Computers, Materials & Continua
DOI:10.32604/cmc.2022.019001
Article

Tech Science Press

Enhancing the Robustness of Visual Object Tracking via Style Transfer

Abdollah Amirkhani[1,*], **Amir Hossein Barshooi**[1] **and Amir Ebrahimi**[2]

[1]School of Automotive Engineering, Iran University of Science and Technology, Tehran, 16846-13114, Iran
[2]School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, 2308, Australia
[*]Corresponding Author: Abdollah Amirkhani. Email: amirkhani@iust.ac.ir

Received: 29 March 2021; Accepted: 18 May 2021

Abstract: The performance and accuracy of computer vision systems are affected by noise in different forms. Although numerous solutions and algorithms have been presented for dealing with every type of noise, a comprehensive technique that can cover all the diverse noises and mitigate their damaging effects on the performance and precision of various systems is still missing. In this paper, we have focused on the stability and robustness of one computer vision branch (i.e., visual object tracking). We have demonstrated that, without imposing a heavy computational load on a model or changing its algorithms, the drop in the performance and accuracy of a system when it is exposed to an unseen noise-laden test dataset can be prevented by simply applying the style transfer technique on the train dataset and training the model with a combination of these and the original untrained data. To verify our proposed approach, it is applied on a generic object tracker by using regression networks. This method's validity is confirmed by testing it on an exclusive benchmark comprising 50 image sequences, with each sequence containing 15 types of noise at five different intensity levels. The OPE curves obtained show a 40% increase in the robustness of the proposed object tracker against noise, compared to the other trackers considered.

Keywords: Style transfer; visual object tracking; robustness; corruption

1 Introduction

Visual object tracking (VOT), which is a subset of computer vision systems, refers to the process of examining a region of an image in order to detect one/several targets and to estimate its/their positions in subsequent frames [1]. Computer vision includes other sub-branches such as object detection [2], classification [3], optical-flow computation [4], and segmentation [5]. Because of its greater challenges and more versatile applications, further attention has been paid to the subject of VOT, and it has become one of the main branches of computer vision, especially in the last two decades [6].

The applications of VOT in the real world can be classified into several categories, including surveillance and security [7,8], autonomous vehicles [9], human-computer interaction [10], robotics [11], traffic monitoring [12], video indexing [13], and vehicle navigation [14,15].

The VOT procedure is implemented in four steps of i) target initialization, ii) appearance model, iii) motion prediction, and iv) target positioning [15]. In the target initialization step, the object/objects we intend to track is/are usually specified by one/several bounding boxes in the first frame. The appearance model itself comprises the two steps of visual representation (which is used in the construction of robust object descriptors with the help of various visual features) and statistical modeling (which is employed in the construction of mathematical models by means of the statistical learning techniques) for the detection of objects in image frames [16,17]. The target positions in other frames are estimated in the motion prediction step. The ultimate position of a target is determined in the final step by different search methods such as greedy search [18] or by the maximum posterior prediction techniques [19].

In any computer vision application, a correct and precise object tracking operation can be achieved by feeding clean data and images to a system; image corruptions in any form can lead to a drop in system performance and robustness. For example, the presence of atmospheric haze can diminish the performance and accuracy of autonomous vehicles and surveillance systems. Mehra et al. [20] showed that the presence of haze or any type of suspended particles in the atmosphere has an adverse snow noise effect on an image, degrading its brightness, contrast and texture features. Also, these suspended particles may sometimes alter the foreground and background of images and cause the failure of any type of computer vision task (e.g., VOT). In another research, the retrieval of lost information in LIDAR images acquired by autonomous vehicles in snowy and rainy conditions has been investigated. The principle component analysis has been used to improve the obtained images [21].

Other issues also influence the VOT robustness and factors such as the quality of camera sensors, requirements for real-time processing, noise, loss of information during the transfer from 3D to 2D space, and environmental changes. Several factors could cause the environmental fluctuations themselves, e.g., the presence of occlusions, illumination problems, deformations, camera rotation, and other external disturbances [15]. In VOT, the occlusions can occur in three forms: self-occlusion, inter-object occlusion, and occlusion by the background; and for each of these occlusions, four different intensity levels are considered: non-occlusion, partial occlusion, full occlusion, and long-term full occlusion [22].

Modeling an object's motion by means of linear and nonlinear dynamic models is one way of dealing with occlusion in object tracking. Such models can be used to predict the motion of an object from the moment of its occlusion to its reemergence. Other methods such as the silhouette projections, color histogram, and optical flow techniques have also been employed for removing the occlusions and boosting the robustness of object trackers [22]. Liu et al. [23] presented a robust technique for detecting traffic signs. They claimed that all the traffic signs with occlusion of less than 50% could be identified by their proposed method. In another study [24], occlusion problem was solved by using particle swarm optimization as a tracker and combining it with Kalman filter.

In this paper, we have proposed a new method for increasing the robustness and preventing the performance drop of object trackers under different ambient conditions. The presented method can be applied to various types of trackers and detectors and it does not impose a heavy computational load on a system. To substantiate our claim, we have implemented our approach on a visual tracker known as the generic object tracking using regression networks. The main challenge we confronted was the lack of a specific benchmark for evaluating the proposed model and comparing it with other existing algorithms. To deal with this deficiency, we tried to create a benchmark that included most of the existing noises. The main contributions of this work can be summarized as follows:

- Building new training data from previous data through style transfer and combining them.
- Modeling and classifying 15 different types of noises in four groups with five different intensity levels and applying them to the benchmark.
- Applying the proposed method on one of the existing object trackers and comparing the obtained results with those of the other trackers.

It should be mentioned that the presented technique can be applied to multi-object trackers as well. In the rest of this paper, a review of the research activities was conducted to improve image quality and suppress the adverse effects of noise on visual tracker performance has been presented in Section 2. The proposed methodology has been fully explained in Section 3. In Section 4, the obtained results have been given and compared with those of other techniques. Finally, the conclusions and the future work have been covered in Section 5.

2 Common Methods of Maintaining Robustness in Object Trackers

Image enhancement and image restoration are usually known as image denoising, deblocking and deblurring [25]. Yu et al. [25] have defined the image enhancement and restoration process as follows:

> "*A procedure that attempts to improve the image quality by removing the degradation while preserving the underlying image characteristics.*"

The works conducted on the subject of robustness in object trackers can be generally divided into two categories: i) denoising techniques and ii) using deep networks. The denoising techniques inflict a high computational cost. Conversely, the low speed of deep networks in updating the weights has become a serious hurdle in the extensive use of these networks in visual tracking [26]. Each of these methods has been explained in the following subsections.

2.1 Denoising Techniques

The first and simplest method for improving a system's accuracy and performance against noisy data is to use a denoising or image restoration technique. In this approach, before feeding the data to the system, different filters and algorithms are used to remove the noise from a corrupted image and to keep the edges and other details of the image intact as much as possible. Some of the more famous of these techniques in the last decade are the Markov random field [27], block-matching and 3D filtering (BM3D) [28], decision-based median filter (DBMF) [29], incremental multiple principal component analysis [30], histogram of oriented gradients [31], local binary pattern human detector [32], co-tracking with the help of support vector machine (SVM) [33], and the nonlocal self-similarity [34] methods [14]. For example, the BM3D filtering technique has been employed in [28] for image denoising, using the unnecessary information of images.

The standard image processing filters have many problems. For example, the median filter acts on all image pixels and restores them without paying attention to the presence or absence of noise. To deal with this drawback, fuzzy smart filters have been developed. These filters have been designed to act more intensely on the noisy regions of images and overlook the regions with no noise. Fuzzy logic was used in [35] for the first time to improve the quality of color images, remove the impulsive noises, and preserve the image details and edges. Earlier, Yang et al. [36] had employed the heuristic fuzzy rules to enhance the multilevel median filters' performance. Despite the mentioned advantages of the fuzzy smart filters, they have two fundamental flaws:

- **New image corruptions:** The mentioned techniques cause new corruptions in the processed images in proportion to the noise intensity levels. For example, in applying the median filter, the edges in the improved images are displaced in proportion to the window size. As another example, in image denoising with a diffusion filter's help, the image details, especially in images with high noise intensities, fade considerably.
- **Application-based:** The mentioned filters cannot be applied to any type of noise. For example, it was demonstrated in [37] that the Weiner filter performs better on speckle, Poisson, and Gaussian noises than the mean and the median filters.

The denoising techniques improved very little during the last decade. The denoising algorithms were believed to have reached their optimal performance, which cannot be further improved [38]. It was about this time that the emergence of machine learning techniques opened a new door to image quality improvement and denoising.

2.2 Learning-Based Methods

First convolutional neural network (CNN), called LeNet, was presented by LeCun et al. [39] to deal with large data sets and complex inference-based operations. Later on, and since the development of the AlexNet, the CNNs have turned into one of the most common and successful deep learning networks for image processing. Jain et al. [40] have claimed that using the CNNs to denoise natural images is more effective than using other image processing techniques such as the Markov random field. For face recognition in noisy images, Meng et al. [41] have proposed a deep CNN consisting of denoising and recognition sub-networks. Contrary to the classic methods, in which the two mentioned sub-networks are trained independently, these two have been trained as a sequence in the above-mentioned work.

Using a CNN and training it without a pre-learned pattern requires a large training dataset. Moreover, even if such data are available, it would take a long time (tens of minutes) for training a network and reaching the desired accuracy [26]. Considering this matter, a CNN-based object tracker consisting of four layers (two convolutional layers and two fully-connected layers) was presented in [26]. This tracker has been proposed by adding a robust sampling mechanism in mini-batches and modifying the stochastic gradient descent to update the parameters, significantly boosting the execution speed and robustness during training.

Based on machine learning knowledge, a prior image is required by the learning-based methods. Despite the simplicity of these techniques, they have two drawbacks:

- **Extra computational cost**—which is imposed on a system due to the optimization of these techniques
- **Manual adjustment**—which has to be performed because of the non-convexity of these techniques and the need to enhance their performance

To deal with these two issues, discriminative learning methods were proposed. Using a discriminative learning approach, Bhat et al. [42] presented an offline object tracking architecture based on the target model prediction network, predicting a model in just a few optimization steps. Despite all the approaches presented so far, the problem of dependency on prior data still remains. Some researchers have tackled this problem with the help of correlation filters. Using the correlation filters in object tracking techniques to improve performance and accuracy is common; this has led to two classes of object trackers: correlation filter-based trackers (CFTs) and non-correlation filter-based trackers (NCFTs). A novel method of vehicle detection and tracking based on the Yolov3 architecture has been presented in [43]. The researchers have used a vision and image quality improvement technique in this work, which includes three steps: illumination enhancement, reflection component enhancement, and linear weighted fusion. In another study, and based on a sparse collaborative model, Zhong et al. [44] presented a robust object tracking algorithm that simultaneously exploits the holistic templates and local representations to analyze severe appearance changes.

Another method for preventing accuracy loss when using corrupted data is to import these data directly into a training set. Zhao et al. [45] focused on blurred images as a particular case of data corruption. They showed that a low accuracy is obtained by evaluating the final model on blurred images, even by using deeper or wider networks. To rectify this problem, they tried to fine-tune the model on a combination of clear and blurred images in order to improve its performance. A review of the different techniques used for enhancing the robustness of object trackers has been presented in Fig. 1.

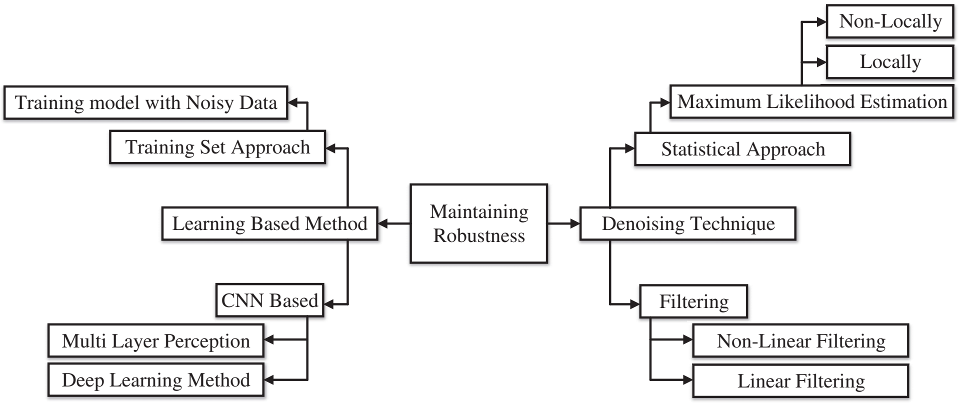


Figure 1: A review of the different techniques presented for boosting the robustness of object trackers

3 Methodology

In this section, we will describe the proposed procedure in full details. At first, we need to introduce an object tracker, which will be used in this work. After selecting a tracker type, the process will be divided into several subsections, which will then be applied in sequence to the model considered.

Our methodology comprises three basic steps. In the first step, we train our network model with a set of initial data and then evaluate it on an OTB benchmark and compare it with other trackers. In the second step, we apply the modeled noises to the benchmark and again evaluate the model on the noisy benchmark. In the third step, we obtain the style transfer of every single training dataset, train the model with a combination of clean and stylized data, apply the trained model on the benchmark of the preceding step, and report the results.

3.1 Selecting an Object Tracker

In the early part of 2016, Held et al. [46] demonstrated that the generic object trackers could be trained in real-time by observing objects' motion in offline videos. In this regard, they presented their proposed model known as the generic object tracking using regression networks (GOTURN). They also claimed this tracker to be the first neural network tracker that was able to complete the learning process at a speed of about 100 frames per second (100 fps). Thus, we decided to implement our method on this tracker and compare the results before and after applying the changes. It should be mentioned that the presented method in this paper can be applied to all the object trackers and detectors that might be affected by various noises. Fig. 2 shows the performances of two of the most common object trackers (the GOTURN and the SiamMask [47]) in the presence of snow noise. Here, we applied the said noise at five different intensity levels on a dataset consisting of 70 image frames and evaluated these two trackers' performances on the noisy images. The figure includes only 18 sample frames (frame numbers 0, 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 60, 65 and 69, starting from top left). As is observed in the figure, The GOTURN tracker fails in frame 30, at the noise intensity level of 3, and the SiamMask tracker fails in frame 52, at the noise intensity of 4. Although the SiamMask tracker shows more robustness than the GOTURN tracker, the tracking operation in both trackers is hampered at different noise intensity levels.

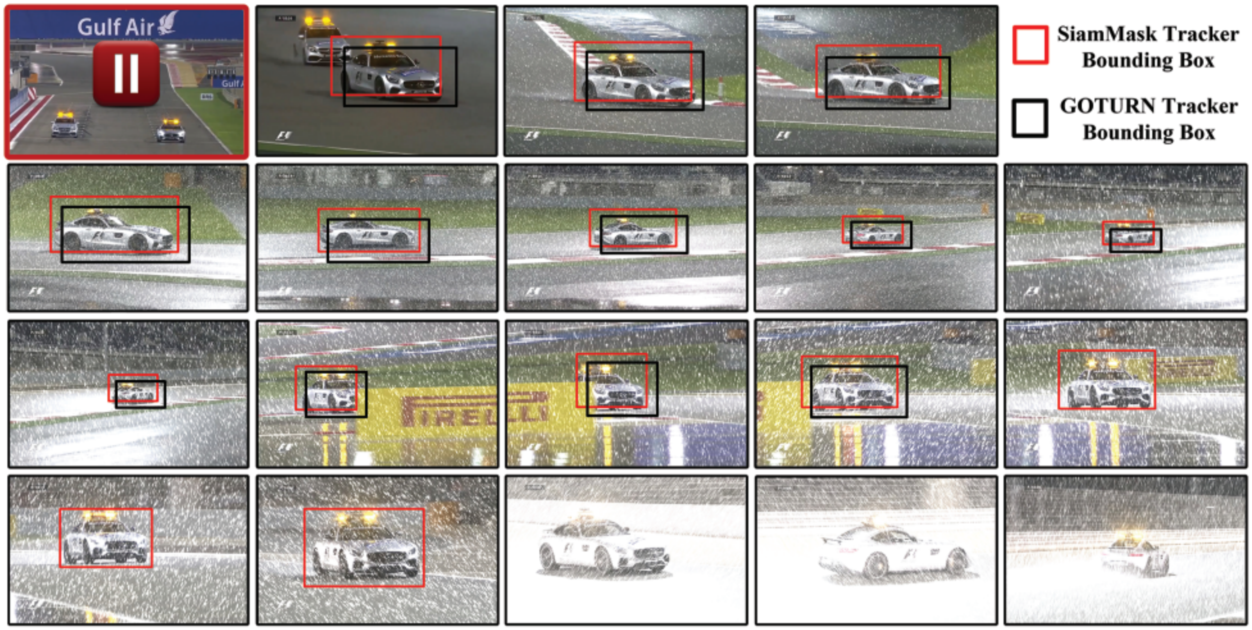


Figure 2: The performances of the GOTURN and the SiamMask trackers on a noisy dataset

3.2 Training/Testing with Clean Data

In this paper, we trained our network with a combination of common images and films. Also, to minimize the error between the predicted bounding box and the ground-truth bounding box, we used the L1 loss function.

The film set: This set contains 314 video sequences, each of which has been extracted from the ALOV300++ dataset [48]. On the average, the 5^{th} frame of each video sequence was labeled according to the position of the object to be tracked, and an annotation file was produced for these frames. The film set was then split into two portions; 20% as the test data and 80% as the training data.

The image set: The first set of images has been taken from the ImageNet detection challenge set, which contains 478807 objects with labeled bounding boxes. The second set of images has been adopted from the common objects in context (COCO) set [49]. This dataset includes 330,000 images in 81 different object categories. More than 200,000 of these images have been annotated, and they cover almost 2 million instances.

3.3 Model Evaluation with Corrupted Data

Most of the benchmarks presented in the literature include either clean data or only specific noises such as the Gaussian noise, while in the real world, our vision is affected by noises of different types and intensities. We needed a noisy benchmark for this work, so we decided to build our own custom benchmark. Note that the mentioned benchmark will only be employed to evaluate the system robustness against different types of noises, and it will never be used to train the proposed object tracker.

In 2019, Hendrycks et al. [50] introduced a set of 15 image corruptions with five different intensities (a total of 75 corruptions). They used it to evaluate the robustness of the ImageNet model in dealing with object detection. The names and details of these corruptions have been displayed in Fig. 3. Based on our viewpoint, we have divided these 15 visual corruptions into the following four categories and interpreted each one by a model of real-world events:

- **Brightness**: We consider the amount of image brightness equivalent to noise and model it with three types of common noises, i.e., the Gaussian noise, the Poisson noise (which is also known as the shot noise), and the impulse noise. For example, the authors in [50] have claimed that the Gaussian noise appears in images under low-lighting conditions.
- **Blur:** Image blurriness is often a camera-related phenomenon, and it can occur via different mechanisms such as the sudden jerking of the camera, improper focusing, insufficient depth-of-field, camera shaking, shutter speed, etc. We modeled these factors' effects on images with four types of blurriness: defocus blur, frosted glass blur, zoom blur, and motion blur.
- **Weather:** One of the most important parameters affecting computer vision systems' quality and reducing their accuracy is the weather condition. We considered a corresponding image corruption for each of the four types of common weather conditions (rainy, snowy, foggy/hazy, and sunny). The snow noise simulates the snowy weather, the frost noise reflects the rainy conditions, the fog noise indicates all the different situations in which a target object is shrouded, and finally, the brightness noise models the sunny conditions and the direct emission of light on camera sensors and lenses.
- **Digital accuracy:** Any type of change in the quality of an image during its saving, compression, sampling, etc., can be considered noise. In this section, such noises will be modeled by the changes of contrast, elastic transforms [51], saving in the JPEG format, and pixelation.

This paper's basic benchmark (the OTB50) includes 50 different sequences such as basketball, box, vehicle, dog, doll, etc. [52]. We apply all the above noises at five different intensity levels (from 1 for the lowest to 5 for the highest intensity) on each of these sequences and build our own custom benchmark. In selecting a benchmark, we must ensure that the data and images in the different sequences of the benchmark don't have any commonality and overlap with the training data; otherwise, the obtained results will be inaccurate and biased and cannot be generalized to other models. For example, the VOT2015 benchmark cannot be used in this paper because of its overlap with the training data.

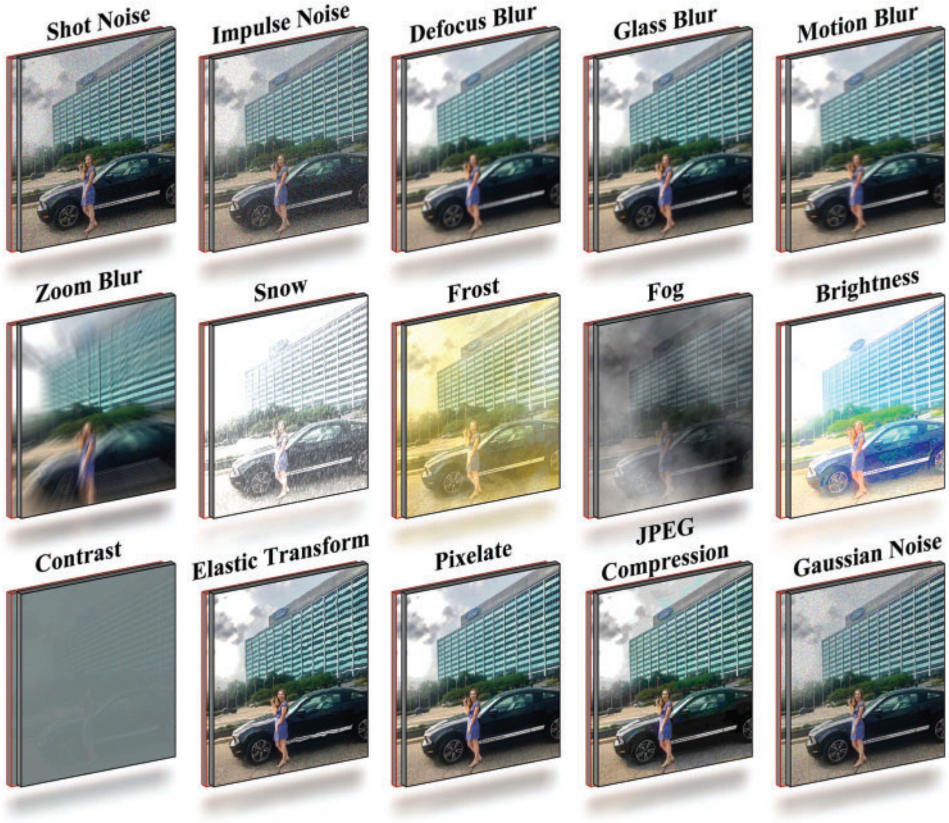


Figure 3: Illustrating 15 types of data corruptions with noise intensity levels of 1 to 5

3.4 Model Training/Testing with Combined Data

One of the applications of deep learning in the arts is the style transfer technique, closely resembling the Deep Dream [53]. This technique was first presented by Gatys et al. [54] in 2016. In this transfer process, two images are used as inputs: the content image and the style reference image. Then, with the help of a neural network, these two images are combined to yield the output image. This network aims to construct a completely new image whose content is provided by the content image and whose style is adopted from the style reference image. This new image preserves the content of the original image in the style of another image.

We employ this technique here and get the style transfer of each of our datasets (with hyperparameter $\alpha = 1$) by means of the adaptive instance normalization (AdaIN) method [55]. Again, as before, an annotation file is created for the new dataset. Finally, we train our object tracker model with a combination of the initial (standard) dataset and the stylized dataset. An example of this transfer and the proposed methodology has been illustrated in Fig. 4. (The style transfer method used for training the proposed model has been taken from https://github.com/bethgelab/stylize-datasets).

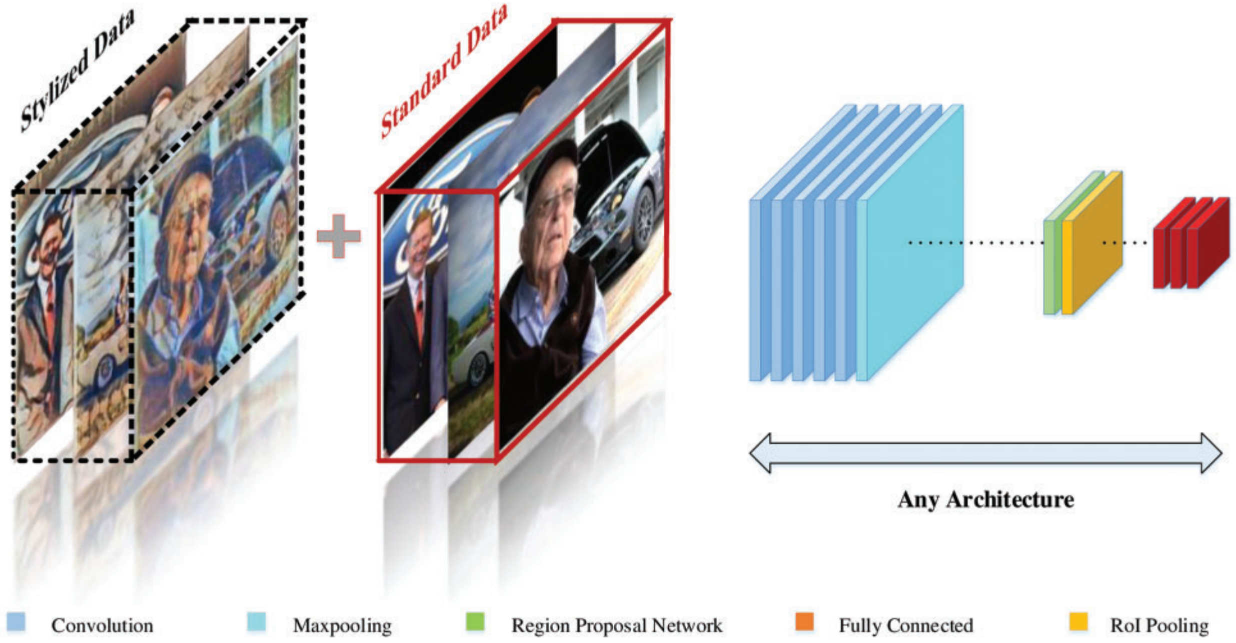


Figure 4: The proposed methodology, along with some samples of style transfer

4 Experimental Results

In order to evaluate the performance of the proposed method and the results achieved by applying it to our custom benchmark, we need to define a specific measure. The most common method used for evaluating the object tracker algorithms is the one-pass evaluation (OPE) approach. In this approach, for each algorithm, the ground-truth of a target object is initialized in the first image frame. Then, the average accuracy or the success rate is reported for the rest of the frames [52]. Considering the many types of noises (15 noise models) and the intensity levels for each noise (5 intensities), a total of 75 graphs will be obtained. Plotting all these graphs and comparing them with one another will not be logical or practical and confuse the reader. Thus, we decided to adopt a criterion that would be appropriate to our approach. In this criterion, the abscissa of each diagram is partitioned into many intervals. The number of these partitions and their intervals are indicated with n and Δx, respectively, so that

$$x_0 = a < x_1 < \cdots < x_{n-1} < x_n = b \tag{1}$$

where a and b represent the lower and the upper bounds of the abscissa and have values of 0 and 1, respectively.

The closer the partitions are, the higher the obtained accuracy. Therefore, we bring n closer to infinity in order to reduce the distance between the partitions. Next, the average value is computed for each of the four noise models (brightness, blur, weather, and digital) and different types of trackers in the OPE diagrams. Thus, we have

$$O\hat{P}E = \frac{1}{N}\lim_{n\to\infty}\sum_{j=1}^{N}\sum_{i=1}^{n} f\left(x_i^*\right), \quad x_i^* \in [x_{i-1}, x_i] \tag{2}$$

$$x_i = a + i\Delta x, \quad \Delta x = (b-a)/n \tag{3}$$

where x is the overlap threshold and f is the success rate. Also, N indicates the number of subsets in each of the four noise models, and its values are 3, 4, 4 and 4 for the brightness, blur, weather, and digital noise types, respectively.

Similar to the Riemann sum theory [56], the above function will converge either to the upper bound (called underestimation in the literature) or the lower bound (called overestimation in the literature), depending on the chosen values of the functions in the partitioned intervals. This notion can also be described by the upper and lower Darboux sum theory. Therefore,

$$O\hat{P}E_{\inf} = L_n\left(f\left(x_i^*\right)\right) = \frac{1}{N}\lim_{n\to\infty}\sum_{j=1}^{N}\sum_{i=1}^{n} \inf f\left(x_i^*\right), \quad x_i^* \in [x_{i-1}, x_i] \tag{4}$$

$$O\hat{P}E_{\sup} = U_n\left(f\left(x_i^*\right)\right) = \frac{1}{N}\lim_{n\to\infty}\sum_{j=1}^{N}\sum_{i=1}^{n} \sup f\left(x_i^*\right), \quad x_i^* \in [x_{i-1}, x_i] \tag{5}$$

Lemma (1): Assuming a large number of partitioned intervals, the underestimated and overestimated values will be equal to each other, and it will be proven that the above function is integrable in the $[a, b]$ interval. Thus

$$for\ n \to \infty: \quad O\hat{P}E_{\inf} = O\hat{P}E_{\sup} = OPE_{new} \tag{6}$$

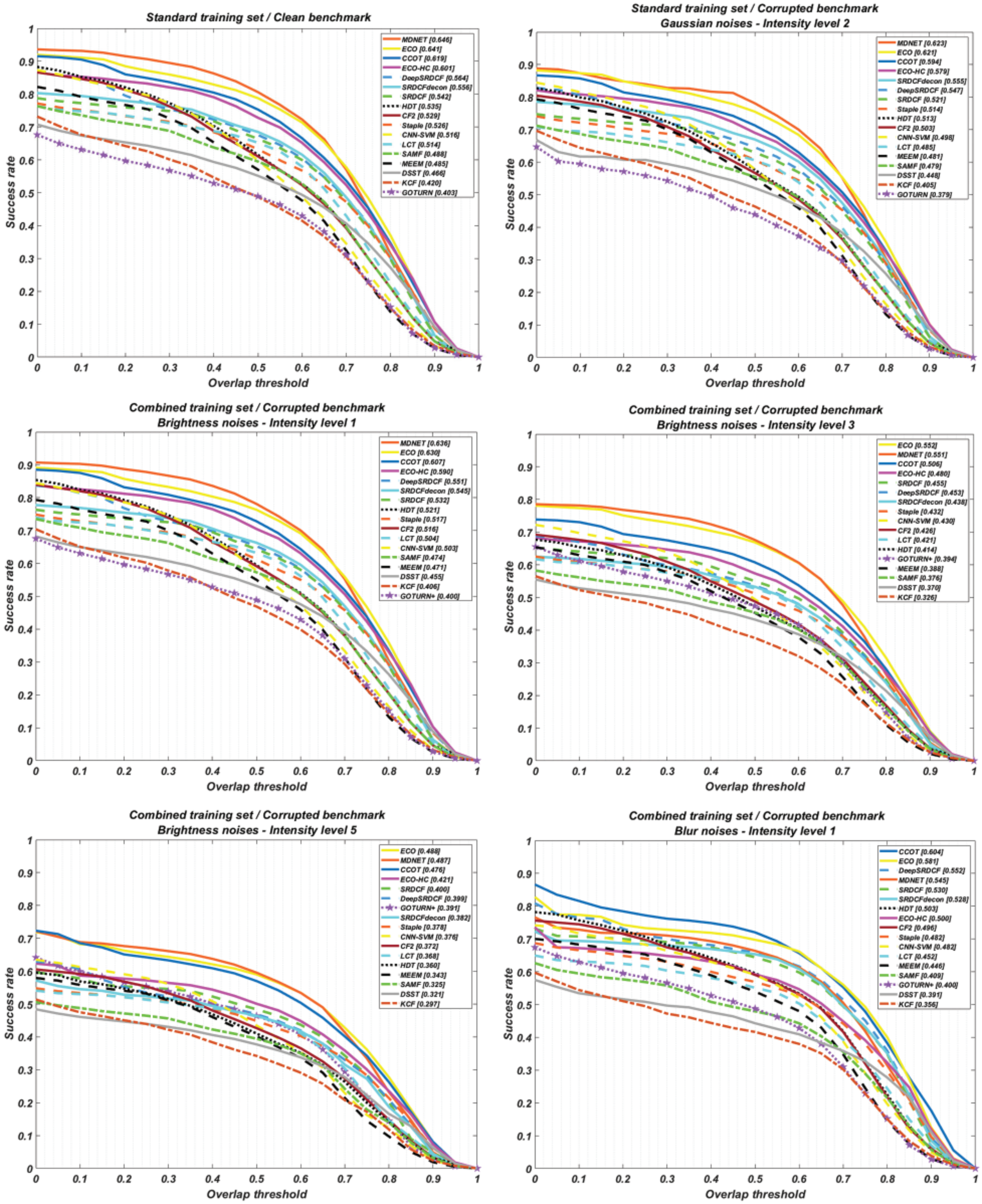
Lemma (2): Using the Riemann sums, the value of a definite integral in the following form can be easily approximated for continuous and non-negative functions:

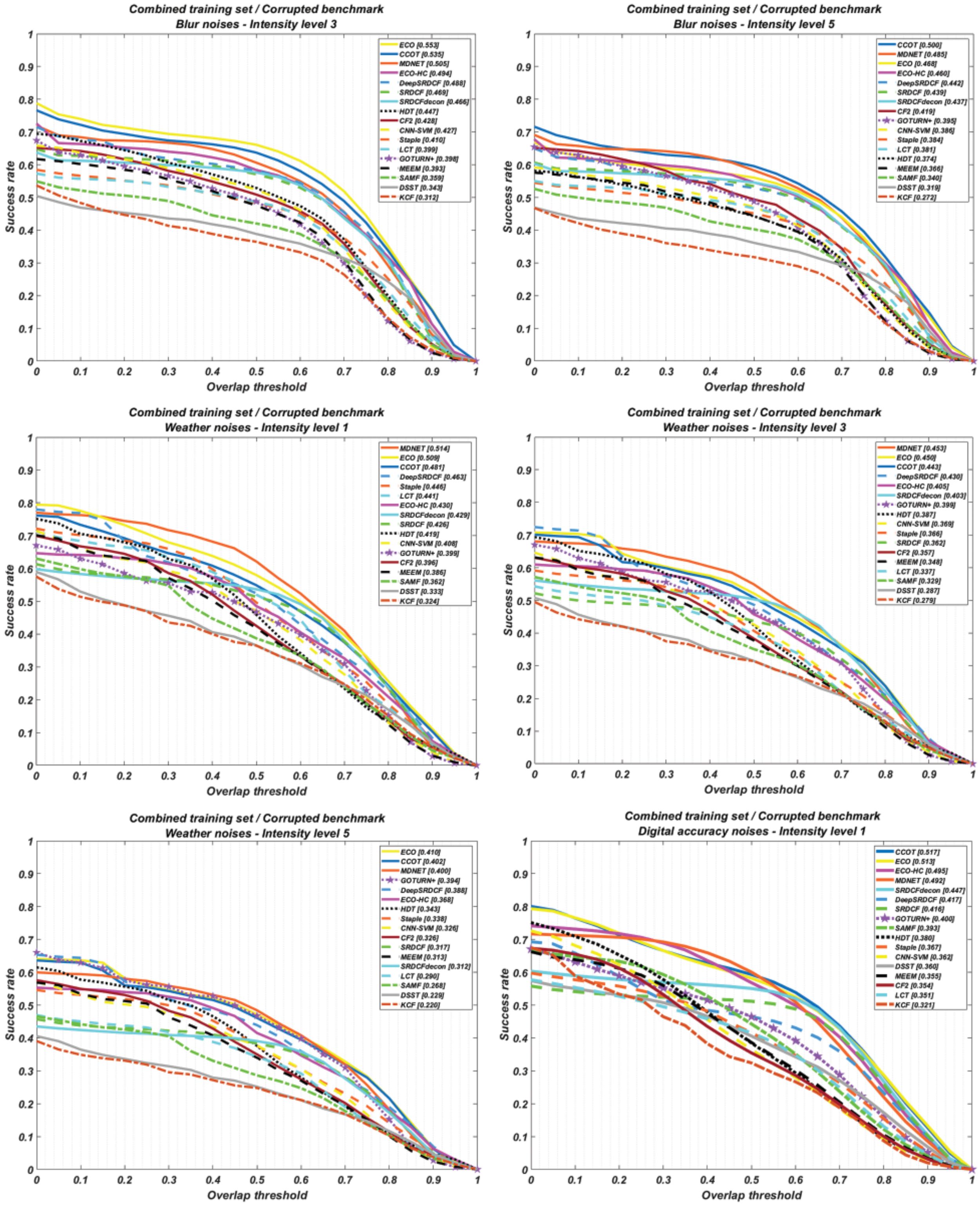
$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{N} f\left(x_i^*\right)(x_i - x_{i-1}), \quad x_i^* \in [x_{i-1}, x_i] \tag{7}$$

Hypothesis (1): A bounded function is Riemann integrable over a compact interval if, and only if, it is continuous almost everywhere. It means that the set of non-continuity points in terms of the Lebesgue size has zero value. This characteristic is sometimes called the "Lebesgue's integrability condition" or the "Lebesgue's criterion for Riemann integrability."

By considering Lemma (2) and Hypothesis (1) and assuming equal lengths for the partitioned intervals, the above equation can be rewritten as

$$n\int_a^b f(x)\,dx \approx \sum_{i=1}^{N} f\left(x_i^*\right) \tag{8}$$


Standard training set / Clean benchmark
Success rate
Overlap threshold
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
MDNET [0.646]
ECO [0.641]
CCOT [0.619]
ECO-HC [0.601]
DeepSRDCF [0.564]
SRDCFdecon [0.556]
SRDCF [0.542]
HDT [0.535]
CF2 [0.529]
Staple [0.526]
CNN-SVM [0.516]
LCT [0.514]
SAMF [0.488]
MEEM [0.485]
DSST [0.466]
KCF [0.420]
GOTURN [0.403]
Standard training set / Corrupted benchmark
Gaussian noises - Intensity level 2
Success rate
Overlap threshold
MDNET [0.623]
ECO [0.621]
CCOT [0.594]
ECO-HC [0.579]
SRDCFdecon [0.555]
DeepSRDCF [0.547]
SRDCF [0.521]
Staple [0.514]
HDT [0.513]
CF2 [0.503]
CNN-SVM [0.498]
LCT [0.485]
MEEM [0.481]
SAMF [0.479]
DSST [0.448]
KCF [0.405]
GOTURN [0.379]
Combined training set / Corrupted benchmark
Brightness noises - Intensity level 1
Success rate
Overlap threshold
MDNET [0.636]
ECO [0.630]
CCOT [0.607]
ECO-HC [0.590]
DeepSRDCF [0.551]
SRDCFdecon [0.545]
SRDCF [0.532]
HDT [0.521]
Staple [0.517]
CF2 [0.516]
LCT [0.504]
CNN-SVM [0.503]
SAMF [0.474]
MEEM [0.471]
DSST [0.455]
KCF [0.406]
GOTURN+ [0.400]
Combined training set / Corrupted benchmark
Brightness noises - Intensity level 3
Success rate
Overlap threshold
ECO [0.552]
MDNET [0.551]
CCOT [0.506]
ECO-HC [0.480]
SRDCF [0.455]
DeepSRDCF [0.453]
SRDCFdecon [0.438]
Staple [0.432]
CNN-SVM [0.430]
CF2 [0.426]
LCT [0.421]
HDT [0.414]
GOTURN+ [0.394]
MEEM [0.388]
SAMF [0.376]
DSST [0.370]
KCF [0.326]
Combined training set / Corrupted benchmark
Brightness noises - Intensity level 5
Success rate
Overlap threshold
ECO [0.488]
MDNET [0.487]
CCOT [0.476]
ECO-HC [0.421]
SRDCF [0.400]
DeepSRDCF [0.399]
GOTURN+ [0.391]
SRDCFdecon [0.382]
Staple [0.378]
CNN-SVM [0.376]
CF2 [0.372]
LCT [0.368]
HDT [0.360]
MEEM [0.343]
SAMF [0.325]
DSST [0.321]
KCF [0.297]
Combined training set / Corrupted benchmark
Blur noises - Intensity level 1
Success rate
Overlap threshold
CCOT [0.604]
ECO [0.581]
DeepSRDCF [0.552]
MDNET [0.545]
SRDCF [0.530]
SRDCFdecon [0.528]
HDT [0.503]
ECO-HC [0.500]
CF2 [0.496]
Staple [0.482]
CNN-SVM [0.482]
LCT [0.452]
MEEM [0.446]
SAMF [0.409]
GOTURN+ [0.400]
DSST [0.391]
KCF [0.356]


Combined training set / Corrupted benchmark
Blur noises - Intensity level 3
Success rate
Overlap threshold
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
ECO [0.553]
CCOT [0.535]
MDNET [0.505]
ECO-HC [0.494]
DeepSRDCF [0.488]
SRDCF [0.469]
SRDCFdecon [0.466]
HDT [0.447]
CF2 [0.428]
CNN-SVM [0.427]
Staple [0.410]
LCT [0.399]
GOTURN+ [0.398]
MEEM [0.393]
SAMF [0.359]
DSST [0.343]
KCF [0.312]
Combined training set / Corrupted benchmark
Blur noises - Intensity level 5
Success rate
Overlap threshold
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
CCOT [0.500]
MDNET [0.485]
ECO [0.468]
ECO-HC [0.460]
DeepSRDCF [0.442]
SRDCF [0.439]
SRDCFdecon [0.437]
CF2 [0.419]
GOTURN+ [0.395]
CNN-SVM [0.386]
Staple [0.384]
LCT [0.381]
HDT [0.374]
MEEM [0.366]
SAMF [0.340]
DSST [0.319]
KCF [0.272]
Combined training set / Corrupted benchmark
Weather noises - Intensity level 1
Success rate
Overlap threshold
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
MDNET [0.514]
ECO [0.509]
CCOT [0.481]
DeepSRDCF [0.463]
Staple [0.446]
LCT [0.441]
ECO-HC [0.430]
SRDCFdecon [0.429]
SRDCF [0.426]
HDT [0.419]
CNN-SVM [0.408]
GOTURN+ [0.399]
CF2 [0.396]
MEEM [0.386]
SAMF [0.362]
DSST [0.333]
KCF [0.324]
Combined training set / Corrupted benchmark
Weather noises - Intensity level 3
Success rate
Overlap threshold
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
MDNET [0.453]
ECO [0.450]
CCOT [0.443]
DeepSRDCF [0.430]
ECO-HC [0.405]
SRDCFdecon [0.403]
GOTURN+ [0.399]
HDT [0.387]
CNN-SVM [0.369]
Staple [0.366]
SRDCF [0.362]
CF2 [0.357]
MEEM [0.348]
LCT [0.337]
SAMF [0.329]
DSST [0.287]
KCF [0.279]
Combined training set / Corrupted benchmark
Weather noises - Intensity level 5
Success rate
Overlap threshold
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
ECO [0.410]
CCOT [0.402]
MDNET [0.400]
GOTURN+ [0.394]
DeepSRDCF [0.388]
ECO-HC [0.368]
HDT [0.343]
Staple [0.338]
CNN-SVM [0.326]
CF2 [0.326]
SRDCF [0.317]
MEEM [0.313]
SRDCFdecon [0.312]
LCT [0.290]
SAMF [0.268]
DSST [0.229]
KCF [0.220]
Combined training set / Corrupted benchmark
Digital accuracy noises - Intensity level 1
Success rate
Overlap threshold
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
CCOT [0.517]
ECO [0.513]
ECO-HC [0.495]
MDNET [0.492]
SRDCFdecon [0.447]
DeepSRDCF [0.417]
SRDCF [0.416]
GOTURN+ [0.400]
SAMF [0.393]
HDT [0.380]
Staple [0.367]
CNN-SVM [0.362]
DSST [0.360]
MEEM [0.355]
CF2 [0.354]
LCT [0.351]
KCF [0.321]

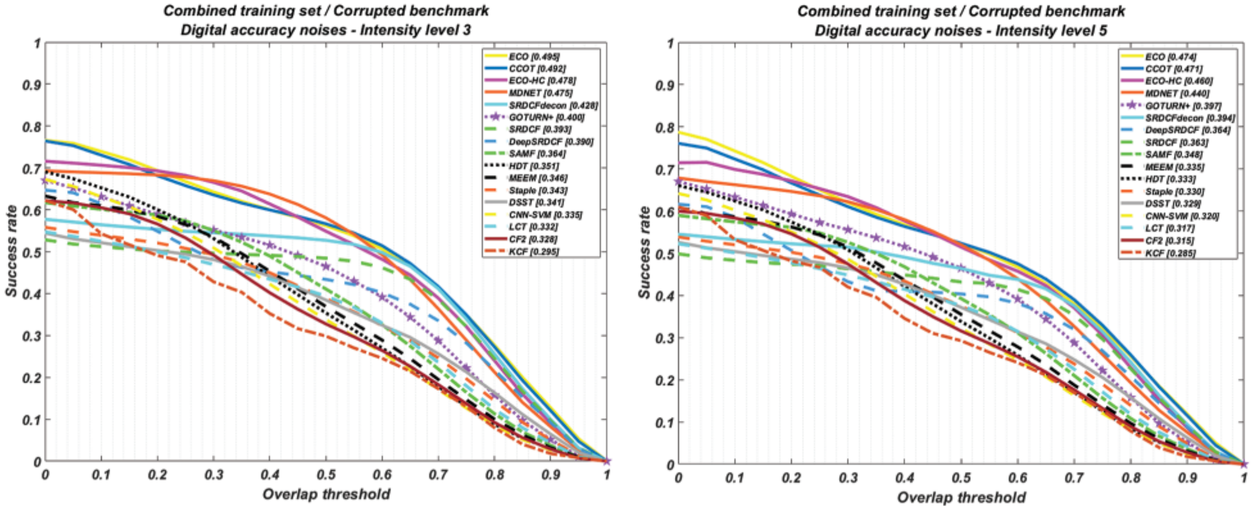


Figure 5: Comparing the performances of different object trackers on the OTB50 benchmark using the proposed criterion

By substituting Eq. (6) into Eq. (8), and according to the transposition property of sigma and integral, we will have

$$OPE_{new} = \frac{n}{N} \lim_{n\to\infty} \int_a^b \sum_{i=1}^{n} f\left(x_i^*\right) dx = \frac{n}{N} \lim_{n\to\infty} \sum_{i=1}^{n} \int_a^b f\left(x_i^*\right) dx \tag{9}$$

The simulation results obtained based on the defined criterion have been displayed in Fig. 5. As is observed in this figure, without altering the structure of a model, the proposed approach has been able to significantly enhance the robustness of the model against different types of noises.

In conclusion, by using the results of Fig. 5, we have calculated the average area under curve (AUC) of each tracker and also calculated the amount of their AUC drop after applying noise in 5 different levels according to the following equations. The results are reported in Tab. 1.

$$AUC_{avg} = L_s = \frac{1}{M} \sum_{i=1}^{M} \int_0^1 f(x)\, dx \tag{10}$$

$$\%AUC_{drop} = (L_0 - L_s) \times 100 \tag{11}$$

where M is equal to the number of noise categories modeled, L_0 is the value of the AUC without noise and s also represents the noise levels in which; $s \in \{1, 2, 3, 4, 5\}$.

Although our work has reached its aims, it has potential limitations. First, due to the combination of clean data and their style transfer, the size of the final data set will be more than doubled, which will increase the network learning time. Second, selecting the proper content layer, style layer, and optimization techniques (e.g., Chimp optimization algorithm [57]), to some extent, might affect the obtained result and performance of the tracker in presence of noise.

According to the results, at the noise level of 1, all trackers showed relatively good robustness, and their AUC drop was less than 18%. At the noise level of 2, a small number of trackers experienced an AUC drop of more than 24%, and the rest of the trackers had a maximum AUC drop of 20%. From the noise level of 3 onwards, there is a significant drop in the trackers' robustness, in which the upper limit of AUC drop between the trackers and in the noise level of 3, 4 and 5 was about 25%, 30% and 40%, respectively. However, the GOTURN trackers training, according to the approach proposed in this paper, showed excellent robustness to all five noise levels, and the maximum AUC drop in all five levels did not exceed 5%.

Table 1: Average AUC of each tracker in the evaluation process on the OTB50 benchmark in five different noise levels

Tracker	L_0	L_1	$\%L_1$	L_2	$\%L_2$	L_3	$\%L_3$	L_4	$\%L_4$	L_5	$\%L_5$
GOTURN+	*0.411*	*0.399*	***2.9***	*0.399*	***2.9***	*0.394*	***4.1***	*0.393*	***4.4***	*0.391*	***4.9***
MDNet	*0.646*	*0.547*	***15.3***	*0.530*	***18.1***	*0.496*	***23.2***	*0.476*	***26.3***	*0.453*	***29.8***
ECO	*0.641*	*0.558*	***12.9***	*0.550*	***14.2***	*0.512*	***20.1***	*0.496*	***22.7***	*0.460*	***28.2***
CCOT	*0.619*	*0.552*	***10.9***	*0.536*	***13.3***	*0.494*	***20.2***	*0.472*	***23.6***	*0.462*	***25.4***
ECO-HC	*0.601*	*0.504*	***16.1***	*0.480*	***20.1***	*0.464*	***22.8***	*0.453*	***24.6***	*0.427*	***28.9***
DeepSRDCF	*0.564*	*0.496*	***12.1***	*0.473*	***16.0***	*0.440*	***22.0***	*0.428*	***24.1***	*0.398*	***29.4***
SRDCFdecon	*0.556*	*0.487*	***12.4***	*0.462*	***16.8***	*0.434*	***22.0***	*0.420*	***24.3***	*0.381*	***31.5***
SRDCF	*0.542*	*0.476*	***12.2***	*0.453*	***16.4***	*0.420*	***22.5***	*0.402*	***25.7***	*0.380*	***29.9***
HDT	*0.535*	*0.456*	***14.8***	*0.430*	***19.7***	*0.400*	***25.2***	*0.370*	***30.9***	*0.352*	***34.2***
CF2	*0.529*	*0.440*	***16.9***	*0.401*	***24.1***	*0.385*	***27.2***	*0.365*	***31.0***	*0.358*	***32.3***
Staple	*0.526*	*0.453*	***13.9***	*0.445*	***15.3***	*0.388*	***26.2***	*0.372*	***29.1***	*0.357*	***32.1***
CNN-SVM	*0.516*	*0.439*	***14.9***	*0.432*	***16.1***	*0.390*	***24.4***	*0.375*	***27.3***	*0.352*	***31.8***
LCT	*0.514*	*0.437*	***15.0***	*0.408*	***20.7***	*0.382*	***25.7***	*0.365*	***28.8***	*0.339*	***34.0***
SAMF	*0.488*	*0.409*	***16.2***	*0.400*	***18.2***	*0.357*	***26.9***	*0.335*	***31.2***	*0.320*	***34.4***
MEEM	*0.485*	*0.414*	***14.6***	*0.391*	***19.3***	*0.369*	***23.9***	*0.360*	***25.8***	*0.339*	***30.1***
DSST	*0.466*	*0.385*	***17.4***	*0.351*	***24.6***	*0.335*	***28.1***	*0.309*	***33.6***	*0.299*	***37.8***
KCF	*0.420*	*0.352*	***16.2***	*0.336*	***19.9***	*0.303*	***27.9***	*0.286*	***31.9***	*0.268*	***36.2***

5 Conclusion and Future Work

Visual noises in images are unwanted and undesirable aberrations, which we always try to get rid of or reduce. In digital images, noises appear as random spots on a bright surface, and they can substantially reduce the quality of these images. Image noises can occur in different ways and by various mechanisms such as overexposure, sudden jerking or shaking of camera, changes of brightness, magnetic fields, improper focusing, and environmental conditions like fog, rain, snow, dust, etc. Noises have negative effects on the performance and precision of computer vision systems such as object trackers. Separately dealing with each of these challenges is an easy task, but it is much more difficult to manage them collectively, which is practically more important. In this paper, a novel method was presented for preserving the performance and accuracy of object trackers against noisy data. In this technique, the tracker model is only trained by a combination of standard training data and their style transfer. To validate the presented approach, an object tracker was chosen from the commonly used trackers available, and the proposed technique was applied to it. This tracker was tested on a customized benchmark containing 15 types of noises at five different noise intensity levels. The obtained results show an increase in the proposed model's accuracy and robustness against different noises than the other considered object trackers. In future work, we intend to apply the Deep Dream technique on our custom training set and train the object tracker with the combination of this dataset and its style transfer. We also intend to test it on both single-object and multi-object trackers. It is worthy of mentioning that this method can be used as a kind of preprocessing block for maintaining robustness in any object detections or computer vision tasks.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, pp. 2544–2550, 2010.

[2] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[3] R. M. Haralick, K. Shanmugam and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.

[4] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1–3, pp. 185–203, 1981.

[5] S. Alpert, M. Galun, A. Brandt and R. Basri, "Image segmentation by probabilistic bottom-up aggregation and cue integration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 315–327, 2011.

[6] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, T. Vojir *et al.,* "The visual object tracking vot2015 challenge results," in *Proc. of the IEEE Int. Conf. on Computer Vision Workshops*, Santiago, Chile, pp. 1–23, 2015.

[7] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed and M. Xu, "A survey on machine learning techniques for cyber security in the last decade," *IEEE Access*, vol. 8, pp. 222310–222354, 2020.

[8] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, S. Chen *et al.,* "Performance comparison and current challenges of using machine learning techniques in cybersecurity," *Energies*, vol. 13, no. 10, pp. 2509, 2020.

[9] J. Janai, F. Güney, A. Behl and A. Geiger, "Computer vision for autonomous vehicles: Problems, datasets and state of the art," *Foundations and Trends® in Computer Graphics and Vision*, vol. 12, no. 1–3, pp. 1–308, 2020.

[10] A. Dix, J. Finlay, G. D. Abowd and R. Beale, *Human-Computer Interaction*, 3rd ed., Halow, UK: Pearson, 2000.

[11] O. G. Selfridge, R. S. Sutton and A. G. Barto, "Training and tracking in robotics," in *Proc. of the Ninth Int. Joint Conf. on Artificial Intelligence*, Los Angeles, California, USA, pp. 670–672, 1985.

[12] K. Kiratiratanapruk and S. Siddhichai, "Vehicle detection and tracking for traffic monitoring system," in *TENCON 2006-2006 IEEE Region 10 Conf.*, Hong Kong, China, pp. 1–4, 2006.

[13] S. W. Smoliar and H. Zhang, "Content based video indexing and retrieval," *IEEE Multimedia*, vol. 1, no. 2, pp. 62–72, 1994.

[14] K. R. Reddy, K. H. Priya and N. Neelima, "Object detection and tracking–a survey," in *2015 Int. Conf. on Computational Intelligence and Communication Networks*, Jabalpur, India, pp. 418–421, 2015.

[15] M. Fiaz, A. Mahmood, S. Javed and S. K. Jung, "Handcrafted and deep trackers: Recent visual object tracking approaches and trends," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–44, 2019.

[16] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick *et al.,* "A survey of appearance models in visual object tracking," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 4, pp. 1–48, 2013.

[17] R. Serajeh, A. E. Ghahnavieh and K. Faez, "Multi scale feature point tracking," in *2014 22nd Iranian Conf. on Electrical Engineering*, Tehran, Iran, pp. 1097–1102, 2014.

[18] H. Pirsiavash, D. Ramanan and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *CVPR 2011*, Colorado Springs, CO, USA, pp. 1201–1208, 2011.

[19] M. M. Barbieri and J. O. Berger, "Optimal predictive model selection," *The Annals of Statistics*, vol. 32, no. 3, pp. 870–897, 2004.

[20] A. Mehra, M. Mandal, P. Narang and V. Chamola, "ReViewNet: A fast and resource optimized network for enabling safe autonomous driving in hazy weather conditions," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2020. https://doi.org/10.1109/TITS.2020.3013099.

[21] M. Aldibaja, N. Suganuma and K. Yoneda, "Robust intensity-based localization method for autonomous driving on snow-wet road surface," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2369–2378, 2017.

[22] B. Y. Lee, L. H. Liew, W. S. Cheah and Y. C. Wang, "Occlusion handling in videos object tracking: A survey," *IOP Conference Series: Earth and Environmental Science*, vol. 18, no. 1, pp. 12020, 2014.

[23] C. Liu, F. Chang and C. Liu, "Occlusion-robust traffic sign detection via cascaded colour cubic feature," *IET Intelligent Transport Systems*, vol. 10, no. 5, pp. 354–360, 2016.

[24] R. Serajeh, K. Faez and A. E. Ghahnavieh, "Robust multiple human tracking using particle swarm optimization and the Kalman filter on full occlusion conditions," in *2013 First Iranian Conf. on Pattern Recognition and Image Analysis*, Birjand, Iran, pp. 1–4, 2013.

[25] G. Yu and G. Sapiro, "Image enhancement and restoration: Traditional approaches," in *Computer Vision: A Reference Guide*. Berlin, Germany: Springer, pp. 1–5, 2019.

[26] H. Li, Y. Li and F. Porikli, "Deeptrack: Learning discriminative feature representations online for robust visual tracking," *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1834–1848, 2015.

[27] S. Z. Li, "Markov random field models in computer vision," in *European Conf. on Computer Vision*, Berlin, Heidelberg, pp. 361–370, 1994.

[28] K. Dabov, A. Foi, V. Katkovnik and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[29] D. A. F. Florencio and R. W. Schafer, "Decision-based median filter using local signal statistics," in *Visual Communications and Image Processing '94*, Chicago, IL, USA, vol. 2308, pp. 268–275, 1994.

[30] T. J. Chin and D. Suter, "Incremental kernel principal component analysis," *IEEE Transactions on Image Processing*, vol. 16, no. 6, pp. 1662–1674, 2007.

[31] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, San Diego, CA, USA, vol. 1, pp. 886–893, 2005.

[32] N. Jiang, J. Xu, W. Yu and S. Goto, "Gradient local binary patterns for human detection," in *2013 IEEE Int. Symp. on Circuits and Systems*, Beijing, China, pp. 978–981, 2013.

[33] M. Tian, W. Zhang and F. Liu, "On-line ensemble SVM for robust object tracking," in *Asian Conf. on Computer Vision*, Berlin, Heidelberg, pp. 355–364, 2007.

[34] Z. Zha, X. Zhang, Q. Wang, Y. Bai, Y. Chen *et al.,* "Group sparsity residual constraint for image denoising with external nonlocal self-similarity prior," *Neurocomputing*, vol. 275, no. 1–4, pp. 2294–2306, 2018.

[35] F. Farbiz, M. B. Menhaj, S. A. Motamedi and M. T. Hagan, "A new fuzzy logic filter for image enhancement," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 30, no. 1, pp. 110–119, 2000.

[36] X. Yang and P. S. Toh, "Adaptive fuzzy multilevel median filter," *IEEE Transactions on Image Processing*, vol. 4, no. 5, pp. 680–682, 1995.

[37] P. Patidar, M. Gupta, S. Srivastava and A. K. Nagawat, "Image denoising by various filters for different noise," *International Journal of Computer Applications*, vol. 9, no. 4, pp. 45–50, 2010.

[38] A. Levin and B. Nadler, "Natural image denoising: Optimality and inherent bounds," in *CVPR 2011*, Colorado Springs, CO, USA, pp. 2833–2840, 2011.

[39] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[40] V. Jain and S. Seung, "Natural image denoising with convolutional networks," *Advances in Neural Information Processing Systems*, vol. 21, pp. 769–776, 2008.

[41] X. Meng, Y. Yan, S. Chen and H. Wang, "A cascaded noise-robust deep CNN for face recognition," in *2019 IEEE Int. Conf. on Image Processing*, Taipei, Taiwan, pp. 3487–3491, 2019.

[42] G. Bhat, M. Danelljan, L. Van Gool and R. Timofte, "Learning discriminative model prediction for tracking," in *Proc. of the IEEE/CVF Int. Conf. on Computer Vision*, Seoul, Korea (South), pp. 6182–6191, 2019.

[43] M. Hassaballah, M. A. Kenk, K. Muhammad and S. Minaee, "Vehicle detection and tracking in adverse weather using a deep learning framework," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2020. https://doi.org/10.1109/TITS.2020.3014013.

[44] W. Zhong, H. Lu and M. H. Yang, "Robust object tracking via sparsity-based collaborative model," in *2012 IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, pp. 1838–1845, 2012.

[45] W. Zhao, B. Zheng, Q. Lin and H. Lu, "Enhancing diversity of defocus blur detectors via cross-ensemble network," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, pp. 8905–8913, 2019.

[46] D. Held, S. Thrun and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conf. on Computer Vision*, Amsterdam, The Netherlands, pp. 749–765, 2016.

[47] Q. Wang, L. Zhang, L. Bertinetto, W. Hu and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, Long Beach, USA, pp. 1328–1338, 2019.

[48] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan *et al.,* "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2013.

[49] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona *et al.,* "Microsoft coco: Common objects in context," in *European Conf. on Computer Vision*, Zurich, Switzerland, pp. 740–755, 2014.

[50] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *Int. Conf. on Learning Representations*, New Orleans, LA, USA, 2019.

[51] G. E. Christensen, "Consistent linear-elastic transformations for image matching," in *Biennial Int. Conf. on Information Processing in Medical Imaging*, Berlin, Heidelberg, Springer, pp. 224–237, 1999.

[52] Y. Wu, J. Lim and M. H. Yang, "Online object tracking: A benchmark," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Portland, Oregon, USA, pp. 2411–2418, 2013.

[53] A. Mordvintsev, C. Olah and M. Tyka, "Inceptionism: Going deeper into neural networks," 2015. [Online]. Available: http://googleresearch.blogspot.com/2015/06/inceptionism-going-deeper-into-neural.html.

[54] L. A. Gatys, A. S. Ecker and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas, Nevada, USA, pp. 2414–2423, 2016.

[55] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. of the IEEE Int. Conf. on Computer Vision*, Venice, Italy, pp. 1501–1510, 2017.

[56] R. Henstock, *The General Theory of Integration*. Oxford, England: Oxford University Press, 1991.

[57] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, no. 1, pp. 113338, 2020.