Neurocomputing 183 (2016) 79–89

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Experimenting WNN support in object tracking systems

Massimo De Gregorio [a], Maurizio Giordano [b], Silvia Rossi [c], Mariacarla Staffa [c,*]

[a] Istituto di Scienze Applicate e Sistemi Intelligenti "Eduardo Caianiello" – CNR, Via Campi Flegrei 34, Pozzuoli, Italy
[b] Istituto di Calcolo e Reti ad Alte Prestazioni – CNR, Via Pietro Castellino 111, Naples, Italy
[c] Università degli Studi di Napoli "Federico II", Via Claudio 21 80125, Naples, Italy

ARTICLE INFO

Article history:
Received 5 September 2014
Received in revised form
22 September 2015
Accepted 23 September 2015
Available online 12 December 2015

Keywords:
Weightless neural network
Object tracking

ABSTRACT

Object tracking is a challenging problem in many computer vision applications, which go from robotics to surveillance systems. When applied to real world conditions, tracking methods found in the literature compete in solving some inherent difficulties of object segmentation and movement prediction, such as camouflage, occlusions, dynamic background, brightness, color and shape changes. To address some of these issues, we propose a general framework for object tracking by exploiting well-known segmentation techniques and a weightless neural network based prediction algorithm. The considered neural computing model is DRASiW, that we, here, extended with reinforcing and forgetting mechanisms. This model has the property of being noise tolerant and capable of learning step-by-step the new appearance of the moving object, by updating the learned object shape through the evolution of its internal representation (called "mental" image). The proposed object tracking framework has been evaluated on different benchmark videos. Experimental results show the viability and the benefits of the proposed DRASiW-based object tracking framework in the chosen case studies in comparison with three state-of-the-art methods. In addition, results provide useful insights about which combination of DRASiW-based operational modes and segmentation techniques improves the performance in the considered cases.

1. Introduction

In the last years, several object tracking techniques have been proposed due to the increasing importance of this topic in different application domains, which go from surveillance systems to robotic applications. The object tracking problem consists in reconstructing the trajectory of an object along a sequence of images. It is considered a basic problem in many computer vision applications and it is inherently difficult especially when applied to real world conditions, whereas unstructured object shapes are considered, real time responses are required, and problems of brightness and not-stationary background may alter the performance of the processing system.

In many applications, Kalman filters and hence Gaussian distributions are used to track a moving object [1] by following its motion from frame to frame. More recently, particle filters have been introduced to estimate non-Gaussian, non-linear dynamic processes [2,3]. On the contrary, tracking-by-detection approaches estimate the object state considering measurements referred only to the current image. In particular, adaptive discriminating trackers, such as [4,5] that build a classifier during tracking, have been demonstrated to handle significant appearance changes, short-term occlusions, and cluttered background [6]. Objects are commonly detected by using point detector mechanisms (e.g., as SIFT [7]), background subtraction (as in [8,9]), segmentation (e.g., as graph-cut in [10] or active contours in [11]), or learning mechanisms (e.g., as Adaboost in [12] or SVM in [13]).

In general, Neural Networks (NNs) are exploited in these contexts since they can express highly non-linear decision surfaces, and they can consequently be used to appropriately classify objects presenting a high degree of shape variation. For example, approaches using NNs for object tracking exist, such as [14–16], that use NNs as global filters to identify object features, and thus to localize salient objects. These approaches mainly concern Convolutional Neural Networks (CNNs) [17], while in [18] the authors proposed a modified segmentation method inspired by [19], which considers artificial creatures, each of which can look at a particular part of the image through its own retina, and an ANN is deployed to decide the next move each creature should perform based on the current input.

Differently from the mentioned approaches, in this paper, we propose a weightless neural network (WNN) system, named DRASiW, as a feature detector for tracking any kind of object. Contrary to tracking methods that employ static appearance models of the target object (by manually tuning or offline

* Corresponding author.
E-mail addresses: massimo.degregorio@cnr.it (M.D. Gregorio), maurizio.giordano@cnr.it (M. Giordano), silvia.rossi@unina.it (S. Rossi), mariacarla.staffa@unina.it (M. Staffa).

http://dx.doi.org/10.1016/j.neucom.2015.09.117
0925-2312/

training), we propose a method based on a dynamic appearance model to cope with difficulties whereas the tracked object exhibits significant appearance changes. Hence, we propose a tracking system that does not need any a priori knowledge or model of the object to track. It can be classified as a tracking-by-detection approach. Systems that use WNNs have already been developed in [20–22] with very good results. However, these works address the tracking problem in specific domains.

In our work, the WNN-based tracker is the main component of a general framework in which different pre-filtering segmentation techniques can be adopted. Our aim is to introduce a general framework, where different combinations of filters and DRASiW operational modes can be deployed in different scenarios. In general, a system, developed and designed to be a good tracker, has to mainly face different problems, such as vibrating or moving cameras, camouflage or occlusions, dynamic background, brightness, color, and shape changes. To deal with such problems, here, we propose different configurations and solutions for guiding the employment of a DRASiW-based tracker depending on the application domain. We will go deep into the testing and evaluation of the proposed approach, by passing through a general analysis of the DRASiW systems applied to some of the main challenging problems within this field, and by comparing its performance with those provided by state-of-the-art techniques.

The paper is organized as follows. After a brief overview on WNN systems for tracking in Section 2, in Section 3 we present the general framework of our approach. The results of the DRASiW-based trackers applied to different tracking challenges are reported in Section 4 in comparison with other methods. Finally, in Section 5, we present our conclusions and give some perspectives.

2. WNN approaches to tracking

The first WNN-based approach for object tracking was proposed in [20], in which the authors developed a system capable of following the movement (rolling and pitching) of a ship. The system is based on reference points (regions) belonging to the ship structure. Since those points are always surrounded by a static background, a simple binarization of the image is enough to guarantee a good performance of the tracking system.

In [23], another tracker based on WNN was designed for ships or enemy vessels tracking for warship tasks.[1] Here, the binarization of the image is carried out by sliding the *Prewitt's* edge detector to find the four target points used for detecting the ship in the sea. Unfortunately, in this paper nothing is said about the background (shimmering water, rough sea, etc.), and the target to be followed is always presented as an almost uniformly colored object in a quiet sea. A parallel implementation of this tracker has been proposed in [24].

The first two trackers, facing the problem of tracking any object that can dynamically change its shape during tracking on dynamic backgrounds, are those proposed in [25] and [21]. In these two similar approaches, the trackers are capable of following both deformable objects and human beings with very good performance. The first one ([25]) has been designed for multiple objects tracking, but framed by a fixed camera, while the second one ([21]) deals even with subjects framed by moving cameras (e.g., a robot following a human being).

The problem of tracking moving objects poses some main requirements a general purpose tracker should have, such as to be very quick and robust in recognizing the subject to follow, and to be equipped with suitable context-dependent filters able to isolate the moving target from the background. In previous work [21], we showed how WNN-based trackers are able to provide a quick recognizing of the moving subject. In this work, we introduce a new reinforcing and forgetting mechanisms in DRASiW, and we assess not only the robustness of this approach, but we also test different filters with the aim of proposing a good domain-dependent configuration for the particular tracking task one is facing with.

[1] We cited this work only as an example of application in this area. We refuse the use of these methodologies and any other methodology for military purposes.

3. DRASiW-based approach for tracking

In this work, a new approach based on a particular WNN for object tracking that does not require a priori model of the object to follow is proposed. The WNN has the property of being noise tolerant and capable of learning step-by-step the new appearance of a moving object on a dynamic background. In addition, we propose a strategy to constantly update the WNN memory allowing to trade-off between the changing of the shape of the target and the possible recovering after short-term occlusions. Furthermore, such a network can be used to deploy virtual sensors that, with a limited use of computational resources, can be used on-board for object tracking and dynamical selection of the desired targets to track.

3.1. The general framework

The proposed framework is depicted in Fig. 1. The object to be tracked is selected by the user by drawing a box (blue box in Fig. 1) containing the whole (or part of) target. Initial target selection, which is not a prior focus in this work, could be carried out (semi-) automatically by exploiting well-known techniques for motion-detection. The box identifies the region of interest (i.e., the central retina) of the frame on which the system works (see Section 3.2). A user can stop the system and select a new target to track at any time of the process.

To transform the input video frame in a suitable format for DRASiW systems the user chooses a specific segmentation filter to be applied to the region containing the target. Examples of such filters are provided in Section 3.4. The filter produces a binary image which is then passed to the DRASiW-system. Such a binary input represents the target (with its initial shape and position) to be followed, and is used to train the DRASiW discriminators, as will be explained in the following sections. The discriminators monitor different parts of the original image to track the movements of the selected target in every direction (see Section 3.3). Frame by frame, the DRASiW system tries to localize the object through the evaluation of the discriminators responses. The discriminator with the highest response identifies the most probable new position of the object in the scene.

Once the system localizes the object in the new (i,j) position (that is, discriminator $d_{i,j}$ has provided the highest response), the memories of the network discriminators are trained on the new appearance of the target in different ways, depending on the two possible execution modes of the DRASiW-based tracking systems (see Section 3.3).

At each new frame, independently from the filter and execution mode used, we compute the coordinates (x,y) of the mass center of the binary image contained in the frame portion controlled by the discriminator that provided the highest response. Hence, the coordinates $(i+x,j+y)$ represent the new (X,Y) position of the target object.

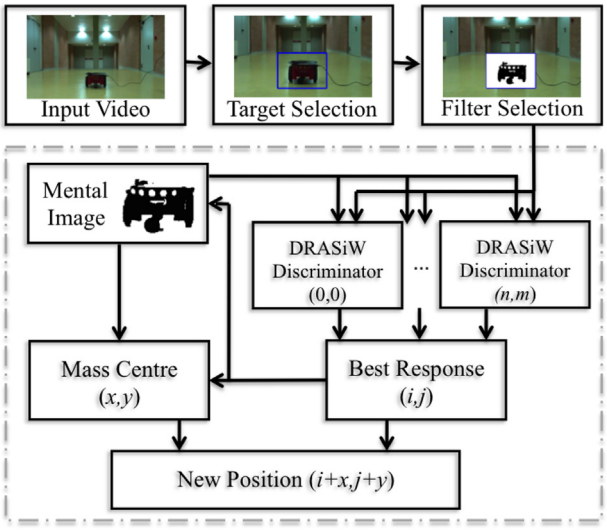


Fig. 1. A general framework for tracking via DRASiW.

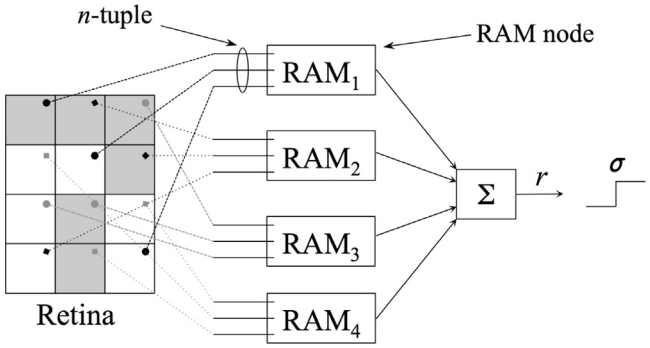


Fig. 2. A 3-bit WiSARD discriminator.

3.2. The DRASiW weightless neural network model

WNNs are neural computing models based on artificial neurons with binary inputs and no weight among them. Neurons are implemented by Random Access Memories (RAMs) functioning as look-up tables [26]. Learning in WNNs consists in changing the RAM neurons content, instead of adjusting synaptic weights. A RAM-based neuron is capable of recognizing n-bit inputs (n-tuples of bits) coming from a specific portion of a binary image (called the *retina*). WNNs are plausible models of biological neural networks because of the similarity between the way WNNs decode RAM addresses and the way excitatory and inhibitory signals are integrated in biological neural networks. An exhaustive survey on WNNs can be found in [27]. In this work, we consider the WiSARD model, that is a particular type of WNN developed in the late 1970s, when the availability of integrated circuit memories, allowed three researchers, named Wilkes, Stonham and Aleksander (from which the acronym WiSARD – Wilkes, Stonham and Aleksander Recognition Device), to patent and to commercially produce the first artificial WNN machines [28]. The WiSARDs can be directly developed on reprogrammable hardware.

In WiSARD systems, given a $m \times n$ retina, a *discriminator* is a set of m RAM-based neurons biunivocally associated to a set of uncorrelated n-tuples of retina pixels. These pixels are selected in a pseudo-random way (see, for example, Fig. 2). Each possible configuration of a n-tuple of bits in the retina identifies a specific address of a RAM. In the example of Fig. 2, by giving the binary image as input (i.e., the "7" represented in the retina) the RAM_1 will be accessed at the location 100.

In the training phase, a discriminator, is first initialized (all RAM memory cells are set to 0) and then a set of $m \times n$ binary patterns (the *training set*) is used to train the discriminators: n-tuples of bits are extracted from each pattern of the training set, and used to access RAM memory cells and to store a 1. Once the training is completed, each RAM memory cell contains 0 or 1 (namely, 1 if the corresponding memory cell has been addressed during the training phase at least one time). The information stored in RAM neurons during the training phase is used to classify new patterns other than those of the training set. When one of these is given as input to a discriminator, n-tuples of bits are extracted from it and then used to access RAM memory cells in read mode: accessed contents are summed by the summing device Σ. The number r computed by Σ, which is called the *discriminator response*, is equal to the number of RAMs that output 1. Intuitively, r reaches the maximum value m if the input pattern belongs to the training set; r is equal to zero if no n-tuple of bits of the input pattern occurs in the training set (no RAM outputs 1). Intermediate values of r express a kind of "similarity measure" of the input pattern with respect to the patterns in the training set. The summing device enables this network of RAM neurons to exhibit – just like other ANN models based on synaptic weights – generalization and noise tolerance [29].

DRASiW [30] is an extension to the WiSARD model provided with the ability of producing pattern examples, or prototypes, derived from learned categories. RAM-based discriminators are modified in what their memory cells may store and, correspondingly, in how the training algorithm works. In particular, memory cells can store q-bit words, representing non-negative integer values. The training algorithm in DRASiW increases by 1 the values of accessed memory cells (instead of storing always a 1 at each access). At the end of the training phase, the values of memory cells will range from 0 and T (where T is the number of patterns in the training set).

Contents of memory cells in DRASiW discriminators can hence be interpreted as sub-pattern frequencies in the training set. One should notice that if the Σ device counts the number of addressed memory cells whose content is not 0, DRASiW performs the same classification process of WiSARD. Thus, while the two models are equivalent in terms of classification capabilities, DRASiW allows an additional capability: to generate synthetic prototypes of training samples. In other words, once the training phase is complete, RAM contents can be (reversely) used to compute a greyscale image representation of each learned category of patterns, namely "Mental Images" (MIs). Each grey pixel in the MI represents how many times that pixel was set to 1 across all patterns of the training set.

3.3. DRASiW for object tracking

The DRASiW that we propose as main component of the tracking system is formed by a set of RAM-based discriminators whose training policy and role is defined as follows. At the beginning of the video sequence, once the box including the object to be followed has been selected, one discriminator, referred to as the *central discriminator*, is trained on the region of the first frame delimited by the box. The frame region defines the retina size which is fixed for all discriminators and throughout the video duration. All other discriminators are trained on the same frame region like for the central discriminator. In the prediction phase, all discriminators, except for the central one, try to match the learned patterns to regions (with the same retina size) of the current frame placed around the central one.

In this work, we consider a configuration where the discriminators work on regions located within a rectangular neighborhood of the central region, as depicted in Fig. 3. Doing so, each

discriminator is identified by its relative coordinates. The set of discriminators, organized in this way, forms what we call *prediction window*, which identifies the image area in which it is possible to predict the target object movements. The maximum displacement of all the retinas from the central one in all the directions is called *prediction window precision* P_w. In particular, since we consider a prediction window precision of 10 pixels, we will use $21 \times 21 = 441$ discriminators (including the central one). Let $(0,0)$ be the retina coordinates of the central discriminator ($d_{0,0}$), N_p the number of pixel forming the retina. The whole set of discriminators is labeled as $d_{i,j}$ (with $i,j \in [-P_w, P_w]$), and each one is formed by N_p/n RAM neurons. The total number of RAM neurons (RN) forming the DRASiW system is given by:

$$RN = (2P_w + 1)^2 \cdot \frac{N_p}{n}.$$

The generic discriminator $d_{i,j}$ is responsible for detecting the object in case its new position is identified by (i,j) in the prediction window. When a new frame is given as input, each discriminator gives a response to that input. The various responses are evaluated by an algorithm which compares them and computes the relative confidence c of the highest response (e.g., the difference between the highest response and the second highest response, divided by the highest response). Depending on the way by which RAM contents of discriminators is updated during the tracking, we distinguish two execution modes of the DRASiW-based tracking systems, which are described as follows.

DRASiW Tracker: The first execution mode we consider in this work has been already proposed in [21]. At each frame, the discriminators of the prediction window are initialized and then trained on a single binary pattern resulting from filtering a region in the current frame image.

On the next frame, on the basis of the one-step training of the previous frame, each discriminator in the prediction window provides a matching response of the moving object. Suppose the discriminator with the highest response is $d_{i,j}$. Then, the system computes the coordinates (x,y) of the mass center in the $d_{i,j}$ retina. Hence, the coordinates $(i+x, j+y)$ will represent the new (X,Y) position of the target object. On the next frame, the central point of the prediction window becomes (i,j) and the retina of the central discriminator is centered on it. Then, all the discriminators are re-initialized (i.e., all memory cells set to 0) and trained on the binary image computed by the filter applied to the image part contained in the new retina of the central discriminator.

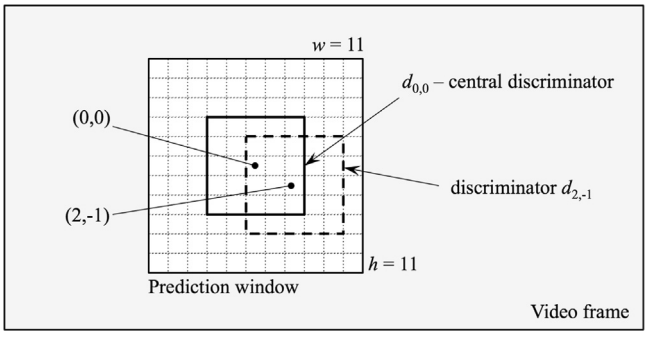


Fig. 3. Prediction window (5–pixel precision) and discriminator retinas.

DRASiW-memory Tracker: In this tracking mode, discriminators are never re-initialized during the video. On each frame the new binary input provided by the filter is used first to detect the moving object displacement (i,j) by identifying the discriminator $d_{i,j}$ giving the highest matching response, then to train all discriminators in the prediction window by means of the *Reinforcing & Forgetting* (*RF*) mechanism. The RF mechanism fully exploits the DRASiW capability to store and update inside RAMs frequency counters of sub-patterns occurrences. In fact, RAM contents addressed by sub-patterns (n-tuples of bits) extracted from the input frame are increased by one (*Reinforcing*), while the other memory cells are decreased by one (*Forgetting*). In other words, the forgetting action consists in a selective unlearning of infrequent sub-patterns. The *Reinforcing & Forgetting* (*RF*) strategy allows to keep in discriminator RAMs an up-to-date "mental image" (MI) of the object shape that changes during the tracking. In this way, the stored and updated MI, during time, represents a sort of target object shape history. This history represents a fundamental facility in the case we need to extract from the tracking a cinematic/dynamic model of the target. In addition, by managing a MI as the shape history of the target object, the system is capable of recovering from occlusions in the short term. This capability strongly depends on the system *Forgetting* time: the more reinforced is the content of the mental image, the longer the occlusion may last with no target lost. Some mental images generated by DRASIW-memory tracker are reported in Fig. 4. The darker is the pixel gray level the longer the system recognize it.

Finally, like in the former execution mode, the predicted position (i,j) becomes the center of retina for the central discriminator, while all discriminators in the neighborhood are shifted accordingly.

Note that, since both execution modes (DRASiW and DRASiW-memory) involve a global rewriting of all RAM cells on each frame (either for re-initialization or for reinforcing/forgetting), the computational complexity of the two modes is similar.

3.4. Image filters

DRASiW systems can be fed only by binary inputs (a set of n-tuples of bits). As a consequence, regardless of whether the input

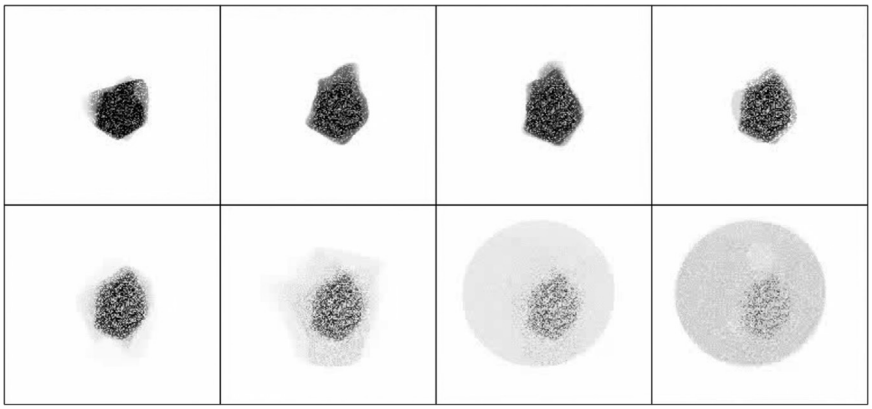


Fig. 4. Some consecutive mental images generated by DRASiW-memory on *deformable target* video.

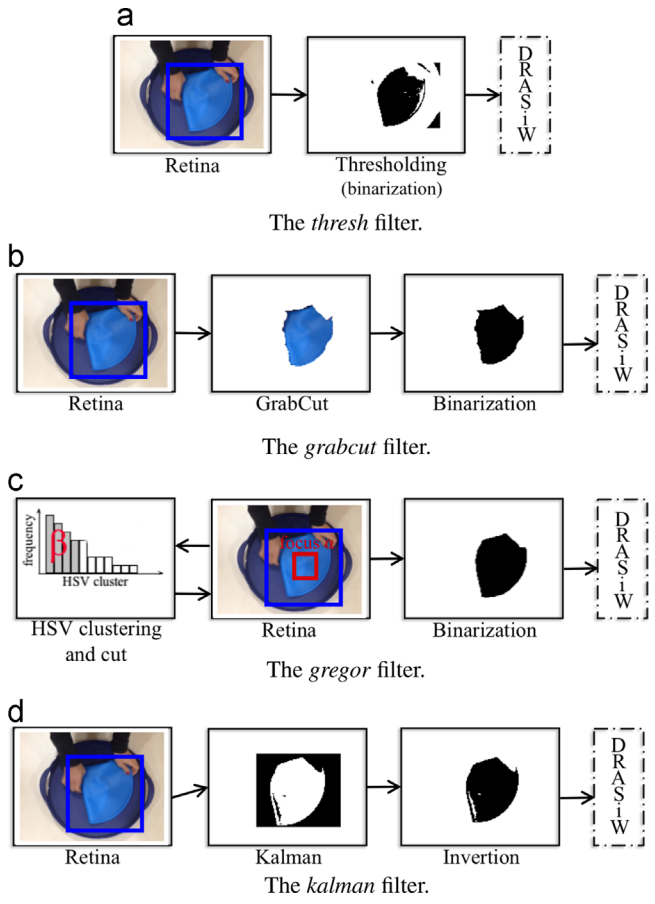


Fig. 5. Tracking filtering techniques. (a) The *thresh* filter. (b) The *grabcut* filter. (c) The *gregor* filter. (d) The *Kalman* filter.

data of a DRASiW are gene expression strings, acoustic signals, histograms, colored images, or any other kind of information, they must be converted into binary data trying to preserve the original information contents and their relation. For this purpose we chose to equip the DRASiW-based tracker with a predefined set of filters for converting video frames into binary images.

It should be noted that, except for the *gregor* filter, all the supported filters are segmentation methods found in the literature, whose implementation is available in the OpenCV Library. Indeed, the *gregor* filter implements a novel segmentation method to separate foreground objects of interest, and its algorithm is described in this work.

Threshold filters: The simplest filter used to segment the foreground of images is the *Threshold Filter* (from now on we call it *thresh* filter). Such a filter, once the threshold value is set, first converts the image area enclosed by the retina into grayscale, then it converts each gray pixel to one (black) if the gray level is over the fixed threshold, to zero (white) otherwise, as represented in Fig. 5a. The threshold can be set at video startup and it is fixed during the process.

It is also possible to use a dynamic threshold computed at runtime according to the Otsu algorithm clustering-based image threshold method [31]. Nevertheless, the performance of dynamic thresholding is not so good in video scenes with very dynamic background since the computed threshold may vary in a sensitive way in these situations.

Grabcut filter: The *grabcut* filter applies the well-known GrabCut method [32] to separate the foreground and background regions of an image enclosed by a bounding box (in our case, the bounding box is the discriminator retina). The main filter steps are depicted in Fig. 5b.

The method evaluates the color distribution of the background and of the target object in the retina using a Gaussian mixture model. When applied to the image in the retina, the GrabCut algorithm returns a new image with all detected background pixels in white, while foreground pixels appear with the original colors. The output image of the GrabCut method is, then, binarized by converting all foreground pixels into black pixels. In this format, the segmented binary images become the inputs for the DRASiW module of the tracking system.

Gregor filter: The *gregor* filter is a novel segmentation technique we developed for the purpose. This filter identifies the more frequent colors in a given region (*focus area*) of the retina. More precisely, the focus area is a box (the red rectangle in Fig. 5c) centered with respect to the retina and whose size is a α percentage of retina size. The filter uses the focus area to compute the histogram representing the pixel color (HSV) frequencies in the focus. The histogram is then ordered and cut to leave only the more frequent pixel colors whose area is a β percentage of the total area of the original histogram. The selected colors are used to identify and to label (black) pixels in the retina as belonging to the tracked object, while all the other pixels are labeled as background (white). This filtering process is repeated at each frame to adapt to the dynamism of the environment conditions.

Kalman filter: The *kalman* filter implements the well-known Kalman method [33] to label as foreground the pixels in the retina whose RGB channels are within a certain fixed range from a reference pixel RGB color. Although the Kalman method has its own tracking mechanism based on a state model propagation algorithm to predict the next target position, in this work we exploited only the image segmentation feature of the Kalman method. In other words, once our tracker has predicted the next position in the frame, it is used as reference pixel RGB color for the next segmentation step. The main filter steps are depicted in Fig. 5d. Since the OpenCV implementation of the Kalman method returns a binary image with background pixels as black, an image inversion was required before passing it to the DRASiW tracking systems.

4. DRASiW–tracker applied to different tracking challenges

In this section, we evaluate the DRASiW tracking system's performance on some challenging benchmark videos. Namely, the chosen videos deal respectively with some of the main problems that can be encountered during object tracking, such as: partial and short-term occlusions (*occlusion*), camouflage effects (*camouflage*), significant changes in the appearance of a deformable object (*deformable target*), color and pose changes of a rigid object with a cluttered background (*rubik*).[2]

We provide a comparative analysis of the proposed DRASiW tracking methods (the DRASiW tracker – D, and the DRASiW-Memory tracker – DM) with respect to some state-of-the-art tracking methods we used as benchmarks. In particular, for the comparison, we consider the Multiple Instance Learning (MIL) method[3] [4], the FragTrack (FRAG) algorithm[4] [5], and the Tracking–Learning–Detection (TLD) method[5] [6].

[2] All the videos are available at http://www.smile.unina.it/wnn, except for the *camouflage* video that is available at http://www.changedetection.net

[3] Code available at http://vision.ucsd.edu/project/tracking-online-multiple-instance-learning

[4] Code available at http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm

[5] Code available at http://personal.ee.surrey.ac.uk/Personal/Z.Kalal/tld.html

Our aim is to observe whether the DRASiW-based tracking methods provide comparable performance with respect to the state-of-the-art methods, and to different configurations (i.e., the choice of the segmentation filter and the use of the memory mechanism in DRASiW). Thus, the discussion on the experiments will also provide some insights about the best DRASiW configuration for each challenge.

4.1. Experimental results

For each video, we evaluate common performance metrics for quantitative comparison. In particular, we consider the *Center Localization Error* (CLE), also referred to as central-pixel error. It represents the positioning error of the tracker, whose value is computed as the Euclidean distance between the object centroid coordinates, as predicted by the tracker, and the real object centroid coordinates (Ground Truth – GT). Then, we consider the *Precision* (*P*) that measures how long a tracker is able to correctly track the target object. Note that, in all videos the target never disappears from the camera view. This implies that we cannot measures both True and False Negatives. Hence, precision is evaluated as the number of True Positives (TPs) with respect to the total number of frames. In detail, we first evaluate the number of TP by considering the Jaccard coefficient. The Jaccard coefficient evaluates the intersection between the bounding box centered on the GT coordinates and the bounding box centered on the new position predicted by the tracker. Such measure defines a TP when the two boxes overlap for more than 50% [6].

For each of the selected video sequences, we show the plots with graphs of CLE performance, and the *P* histogram, where FRAG, TLD and MIL precision values are reported as horizontal lines. Finally, in Table 1, the average values of the CLE (ACLE), and of the *P* values are reported, all together for their comparison. "ACLE total" and "ACLE on TP" are respectively the average CLE evaluated on the entire video sequence, and the average CLE evaluated only on frames in which the method achieved a TP. In particular, we highlight in bold the best performance for each video, and in italic the second best performance. In some cases (*occlusion* and *deformable target* video sequences) the second best method performance is very close to the first one.

Occlusion: In this video sequence, filmed by a fixed camera, a partial and short-term occlusion is evaluated in a highly controlled and static environment (snapshots of the video sequence are reported in Fig. 6).

In Fig. 7 (right), we show the precision achieved by the proposed tracking systems when using different filters. We can note that the DRASiW-memory trackers achieve similar results, regardless of the choice of filter (except when combined with the *kalman* filter). On the contrary, the precision of the DRASiW tracker depends on the filter choice. As we expected, in cases of short-term occlusions, the reinforcing & forgetting mechanism allows the system to keep track of the partially occluded object. Indeed, although the occluding object starts being recorded in the mental image, the memory of the target object is still strong due to a long previous reinforcement, thus driving the system to follow the latter instead of the former. Overall, DM and FRAG show the best performance in terms of precision. Note that *D* precision values are comparable with TDL and MIL. Fig. 7 (bottom-left) shows clearly that the DM trackers are able to keep the target correctly while being occluded.

As shown in Table 1, the minimum average CLE is 3.1 (as for the FRAG) and it was achieved by the DM tracker combined with the *thresh* filter. Furthermore, DM achieves a similar performance also when combined with *grabcut* (3.5) and *gregor* (5.5) filters. A similar ACLE was achieved also by the *D* tracker, although only in combination with the *grabcut* filter (3.7). Eventually, the best tracking method to cope with this kind of occlusion is the DM tracker. In fact, it provides a high precision and the lowest ACLE values with respect to the other trackers (see Table 1).

Camouflage: In this video sequence a man carries a carton from one side of a room to the other, then he leaves the carton on a table, where a carpet with the same box color is in the background. After a while, the man comes back to pick the carton up and brings it away (snapshots of the video sequence are reported in Fig. 8).

With respect to the *occlusion* video, this video sequence shows a real environment, and hence more challenging for the trackers. The FRAG tracker gets the best performance on this video, by achieving 0.83 in precision and 7.9 in ACLE. The FRAG method is even able to recover the target object after a loss. This is proved by the small difference between the ACLE on the entire video (ACLE total) and the ACLE limited to successful cases (ACLE on TP), which are reported in Table 1. As regards the DRASiW methods, the choice of the *grabcut* filter can significantly improve the performance. With respect to the FRAG method, D and DM trackers have comparable precision, even better than TLD and MIL methods, when using the *grabcut* filter, while they have poor precision in combination with the other filters (see Fig. 9 (right)). Snapshots of the trackers behavior are shown in Fig. 8.

The trend of the CLE obtained by the trackers (in Fig. 9 (left)) highlights the challenging nature of this video sequence. Note that, the DRASiW-memory has a more stable trend of the CLE thanks to the memory reinforcement mechanism, which is favored by the fact that, in

Table 1
Precision (*P*), average center localization error on all frames (ACLE total), and average center localization error on true positives (ACL on TP).

	Occlusion (228 frames)			**Camouflage** (2014 frames)			**Deformable** (1390 frames)			**Rubik** (875 frames)		
Tracking Method	P	ACLE total	ACLE on TP	P	ACLE total	ACLE on TP	P	ACLE total	ACLE on TP	P	ACLE total	ACLE on TP
D–Grabcut	*1.00*	*3.7*	*3.7*	*0.76*	*21.3*	*9.8*	**1.00**	**7.8**	**7.8**	*0.65*	*17.7*	*8.7*
D–Gregor	0.87	19.4	10.4	0.04	62.6	6.7	0.46	37.7	17.5	0.46	40.5	7.8
D–Kalman	0.76	26.7	11.1	0.00	183.6	–	0.77	24.3	5.4	0.00	246.6	–
D–Thresh	0.86	12.6	6.1	0.23	49.7	5.4	*0.99*	*9.4*	*9.2*	0.12	196.0	10.7
DM–Grabcut	*1.00*	*3.5*	*3.5*	0.70	13.8	9.5	*0.99*	*8.3*	*8.1*	**0.80**	**11.7**	**9.6**
DM–Gregor	*1.00*	*5.5*	*5.5*	0.01	96.6	8.2	0.28	65.0	16.5	0.52	36.0	8.1
DM–Kalman	0.51	24.4	10.5	0.00	205.9	10.7	0.81	17.4	6.3	0.00	246.6	–
DM–Thresh	**1.00**	**3.1**	**3.1**	0.22	52.1	5.9	*0.99*	*10.8*	*10.6*	0.22	33.6	11.0
MIL	0.67	21.1	9.6	0.02	75.9	6.4	0.48	27.8	9.9	0.60	14.0	9.5
FRAG	**1.00**	**3.1**	**3.1**	**0.83**	**7.9**	**5.7**	0.50	24.3	10.8	0.21	133.8	8.9
TLD	0.89	12.4	7.9	0.24	140.9	9.1	0.97	12.8	11.5	0.45	49.2	12.3

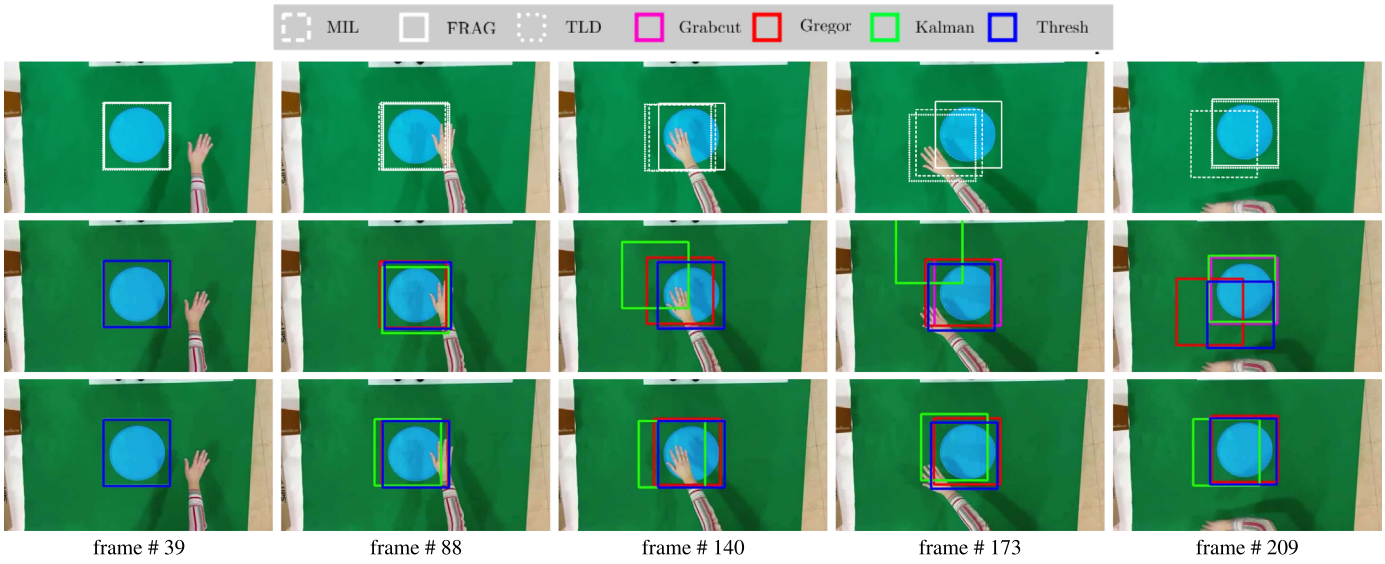


Fig. 6. Snapshots from the *occlusion* video: MIL, FRAG, and TLD (top) – DRASiW (middle) – DRASiW-memory (bottom).

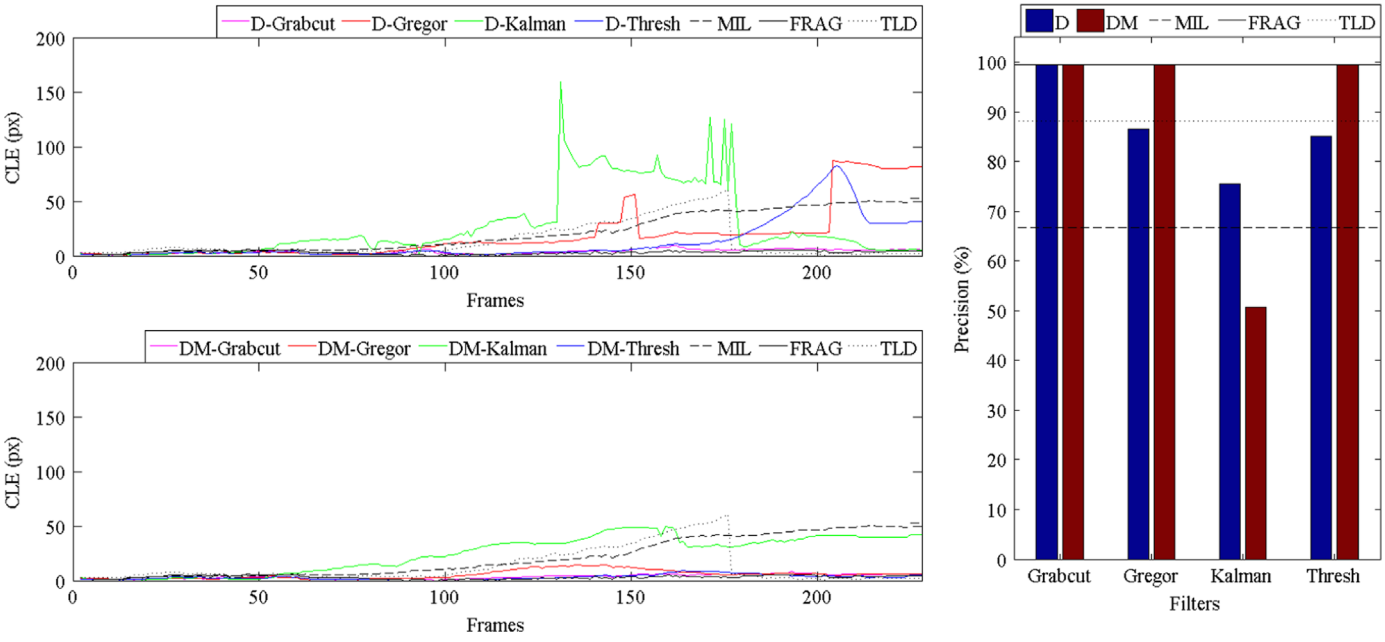


Fig. 7. Trends of Center Localisation Error – CLE (left), and tracker precision histogram (right) for the *occlusion* sequence.

most of the video sequence, the object stands still. On this video several trackers fail after a while. In conclusion, we can state that the *grabcut* filter definitely improves the performance of D and DM trackers.

Deformable Target: The video shows the manipulation of pizza dough in a controlled environment. Here, different actions performed by human hands are considered, such as moving around, manipulating, stretching and seasoning the dough (snapshots of the video sequence are reported in Fig. 10).

Even in this case (see Table 1), we observe that the best filter choice on the average is *grabcut*, and the best ACLE (7.8) is achieved when it is combined with the D tracker. Comparable precisions are achieved with D and DM trackers when combined with the *thresh* filter (see Fig. 11 (right)), but with a slightly greater ACLE (9.4 and 10.8 for *thresh* with respect to 7.8 and 8.3 for *grabcut*). D and DM methods have similar performances since the use of DRASiW's mental image as a long-term memory of the object to follow is not useful when tracking an object whose shape drastically changes (as in case of pizza dough manipulation), on the contrary, a step-by-step relearning mechanism of the object appearance is a desirable requirement for such settings (see Table 1). Moreover, the proposed methods outperform MIL and FRAG trackers in any filter configuration, while they behave similarly to the TLD tracker when using the *grabcut* and *thresh* filters. Only the TLD results approach those obtained by D and DM with *grabcut* and *tresh* filters.

Snapshots of trackers behavior are shown in Fig. 10. As we expected, in such controlled environment, the D and DM trackers, when using the *kalman* filter, are able to track the object, due to its well defined color, achieving the best ACLE (6.3). These trackers lose the target at the seasoning step due to the occurrence of long-lasting partial occlusions on the target (see Fig. 11 (left)). The worst ACLE is obtained by using the *gregor* filter that fails when the background significantly overlaps the focus area.

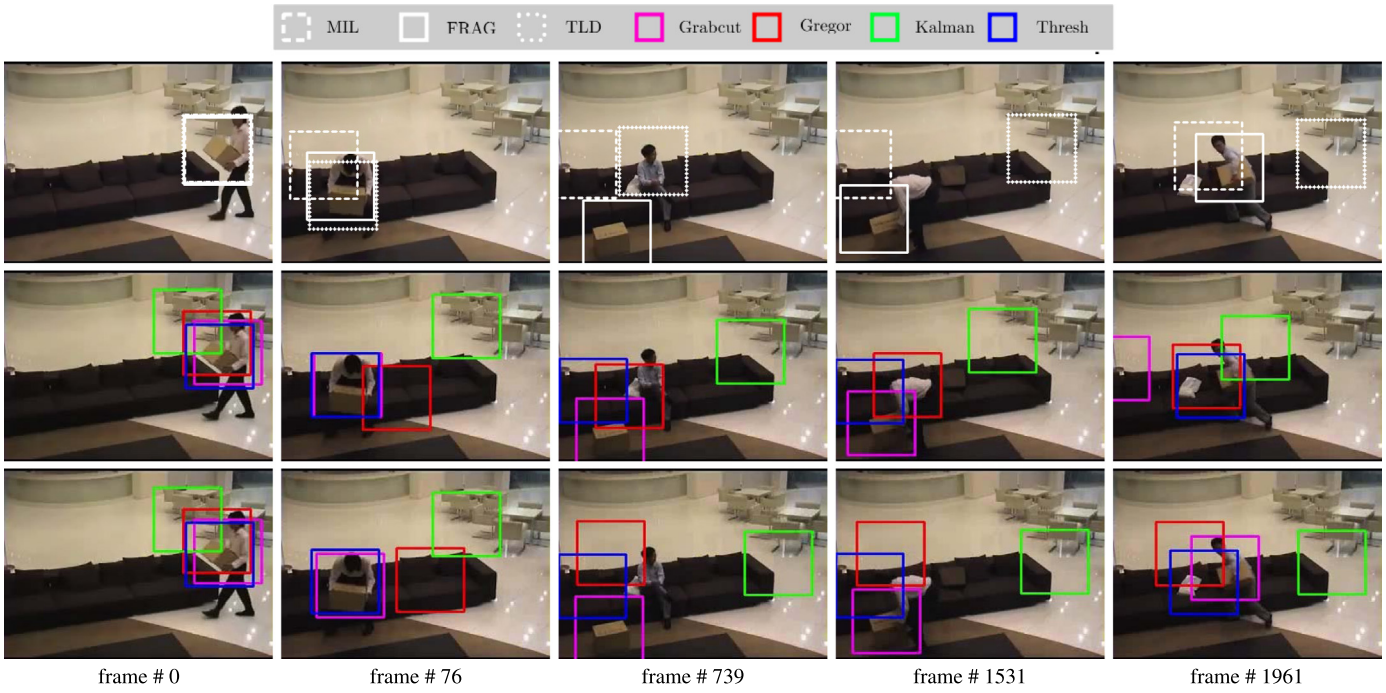


Fig. 8. Snapshots from the *camouflage* video: MIL, FRAG, and TLD (top) – DRASiW (middle) – DRASiW-memory (bottom).

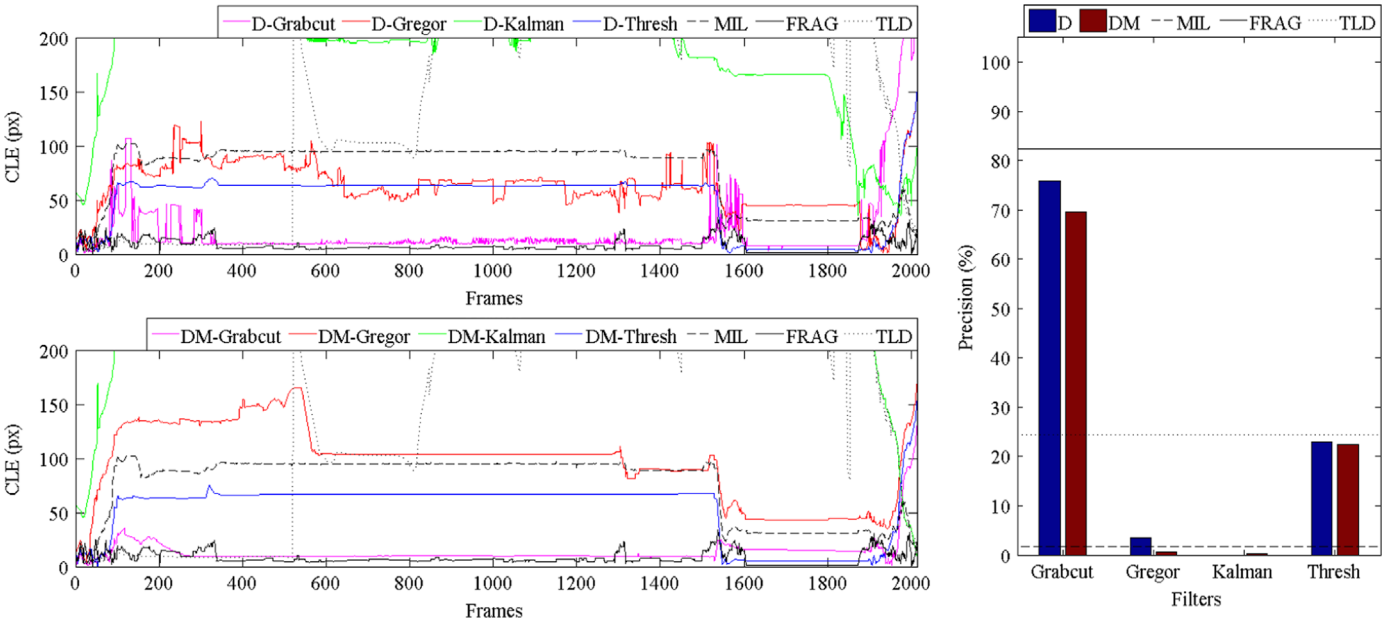


Fig. 9. Trends of Center Localisation Error – CLE (left), and tracker precision histogram (right) for the *camouflage* sequence.

In conclusion, the problem of drastic changes of object's appearance is still challenging for MIL and FRAG methods, while it is well addressed by TLD and DRASiW-based trackers.

Rubik: The last video shows a person who moves and rotates a rigid object, i.e., a *Rubik Cube*, over a cluttered background (see Fig. 12). In addition, while rotating, the target object changes color and pose. The cube is the object to be tracked.

Even though the best ACLE (7.8) is achieved by the D tracker when combined with the *gregor* filter, the DM tracker combined with the *grabcut* filter is the best choice in terms of both ACLE (11.7) and precision (0.80) (see Table 1). In Fig. 13 (right), it is quite evident that the *grabcut*-based DM tracker achieves the best performance with respect to the benchmark methods (MIL, FRAG and TLD) as well as to the other configurations of D and DM trackers. On the other side, the *gregor* filter choice loses to MIL, while it wins against TLD and FRAG. Fig. 13 (right) shows how DM trackers behave better than D trackers in all filter configurations, except for the *kalman* filter. This is mainly due to the fact that the constant change of color of the target object causes the *kalman* filter to fail after a few frames from the beginning of video.

Concluding, the DM tracker combined with the *grabcut* filter shows the best performance in terms of both ACLE and precision.

5. Conclusions

In this work, we proposed a particular model of a weightless neural network to address some of the challenges one can encounter in video tracking applications. In particular, a DRASiW-

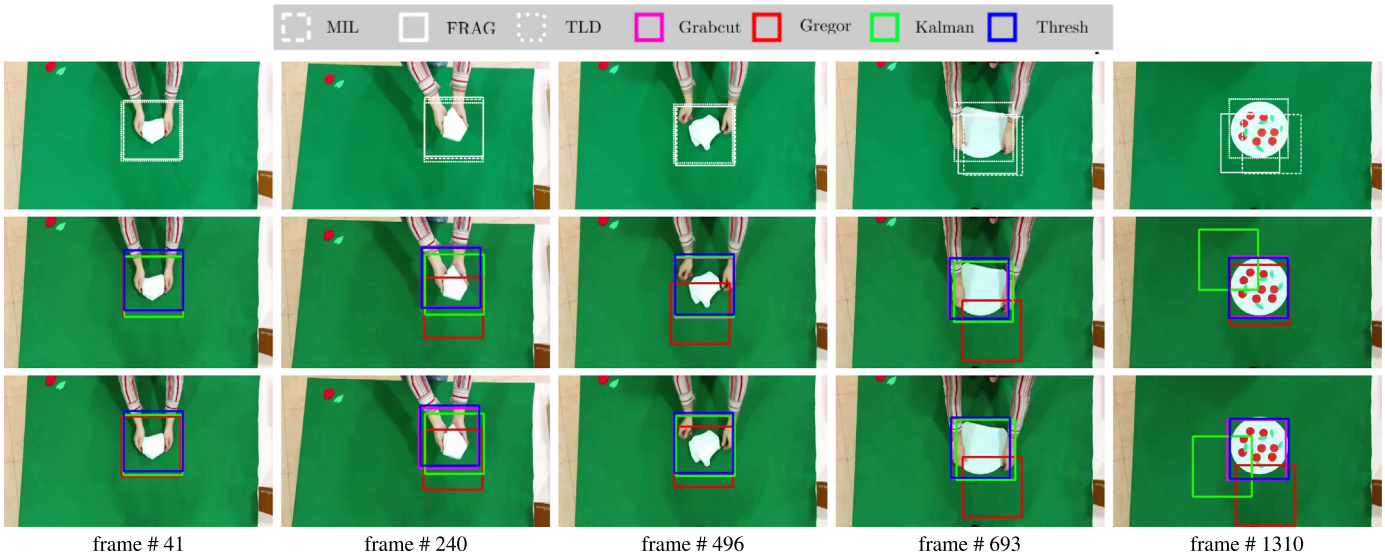


Fig. 10. Snapshots from the *deformable target* video: MIL, FRAG, and TLD (top) – DRASiW (middle) – DRASiW-memory (bottom).

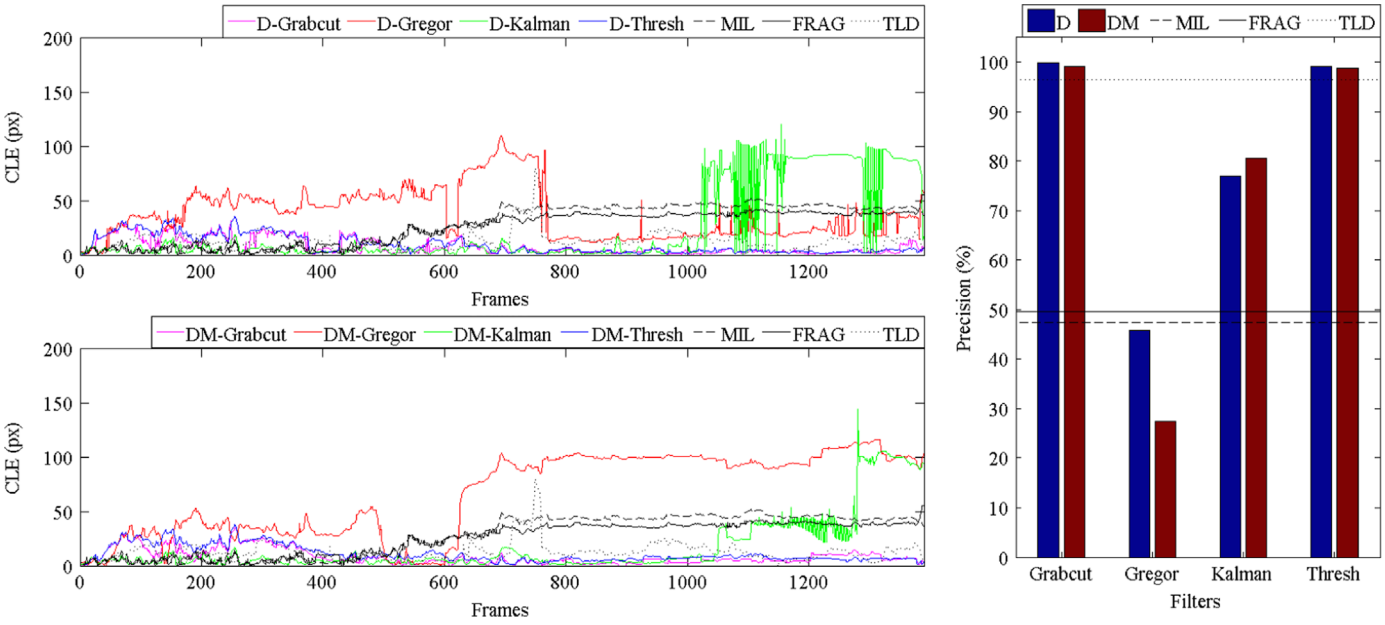


Fig. 11. Trends of Center Localisation Error – CLE (left), and tracker precision histogram (right) for the *deformable target* sequence.

based tracker system, whose behavior was extended by using a strategy of memory reinforcement and forgetfulness, is presented within a global framework, in which several segmentation techniques can be exploited in the image pre-filtering phase. Filters as the well-known *grabcut*, *kalman* and *thresh* have been considered, as well as a new filter named *gregor*. Our aim was to investigate the use of different settings of the proposed framework (i.e., combination of filters and DRASiW or DRASiW-memory methods) to cope with different challenges occurring during tracking, such as short-term occlusions, camouflage, appearance/pose/color changes of the target and cluttered backgrounds. We aimed at evaluating what is the best setting for addressing each considered tracking problem and in which context the use of the DRASiW-based trackers offers a significant contribution with respect to other tracking-by-detection methods (MIL, FRAG and TLD).

Concerning the general analysis, we attest that the DRASiW-based trackers well accomplish the feature, that all general purpose trackers should have, to be very robust in recognizing and tracking the subject to be followed. The robustness is proved by the high precision obtained in most of the experiments by the DRASiW-based trackers, performed either better or in a comparable way with the other trackers. This is due to the capability of DRASiW to be retrained online, frame-by-frame, and hence, to adapt to shape/color/pose target changes, and, on the other hand, to take advantage (in case of DRASiW-memory) of the mental image that allows to cope with short-term occlusions.

With respect to benchmark trackers, our proposed system is very versatile. Regarding the object segmentation problem to isolate the foreground, in different scenes, different pre-filtering solutions were tested and compared. As we expected, there is no single filter that provides optimal results in all cases. On average on all the video sequences, the *grabcut* filter provides the best performance, especially when combined with the DRASiW-memory system. This is quite evident from the performance analysis on the *camouflage* video sequence. In fact, such combination represents the only applicable approach in case of camouflage problem, that comes out to be the most difficult to solve by the benchmark trackers (except for FRAG). Furthermore, in the case of the *rubik* video the DRASiW-memory

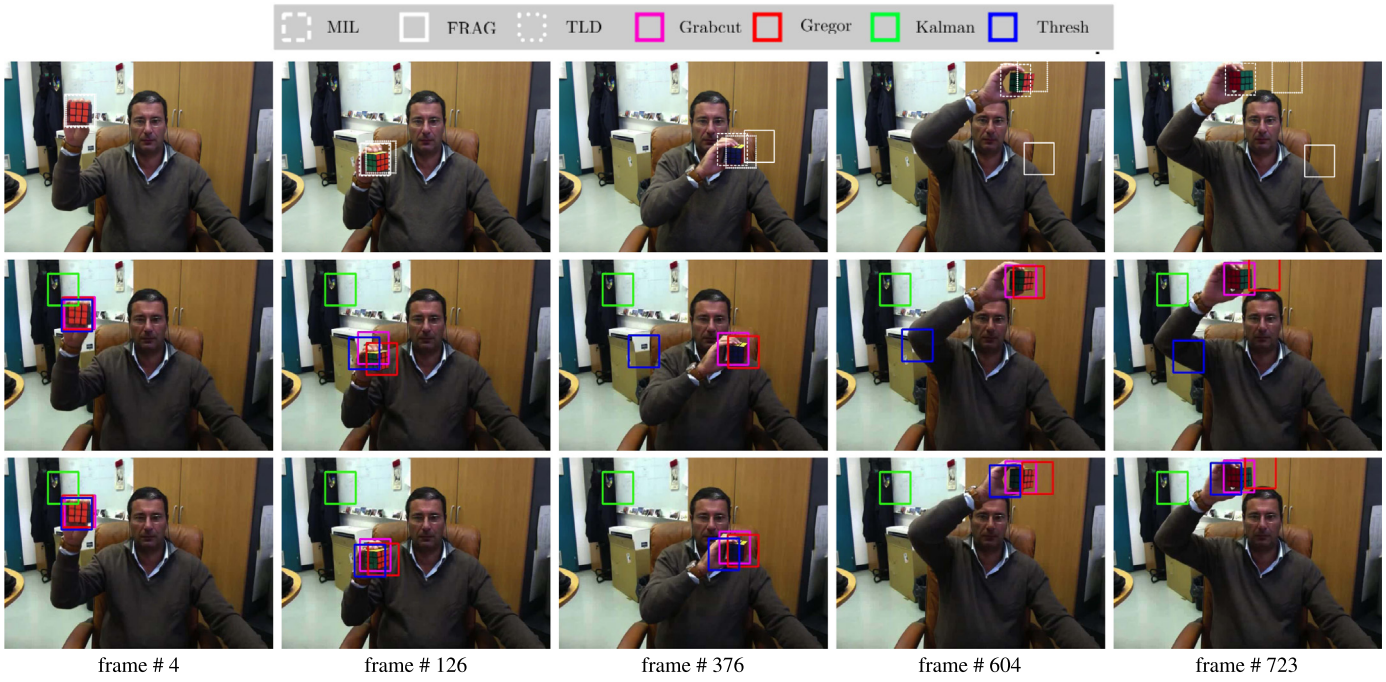


Fig. 12. Snapshots from the *rubik* video: MIL, FRAG, and TLD (top) – DRASiW (middle) – DRASiW-memory (bottom).

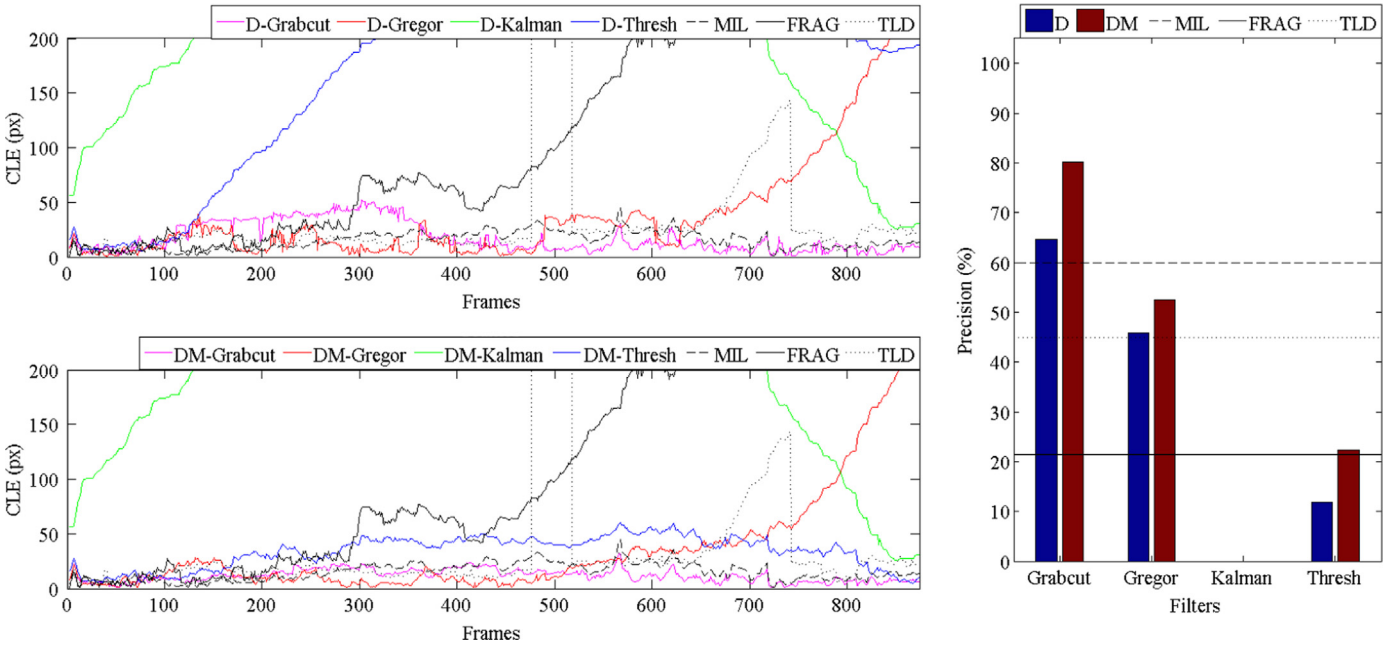


Fig. 13. Trends of Center Localisation Error – CLE (left), and tracker precision histogram (right) for the *rubik* sequence.

combined with the *grabut* filter achieves even better results with respect to the benchmark trackers. From the performance evaluation, it also arises that the use of DRASiW-memory often improves the performance with respect to the DRASiW approach (e.g., in the *occlusion* and the *rubik* video sequence), by permitting to longer follow the target. Hence, on average, the combination of DRASiW-memory and *grabcut* is the one performing better.

Concluding, we can state that there is not a general method to address all the proposed challenges. In particular, while the occlusion problem seems to be well addressed from all the trackers, the camouflage problem is still a challenge. In fact, in the latter case, only the TDL and the DRASiW-memory with the *grabcut* filter are applicable. On the contrary, our approach prominently shows its competitiveness when dealing with drastic changes of shape and pose of the target object. Finally, differently from the considered state-of-the-art methods, the proposed approach is characterized by the advantage to be easily and quickly adjustable (by changing the pre-filtering strategy and eventually by using the memory support). This versatility is not often present in other approaches.

Acknowledgements

This work has been partially funded by the European Commission's 7th Framework Programmes as part of the project

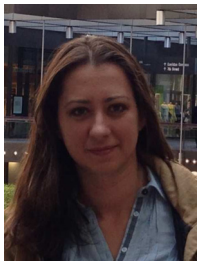
SAPHARI under grant 287513, and RODYMAN under ERC–grant agreement no. 320992.

References

[1] N. Gordon, A hybrid bootstrap filter for target tracking in clutter, IEEE Trans. Aerosp. Electron. Syst. 33 (1) (1997) 353–358.
[2] M. Pitt, N. Shephard, Filtering via simulation: auxiliary particle filters, J. Am. Stat. Assoc. 94 (1999) 446.
[3] M. Black, A. Jepson, A probabilistic framework for matching temporal trajectories: condensation-based recognition of gestures and expressions, in: LNCS 1406, 1998, pp. 909–924.
[4] B. Babenko, M. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1619–1632.
[5] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, 2006, pp. 798–805.
[6] Z. Kalal, K. Mikolajczyk, J. Matas, Tracking-learning-detection, IEEE Trans. Pattern Anal. Mach. Intell. 34 (7) (2012) 1409–1422.
[7] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[8] C. Stauffer, W.E.L. Grimson, Learning patterns of activity using real-time tracking, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 747–757.
[9] A. Monnet, A. Mittal, N. Paragios, V. Ramesh, Background modeling and subtraction of dynamic scenes, In: Proceedings of the Ninth IEEE International Conference on Computer Vision—vol. 2, ICCV'03, IEEE Computer Society, Washington, DC, USA, 2003, pp. 1305–1312.
[10] N. Xu, N. Ahuja, Object contour tracking using graph cuts based active contours, in: 2002 International Conference on Image Processing, vol. 3, 2002, pp. III–277–III–280.
[11] N. Paragios, R. Deriche, Geodesic active regions and level set methods for supervised texture segmentation, Int. J. Comput. Vis. 46 (3) (2002) 223–247.
[12] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1) (1997) 119–139.
[13] C. Papageorgiou, M. Oren, T. Poggio, A general framework for object detection, in: Sixth International Conference on Computer Vision, 1998, 1998, pp. 555–562.
[14] G. Valli, R. Poli, S. Cagnoni, G. Coppini, P. G. Valli, Neural networks and prior knowledge help the segmentation of medical images, 1998.
[15] E. Cuevas, D. Zaldivar, R. Rojas, Lvq color segmentation applied to face localization, in: 1st International Conference on Electrical and Electronics Engineering, 2004, ICEEE, 2004, pp. 142–146.
[16] A.D. Kulkarni, Artificial Neural Networks for Image Understanding, 1st ed., John Wiley & Sons, Inc., New York, NY, USA, 1993.
[17] J. Fan, W. Xu, Y. Wu, Y. Gong, Human tracking using convolutional neural networks, IEEE Trans. Neural Netw. 21 (10) (2010) 1610–1623.
[18] L. Mussi, R. Poli, S. Cagnoni, Object tracking and segmentation with a population of artificial neural networks, in: Workshop Italiano di Vita Artificiale e Computazione Evolutiva (WIVACE'07), 2007.
[19] R. Poli, G. Valli, Neural inhabitants of mr and echo images segment cardiac structures, In: Proceedings of Computers in Cardiology 1993, 1993, pp. 193–196.
[20] H. L. França, J. C. P. da Silva, O. Lengerke, M. S. Dutra, M. De Gregorio, F. M. G. França, Movement persuit control of an offshore automated platform via a ram-based neural network, in: 11th International Conference on Control, Automation, Robotics and Vision, ICARCV, IEEE, 2010, pp. 2437–2441.
[21] M. Staffa, M. De Gregorio, M. Giordano, S. Rossi, Can you follow that guy? in: 22th European Symposium on Artificial Neural Networks, ESANN, 2014, pp. 511–516.
[22] M. Staffa, S. Rossi, M. Giordano, M. De Gregorio, B. Siciliano, Segmentation performance in tracking deformable objects via wnns, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 2462–2467.
[23] R.S. Moreira, N.F. Ebecken, A.S. Alves, F.M. França, Tracking targets in sea surface with the WiSARD weightless neural network, in: BRICS Conference on Computational Intelligence, IEEE, 2013, pp. 166–171.
[24] R.d. Silva Moreira, N.F. Favilla Ebecken, Parallel wisard object tracker: a ram-based tracking system, Comput. Sci. & Eng.: Int. J. 4 (1) (2014) 1–13.
[25] R. de Carvalho, D.S. C. Carvalho, F. Mora-Caminon, P. Lima, F. França, Online tracking of multiple objects using wisard, in: 22th European Symposium on Artificial Neural Networks, ESANN, 2014, pp. 541–546.
[26] I. Aleksander, M. De Gregorio, F.M.G. França, P.M.V. Lima, H. Morton, A brief introduction to weightless neural systems, in: European Symposium on Artificial Neural Networks, ESANN, 2009, pp. 299–305.
[27] T. Ludermir, A. Carvalho, A. Braga, M. Souto, Weightless neural models: a review of current and past works, Neural Comput. Surv. 2 (1999) 41–61.
[28] I. Aleksander, W.V. Thomas, P.A. Bowden, WISARD a radical step forward in image recognition, Sens. Rev. 4 (1984) 120–124.
[29] I. Aleksander, H. Morton, An Introduction to Neural Computing, Chapman & Hall, London, 1990.
[30] B.P. Grieco, P.M. Lima, M. De Gregorio, F.M. França, Producing pattern examples from "mental" images, Neurocomputing 73 (7–9) (2010) 1057–1064.
[31] Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst. Man Cybern. 9 (1) (1979) 62–66.
[32] C. Rother, V. Kolmogorov, A. Blake, "grabcut": Interactive foreground extraction using iterated graph cuts, ACM Trans. Graph. 23 (3) (2004) 309–314.
[33] G. Welch, G. Bishop, An introduction to the Kalman filter, Technical Report, 1995, Chapel Hill, NC, USA.

Massimo De Gregorio was born in Torre del Greco (Naples – Italy) in 1962. Upon completion of his graduate education (Laurea in Fisica – Cibernetica) at the University of Naples), in the 1994 he began working as researcher at the Istituto di Cibernetica of Consiglio Nazionale delle Ricerche (CNR) in the Research Group "Theory and Techniques of Knowledge Representation". Between 1999 and 2003, he held visiting teaching positions at University of Naples "Federico I" (2000–2003) and Second University of Naples (1999–2000). His research work has been mainly concerned with Artificial Intelligence, Artificial Neural Networks (in particular Weightless Neural Systems), Neuro-Symbolic Hybrid Systems, with Extensive applications in Expert Systems, Pattern Recognition, Active Video Surveillance and more recently in Behaviour Based Robotics. He is the co-author of different invited papers of several international conferences on Artificial Intelligence and Artificial Neural Networks.

Maurizio Giordano was born in Naples on 6th of August 1966. In 1992 he graduated (cum laude) in Physics with specialization in Cybernetics at the University of Naples "Federico II". From 1991 to 2000 he has been working at the Cybernetics Institute of CNR (CNRICIB) with grant supports and temporary research positions. From 2001 he is a CNR research scientist with a permanent position. From 2000 to 2013 he taught Operating Systems and Programming Languages at the Computer Science course of the Faculty of Science (University of Naples "Federico I"). From 1996 to 2005 he participated to several European Projects (Nanos & VHF Esprit Projects FP4, POP LTR Project FP5) in High Performance and Grid Computing. From 2009 he is responsible of the CNR-ICIB research unit of the HPCLAB of CNR-ISTI (Pisa). The activities he is leading are in the Service-Oriented Architecture research area and they are part of the MiDAS & S-Cube FP7 Project workplans and supported by EU fundings. From March 2014 he works at the High Performance Computing and Networking Institute of CNR (ICAR-CNR). He is coauthor of fifty papers published in international journals and conferences. He has been member of conference committees and evaluation boards for research grant and acquisition of HPC infrastructures.



Silvia Rossi is currently assistant professor at the University of Naples "Federico II" (department of Electrical Engineering and Information Technologies). She received the M.Sc. degree in Physics from University of Naples "Federico II", Italy, in 2001, and the Ph.D. in "Information and Communication Technologies" from University of Trento, Italy, in 2006. Her research interests include Artificial Intelligence, Multi-agent System, Cognitive Robotics and Human–Robot Interaction.

Mariacarla Staffa is currently Research Fellow at the University of Naples Federico II (Department of Electrical Engineering and Information Technology). She received her B.Sc. and M.Sc. degrees in Computer Science both with honours (magna cum laude) from the University of Naples Federico II in 2004 and 2008, respectively, and her Ph.D. degrees thesis entitled "Attentional Mechanism for Sensory-motor Coordination in Behavior-based Robotic Systems" under the supervision of Professors Bruno Siciliano in 2011. She is member of the PRISMA (Projects of industrial and service robotics, mechatronics and automation) and of the PRISCA (Projects of intelligent robotics and advanced cognitive systems) Laboratories, making research in the fields of Cognitive Robotics, Artificial Intelligence and Human–Robot Interaction.