Object Tracking with Adaptive Particle Filter Tracker Using Convolutional Neural Network

Azdoud Youssef, Amine Aouatif, Hachimi Hanaa
Systems Engineering Laboratory, National School of Applied Sciences
Ibn Tofail University of Kenitra - Morocco
youssef.azdoud@uit.ac.ma, aouatif.amine@uit.ac.ma, hanaa.hachimi@uit.ac.ma

Abstract
Machine learning in today's computer vision domains has shown their accuracy and performance and that has taken the attention of the community to dive deeper into its application. Thus these machine learning algorithms have guided this research to introduce our tracking object system, which relies on a probabilistic algorithm that gives an accurate location of the target within the area of research, combined with the technique that propagates searching elements according to a distribution, where the probabilistic algorithm adopted in our case is Particle Filter (PF), it is a meta-heuristic method, which searches for the perfect state estimate within the area of research. It approximates the filtered posterior generation by a set of weighted particles, it weights the particles based on a likelihood score and then propagates these particles according to a movement model which is a kernel distribution, then we classify the particles that are under a specific threshold with the pre-trained AlexNet CNN. The best candidate is the one that belongs to the same class as the target and it has a good weight. We have conducted qualitative and quantitative experiments on our approach, where it achieves 24.3 pixels in average error localization.

Keywords
Visual object tracking, kernel distribution, metaheuristic, particle filter, deep learning, convolutional neural network.

1. Introduction
Computer vision community have been studying visual tracking object for decades due to its importance in numerous applications such as iris recognition border-crossing, detect vehicle registration plates, autonomous driving, and much more. Despite the state of the art of object tracking algorithms, the research in the field is inexhaustible. Because the challenges that they face like changes in the object's appearance or large lighting variation in the background makes these algorithms fail to observe the object motion. Thus, offer new studies a large place to give improvements. By definition, object tracking is the localization of a detected target as long as it is presented on each frame in the sequence. Most existing tracking approaches focus on the recognition process, which is usually based on the main visual descriptors, for instance, color, shape, texture, or features presented in a frequency space (Tayb 2016). In the literature, we can categorize the tracking algorithms into tree classes, particularly deterministic approaches, probabilistic approaches, and deep learning approaches. Deterministic approaches search iteratively for the area most similar to the target window area via maximizing or minimizing measures between those areas. A typical deterministic mothed is Mean Shift (Comaniciu et al. 2003), it uses histogram to represent the target and for the similarity function, they use the Bhattacharyya coefficient. other deterministic approaches adopted integrating another feature description such as texture feature (Bousetouane et al. 2013), SIFT (Jansen et al. 2015), Cross-Bin metric (Leichter 2012). On the other hand, probabilistic approaches based on uncertainty and randomized logic to find the fulfillment solution. In contrast to most existing probabilistic metaheuristic algorithms (Davis 1991), (Carpenter et al. 1999), (Kennedy and Eberhart 1995), (X.S Yang et al. 2008) and (X.S Yang 2011) they maneuver elements that attempt to locate the global minima or maxima within a searching space, these elements are kept or replace with a new one each generation, and that depends on the fitness function. There is however a nuance between these algorithms in the number of elements and parameters. Recently, the researchers are more and more interested in deep learning due to their good achievements and their performance in image classification, object tracking, and image segmentation. Hence, this has created a new category of tracking algorithms based on deep convolutional neural networks (C. Ma et al. 2015), (S. Hong et al. 2015) and (N. Wang and D.-Y. Yeung 2013). Benchmark evaluations on object trackers show that of CNN

based trackers are surpassing traditional tracker base on color histogram (G. Tian et al. 2009), (H. Chu et al. 2007), HOG (N. Dalal and B. Triggs 2005) and SIFT (Jansen et al. 2015). The rest of the paper is organized as follows. Section 2 discusses our proposed method named CNN Adaptive Particle Filter (CAPF) using kernel distribution. Section 3 provides experimental results. Finally, section 4 portrays a conclusion and future work.

2. Proposed method

In this work, we have added a deep learning layer into our system which relies on a probabilistic algorithm that gives an accurate location of the target within the area of research, combined with the technique that propagates searching elements according to a distribution. The tracking process goes through the different stages as explained later, and it is dedicated to tracking object applications in stationary camera sequences.

2.1 Particle Filter application

The probabilistic algorithm adopted in here is Particle Filter (PF), it is a metaheuristic method, which searches for the perfect state estimate in a dynamical system. It is a hypothetical algorithm that approximates the filtered posterior generation by a set of weighted particles, it weights the particles based on a fitness function or likelihood score and then propagates these particles according to a movement model (Carpenter et al. 1999).
Our CNN Adaptive Particle Filter tracker relies on a deterministic search window, whose feature content corresponds to the histogram color model. Indeed, likelihood weights are calculated with a normalized Euclidean distance between target color histogram and particle color histogram. Actually, the particle state modeling is the location of each state within the image in this application, where the state space is represented in the spatial domain $P_i = (x, y)$. Figure 1 represents particle modeling on a frame.

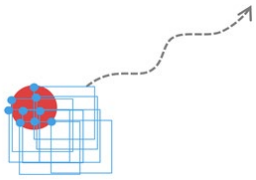


Figure 1: The spatial representation of particles within a frame

Firstly, we generate a set of N particles, then in the first image I_0 we initialize the states randomly. Next, we calculate for each particle state $P_{1,\dots,N}$ its RGB normalized histogram $hist_{rgb}^{P_i} = \left[H_r^{P_i} H_g^{P_i} H_b^{P_i}\right]$. Finally, we calculate the Euclidean distance between each color channel of the inputs, which gives us tree distances:

$$d_r^{P_i} = \sqrt{\sum \left(Tr_r - H_r^{Pi}\right)^2} \qquad (1)$$

$$d_g^{P_i} = \sqrt{\sum \left(Tr_g - H_g^{Pi}\right)^2} \qquad (2)$$

$$d_b^{P_i} = \sqrt{\sum \left(Tr_b - H_b^{Pi}\right)^2} \qquad (3)$$

Where d_r^{Pi}, d_g^{Pi} and d_b^{Pi}, are the Euclidean distances between iTh particle state and target histogram for channel R, G, and B, and these three distances represent the likelihood weights. Whereas Tr_r, Tr_g, and Tr_b are the target's histogram for the R, G, and B channels successively.

After the previous step, it comes filtering stage PF where we group the three weight values in one normalized vector see equations (4) and (5), these equations represent weight's vector for a particular state.

$$w_{1,\dots,N}^t = \sqrt{d_r^{Pi\,2} + d_g^{Pi\,2} + d_b^{Pi\,2}} \qquad (4)$$

$$w_{1,\dots,N}^t = \left\| d_{rgb}^{Pi} \right\| \qquad (5)$$

After processing all particle states, we minimize the set of weights calculated, in order to find the nearest state P_i^t similar to the target for the iTh state particle at the frame I_t. All the values and the positions on the frame are saved in a vector. On the resampling stage, we apply our proposed distribution resampling technique, which aims to generate new particle state for the next generation. The function defined as follows:

$$P_{1,\dots,N}^{t+1} = DS(p_i^t) \quad (6)$$

$$P_{1,\dots,N}^{t+1} = \{P_1^{t+1}, \dots, P_N^{t+1}\} \quad (7)$$

2.2 Distribution resampling contribution

At the PF resampling phase, we generate new particles for states that have low weights, to be used in the next generation. Our purpose is to improve this resampling process, by making new particles that could give higher weights. Consequently, it offers more chances to find the nearest estimate of the target location within the Area Of Research (AOR).

Normally, by using normal random propagation, it will not spread particle states in AOR that could give good candidates similar to the target. Therefore, we propose to propagate particles within AOR by prospecting a kernel distribution shape for many reasons. First, we will get denser particles that could give good fitness values. Second, since the movement of the object between two successive frames is small, we ensure a high density of particles in the current generation around the previous location of the target. Finally, we spread a few particles on the edges to guarantee that we have the probability to find the target in case it makes a larger step motion. In this work, we have tested three kernels, Gaussian, Epanechnikov and Triangular. Where their definitions are in the following equations and the values of distribution are between [0 and 1] for computational purposes.

$$f(x,y) = exp\left(-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)\right) \quad (8)$$

$$f(x,y) = \frac{3}{4} * \left(1 - \left(\frac{x^2}{\sigma} + \frac{y^2}{\sigma}\right)\right) \quad (9)$$

$$f(x,y) = \left(1 - ||x-y||\right) \quad (10)$$

In this phase of resampling, we take the coordinates of the best estimate particle as input parameters into the resampling function. In order to generate samples for the next frame within the AOR. Where the best estimate candidate represents the lowest Euclidean distance within our vector of likelihood weights.

The distribution resampling technique is using fields. A field is a region in which we spread a precise number of particles. where its shape and size are depending on the target dimensions. Thus, we adopt an elliptical shape because it suits kernel distribution, and it ensures covering the whole object with particles.

Figure **2** illustrates this definition.

Figure 2: The target shapes

We illustrate the detailed procedure of distribution resampling, starting by kernel distribution then finishing with a set of particle states propagated with respect of distribution form. The main steps are described as follows:

- *Build the distribution*: First, according to the size of the AOR and kernel type we build the distribution, where the values are within 0 and 1.
- *Find the peak of the distribution*: After the previous step, we try to locate its global Maxima and we save its position.
- *Calculate the horizontal and vertical means:* Based on the global Maxima position, we define path from the peak to the null value vertically and horizontally, then we calculate the mean of each direction.
- *Based on the means, we calculate the number of fields with their width and height*: In this step, we specify the perimeters of each field. Since the equation of elliptical shape needs the radius of the x-axis (width) and the radius of the y-axis (height) which will be calculated from horizontal cross-section and vertical cross-section respectively Figure 3 shows a horizontal cross-section of the kernel, we consider it as a table of N elements. Where the global Maxima is presented in yellow, the blue represents the edge of the kernel, and the green represents the elements between the peak of the kernel and its last null value. Firstly, we calculate the number of elements constituting the radius of the first field from the global Maxima. On x-axis we check if the value of the first element is greater than

or equal to the horizontal means, if it is true, then this field will have a width radius of one single element, otherwise we add the next element and the previous element and we check if their sum is greater than or equal to the horizontal means, if this is the case then the width of this field has two elements, otherwise we add a third element to the two previous ones and we iterate again. This process is repeated until we find the length of the first field, so on so forth for the next fields. However, the element of the previous field is not re-used, instead we start from the element following the last element of the previous field. Secondly, we move to the second field and we apply the same process, however, the element of the first field is not re-used, instead we start from the element following the last element of the previous field. In fact, we calculate the width of the field by accumulating the previous field with the elements of the current field.

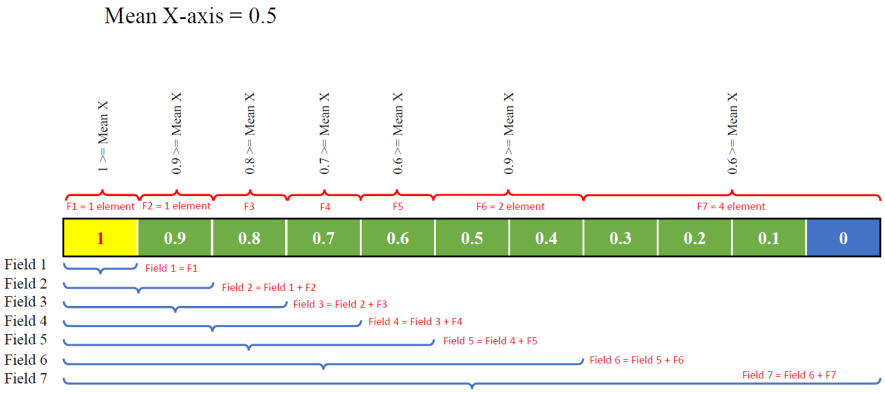


Figure 3: Overview of the field calculation process

- *Calculate the particle's portions on each field*: We calculate the sum of the cross-section of the kernel, where the direction of the cross-section has a negligible impact on the calculation. Then, we calculated for each field its percentage according to the following equation:

$$Pr_{field\ t} = \frac{\sum_1^{N_{field\ t}} V_i}{\sum_1^{N_{cs}} T_i} * 100 - Pr_{field\ t-1} \qquad (11)$$

  Where $Pr_{field\ t}$ and $Pr_{field\ t-1}$ are percentages of the current field and the previous field respectively, $N_{field\ t}$ is the number of elements of the current field, N_{cs} is the number of element of the cross-section table.
- *Propagate particles on each field*: After finding the portion of each field, we try to build these fields using the pair of radius found previously, then propagate their portion inside them.
- *Combine the results*: Finally, we will have several fields of particles superimposed, we combine them in one single structure of particles distributed according to kernel norm, Figure 4, illustrates this process.

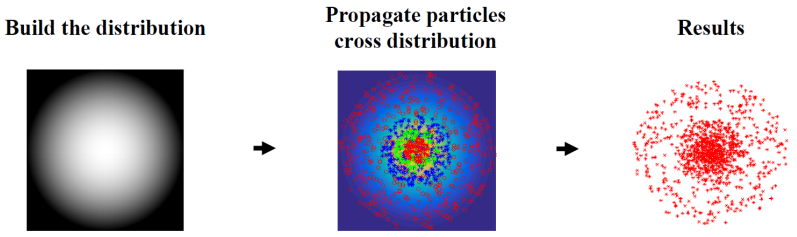


Figure 4: Distribution resampling base on Epanechnikov kernel

2.3 CNN application

In our work we felt to enhance the accuracy of tracking by integrating the deep learning in our process. The 8-layer deep convolution neural network AlexNet is composed of five convolutional layers and three fully connected layers where the output of the last layer is passed to a 1000-way softmax which gives a distribution over the 1000 class labels. We choose this pretrained network AlexNet CNN, because it was trained on more than a million images from

the ImageNet database, it has learned rich feature representations for a wide range of images, furthermore the processing of layers was separated and carried out on two GPUs. In addition, it can classify inputs into 1000 object categories (A. Krizhevsky et al. 2012).

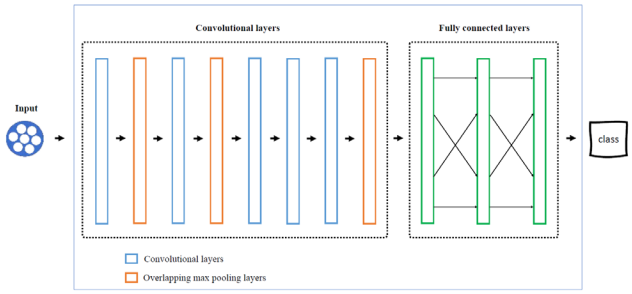


Figure 5: Overview of the AlexNet CNN

In this work, we apply this CNN after we finish calculating the likelihood weights of the particles. First, we select the particles that their weight is less than the similarity threshold set in the input. Then we take their positions and we build their images by taking the same width and height as the target. These images candidates are inputs of the CNN network that will be classified to get the list of labels. Next, we select the best elements by comparing the labels with the label of our original target, and we keep only those who are similar to the target. Then, we finish with a set of candidates that have more chances to be the target in the current frame. Finally, from this set we minimize their weights in order to get the best candidate.

2.4 Proposed tracking algorithm

Many tracking systems rely on object appearance and shapes, and they are the main points that affects their performance. This work presents a new approach based on PF, CNN network and an estimation technique to construct a framework base on a statistical multi-reference histogram of the object appearance. These channels histograms are considered as a feature representation. Our proposed method contains the following steps. After initializing the parameters and locating the target by HOL detector (Y. Azdoud et al. 2015), we predict the positions by using a random uniform model to give for each particle state its coordinates inside the AOR, which we specify it according to the initial location detected. Then we filter state particles using the likelihood weight and CNN output, Next, we pick the greatest state as a result, which will be considered as the new location of the target on the current frame. Lastly, in the resampling step, we apply our proposed distribution resampling technique in order to generate more distinctive and effective particle state using a specific kernel distribution. The process of the proposed algorithm for object tracking is summarized in the following algorithm.

Algorithm 1: Adaptive Particle Filter Tracker Using Convolutional Neural Network

INPUT: kernel type, Target size, Target location, number of particles, Similarity Threshold, CNN classifier.
OUTPUT: Target locations $C_{\{0,\dots,n\}}$ at each frame $I_{\{0,\dots,n\}}$.
Begin

- Locate the target by a detector or locate it manually, get its location and size.
- Classify the target image with AlexNet CNN network.
  $Label_{target} = CNN(Target)$
- Compute the histogram of each color $Tr_{\{rgb\}}$, Normalize the multi-color reference histogram by the size of target.
- Set the AOR according to the target location.
  $[\, border_y \; border_x] = AOR(Tr_y, Tr_x)$
- Create and initialize the particles.

While $t = I_{0\$}: I_n$ **do**

- Predict the position of the particles by distributing them in the AOR in the first frame.

If first frame

- $P^t_{\{1,\dots,N\}} = predicting(border_y, border_x, N)$

End if

- Calculating the likelihood w_t of the particles.
  $w_{\{1,..,N\}}^{t} = likelihood\left(P_t^{\{1,...,N\}}, I_t, size_{window}, hist_{\{rgb\}}^{\{Tr\}}\right)$
- Select particles where their weights < Similarity Threshold.
- Classify the selected particles with CNN.
- Filter particle set and keep those who their labels equal to $Label_{target}$
- Find the new location C_t by minimizing the particle set using their weights.
  $C_t = get\_new\_location\left(P_{\{1,...,N\}}^{t}, w_{\{1,...,N\}}^{t}\right)$
- Update the AOR according to new location.
  $\left[border_y\ border_x\right] = AOR\left(C_y, C_x\right)$
- Save Target location C_t.
- Applying the distribution Resampling on particles.

End

3. Experimental results

In this section, we will show the performance of our CAPF approach. In fact, we have conducted qualitative and quantitative experiments. Indeed, the experiments were implemented over an Intel Core i5-2450M CPU with 2.50GHz and RAM 1600 MHz 12GB DDR3. we have compared our results with three probabilistic methods. Namely, CS, PF, and PSO. Object detection is done manually by marking a bounding box within the first image for experiment purposes. Then, our algorithm runs iteratively to find the best object matching region for every new frame.

3.1 Dataset

Evaluation were conducted on several challenging video sequences. Namely, OTB50, OTB100[1] and LITIV[2]. The datasets are used from object tracking benchmark, furthermore, frequently processed by researchers in computer vision tracking. Obviously, these datasets guarantee a heterogeneous environment and background. Overall, the sequences used can group two or more difficulties such as illumination variation(IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), out-of-view (OV), in-plane rotation (IPR), out-of-plane rotation (OPR), background clutters (BC), and low resolution(LR). The ground truth position has been used in target initialization in the first frame for each compared algorithm.

3.2 Evaluation

In this section our approach and the state-of-the-art methods have undergone several tests is to ensure an impartial comparison. We discuss in detail these tests and their results are mentioned in Table 2, Table 3, Figure 6 and Figure 7. We used Center Location Error (CLE) (W. Tang et al. 2017) as a quantitative evaluation, this criterion allows to calculate the localization errors between the center of the tracking results and the ground truth annotations.

$$CLE = \frac{1}{N_{gr}} * \sum_{i=0}^{N_{gr}} \sqrt{(Xgi - Xri)^2 + (Ygi - Yri)^2} \qquad (12)$$

Where N_{gr} is the total number of images for each frame, (Xg_i, Yg_i) is the position of the ground truth, and (Xr_i, Yr_i) is the tracking result at frame I_i, respectively. According to the equation **(12)**, the algorithm performs better when the CLE value is small. Moreover, as defined in this equation, the size of the bounding boxes is neglected by this quantitative metric.

Furthermore, we also use the Overlap Ratio (OR) (W. Tang et al. 2017) as a second evaluation, it measures the success rate, and it also shows how our proposal performers over sequences in which the object changes its size. The OR is defined by:

$$OR_i = \frac{area(R_i \cap G_i)}{area(R_i \cup G_i)} \qquad (13)$$

[1] OTB dataset homepage, http://www.visual-tracking.net.
[2] LITIV dataset homepage, http://www.polymtl.ca/litiv/en/codes-and-datasets.

Where R_i is the tracked bounding box, G_i is the ground truth bounding box, where $R_i \cap G_i$ represents the intersection, and $R_i \cup G_i$ is the union of two regions. We consider the tracking has succeeded, if the $OR \geq 0.5$.

Moreover, we calculate the Success Ratio (SR) by setting an overlap score r which is defined as the minimum overlap ratio, where can decide whether an output bounding box is correct or not, it is calculated by the following equation:

$$SR = \sum_{i=0}^{N} \frac{u_i}{N_{gr}}, \quad SR \in [0,1] \qquad (14)$$

$$u_i = \begin{cases} 1 \; if \; OR_i \geq r \\ 0 \; if \; not \end{cases}$$

Where N is the frame number, this overlap ratio threshold r who decide whether a corresponding tracking result is good or bad. As defined in equation **(14)**, we calculate the percentage of success location according to the threshold. If SR achieves maximum score and OR is higher in this case the tracker performs great. Otherwise, it performs the worst (K. Hu et al. 2017).

3.3 Results and Discussion

In this section, we discuss the results of the metrics that we presented above. The evaluations were made on 17 sequences, we have made multiple tests with the three kernels distribution to see their influence on the accuracy (Epanechnikov, Gaussian and Triangular), with different particles set sizes between 100 and 600, and we do not get a good result if the number of particles is less than 100 and when we increase the number it does not give a convincing accuracy. Table 1 summarizes all these tests and highlights the performance of the proposed approach with different parameters. Indeed, the overall performance is summarized by the average values in the last row.

As we can see the results of CLE over the chosen sequences, indicates that the change of kernels has a little influence on the accuracy of our tracker. As a result, the overall performance evaluation demonstrates that the CAPF method which implements Epanechnikov kernel with 600 particles takes the first place with 24.04 pixels, while CAPF Gaussian with 200 particles achieves 24.30 pixles which is nearby the first one but with fewer particles and less time and resource consuming. Lastly comes the CAPF with 600 states with Triangular kernel, it achieves the third best overall performance with an average of 24.35 pixels.

Table 1: An overview of the sequences with three kernels Epanechnikov, Gaussian and Triangular

Proposed method	CAPF E			CAPF G			CAPF T		
Nb particles / Sequences	100	200	600	100	200	600	100	200	600
BlurCar3	26.12	25.14	25.56	24.45	25.88	26.26	25.13	23.87	26.43
BlurFace	29.71	32.11	32.51	29.6	29.72	62.86	31.4	30.4	34.46
BlurOwl	35.09	53.71	34.62	58.25	46.93	26.26	29.96	26.85	42.51
CarScale	135.19	30.91	27.42	30.28	29.21	27.33	31.3	30.59	28.21
Dog	29.79	30.81	29.2	29.59	27.57	30.17	28.52	29.16	28.97
Doll	29.74	25.77	25.19	27.72	25.37	25.74	37.06	26.13	25.43
DragonBaby	22.47	29.86	31.06	33.8	31.7	28.54	30.09	29.43	29.03
FaceOcc1	23.55	23.17	21.81	22.38	21.58	20.84	21.59	21.39	21.14
Football1	25.37	28.14	24.44	26.76	24.92	26.3	26.91	25.53	27.23
Girl	20.08	27.93	22.58	81.76	23.19	19.79	23.54	20.66	21.79
Gym	11.44	11.18	10.86	11.13	11.02	10.86	11	10.95	10.85
Jogging	13.21	118.01	12.46	11.91	11.58	11.69	11.96	87.23	11.56
jp1	21.33	25.13	24.68	21.86	21.38	26.03	21.56	23.5	23.26
jp2	23.09	21.96	21.22	20.92	21.16	20.44	22.58	20.49	21.59
Skater	17.2	16.09	15.14	15.49	14.49	13.71	15.71	13.55	13.87
Surfer	32.84	25.25	26.42	26.26	23.84	24.14	28.14	24.16	24.36
wbook	23.93	24.22	23.43	23.09	23.55	22.57	23.8	23.23	23.22
Mean	30.60	32.32	***24.04***	29.13	24.30	24.91	24.72	27.48	24.35

The second evaluation was conducted on three meta-heuristic algorithms. Namely, PF, PSO and CS plus our proposed method APF without CNN with 600 states using Triangular kernel. Table 2 and Table 3 demonstrate respectively the average CLE and the average SR on 17 sequences compared CAPF with 200 particles and Gaussian kernel.
Table 2 and Table 3 shows that our APFT approach without deep learning layer gets good results it has the highest average SR of 56.28% and with 20.06 pixels in average CLE, followed by CAPFG in the second place with performance of 49.66% in average SR and an average CLE of 24.3 pixels. Then comes PSO, PF and CS in the average SR and average CLE respectively.

Table 2: Average success rate (%) tracking of the sequences for CARF 200G, APF 600T, PF, CS and PSO

Sequence	CAPFG	APFT	PF	CS	PSO
BlurCar3	49.3	57.42	54.34	24.09	22.41
BlurFace	63.89	82.76	60.93	67.75	17.16
BlurOwl	62.6	85.42	44.69	51.35	65.77
CarScale	25	35.71	34.92	14.68	24.6
Dog	16.54	16.54	14.96	6.3	6.6
Doll	37.99	52.27	50.54	35.38	60.46
DragonBaby	48.67	52.21	48.67	50.44	68.14
FaceOcc1	93.16	99.55	98.88	69.62	90.36
Football1	20.27	25.68	16.22	9.46	14.86
Girl	48	51.2	29.8	53.6	14.6
Gym	30.64	31.29	31.81	25.55	31.03
Jogging	66.45	67.1	66.45	52.12	54.07
jp1	55.43	46.71	45.23	66.12	85.36
jp2	53.28	72.93	53.71	48.03	95.2
Skater	68.75	78.13	78.75	65	79.38
Surfer	27.39	28.46	3.72	6.91	21.28
wbook	76.94	73.32	71.94	74.18	63.34
Mean	**49.66**	56.28	47.39	42.39	47.92

Table 3: Average errors for target localization (by pixel) in the sequences for CARF 200G, APF 600T, PF, CS and PSO

Sequence	CAPFG	APFT	PF	CS	PSO
BlurCar3	25.88	24.99	26.63	73.33	75.86
BlurFace	29.72	19.57	21.35	26.15	31.3
BlurOwl	46.93	14.04	140.07	27.12	24.84
CarScale	29.21	27.22	31.63	59.05	22.33
Dog	27.57	28.89	31.21	61.33	29.56
Doll	25.37	12.58	20.36	65.74	29.56
DragonBaby	31.7	20.86	33.56	20.52	14.23
FaceOcc1	21.58	18.17	18.05	35.18	27.95
Football1	24.92	27.18	32.51	59.95	26.89
Girl	23.19	18.99	16.57	11.56	27.93
Gym	11.02	10.67	10.98	22.09	19.79
Jogging	11.58	11.27	11.76	22.37	18.49
jp1	21.38	28.75	29.55	38.63	11.72
jp2	21.16	18.33	57.06	50.29	7.05
Skater	14.49	12.73	12.67	22.92	17.1
Surfer	23.84	23.15	188.79	131.32	24.35
wbook	23.55	23.63	24.03	20.31	23.45
Mean	**24.3**	20.06	41.57	43.99	25.43

Furthermore, we highlight the overlap ratio plots, and the success ratio plots in Figure 6 and Figure 7 over 4 sequences. Apparently, in Figure 6, we can see that our tracker CAPF almost maintains a high and stable overlap score along the sequences.

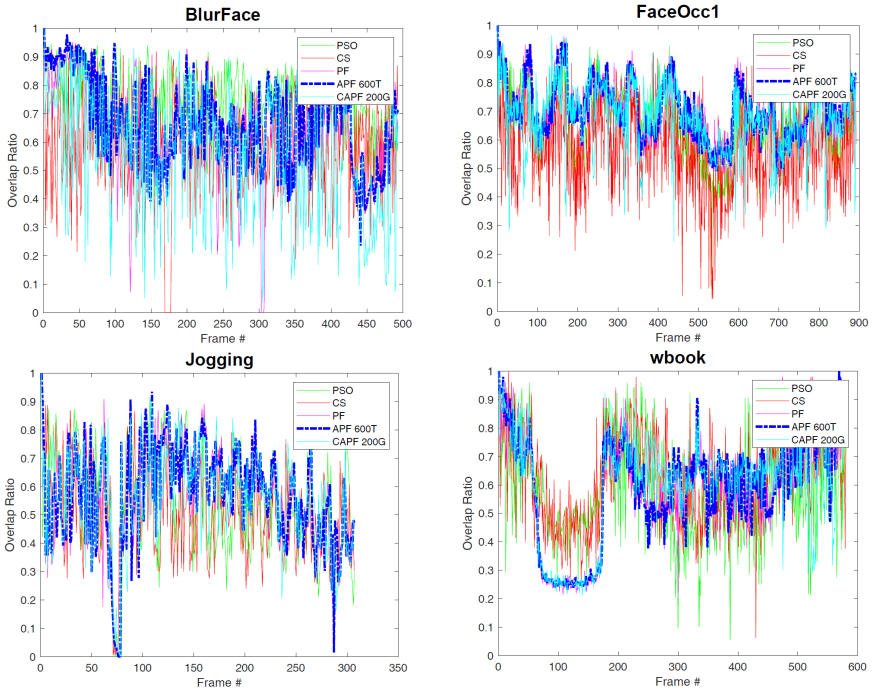
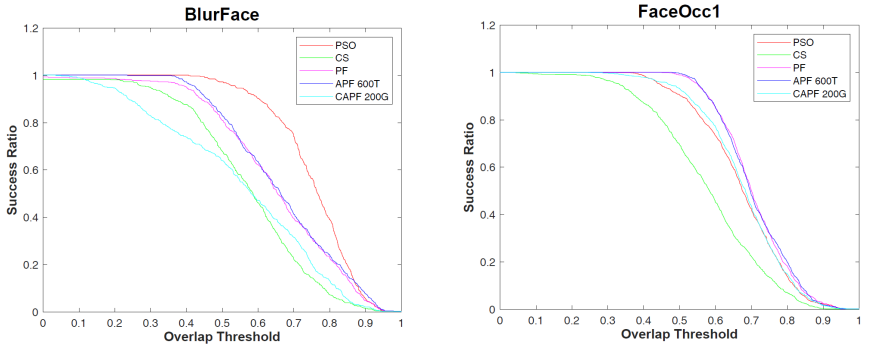


Figure 6: Overlap ratio plots of Tracker CARF 200G, APF 600T, PF, PSO and CS tracker on 4 video sequences

Additionally, we have used the success ratio plot to figure out the average performance of trackers on every sequence from datasets. The success plot gives the percentage of frames where the overlap ratio is higher than a threshold. As illustrated in Figure 7 CAPF approach achieves the good score in terms of this quantitative evaluation versus the state of the art trackers. The above analysis implies that our approach performs accurate and gives stable results.



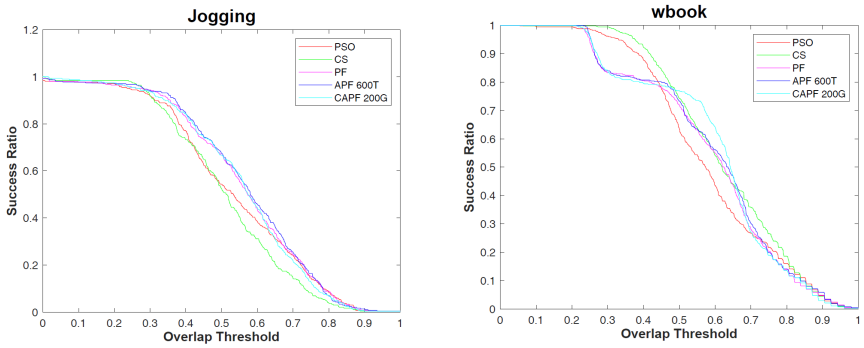


Figure 7: Success ratio plots of Tracker CARF 200G, APF 600T, PF, PSO and CS tracker on 4 video sequences

4. Conclusion

In the literature the deterministic approaches have proven their efficiency in many tracking cases, however they have shown their disadvantages especially at the time running, resource use and on accuracy rate. In this paper, a novel object tracker approach has been proposed by integrating kernel distribution and CNN deep learning layer inside PF probabilistic algorithm, in order to give more intense candidate around the region of interest. Thus, three kernels were examined, namely Epanechnikov, Gaussian and Triangular and the method was experimentally viewed in many sequence scenarios. They showed that the influence of the kernel type on the CLE results has a slight difference and the optimal kernel is the Gaussian with size particle population of n=200. Lastly, the CAPF is compared with three state-of-art algorithms especially PSO, PF and CS plus our proposition APF where the optimal kernel was the Triangular with size particle population of n=600 with without CNN, the examination results show that the success rate of APF tracker outperforms CAPF, PSO, PF, and CS and the target localization error is much higher than these algorithms. However, the CAPF comes in the second place by implementing Gaussian kernel with less particles n=200, it does not achieve the first place, but it has proven its effectiveness in time and resource consuming. In future work we will integrate trained CNN instead of pre-train one.

Acknowledgements

This work, and my previous articles have been supported in part by a grant (CNRST scholarship awarded to Azdoud Youssef N : 3UlT2O15) from the “Centre National de la Recherche Scientifique et Technique,” Ministry of Higher Education and Scientific Research CNRST, Morocco, also this research work is supported by the "SAFEROAD: Métaplateforme de la Sécurité Routière" project under Contract No: 24/2017.

References

S. Tayb, Y. Azdoud, A. Amine, B. Nassih, H. Hachimi and N. Hmina, "HOL, GDCT and LDCT for pedestrian detection," in *International Conference on Signal, Image Processing and Pattern Recognition*, 2016.

D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, p. 564–577, 5 2003.

F. Bousetouane, L. Dib and H. Snoussi, "Improved mean shift integrating texture and color features for robust real time object tracking," *The Visual Computer,* vol. 29, p. 155–170, 3 2013.

S. Jansen, A. Shantia and M. A. Wiering, "The neural-SIFT feature descriptor for visual vocabulary object recognition," in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015.

I. Leichter, "Mean Shift Trackers with Cross-Bin Metrics," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 34, p. 695–706, 4 2012.

L. Davis, "Handbook of genetic algorithms," 1991.

J. Carpenter, P. Clifford and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Proceedings-Radar, Sonar and Navigation,* vol. 146, p. 2–7, 1999.

J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, 1995.

X.-S. Yang, "Bat algorithm for multi-objective optimisation," *International Journal of Bio-Inspired Computation,* vol. 3, p. 267–274, 2011.

X.-S. Yang and others, "Firefly algorithm," *Nature-inspired metaheuristic algorithms,* vol. 20, p. 79–90, 2008.

C. Ma, J.-B. Huang, X. Yang and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE international conference on computer vision*, 2015.

S. Hong, T. You, S. Kwak and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *International conference on machine learning*, 2015.

N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Advances in neural information processing systems*, 2013.

G. Tian, R. Hu, Z. Wang and Y. Fu, "Improved object tracking algorithm based on new HSV color probability model," in *International Symposium on Neural Networks*, 2009.

H. Chu, S. Ye, Q. Guo and X. Liu, "Object tracking algorithm based on camshift algorithm combinating with difference in frame," in *2007 IEEE International Conference on Automation and Logistics*, 2007.

N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, 2005.

A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *2009 World congress on nature & biologically inspired computing (NaBIC)*, 2009.

Y. Azdoud, A. Amine, N. Alioua and M. Rziza, "Pre collision detection system for pedestrian safety based on HOL," in *IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, 2015.

W. Tang, B. Liu and N. Yu, "Deep Scale Feature for Visual Tracking," in *Image and Graphics*, 2017.

K. Hu, E. Fan, J. Ye, C. Fan, S. Shen and Y. Gu, "Neutrosophic Similarity Score Based Weighted Histogram for Robust Mean-Shift Tracking," *Information,* vol. 8, 2017.

Biography

AZDOUD Youssef Ph.D student in Computer Engineering, affiliated to the Systems Engineering Laboratory in the National School of Applied Sciences (ENSA), Ibn Tofail University, Kenitra, Morocco. He received his master's degree in computer engineering: Master of Information Systems Security from the National School of Applied Sciences of Kenitra, Ibn Tofail University, Kenitra, Morocco. He is currently working in the field of Computer Vision.

Prof. Aouatif AMINE has been an associate professor in the Ibn Tofail University - Kenitra, Morocco, since 2014. In July 2010, she joined the ENSA of Kenitra. Ex Vice President of IEEE SPS Morocco Chapter, Secretary General of the Moroccan Association for the Development of Electronics, Electrical Engineering, Computer Science and Automation (AMADEIA). Her areas of research include data classification, object tracking, road safety for intelligent vehicles and pattern recognition.

Prof. Hanaâ Hachimi, Ph.D. in Applied Mathematics and Computer Science and a Ph.D. in Mechanics and Systems Reliability. She is a professor and researcher in the ENSA of Kenitra, Ibn Tofail University. She is the Secretary General at Sultan Moulay Slimane University. She is affiliated with the Systems Engineering Laboratory since October 2013, precisely with the BOSS team. She is responsible and coordinator of the courses: Operational Research, Graph Theory, Statistics, and Probability.