2D Recurrent Neural Networks: a high-performance tool for Robust Visual Tracking in dynamic scenes

Giovanni Masala[1*], Filippo Casu[2] , Bruno Golosio[2], and Enrico Grosso[2]
[1]School of Computing, Electronics and Mathematics, Plymouth University, UK, Portland Square, Drake Circus, Plymouth, PL48AA, *corresponding author: giovanni.masala@plymouth.ac.uk, +44 (0)1752 586273.
[2]Department of Political Science, Communication, Engineering and Information Technologies, Computer Vision Laboratory, University of Sassari, Viale Mancini, 5, 07100, Sassari, Italy.

Abstract

This paper proposes a novel method for robust visual tracking of arbitrary objects, based on the combination of image-based prediction and position refinement by weighted correlation. The effectiveness of the proposed approach is demonstrated on a challenging set of dynamic video sequences, extracted from the final of triple jump at the London 2012 Summer Olympics. A comparison is made against five *baseline* tracking systems. The novel system shows remarkable superior performances with respect to the other methods, in all considered cases characterized by changing background, and a large variety of articulated motions.

The novel architecture, from here onwards named 2D Recurrent Neural Network (2D-RNN), is derived from the well-known Recurrent Neural Network model and adopts nearest neighborhood connections between the input and context layers in order to store the temporal information content of the video. Starting from the selection of the object of interest in the first frame, neural computation is applied to predict the position of the target in each video frame. Normalized cross-correlation is then applied to refine the predicted target position.

2D-RNN ensures limited complexity, great adaptability and a very fast learning time. At the same time, it shows on the considered dataset fast execution times and very good accuracy, making this approach an excellent candidate for automated analysis of complex video streams.

Keywords: Recurrent Neural Network, Convolutional Network, Video Tracking, Automated Video Analysis.

1 Introduction

The visual tracking is the generic process of locating one or more objects in the visual field. Tracking is done instinctively and without any effort by humans, but for artificial systems that use cameras, can be a very difficult and time-consuming process. Errors in tracking are often due to sharp changes in the motion of objects or camera motion, partial occlusions, changes in the appearance of the objects; for non-rigid objects, the changes in the appearance are most of the time the main problem to be addressed [1]. From this point of view, even though assumptions are often made to constrain the tracking problem within the context of a particular application, it is nowadays clear that a robust representation of the target appearance is a crucial issue in order to successfully implement tracking methods.

Based on the learning strategy adopted, tracking algorithms can be classified into two main categories: generative and discriminative methods. Generative methods rely

on a statistical model of the target appearance, usually estimated from training frames. In order to guarantee robustness of the representation and to maintain the integrity of the target appearance, various approaches have been proposed, including sparse representation [2-6] and on-line density estimation [7]. Conversely, discriminative methods rely on a direct implementation of classifiers aimed at discriminating the target from the surrounding background [8-10]. Also in this case several significant implementations have been proposed, including multiple instance learning [8], structured support vector machines [9], on-line boosting [10], random forests [11-12] and Kernel Regularized Least Squares [13-14]. These approaches are often characterized by the adoption of very peculiar features for object representation, such as Haar-like wavelets, color histograms and orientation histograms, which generally improve the detection of rigid objects but may not generalize well in presence of deformable targets and other challenges arising from complex video sequences.

In this paper, an original method is proposed for robust visual tracking, based on a combination of image prediction and weighted correlation matching techniques. The image prediction is made through a novel neural networks architecture named Two-dimensional Recurrent Neural Network (2D-RNN). 2D-RNN is derived from both Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs).

RNN is an artificial neural network with feedback connections between nodes. Due to this peculiar structure, a RNN has the capability to model dynamic systems [15]. Among RNNs, Elman's neural network (also known as Simple Recurrent Network or SRN) is a partially recurrent network that has a simple three-layer structure and a set of additional context unit receiving input from and feeding output to the intermediate (hidden) layer [16]. Training and convergence of SRNs usually take a long time, which makes them useless when dealing with high resolution images and in case of time-critical applications [4]. Authors in [31] propose deep tracking through RNN and unsupervised learning (but only using synthetic dataset) on simulated scenario representing a robot equipped with a 2D laser scanner.

A CNN is a feed-forward artificial neural network where the arrangement of individual neurons is biologically inspired by the concept of "receptive visual field". A CNN exploits local correlation of data by enforcing a local connectivity pattern between neurons of adjacent layers: neurons of "following layers" take as input small overlapping neurons of the "preceding layers", ensuring a good representation of the original image and a reasonable invariance to planar translation of the image data [17]. Due to their representation power, CNNs have recently attracted a considerable attention in the Computer Vision community [4], particularly for image- and video-based object recognition. CNNs reach excellent results in object detection/recognition (see for example [24-28]) but an extensive use of CNNs in this domain requires long training phases and good GPUs [27].

More questionable is the adoption of CNNs for generic visual tracking; performing on-line learning is possible but it is not straightforward, due to the usually large network size; moreover, according to Hong and colleagues [4], the extraction of features from the deep structure may not be appropriate for visual tracking due to the relatively poor localization accuracy and due to the function of deep layers, mainly related to the semantic content of the image.

With this respect, a recent successful attempt has been proposed by Bertinetto and colleagues[29]; they introduce a fully convolutional network in which a deep

conv-net is trained to address a more general similarity learning problem in an initial offline phase, and then this function is simply evaluated online during tracking.

The architecture proposed in this paper, the 2D-RNN, is a variation of the Elman's architecture. More in detail, this neural architecture is derived from a CNN where the input layer captures small areas of the input image. This mapping of the image pixels allows to reduce both the training time and the network dimension, yet keeping the temporal information embedded in the video and the image details unaltered.

With respect to the seminal work presented in [23], this paper focuses on the performance of 2D-RNN and gives a detailed comparison of the proposed tracker against some *baseline* trackers. In particular, three discriminative and two generative methods have been tested on the same dataset:
Boosting Tracking - a real-time object tracking based on a novel on-line version of the AdaBoost algorithm [10];
MIL Tracking - a real-time tracker based on Haar features and multiple instance learning [8];
KCF Tracker - a novel tracking framework that utilizes intensity images and founds on some mathematical properties of the circulant matrix to enhance the processing speed [13-14],
Cross Correlation - a baseline generative tracker implemented by using a normalized cross correlation and frame by frame search [18],
Meanshift tracker - a well know generative tracker based on color histograms and the meanshift procedure in order to optimize the location of the target [19].

The paper is organized as follows: in section 2, the tracking problem is analytically stated, the solution based on the novel 2D-RNN architecture is described and compared with the Elman's SRN. A case study for video tracking (triple-jumping runner and related dataset) is first introduced in section 3; then experimental steps and experimental protocols are defined. Section 4 is devoted to the comparison and discussion of the experimental results. Conclusions and future developments are finally discussed in section 5.

2 Object tracking in real-time video

In this paragraph we discuss about the tracking problem for scenes including non-rigid and articulated bodies; thereafter the two types of neural networks utilised in the experimental section, the original Elman's SRN and the proposed 2D-RNN are detailed.

2.1 Tracking

In a tracking problem, an object can be defined as "anything that is of interest for further analysis"[3]. Objects can be represented by their varying shapes and appearances; the position of a single object can be traced through a single point as the centroid or by a set of points related to a small region in the image; for example primitive geometric shapes (suitable for rigid object but also used for tracking of non-rigid objects), object silhouette and contour, articulated shape models or skeletal models. In the proposed method a simple rectangular shape (bounding box or BB) is used. The BB has a fixed dimension for all frames of the database. Note that for the

purposes of this paper, the initialization of the tracking process, for example by moving objects detection or direct object recognition, it is not explicitly considered; as a consequence the object of interest must be defined at the initial time step by manually placing a starting BB in the first frame. Afterward, the tracking algorithm iteratively determines the object position in the next frame by the following three steps:

– Step 1: prediction of the next-frame by 2D-RNN.

The past *i* images inside the BB are fed as input to a 2D-RNN, which produces as output the prediction of the expected image content of the bounding box (sub-image) for the current time step.

– Step 2 : correlation-based refinement.

The expected position of the bounding box for the current time step can be evaluated and refined through the correlation between the predicted sub-image (2D-RNN output) and the current image. Let C(x,y) be the correlation matrix between the predicted sub-image and the current entire image. The arguments of the correlation matrix represent the relative coordinates between the sub-image and the entire image. Each element represents the value of the correlation between the sub-image and the entire image for those relative coordinates. The correlation matrix is computed by convolution in the Fourier domain; the position of the maximum of the correlation matrix corresponds to the best prediction of the BB position for the current time step.

– Step 3 : Computation of the expected position of the BB

In general, the correlation matrix can have more than one local maximum, and it can happen that the target BB position is close to a local maximum that is not the absolute maximum. Such issue is quite relevant when the dynamic object of interest is subject to partial occlusions, abrupt deformations, etc. With the purpose to solve this issue, in our method the correlation matrix C(x,y) is weighted with a two-dimensional Gaussian function centered at the extrapolated center position $(\hat{X}_c^{t,extr}, \hat{Y}_c^{t,extr})$, which is evaluated as follows: at each time step *t*, it can be assumed that the object position has been detected in the previous *t*−*i* time steps, through the centroid of the bounding box. The system then computes the extrapolated position of the BB based on the measured velocity of the two most recent frames. In particular, the coordinates $X_c^{t,extr}$ and $Y_c^{t,extr}$ of the extrapolated center position are defined through :

$$\hat{X}_c^{t,extr} = X_c^{t-1} + \Delta X_c^{t-1} \quad (1)$$

$$\hat{Y}_c^{t,extr} = Y_c^{t-1} + \Delta Y_c^{t-1} \quad (2)$$

Where

$$\Delta X_c^{t-1} = X_c^{t-1} - X_c^{t-2} \quad (3)$$

$$\Delta Y_c^{t-1} = Y_c^{t-1} - Y_c^{t-2} \quad (4)$$

The weighted correlation matrix is evaluated as:

$$C_w(x,y) = C(x,y) \cdot G(\hat{X}_c^{t,extr}; \hat{Y}_c^{t,extr}; x, y) \quad (5)$$

Where $G(\hat{X}_c^{t,extr}; \hat{Y}_c^{t,extr}; x, y)$ is a two-dimensional Gaussian function centered on $(\hat{X}_c^{t,extr}, \hat{Y}_c^{t,extr})$.

The refined center position is given by the coordinates at which the weighted correlation matrix has a maximum:

$$(X_c^t, Y_c^t) = \arg\max[\mathrm{C_w}(\mathrm{x},\mathrm{y})] \quad (6)$$

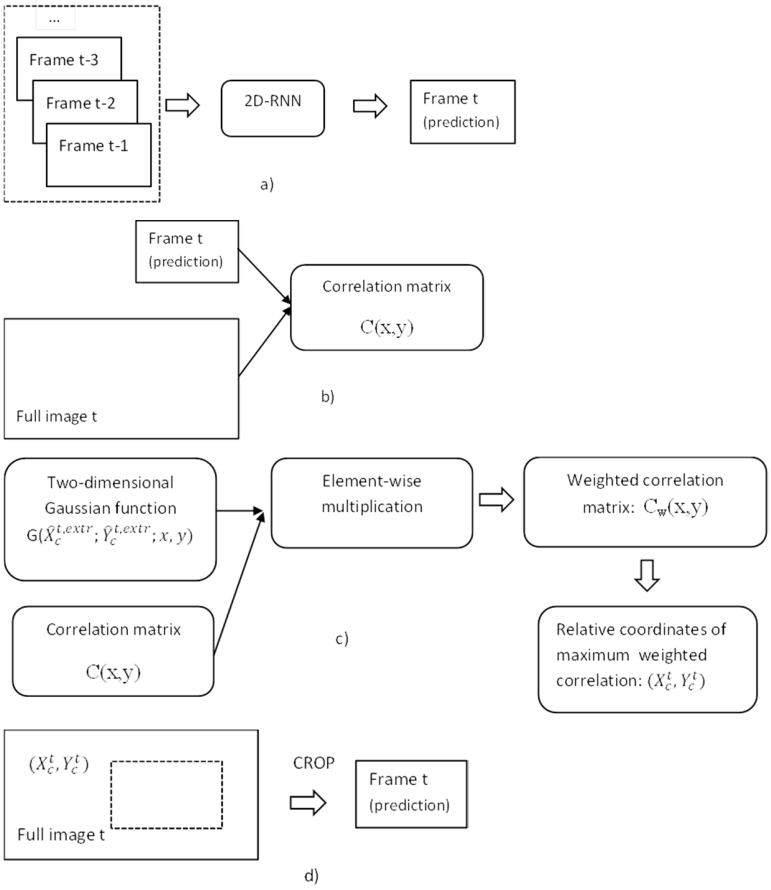

Fig. 1. a) The 2D-RNN takes as input the images in the bounding box at previous time steps and yields as output a prediction of the image in the bounding box at time step t. b) The correlation between the predicted bounding box image and the entire image is evaluated. (X_c^t, Y_c^t) are the relative coordinates between the bounding box and the entire image for which the correlation is maximum. c) The correlation matrix C(x,y) is weighted with a two-dimensional Gaussian function centered at the extrapolated center position; (X_c^t, Y_c^t) are the relative coordinates between the bounding box and the entire image for which the weighted correlation is maximum. d) The bounding box image at time step t is obtained by cropping the entire image at coordinates (X_c^t, Y_c^t).

Note that using the above tracking scheme also the predicted content of the BB can be evaluated (both the dynamic background and the object of interest) by considering the residual error corresponding to the maximum of the correlation.

Figure **1** shows in detail the tracking scheme based on the 2D-RNN (or SRN to have a comparison) next frame prediction.

2.2 The Elman's neural network

In a Elman's Simple Recurrent Network (SRN) an input, hidden, context, and an output layers are defined. The outputs of the context neurons and the external input neurons are fed to the hidden neurons. Context neurons are known as memory units as they store the previous output of hidden neurons. At the time step t, the context layer nodes carry the output of hidden layer nodes of the time step t−1 iteration and supply that as input during processing of the time step t data. The SRN architecture is shown in Fig**. 2**.

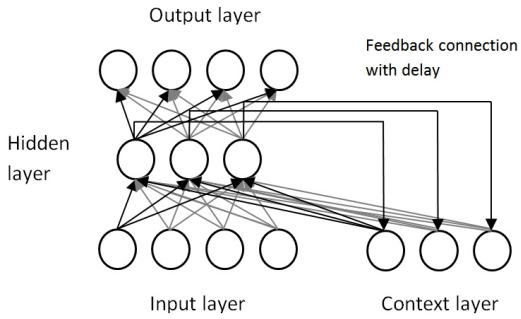

Fig. 2. Architecture of the SRN. The layers are fully connected with a feedback connection between the hidden and the context layers. The context layer provides both actual and delayed inputs to the hidden layer.

Considering I, S, C and O as input, hidden, context and output layer vectors, respectively, the vector components at the t[th] iteration can be written as [20]:

$$i_p^t \in I, p = 1,2,..,n \quad (7)$$
$$s_q^t \in S, q = 1,2,..,m \quad (8)$$
$$o_r^t \in O, r = 1,2,..,l \quad (9)$$
$$c_q^t = C \quad (10)$$
$$s_q^t = f\left(b_q^t\right) \quad (11)$$
$$c_q^t = s_q^{t-1} \quad (12)$$

In the above equations, n, m and l represent the numbers of nodes of input, hidden, and output layer, respectively, $f(\cdot)$ indicates the activation function of the q^{th} hidden node at the t^{th} iteration, while c_q^t denotes the input of the q^{th} context layer node at the t^{th} iteration and b_q^t is the linear output of the hidden node q at t[th] iteration.

Let $W1$, $W2$ and $W3$ be the weight matrices between input and hidden layer, hidden and context layer and hidden and output layer , respectively. The output of hidden layer and output layer nodes at the t^{th} iteration with these weight matrices can be represented by the following equations :

$$s_q^t = f\left(\sum_{p=1}^{n} w_{1qp} i_p^t + \sum_{j=1}^{m} w_{2qj} s_j^{t-1}\right) \quad (13)$$

$$o_r^t = f\left(\sum_{q=1}^{m} w_{3rq} s_q^t\right) \quad (14)$$

where the w_{1qp}, w_{2qj}, w_{3rq} are the elements of the weight matrices $W1$, $W2$, and $W3$, respectively.

The backpropagation algorithm is used to perform the neural network training [21]. In such algorithm, the error is minimized to converge to the target value by updating the link weights at each iteration through the equation (15).

$$W^{new} = W^{old} + \alpha \Delta W \tag{15}$$

where α is the learning rate.

The error E expresses the difference between the set target at the output nodes and the actual output obtained as defined in equation (16):

$$E(W) = \frac{1}{2} \sum_{k=1}^{e} \sum_{r=1}^{l} (O_r^t - o_r^t)^2 \tag{16}$$

where $O^{t,r}$ and $o^{t,r}$ represents the set target and the actual output from the network at the t^{th} iteration, respectively, and e is the number of epochs.

2.3 Two-dimensional Recursive Neural Networks

In the proposed 2D-RNN, hidden, context and output layers are organized in two-dimensional arrays all having the same dimensions as the input image. Unlike the Elman's network, the layers of the proposed network are not fully connected to each other. In particular , denoting by (x,y) the index of row and column of the matrix of the hidden layer, respectively, 2D-RNN uses for each element (x',y') of the input matrix also its nearest elements in the connection with the correspondent element of the hidden layer (x,y). This type of association is replicated in the connection of the context layer with the hidden layer and in the connection between the hidden layer and the output layer, as shown in figure **3**.

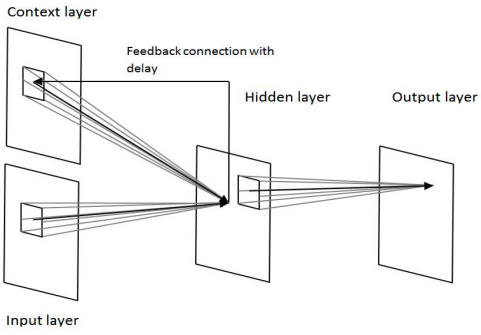

Fig. 3. Architecture of the 2D-RNN. Mapping of the image pixels from the input and context layers. Each node in the hidden layer receives input from both the actual and delayed image. Spatial information is preserved through the layers

Note that neuron (x,y) of the hidden layer is connected to all neurons (x',y') of the input layer and to all neurons (x'',y'') of the context layer with:

$$x - k \leq x' \leq x + k \, , \, y - k \leq y' \leq y + k \tag{17}$$

$$x - k \leq x'' \leq x + k \, , \, y - k \leq y'' \leq y + k \tag{18}$$

Anyway, the neuron at position (x,y) of the hidden layer is connected to the corresponding neuron of the input layer and to its nearest neighbors, and to the corresponding neuron of the context layer and to its nearest neighbors. Similarly, each neuron of the output layer is connected to the corresponding neuron of the hidden layer and to its nearest neighbors.

Also in this case the backpropagation algorithm is used to perform the training of the network. Also the additional k parameter (dimension of the neighborhood) requires an optimization, and the equations (13) and (14) are modified in this forms:

$$s_{xy}^{t} = f\left(\sum_{u=x-k}^{x+k}\sum_{w=y-k}^{y+k} w_{1xyuw}\, i_{uw}^{t} + \sum_{u=x-k}^{x+k}\sum_{w=y-k}^{y+k} w_{2xyuw}\, S_{uw}^{t-1}\right) \tag{19}$$

$$o_{xy}^{t} = f\left(\sum_{u=x-k}^{x+k}\sum_{w=y-k}^{y+k} w_{3xyuw}\, S_{uw}^{t}\right) \tag{20}$$

3 Experimental results

3.1 Basic assumptions and datasets

A limited but challenging dataset of sequences, extracted from the final of triple jump at the London 2012 Summer Olympics, is used in order to validate the proposed tracking algorithm. The original video data is freely available on the YouTube platform [32]. As illustrated by few frames in fig. 4, the dataset is characterized by severe conditions that strongly affect the application of tracking techniques. In particular it is worth noting the presence of a moving target (the athlete) over a dynamic background (due to the continuous motion of the camera); additional critical issues are also present such as noise, articulated motion and scene illumination changes.

Fig. 4. Frames extracted from the triple jump sequence. Several visual artifacts can be noticed, such as moving background, changes in the object (the runner) shape, changes in lighting and occlusions.

A total number of 10 sequences is used in the experimental phase; each sequence relates to a different athlete, as shown in table 1.

Sequence	Athletes' name	Tot frames
1	Platniski	97
2	Laine	103
3	Dong	108
4	Copello	113
5	Oke	117
6	Compaore	96
7	Sands	98
8	Greco	119
9	Donato	119
10	Claye	127

Table 1: Sequences considered in the MP4 video format (stored in a single file)

The sequences are in MP4 video format (all are stored in a single file) and are characterized by a frame rate of 29 images/s; the dimension of each original frame is 1280×720 pixels. Each sequence has a duration of about 45 seconds but temporal subsampling is applied taking only one frame every ten for further processing. Therefore, for each sequence the number of frames processed varies between 97 and 127.

Our approach to visual tracking is not in terms of object detection performed frame by frame, but it rather refers to the ability of continuously establishing the correspondence of a given foreground area (containing the object of interest) between two subsequent frames[33]. As stated in the introduction, object-tracking methods often impose external constraints in order to guarantee adequate tracking performances; these constraints almost always concern the appearance of objects. Most tracking algorithms also assume some a priori knowledge on the motion of the objects, for instance stating constant velocity or constant acceleration of the target. Finally, as discussed in the previous section, prior knowledge about the number and the size of objects, or the object appearance and shape, can be used to simplify the problem. The proposed method does not make assumptions. Furthermore, it does not use any pre-processing of the image to remove external objects (i.e. TV-written) and it does not apply any pre-processing such as band-pass filtering or segmentation. The result of the developed object tracker is shown by a simple bounding box that contains the athlete in all different frames of the video (see figure 5). The gold standard for each frame is provided through manual labeling of the region of interest and more specifically by defining the position of the pelvic bones of the athlete.

In summary, the main processing steps for the experimental phase are the following:

- temporal subsampling and extraction of the single JPG frames for each sequence;
- resizing by interpolation of all the frames from the original number of pixels (1280×720) to a small image dimension (128×72pixels) more convenient for further processing ;
- conversion of color frames to grayscale;
- 2D-RNN training and validation.

Training and validation are repeated following a cross-validation scheme; for each combination of training and validation sets, a comparison between the original SRN with respect to the novel 2D-RNN is performed. Input data are the same for both networks.

3.2 Configuration

Both networks require a careful evaluation of configuration parameters. The SRN can identify the single-order dynamic system using fixed coefficients in the context neurons, using weight = 1 in the feedback connections with the context layer; SRN best architecture needs 2500 input, 250 hidden, 250 context and 2500 output neurons. Note that the input and output layers are related to the frame input matrix (the 50x50 pixels bounding box) while the number of neurons of the context and hidden layers have been optimized trying several configurations.

2D-RNN is not fully connected as the SRN; it requires 2500 input, hidden, context and output neurons (the numbers of neurons for all layers is fixed with respect to the frame input matrix). Best results are obtained for a number of nearest neighbors k=3, using weight=1 in the feedback connections with the context layer. For both RNNs and for all neurons a logistic standard transfer functions has been adopted.

4 Results and Discussion

4.1 Performances of RNNs

In order to check the independence from the sampling of the dataset, a k-folder cross validation (5x2) has been used in the simulations. First round of cross-validation involves partitioning a sample of data in two complementary subsets, and it carries out an investigation on one subset (train set of 5 videos), while the validation is made on the other subset (test set of 5 videos); after that the experiment is repeated swapping train and test sets. To reduce the variance, 5 rounds of cross-validation are carried out choosing random different partitions, and the validation results are averaged over the 10 (5x2) rounds.

In table 2 the comparison of the best setting for both RNNs on the same random train test and blind test set is shown; learning times refer to a simple desktop architecture based on an Intel CoreTM 2 DUO CPU E 8400 @3.00 GHz and 4 GB RAM.

Configuration Parameters	2D-RNN	SRN
Input Neurons	2500	2500
Output Neurons	2500	2500
Hidden Neurons	2500	250
Context Neurons	2500	250
Learning rate	0.05	0.005
Number of Epochs	130	280
Number of Connections	367500	1312500
Learning time (s)	1092	9230
Best rmse	0.104	0.114

Table 2: Performance analysis of the 2D-RNN and SRN for the same blind test set

Table 2 clearly indicates that the learning phase of 2D-RNN is faster than SRN, and 2D-RNN produces the best results. The best learning rate for both RNNs are reported. In particular, the root-mean-square deviation (rmse) is repeatedly computed on the test set after a random selection of the training set followed by the learning phase. The results for the 2D-RNN, in 5x2 cross validation, is a mean rmse = 0.105 ± 0.003 . In summary, table 2 proves that 2D-RNN, compared to SRN on the same dataset, provides a better rmse; the results are stable for the 5x2 cross-validation and 2D-RNN is faster than SRN in terms of learning time and epochs. The complexity of the 2D-RNN is lower than SRN in terms of connections.

4.2 Introduction to tracking results

Visual tracking outcomes can be displayed through the distances between the center of manual annotation (the pelvic bones) of the athlete and the center of bounding box in the 2D-RNN next frame prediction. Using only one frame every ten and starting from the original frame rate information, the RNN predictions correspond to one image every 0.344 s . In figure 5 corresponding samples for the SRN (left) with actual frames and next frame prediction are indicated, together with the correlation diagram. The same results are manifested for the 2D-RNN in figure 5 (right). In the surface plot, the peak of the cross-correlation matrix occurs where the sub images are best correlated.

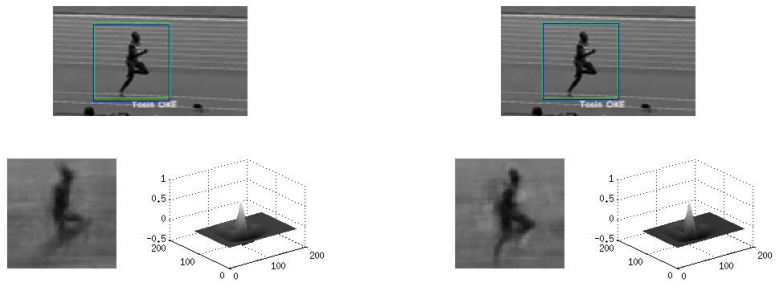

Fig. 5. In this composed pictures with both SRN (left) and 2D-RNN (right), are shown the current frame, predicted next frame and the correlation diagram. The blue bounding box displays the gold standard while the green box displays the position computed by the systems.

It should be highlighted that all the next frames prediction in all pictures are blurred because the RNNs produce a probabilistic distribution of intensity values. This distribution reflects the variability of the images used to train the RNN. In a natural manner, the athletes move their limbs in different ways during the run-up. The issue that the body image is blurred is the consequence of the inability to obtain an accurate prediction. Moreover images with clear prediction of the part of the body with respect to blurred images, would reduce correlations on the average, and consequently larger mean errors, due to the variability and not exact predictability of the next image. This is a compromise viable because in the tracking problem it is only necessary to have an accurate prediction of the center of the BB to follow the object of interest.

In this paper object tracking is evaluated at pixel level [22]. There are no lost frames in the proposed approach, therefore evaluation metrics based on accuracy are

not used. With the aim to show the performances on the correct location of the BB, a position based measure (PBM) can be defined as in [22]:

$$PBM = \frac{1}{N_f}\sum_i \left[1 - \frac{D(i)}{T_h}\right] \quad (21)$$

Where

$$T_h = \frac{1}{2}[(BB_w) + (BB_h)] \quad (22)$$

depends on the dimensions (width and height) of the bounding box. In equation 21, N_f is the total number of frames considered whilst $D(i)$ is the L1-norm distance between the gold standard and the BB predicted by RNN. Using such index in our dataset the resulting mean of PBM is expressed always for the first 85 frames. After the frame 85 there is a large deviation, due to the runner landing on the sand.

A further measure quite convenient for comparison is the deviation index. Deviation defines the capability of a tracker to determine the correct position of the target and measures the accuracy of tracking [22]. In particular, by using Deviation as the error of the center location expressed in pixels as a tracking accuracy measure:

$$Deviation = 1 - \frac{\sum_{i \in Ms} d(T^i, GT^i)}{|Ms|} \quad (23)$$

where $d(T^i, GT^i)$ is the normalized distance between the centroids of the bounding box (BB) and the gold standard while |Ms| denotes the set of frames in a video where the tracked BB overlaps for more than 50% the golden standard bounding box.

In according with [23] another viable measure for comparison is the Euclidean Distance (number of pixels) from the center of the bounding box and the gold standard.

4.3 Results for tracking: 2D-RNN vs *baseline* trackers

In the following comparison, five *baseline* trackers, all freely available on the OpenCV platform, have been considered. A short list of the main features of these trackers is given in the table 3.

Most of times, the configuration of the trackers did not require special effort: in fact, we did not observe substantial effects of small variations of the configuration parameters on the performance of the trackers.

Default configurations have been used for Boosting [10], MIL [8], KCF [14], Cross Correlation [18].For the Meanshift tracker [19], the optimal number of histogram bins has been set in the range 120 to 200, depending on the sequence.

All the metrics used in the results are calculated for the resized video (1/10), for a bounding box of 50x50 pixels, and taking a frame every 10 frames.

Tracker	Target Region	Appearance Model	Motion Model	Method	Update
2D-RNN	Bounding Box	Recursive neural networks (gray level intensity)	Uniform search,No motion model	Recursive neural networks, and weighted correlation (generative)	Incremental update
Boosting	Bounding Box	Haar features, Orientation histograms, Local Binary Patterns	Uniform search	Ada Boost Classifier (discriminative)	Incremental update
MIL	Multiple Boxes	Haar features	Uniform search	MIL Boost Classifier (discriminative)	Incremental update
KCF	Bounding Box	Array of dense samples (intensity)	Gaussian Search	Kernel Regularized Least Squares with classifier (discriminative)	Incremental update
Cross Correlation	Bounding Box	Intensity	Uniform search	Normalized Cross Correlation (generative)	No update
Meanshift	Bounding Box	Color Histograms	Mean shift moments	Mean shift maximum (generative)	Continuous

Table 3: Main characteristics of the *baseline* trackers used for comparison

In table 4 the results in terms of PBM, Deviation and Euclidean Distance are reported for the 2D-RNN and all baseline trackers, with their standard deviations, respectively. Such values are calculated for the first 85 frames of each sequence in the whole test set.

Tracker	PBM	St. dev.	Deviation	St. dev.	Euclidean Distance	St. dev.
2D-RNN	0.95	±0.03	0.95	±0.03	2.4	±1.5
Boosting	0.88	±0.10	0.89	±0.10	5.9	±4.9
MIL	0.52	±0.17	0.85	±0.17	24.3	±8.5
KCF	0.43	±0.09	0.90	±0.09	29	±3.8
Cross Correlation	0.53	±0.16	0.84	±0.16	23.9	±7.9
Meanshift	0.90	±0.09	0.93	±0.09	4.8	±4.4

Table 4: Main results of the *state of the art* trackers among 2D-RNN.

For the algorithms KCF and cross correlation the object of interest is lost in some frames, therefore for these methods the values of the indexes in table 4 are not calculated in these frames. However, the Deviation is calculated as general formula, only for the frame in which the overlapping between BB and object is not less than 50% (that is, when the object is not lost).

In any case, the 2D-RNN shows better results than the other methods used for all indices. Further comparisons between SNR and 2D-RNN are reported in [23], showing that 2D-RNN performs better than SNR in all considered cases.

In fig. 6 the Euclidean Distance (number of pixels) from the center of the bounding box and the gold standard for five sequences randomly selected is shown . In the right side of the chart the average value for each tracker is plotted. In fig. 7 a similar comparison based on PBM is presented.

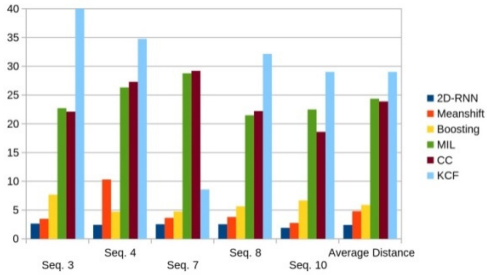

Fig 6. Performance of the trackers measured by the Euclidean distances (number of pixels) from the gold standard (five random sequences and overall average).

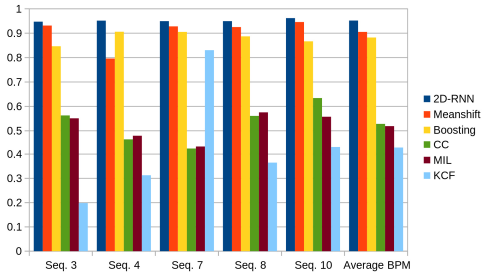

Fig.7. Performance of the trackers measured by the PBM (five random sequences and overall average).

In fig. 8 the comparison of the trackers is shown by using the deviation index;

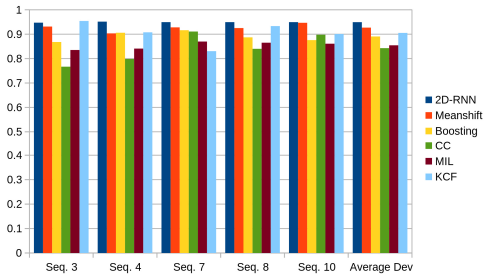

Fig.8. Deviation (five random sequences and overall average).

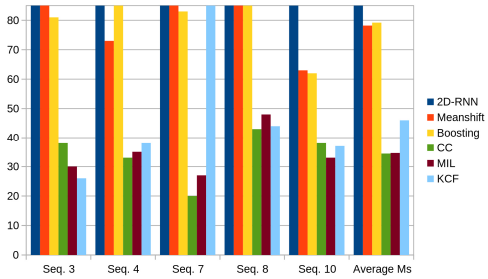

Fig.9. |Ms|, number of frames where the Bounding Box overlaps for more than 50% of the golden standard bounding box.

Fig. 9 gives some additional details on the deviation measures showing that a similar performance in figure 8 does not correspond to an effective tracking performance. In fact for CC, MIL and KCF the number of "lost" frames is significant. This behavior is further clarified by fig. 10, reporting the average distance (for the five sequences) between the center of the BB and the gold standard. Weaknesses of CC, MIL and KCF clearly emerge, but also for Meanshift and Boosting it is possible to observe slight deviations from the golden standard in some parts of the trajectory. For 2D-RNN results are clearly stable.

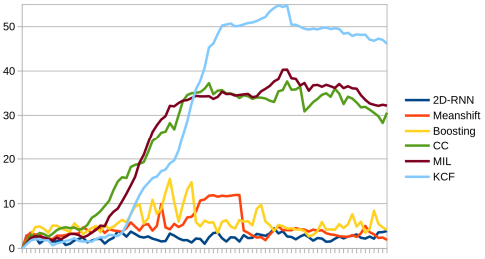

Fig.10. Temporal evolution of the average Euclidean distance (five sequences).

5 Conclusion

A novel tracking algorithm based on the fusion of two complementary neural networks architectures has been presented. The temporal memory of a recursive neural networks is used to keep the correlation among processed pixels, and to perform the next frame prediction at the temporal distances of ten frames, with respect to the frame of interest.

The novel algorithm is called 2D-RNN because a two dimensional approach is proposed: For each pixel of the input image also the information of its k nearest pixels are considered, without any pre-processing of such image.

A quantitative comparison against five *baseline* approaches on the same datasets is made, obtaining superior performances of the proposed method, on the base of recognised important indexes of the video tracking literature. 2D-RNN has also superior performances in terms of rmse and learning times with respect to the classical SRN architecture.

The extension of this approach will be applied in the future to large benchmark datasets, for example ILSVRC (e.g. [29]), with different types of object of interest, and replacing the manual selection of the BB in the first frame with an automatic procedure designed to recognize objects belonging to predefined classes. In future works we will also explore the same optimisation of the SRN shown here, using instead LSTMs [30], which recently are receiving increasing attention in the image processing community due to their better capability of learning long-term dependencies. The system could be useful for the analysis of the athlete errors in the jump, following the paradigm of the computer aided coaching, or for the generation of real-time TV highlights. Moreover, as the new method doesn't require any information related to the object of interest it is therefore suitable for a large set of applications from automated analysis of sport activities to intelligent video-surveillance.

Disclosure of potential conflict of interest

Conflict of Interest: The authors declare that they have no conflict of interest.

Note

This is the author's accepted manuscript. The final published version of this work is published by Springer in Neural Computing & Application 13-10-2017 available at: DOI 10.1007/s00521-017-3235-x. This work is made available online in accordance with the publisher's policies. Please refer to any applicable terms of use of the publisher."

References

[1] H. Denman, N. Rea, A. Kokaram, Content-based analysis for video from snooker broadcasts, Computer Vision and Image Understanding 92 (2) (2003) 176–195.
[2] A. Kokaram, F. Pitie, R. Dahyot, N. Rea, S. Yeterian, Content controlled image representation for sports streaming, Proc. of Content-Based Multimedia Indexing (CBMI05).
[3] A. Yilmaz, O. Javed, M. Shah, Object tracking: A survey, Acm computing surveys (CSUR) 38 (4) (2006) 13.
[4] S. Hong, T. You, S. Kwak, B. Han, Online tracking by learning discriminative saliency map with convolutional neural network, arXiv preprint arXiv:1502.06796.
[5] C. Bao, Y. Wu, H. Ling, H. Ji, Real time robust l1 tracker using accelerated proximal gradient approach, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1830–1837.
[6] X. Jia, H. Lu, M.-H. Yang, Visual tracking via adaptive structural local sparse appearance model, in: Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1822–1829.
[7] X. Mei, H. Ling, Robust visual tracking using L1 minimization, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 1436–1443.
[8] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (8) (2011) 1619–1632.
[9] S. Hare, A. Saffari, P. H. Torr, Struck: Structured output tracking with kernels, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 263–270.
[10] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting., in: BMVC, Vol. 1, 2006, p. 6.
[11] J. Gall, A. Yao, N. Razavi, L. Van Gool, V. Lempitsky, Hough forests for object detection, tracking, and action recognition, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (11) (2011) 2188–2202.
[12] S. Schulter, C. Leistner, P. M. Roth, H. Bischof, L. J. Van Gool, On-line hough forests., in: BMVC, 2011, pp. 1–11.

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *proceedings of the European Conference on Computer Vision*, 2012.
[14] M. Danelljan, F.S. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1090–1097, June 2014
[15] X. Wang, L. Ma, B. Wang, T. Wang, A hybrid optimization-based recurrent neural network for real-time data prediction, Neurocomputing 120 (2013) 547–559.
[16] J. L. Elman, Finding structure in time, Cognitive science 14 (2) (1990) 179–211.
[17] K. Korekado, T. Morie, O. Nomura, H. Ando, T. Nakano, M. Matsugu, A. Iwata, A convolutional neural network vlsi for image recognition using merged/mixed analog-digital architecture, in: Knowledge-Based Intelligent Information and Engineering Sys tems, Springer, 2003, pp. 169–176.
[18] K. Briechle, U. D. Hanebeck, Template matching using fast normalized cross correlation, in: Aerospace/Defense Sensing, Simulation, and Controls, International Society for Optics and Photonics, 2001, pp. 95–102.
[19] Bradski, G.R., "Real time face and object tracking as a component of a perceptual user interface" Applications of Computer Vision, 1998. WACV '98. Proceedings., Fourth IEEE Workshop on , vol., no., pp.214,219, 19-21 Oct 1998.
[20] S. Şeker, E. Ayaz, E. Türkcan, Elman's recurrent neural network applications to condition monitoring in nuclear power plant and rotating machinery, Engineering Applications of Artificial Intelligence 16 (7) (2003) 647–656.
[21] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.
[22] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, M. Shah, Visual tracking: an experimental survey, Pattern Analysis and Machine Intelligence, IEEE Transactions on 36 (7) (2014) 1442–1468.
[23] Masala, G. L., Golosio, B., Tistarelli, M., & Grosso, E. (2016, September). 2D Recurrent Neural Networks for Robust Visual Tracking of Non-Rigid Bodies. In International Conference on Engineering Applications of Neural Networks (pp. 18-34). Springer International Publishing.
[24] Sermanet, P. et al,"Overfeat: Integrated recognition, localization and detection using convolutional networks",International Conference on Learning Representations (ICLR 2014),16,2013,CBLS
[25] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, CVPR, 2014.
[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, ECCV, 2014.
[27] Shaoqing R. et al, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Advances in Neural Information Processing Systems(NIPS),2015.
[28] Redmon, J. et al, You Only Look Once: Unified, Real-Time Object Detection, arXiv:1506.02640 , 2015
[29] Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., & Torr, P. H. (2016, October). Fully-convolutional siamese networks for object tracking. In European Conference on Computer Vision (pp. 850-865). Springer International Publishing.
[30] Russell Stewart and Mykhaylo Andriluka, "End-to-end People Detection in Crowded Scenes", in 29th IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, Los Alamitos, CA, 2016

[31] Ondruska, P., & Posner, I. Deep tracking: Seeing beyond seeing using recurrent neural networks. AAAI-16 conference, Feb 12-17, Phoenix, Arizona USA, 2016.
[32] Dataset: final of triple jump at the London 2012 Summer Olympics available on the YouTube platform: https://www.youtube.com/watch?v=GeYfshPYyZ8
[33] Liu R, Wang D, Han Y, Fan X, Luo Z, Adaptive low-rank subspace learning with online optimization for robust visual tracking, Neural Networks, Volume 88, April 2017, Pages 90-104, ISSN 0893-6080, https://doi.org/10.1016/j.neunet.2017.02.002.