors at different operational points, the improvements are less remarkable.

The *HierarchyEnsemble* algorithm [12] has the lowest number of tracks between the methodologies. It is related to the use of several color templates for each track. Therefore, a new track is created with a *novice* state if a detection has not matched the active tracks and the saved templates. It is also robust when occlusions happen, or when a pedestrian was lost from sight for some frames. This is the reason why the *Hierarchy Ensemble* has the highest MT score on TownCentre set. This methodology is very sensitive to the detectors with a high number of FN and FP. The analysis shows that those detections are noisy (central point and size of the roi), hindering the recollection of the pedestrian templates, and increasing the number of false tracks.

The *CEM* algorithm, developed by Milan *et al.* [16], uses a backward and forward growing path to complete non-detected pedestrian positions, obtaining a high score on MT. In general, it shows a better performance than Zhang's algorithm. It is also robust against noisy detections, because the track paths are obtained using interpolations (linear or spline). The main problem of this algorithm arises when the number of FP is high, such as in the case of the *det_autre* file for the TownCentre dataset. In those situations, the non-filtered FP draws an erroneous path, which, in turn, increases the FP number.

Table 2.
Evaluation scores for tracking results. The best scores are in bold and the second best results are underlined.

Method	MOTA (%)	MOTP (%)	FPPI	MT	FN	FP	SW id
PETS2009 S2.L1 (19 pedestrians)							
Zhang [12]							
det_opencv	67.9	71.2	0.75	13	877	593	21
det_autre	68.9	70.8	1.05	15	584	832	29
ada_mfs_color	78.6	70.5	0.44	15	609	346	42
Milan [16]							
det_opencv	72.9	65.4	0.38	11	926	301	35
det_autre	79.6	75.4	0.67	**18**	**367**	533	48
ada_mfs_color	82.2	71.0	0.18	14	616	147	67
Proposed Methodology: MFS-TF							
det_opencv	66.9	63.5	0.63	13	990	497	45
det_autre	74.3	66.7	0.78	15	525	620	41
TownCentre (225 pedestrians)							
Zhang [12]							
ada_mfs_color	**87.2**	**76.9**	0.18	17	402	143	48
det_opencv	62.1	71.5	2	117	15261	9016	168
det_autre	64.3	69.4	2.54	148	11356	11422	215
ada_mfs_color	51.1	69.4	4.85	**162**	9250	21834	434
Milan [16]							
det_opencv	62.0	68.2	1.36	107	17718	6142	628
det_autre	60.5	**76.2**	3	145	11427	13495	522
ada_mfs_color	70.1	69.2	1.24	137	12992	5591	683
Proposed Methodology: MFS-TF							
det_opencv	63	70.1	1.23	99	17763	5550	514
det_autre	68.3	74.6	1.67	141	12452	7531	472
ada_mfs_color	**70.4**	67.7	1.25	125	12891	5628	543

Source: The Authors

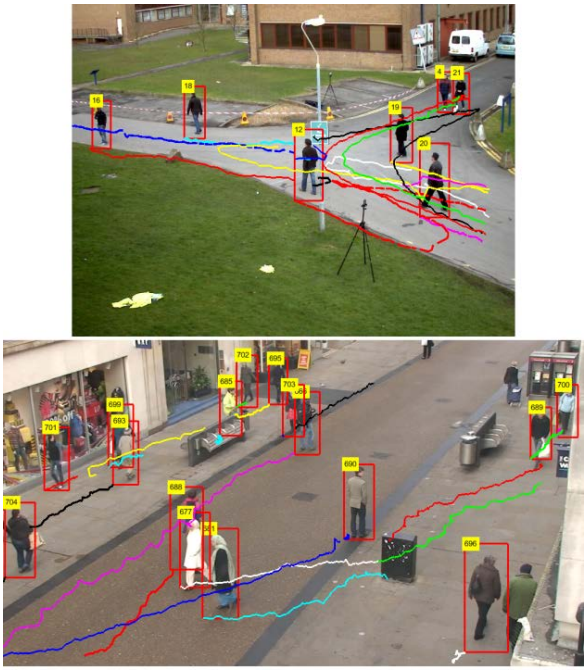


Figure 9. The figure shows the targets and their trajectories computed by our analysis. Each target trajectory is plotted with a different color.
Source: The Authors.

The results of our analysis are plotted on Fig. 9, where the pedestrians' trajectories are depicted using different colors indicating that a new target captured their paths. Table 2 reports the scores of the Target Framework, and the other two tracking methodologies applied on the four sequences following the CLEAR MOTS metrics [19]. The best results are in bold and the second best results are underlined.

8. Conclusions

The pedestrians' dynamics was successfully detected and tracked using the proposed Target Framework. The MFS used on the detection stage obtained accurate hypothesis detections. It also built an Appearance field which correctly followed pedestrians lost by the detector.

Future work will involve improving the overall system. For tracking purposes, the development of additional features should help to perform the tracking on the MFS without the necessity to work on still spaces (color histograms, etc.). Furthermore, the target state machine could be improved by adding other states, for example, to detect complex pedestrian behavior.

Acknowledgments

This paper was supported by PICT-2283 of ANPCyT, CONICET (Argentina), and ACyT A15T14 of UADE.

References

[1] Smeulders, A.W., Chu, D., Cucchiara, R., Calderara, S., Dehghan, A. and Shah, M., Visual tracking: An experimental survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 36(7), pp. 1442-1468, 2014. DOI: 10.1109/TPAMI.2013.230

[2] Goldhammer, M., Gerhard, M., Zernetsch, S., Doll, K. and Brunsmann, U., Early prediction of a pedestrians trajectory at intersections. IEEE Proceedings on Intelligent Transport Systems. pp. 237-242, 2013. DOI: 10.1109/ITSC.2013.6728239

[3] Zhang, Y., Ji, Q. and Lu, H., Event detection in complex scenes using interval temporal constraints. In: IEEE International Conference on Computer Vision. pp. 3184-3191, 2013. DOI: 10.1109/ICCV.2013.395

[4] Jodoin, J., Bilodeau, G. and Saunier, N., Urban tracker: Multiple object tracking in urban mixed traffic. In: IEEE Winter Conf. on App. of Comp. Vision. pp. 885-892, 2014. DOI: 10.1109/WACV.2014.6836010

[5] Keller, C. and Gavrila, D., Will the pedestrian cross?. A study on pedestrian path prediction. IEEE Trans. on Intell. Transp. Systems 15(2), pp. 494-506, 2014. DOI: 10.1109/TITS.2013.2280766

[6] Lucas, B. and Kanade, T., An iterative image registration technique with an application to stereo vision. In: Proc. Int. Joint Conf. on Artificial Intell. 2, pp. 674-679, August, 1981.

[7] Shi, J. and Tomasi, C., Good features to track. In: IEEE Proc. Comp. Vis. and Pattern Recognit. pp. 593-600, June, 1994. DOI: 10.1109/CVPR.1994.323794

[8] Comaniciu, D., Ramesh, V. and Meer, P., Real-time tracking of non-rigid objects using mean shift. In: IEEE Proc. Comp. Vis. and Pattern Recognit. 2, pp. 142-149, 2000. DOI: 10.1109/CVPR.2000.854761

[9] Kalal, Z., Mikolajczyk, K. and Matas, J., Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach. Intell. 34(7), pp. 1409-1422, 2012. DOI: 10.1109/TPAMI.2011.239

[10] Dalal, N. and Triggs, B., Histograms of oriented gradients for human detection. In: IEEE Proc. Comp. Vis. and Pattern Recognit.1, pp. 886-893, 2005. DOI: 10.1109/CVPR.2005.177

[11] OpenCV library v. 2.4.9.0, [online]. [Accessed April 2016]. Available at: http://opencv.org/

[12] Zhang, J., Presti, L. and Sclaroff, S., Online multi-person tracking by tracker hierarchy. In: Proc. Int. Conf. on Adv. Video and Signal-Based Surveillance. pp. 379-385, September, 2012. DOI: 10.1109/AVSS.2012.51

[13] Breitenstein, M., Reichlin, F., Leibe, B., Koller-Meier, E. and Van Gool, L., Online multiperson tracking-by-detection from a single, uncalibrated camera. IEEE Trans. Pattern Anal. Mach. Intell. 33(9), pp. 1820-1833, 2011. DOI: 10.1109/TPAMI.2010.232

[14] Leibe, B., Schindles, K., Cornelis, N. and Van Gool, L., Coupled object detection and tracking form static cameras and moving vehicles. IEEE Trans. Pattern Anal. Mach. Intell., 30(10), pp. 1683-1698, 2008.

[15] Ben-Shitrit, H., Berclaz, J., Fleuret, F. and Fua, P., Tracking multiple people under global appearance constraints. IEEE Int. Conf. on Comp. Vision. pp. 137-144, 2011. DOI: 10.1109/ICCV.2011.6126235

[16] Milan, A., Roth, S. and Schindler, K., Continuous energy minimization for multitarget tracking. IEEE Trans. Pattern Anal. Mach. Intell., 36(1), pp. 58-72, 2014. DOI: 10.1109/TPAMI.2013.103

[17] Negri, P., Estimating the queue length at street intersections by using a movement feature space approach. IET Image Processing 8, pp. 406-416, 2014. DOI: 10.1049/iet-ipr.2013.0496

[18] Negri, P., Goussies, N. and Lotito, P., Detecting pedestrians on a movement feature space. Pattern Recognition 47(1), pp. 56-71, 2014. DOI: 10.1016/j.patcog.2013.05.020

[19] Everingham, M., Gool, L., Williams, C.K., Winn, J. and Zisserman, A., The PASCAL Visual Object Classes (VOC) Challenge. Int. J. Comp. Vis. 8(2), pp. 303-338, 2010.

[20] Deriche, R. and Faugeras, O.D., Tracking line segments. In: European Conf. on Comp. Vis. pp. 259-268, 1990. DOI: 10.1016/0262-8856(90)80002-B

[21] Negri, P. and Garayalde, D., Concatenating multiple trajectories using kalman filter for pedestrian tracking. In: IEEE Biennial Cong. of Argentina. pp. 364-369, 2014. DOI: 10.1109/ARGENCON.2014.6868520

[22] PETS2009. [online]. [Accessed on June 2016], Available at: http://www.cvg.reading.ac.uk/PETS2009/a.html

[23] Benfold, B. and Reid, I., Stable multi-target tracking in real-time surveillance video. In: IEEE Proc. Comp. Vis. Pattern Recognit., pp. 3457-3464, 2011. DOI: 10.1109/CVPR.2011.5995667

[24] Town Centre video and data, [online]. [Accessed June 2016], Available at http://www.robots.ox.ac.uk/ActiveVision/Research/Projects/2009bbenfold_headpose/project.html

[25] Negri, P., Clady, X. and Prevost, L., Benchmarking haar and histograms of oriented gradients features applied to vehicle detection. In: Int. Conf. on Informat. in Control, Automat. and Robotics. pp. 359-364, 2007.

[26] Negri, P., Clady, X., Hanif, S. and Prevost, L., A cascade of boosted generative and discriminative classifiers for vehicle detection. EURASIP JASP 2008, pp. 1-12, 2008. DOI: 10.1155/2008/782432.

[27] Pedestrian Patches-PANKit. [online]. [Accessed on June 2016]. Available at http://pablonegri.free.fr/Downloads/PedestrianPatchesDataset-PANKit.htm

[28] PASCAL VOC2012 dataset. [online]. [Accessed June 2016], Available at (register required) http://host.robots.ox.ac.uk:8080/.

[29] Online multi-person tracking by tracker hierarchy code. [online]. [Accessed June 2016]. Available at: http://cs-people.bu.edu/jmzhang/tracker_hierarchy/Tracker_Hierarchy.htm

[30] Continuous energy minimization for multi-target tracking matlab code. [online]. [Accessed on June 2016]. Available at: http://www.milanton.de/contracking/

[31] Negri, P., and Lotito, P., Pedestrian detection using a feature space based on colored level lines. In: Alvarez, L., Mejail, M., Gomez, L. and Jacobo, J. (Eds). Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. CIARP 2012. Lecture Notes in Computer Science, 7441, pp 885-892, Springer, Berlin, Heidelberg, 2012. DOI: 10.1007/978-3-642-33275-3_109.

P. Negri, is BSc. graduated in Electronic Engineering from the Universidad Nacional de La Plata, Argentina in 1998. He received his MSc. in Robotics from Univ. Pierre et Marie Curie-Paris VI in 2003, and he obtained his PhD from Univ. Paris VI in computer vision in 2008. Since 2010 he is researcher of CONICET and joined the Institute of Technologies at Univ. Argentina de la Empresa. He received the Best Paper Award in CIARP 2012. His research interests are machine learning and pattern recognition applied to computer for intelligent traffic or biomedical applications.
ORCID: 0000-0003-0250-5208

D. Garayalde, is BSc. graduated in Electronic Engineering from the Instituto Tecnológico de Buenos Aires (ITBA), Argentina in 2011. He collaborated as software developer for control systems through machine vision and tracking algorithms. Nowadays he also runs Hornero Makers, a company providing 3D Printing professional solutions for the local industry and professional institutions. His main interests are in the application of new technologies to healthcare industry and prosthetic devices.
ORCID: 0000-0001-9213-9461