Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6711561, 15 pages
https://doi.org/10.1155/2021/6711561

Research Article

A Real-Time and Long-Term Face Tracking Method Using Convolutional Neural Network and Optical Flow in IoT-Based Multimedia Communication Systems

Hanchi Ren,[1,2] Yi Hu,[2] San Hlaing Myint,[3] Kun Hou,[1] Xiuyu Zhang,[4] Min Zuo,[1] Chi Zhang,[2] Qingchuan Zhang,[1] and Haipeng Li[5]

[1]*National Engineering Laboratory for Agri-Product Quality Traceability, Beijing Technology and Business University, Beijing 100048, China*
[2]*Swansea University, Swansea SA1 8EN, UK*
[3]*Global Information and Telecommunication Institute, Waseda University, Tokyo 169-8050, Japan*
[4]*Guizhou Academy of Testing and Analysis, Guiyang 550000, China*
[5]*Capinfo Company Ltd., Beijing 100010, China*

Correspondence should be addressed to Kun Hou; houkun@btbu.edu.cn

Received 12 August 2021; Accepted 1 October 2021; Published 31 October 2021

Academic Editor: Deepak Gupta

The development of the Internet of Things (IoT) stimulates many research works related to Multimedia Communication Systems (MCS), such as human face detection and tracking. This trend drives numerous progressive methods. Among these methods, the deep learning-based methods can spot face patch in an image effectively and accurately. Many people consider face tracking as face detection, but they are two different techniques. Face detection focuses on a single image, whose shortcoming is obvious, such as unstable and unsmooth face position when adopted on a sequence of continuous images; computing is expensive due to its heavy reliance on Convolutional Neural Networks (CNNs) and limited detection performance on the edge device. To overcome these defects, this paper proposes a novel face tracking strategy by combining CNN and optical flow, namely, C-OF, which achieves an extremely fast, stable, and long-term face tracking system. Two key things for commercial applications are the stability and smoothness of face positions in a sequence of image frames, which can provide more probability for face biological signal extraction, silent face antispoofing, and facial expression analysis in the fields of IoT-based MCS. Our method captures face patterns in every two consequent frames via optical flow to get rid of the unstable and unsmooth problems. Moreover, an innovative metric for measuring the stability and smoothness of face motion is designed and adopted in our experiments. The experimental results illustrate that our proposed C-OF outperforms both face detection and object tracking methods.

1. Introduction

With the development of AI technology [1–3], IoT [4–7] is receiving more and more attention from academia. It emphasizes that all objects connected to the internet (including people and machines) have unique addresses and communicate through wired and wireless networks and have been deeply integrated into humans' daily life. For example, a doctor can conduct the diagnosis remotely or even complete the surgery via a telemedical system [8, 9]; by collecting personal information, smart devices may provide personal recommendations which are most suitable for him/her [3, 10]; and even the satellite in the universe can be utilized more efficiently for better serving mankind [11]. However, the smarter the humans' life is, the more dangerous the privacy is. Every smart device is "monitoring" you, so personal data protection and privacy-preserved problems should be paid more attention to. Especially the release of GDPR in EU and EEA in 2016, more and more researchers have been digging into privacy-related works [12–19].

The development of IoT-based MCS drives a sharp increase of human face-related techniques, such as face detection, face tracking, and face recognition. Applications of beauty cameras, security access, surveillance and tracking suspect, etc., have been widely used around people's life, for example, smart city and smart campus. It is with no doubt that accurately detecting and tracking faces are essential steps for the aforementioned missions. Additionally, stably and smoothly tracking face bounding boxes from a sequence of continuous images is also required for some special missions in the field of IoT-based MCS, e.g., face biological signal extraction, silent face antispoofing, and facial expression analysis, as stable and smooth face bounding boxes captured along frames can reduce the signal noise significantly.

Regarding the traditional visual methods, a lot of prior face tracking methods [20–25] take tremendous spirits on feature engineering and color spaces. For the long-term tracking of human faces in the unconstrained video, face tracking has been generally treated as common object tracking, e.g., [26] is a typical method which comes from TLD [27, 28] and also is one of the earliest attempts to apply the tracking-by-detection diagram for the face tracking task. Although common TLD can also deal with face tracking work, [26] upgraded it to be more robust even when viewpoints change. In detail, it adapted a frontal face detector from [29] which is the state-of-the-art method at that time. A validator was deployed on the top of the detector outputting confidence that is how the current image patch corresponds to a face. [30] proposed a face tracking approach where optical flow information is incorporated into the *Viola-Jones* face detection algorithm [31]. Its outputs proceed to build a likelihood map where face bounding boxes are extracted. FT-RCNN [32] is an efficient face tracking method based on Faster R-CNN [33]. A tracking branch is conducted into Faster R-CNN and jointly performs face detection and tracking, but its running time cost is expensive.

Face detection methods are eligible to do face tracking. However, face tracking turns to more concentrate on frame-wise face pattern connection. Thus, as for face tracking, the relationships of the patterns between frames are taken into consideration rather than detecting faces in each individual frame naively. In this paper, we present a novel method for super real-time and long-term face tracking by combining CNN and optical flow (see Figure 1). There are three principal components: a cascade lightweight face detector that takes responsibility for generating an initial face bounding box, a face tracker based on optical flow [28], and a face identifier (a very shallow FCN) who provides face confidence for binary classification. The face identifier guarantees that the face tracker does not focus on nonface patch. The optical flow field is always continuous and uniform; the face bounding boxes generated from our method are extraordinarily stable and smooth. Additionally, C-OF can be easily transferred to any other missions which meet the stable tracking requirement, such as object tracking and person reidentification. Overall, we make five main contributions:

(i) We novelly combine lightweight deep CNN with the optical flow to substantially reduce the running time cost, which achieves stable, smooth, super real-time, and long-term face tracking on both CPU and edge computing devices

(ii) Compared with the deep CNN face detection method, C-OF output fairly stable and smooth bounding boxes and enhance the performance of many applications, such as face biological signal extraction, silent face antispoofing, and facial expression analysis

(iii) A lightweight FCN which only contains five convolutional layers is designed for face identification to guarantee the tracking accuracy

(iv) We innovatively design a metric to quantify bounding box stability and smoothness regarding the scale and position changing of bounding boxes. The experimental results illustrate that our proposed C-OF outperforms both face detection and object tracking methods

(v) The implementation of C-OF is released on GitHub (https://github.com/HandsomeHans/C-OF) for those who are interested in further research work and application on the commercial product

The rest of the paper is organised as follows: in the next section, related work on face detection and object tracking is presented. Our method of how to combine optical flow on a CNN face detection method and insert a lightweight FCN to identify the face box is described in Proposed Method. The details of the experiment and discussion on the results are provided in Experiment and Discussion followed by Conclusion.

2. Related Work

2.1. Face Detection. As the huge success of deep learning, traditional methods face a tough situation in some particular missions. However, they still matter and have many advantages that are worth to be learnt from. Commonly, they extract hand-crafted features to train a classifier and then deploy a kind of sliding-window method to locate the face. For example, with the combination of Haar features and AdaBoost [34], Viola and Jones [35] deployed a cascaded face detector, which performs high recognition accuracy and fast running time. Benefitting from [34], many excellent methods [36–38] are proposed afterwards. Felzenszwalb et al. [39] used mixtures of multiscale models to detect an object, which inspires many face detection approaches, e.g., [40–42]. As the aforementioned methods use hand-crafted features, they all have bad generalization. That is to say, in complex scenarios, the performance of those methods slumps sharply.

Since Krizhevsky et al. [43] won the ILSVRC, deep learning and CNN have had explosive progress on vision missions. CMS-RCNN [44], Face R-CNN [45], and FDNet [46] adopt many novel strategies with regard to face detection based on Faster R-CNN [33]. SSD [47] is another commonly used way of face detection. Methods based on SSD

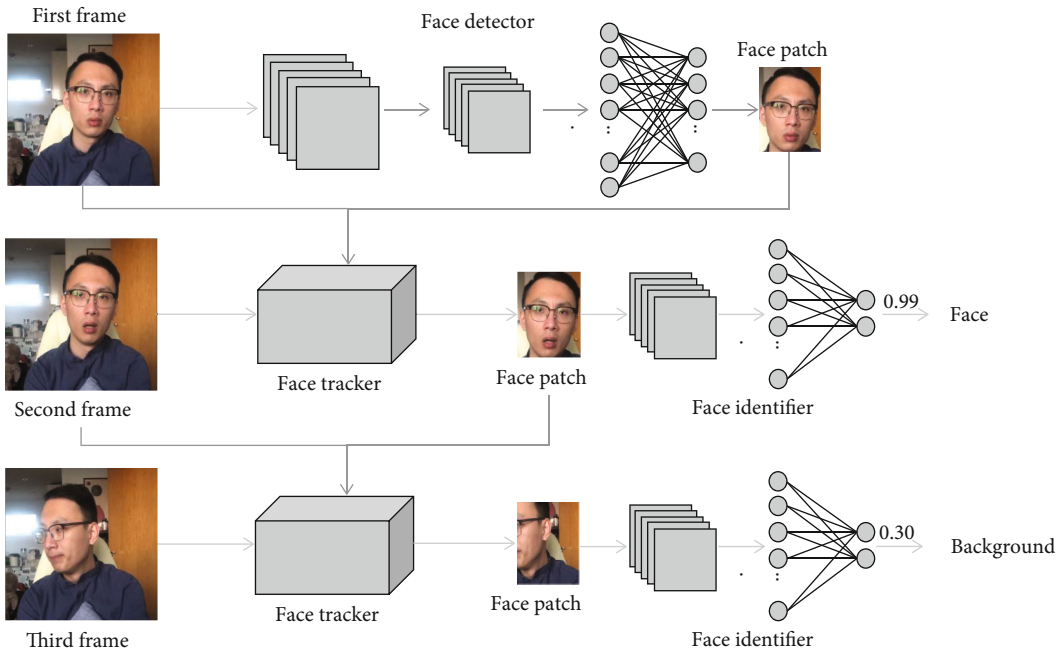

Figure 1: The overall architecture of C-OF.

usually lead to high accuracy and efficiency, e.g., a tiny network is designed in FaceBoxes [48], which attain real-time performance on CPU; FANet [49] uses FPN [50] and merges high-level and low-level features in a low computing cost to train a face detector. On the other hand, cascade CNN methods also show their superiority on the face detection mission. MTCNN [51] consists of three lightweight cascade CNN models to jointly detect face and landmark. PCN [52] upgrades MTCNN by adding an orientation branch to be able to output the face rotated angle. Deng and Xie [53] proposed a nested CNN-cascade learning algorithm that adopts shallow CNN architectures. All these are face detection methods, which all focus on single image representation only. That is why on a continuous video, bounding boxes generated by them are unstable and unsmooth.

2.2. Object Tracking. Face tracking can be considered to be a special category of object tracking. Most scientists implement object tracking methods for face tracking as control experiments. In the beginning, an initial state such as a bounding box of a target object is given, and then, feature extracting and pattern matching methods are conducted in all the subsequent frames. Object tracking has been progressing all the time, as the release of many benchmark datasets and competitions including RGB-T [54], MOT16 [55], and Lasot [56], as well as the development of deep learning. CF [57] has been widely used and inspired a lot of good work in tracking missions. It proved, for the first time, that there is a connection between ridge regression and classical correlation filters. The work accelerated the cost expensive matrix algebra to fast Fourier transforms with $O(n \log n)$ computational complexity. In the meantime, the KCF was first presented and a solution of computing kernels on shifts was proposed as well based on radial basis and dot product. [58] proposed MOSSE which greatly improved the performance of tracking methods with respect to CF. It reduced the computational complexity, and the accuracy increased at the same time. However, it only concerned gray-scale features that this kind of low dimension feature space does not have a good representation. On the other hand, it is unable to adapt object scale variance as it concentrates on translational motion of the center point of the target object between frames and does not take into consideration the scale change of the target object reflected on the screen in the process of moving scale variance. To this end, Danelljan et al. proposed DSST [59] making an improvement on MOSSE by deploying fHOG [39] features instead of gray-scale features to increase the dimension of features from 2 to 28. What is more, the object scale variance is concerned in DSST.

Apart from traditional object tracking methods, CNN-based methods have had great progress and outperform traditional methods a lot on the public benchmarks. [60] introduced a generic object tracking network using a regression mechanism by watching videos offline of objects moving in the world. To be specific, the regression-based tracking network only requires a single feed-forward pass through the network to directly regress the location of the target object. Zhu et al. [61] made progress on Siamese networks, which conduct tracking through similarity comparison strategy, by learning distractor-aware Siamese networks for accurate and long-term tracking. MDNet [62] is one of the most successful generic object tracking methods. It consists of a shared CNN, which is trained on a large set of videos with tracking ground truths, for feature representation extraction. After training, all the branches of domain binary classification layers are replaced. Then, the model was fine-tuned online during tracking to adapt to the new domain. Regarding bounding box regression, they set up an online training linear model to generate the final bounding box.

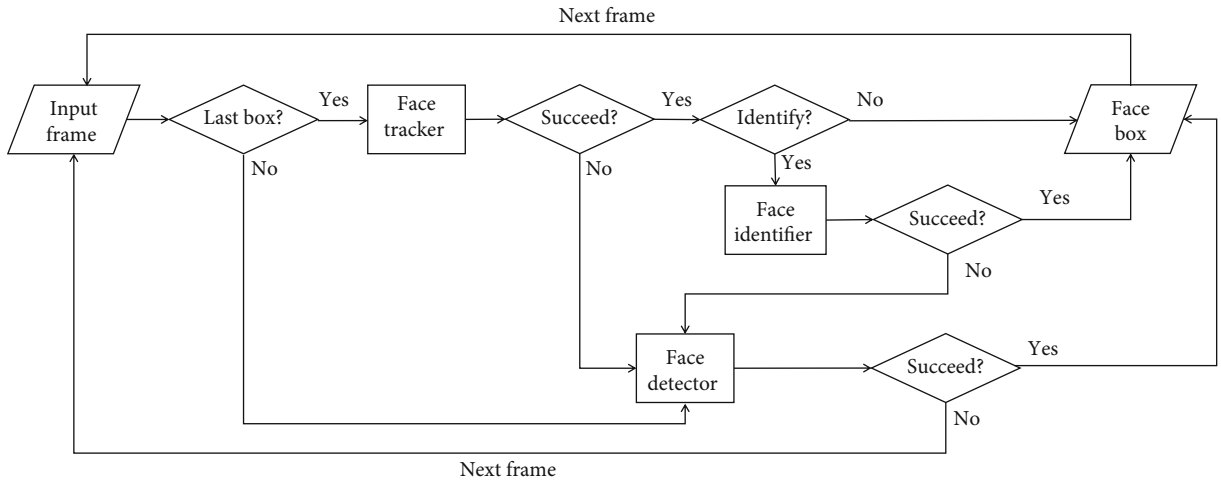

Figure 2: The proposed C-OF face tracking system.

Table 1: Architectures of cascade convolutional networks. "k" stands for kernel size, "s" means stride, and "p" is padding number.

Stage 1	
Input	Color-scale image
Convolution	Outputs: 10, k: 3×3, s: 1, p: 1
MaxPooling	k: 3×3, s: 2
Convolution	Outputs: 16, k: 3×3, s: 1, p: 1
Convolution	Outputs: 32, k: 3×3, s: 1, p: 1
Convolution	Outputs: 6, k: 1×1, s: 1, p: 1
Stage 2	
Input	Color-scale image
Convolution	Outputs: 28, k: 3×3, s: 1, p: 1
MaxPooling	k: 3×3, s: 2
Convolution	Outputs: 48, k: 3×3, s: 1, p: 1
MaxPooling	k: 3×3, s: 2
Convolution	Outputs: 64, k: 2×2, s: 1, p: 1
Dense	Outputs: 128
Dense	Outputs: 6
Stage 3	
Input	Color-scale image
Convolution	Outputs: 32, k: 3×3, s: 1, p: 1
MaxPooling	k: 3×3, s: 2
Convolution	Outputs: 64, k: 3×3, s: 1, p: 1
MaxPooling	k: 2×2, s: 2
Convolution	Outputs: 128, k: 2×2, s: 1, p: 1
Dense	Outputs: 256
Dense	Outputs: 6

3. Proposed Method

The details of the proposed C-OF face tracking method are shown in Figure 2. It consists of three principal components: face detector, face tracker, and face identifier. In the following parts of this section, the first part presents the lightweight face detector. In the second part, the implementation logistics of optical flow are given. The face identifier is provided in the last part.

3.1. Lightweight Face Detector. Same with other face tracking methods, a face detector is essential to figure out an initial face bounding box. We adopt a cascade CNN referring to [51], which is only for face detection without facial landmark prediction. In the first stage, a FCN [63] is deployed to obtain the candidate face bounding boxes. As the networks output not only vast bounding boxes but also confidence density for binary classes of face and background, we predefine a threshold to filter some boxes which have low confidence. Then, highly overlapped boxes are merged by NMS. In stage two, the candidate faces cropped from the input image are fed into the second CNN. Abundant false positive faces are dropped out, and NMS is conducted again. In the last stage, the output is generated the same way as stage two. After NMS, we have the final face box. In our face detector, each part takes its own attention on tackling the detection problem. The first part more focuses on outputting vast face candidates. Then, for the second part, it has to filter false positive faces which means this part is concentrated on face identification. The last part not only focuses on face identification but also puts a lot of attention on box regression. More details of the three CNNs are shown in Table 1.

3.2. Optical Flow Face Tracker. Once the initial face bounding box is obtained, it comes into the tracking part. For the general face detection methods, every frame is handled separately, and there is no more temporal information taken considered, so face bounding boxes perform to be very unstable. The sharp shaking of face bounding boxes makes it more difficult to tackle the problem of critical face analysis missions. To this end, a Median-Flow tracker is deployed and collaborates with a bounding box regression module to locate the position of the face in the current frame with respect to the last frame. Basically, a grid of 10×10 points is uniformly selected from the last face patch, namely, P^l.

Table 2: Architectures of face identifier. "k" stands for kernel size, "s" means stride, and "p" is padding number.

Input	Color-scale image
Convolution	Outputs: 28, k: 3×3, s: 1, p: 1
MaxPooling	k: 3×3, s: 2
Convolution	Outputs: 48, k: 3×3, s: 1, p: 1
MaxPooling	k: 3×3, s: 2
Convolution	Outputs: 64, k: 2×2, s: 1, p: 1
Convolution	Outputs: 128, k: 1×1, s: 1, p: 1
Convolution	Outputs: 2, k: 1×1, s: 1, p: 1

Table 3: Stability values with respect to Equation (3) for MDNet, MTCNN, and C-OF on aforementioned four videos with different conditions.

Method	Active camera				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	4.106844	4.387077	5.088164	5.704352	4.977256
MTCNN	6.679212	6.522255	6.166027	8.261213	9.037096
C-OF (ours)	2.936315	3.350025	3.372295	3.907079	3.864539
Method	Active human				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	3.045595	2.81129	3.305667	3.098207	3.322302
MTCNN	4.064902	3.55413	3.513855	4.050621	3.746015
C-OF (ours)	0.851283	0.993135	0.728653	1.208135	1.403328
Method	Static human				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	1.425355	1.47768	1.453236	1.518395	2.260055
MTCNN	1.994141	2.51177	2.318275	2.333235	2.237117
C-OF (ours)	0.310882	0.308686	0.295996	0.245097	0.321575
Method	Active illumination				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	1.961921	2.130076	2.3661	3.036336	2.243076
MTCNN	2.831262	2.758897	3.116406	4.073383	2.963807
C-OF (ours)	0.452622	0.579371	0.696979	1.006401	0.392867

Then, the motions of these points between the current frame and last frame are estimated by the pyramidal Lucas-Kanade tracker [64] in two directions. $P^f = [p_1^f, p_2^f, \cdots, p_N^f]$ and $P^b = [p_1^b, p_2^b, \cdots, p_N^b]$, where p_n^f and p_n^b represent the predicted points in forward (from last to current fame) and backward (from current to last frame) directions, respectively, from pyramidal Lucas-Kanade tracker, where N is the number of points. In other words, the forward predicted points P^f is calculated from the last frame, current frame, and last frame's uniformly selected points P^l; the backward predicted points P^b are from the two frames and the current frame's uniformly selected points P^c. As for the pyramidal Lucas-Kanade tracker, we set the size of the search window at each pyramid level to be 4 by 4, and two pyramid levels are used. The termination criteria of the iterative search algorithm are set to have a 20 maximum iteration number and 0.03 convergence threshold. In order to filter the points P^l and P^f to estimate the offset of face patch, normalization cross-correlation between last and current frames is performed firstly. Any point whose value is smaller than the median similarity value is dropped out. Then, the median value of P^b is used to further filter points in sets P^l and P^f. The point whose value is larger than the median value is dropped out. While having the filtered points $\widehat{P}^l$ and $\widehat{P}^f$, the coordinate offset of the current face box against the last face box is the median *Euclidean* distance between $\widehat{p}_n^l$ and $\widehat{p}_n^f$:

$$\text{offset}\left(\widehat{P}^l, \widehat{P}^f\right) = \text{median}\left(\left|\left|\widehat{p}_m^l - \widehat{p}_m^f\right|\right|\right); \forall m \in M, M \leq N. \quad (1)$$

Hence, from the last face box and coordinate offsets, we have the current face box. In this way, face boxes are significantly stable and smooth along with frames than those generated from the common face detection method.

Table 4: Smoothness values with respect to Equation (5) for MDNet, MTCNN, and C-OF on four videos with different conditions.

Method	Active camera				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	0.152312	0.146476	0.303847	0.270567	0.088137
MTCNN	1.071629	0.683341	0.650429	0.581538	0.656829
C-OF (ours)	0.005137	0.011361	0.010636	0.005119	0.009432
Method	Active human				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	0.678967	0.491966	0.586164	0.699405	0.402297
MTCNN	1.145627	0.847167	0.93854	0.977966	0.971969
C-OF (ours)	0.001103	0.003444	0.001304	0.004872	0.021336
Method	Static human				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	0.393428	0.43518	0.444789	0.521464	0.612867
MTCNN	0.5484	0.673217	0.729767	0.625556	0.566896
C-OF (ours)	0.004908	0.000072	0.0	0.000008	0.0
Method	Active illumination				
	Clip 1	Clip 2	Clip 3	Clip 4	Clip 5
MDNet	0.417814	0.3991	0.564509	0.531881	0.48117
MTCNN	1.020253	0.616373	0.805697	1.110985	1.045736
C-OF (ours)	0.010921	0.000995	0.0	0.007117	0.000901

3.3. Face Identifier. Face recognition is in general brought into vast focus by the success of CNN on vision missions, and various methods have been presented already. In order to filter the representation in the bounding box generated by the optical flow face tracker, we show a very simple face identifier that works very well in distinguishing the background. Table 2 shows the architecture of the proposed face identifier which has a minor difference against the second part of the aforementioned face detector. The second part of the face detector outputs six values, four for face box coordinate and two for face and background probabilities. We

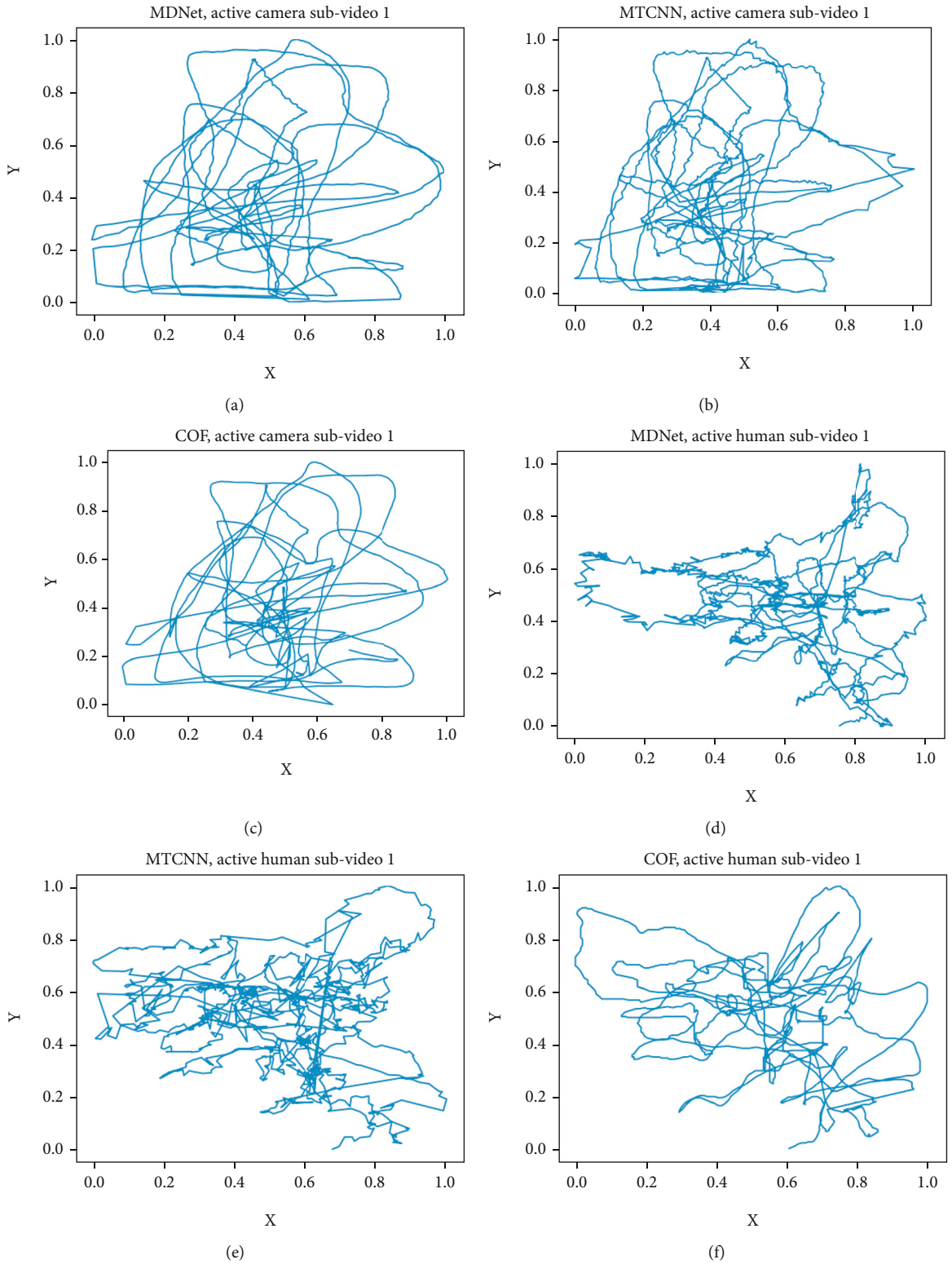

Figure 3: Continued.

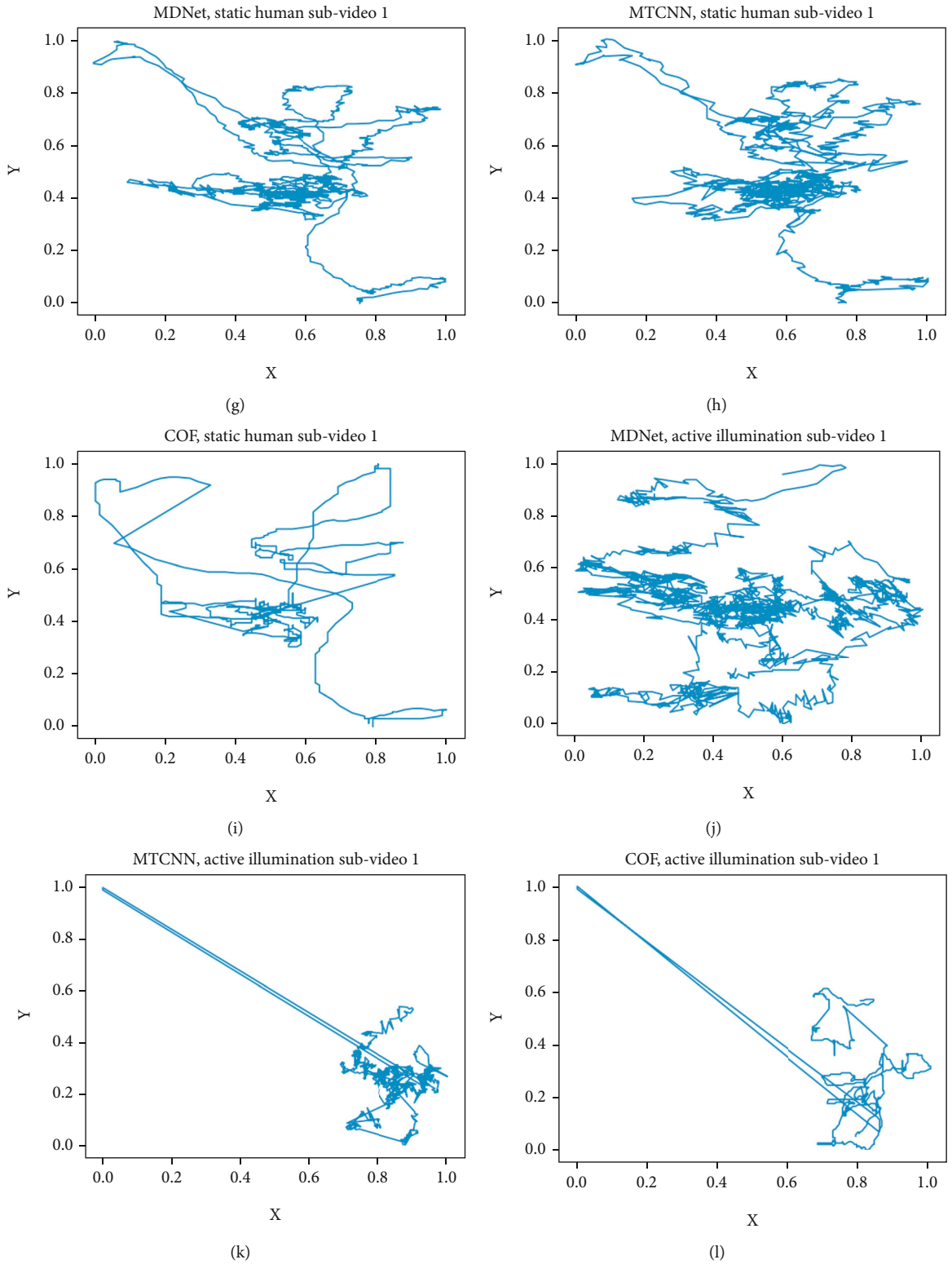

Figure 3: Selected visualization for the motion tracking of center point in the bounding box.

change the output layer to only generate binary probabilities. As the face in a sequence of frames may vary a lot, the face tracker may fail in tracking the face and output a wrong object. So, the face identifier is essential; the main goal of it is to filter the false positive candidates to fit well with its motivation. Therefore, we change the number of output values to two, and the probability distribution for face and background is obtained by a *Softmax* layer which follows the last convolutional layer. In order to make the face identifier capable of dealing with different sizes of face bounding

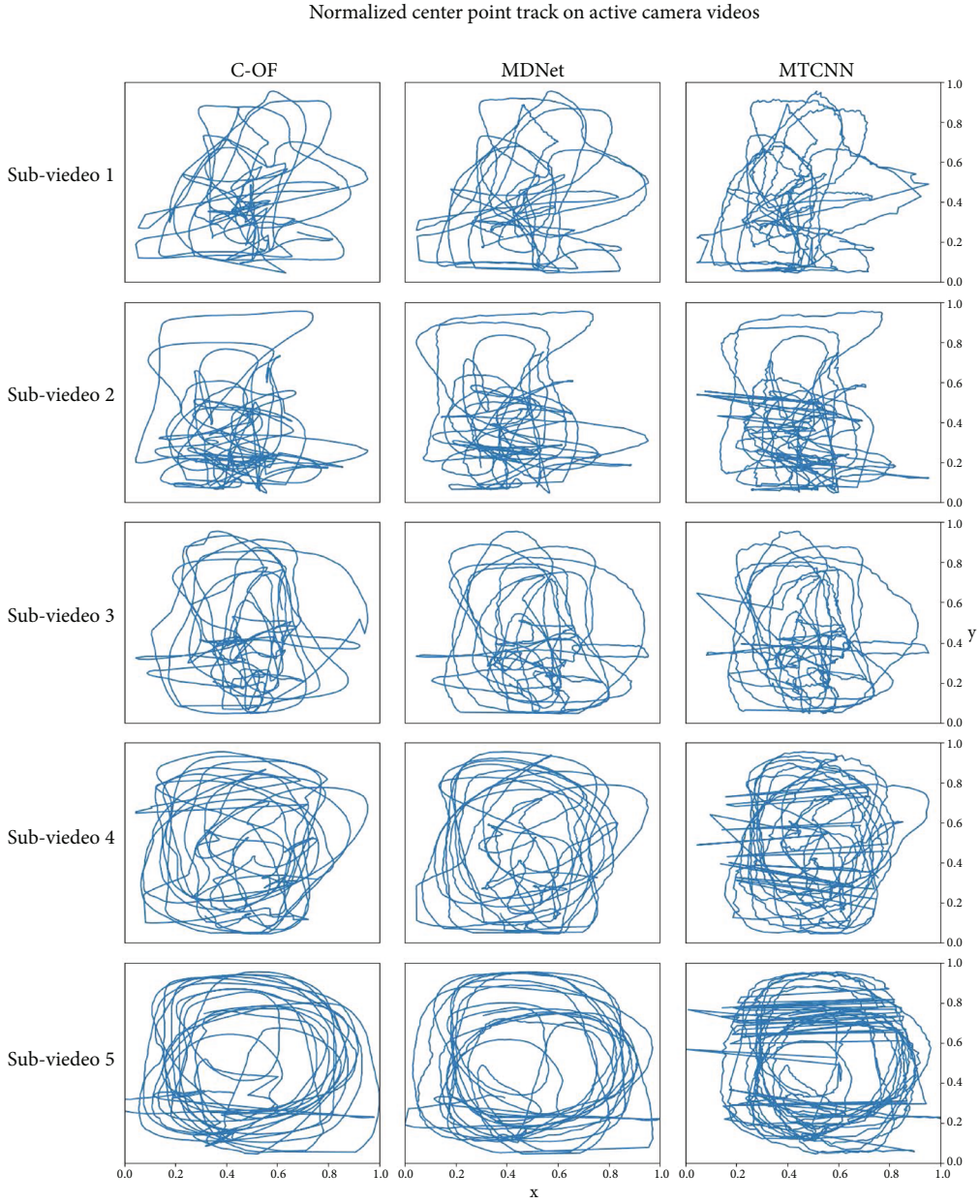

Figure 4: Center point motion tracking on active camera videos.

boxes, we replace two fully connected layers with two convolutional layers to make the model be a FCN [63]. FCN is a specially designed neural network for semantic segmentation. By replacing all the fully connected layers with convolutional layers, FCN breaks the limitation of fixed input size. That is because a fully connected layer needs an input with a fixed dimension to fit with its weights, while a convolutional layer with a 1 by 1 kernel size has no need of fixing the input's dimension. Let the kernel number in a convolutional layer be the same as the hidden node number in a fully connected layer; then, this convolutional layer can replace the fully connected layer directly and break the limitation of fixed input size. Its ability to deal with the arbitrary scale of input images has been spread to other vision missions, such as object classification and detection.

4. Experiment and Discussion

4.1. Experimental Setting and Dataset. A number of experiments on face tracking were conducted to evaluate our proposed C-OF. Other than C-OF, we reproduced MTCNN [51] and MDNet [62] for comparing the tracking performance. For Python implementation, all neural network models are implemented using PyTorch [65] framework, and the source code has been made publicly available (https://github.com/HandsomeHans/C-OF) for reproducing the results. The hyperparameters for MDNet and MTCNN are the same as their official implementation. We also implement C-OF via C++ with ncnn (https://github.com/Tencent/ncnn) which is a high-performance neural network inference computing framework optimized for mobile

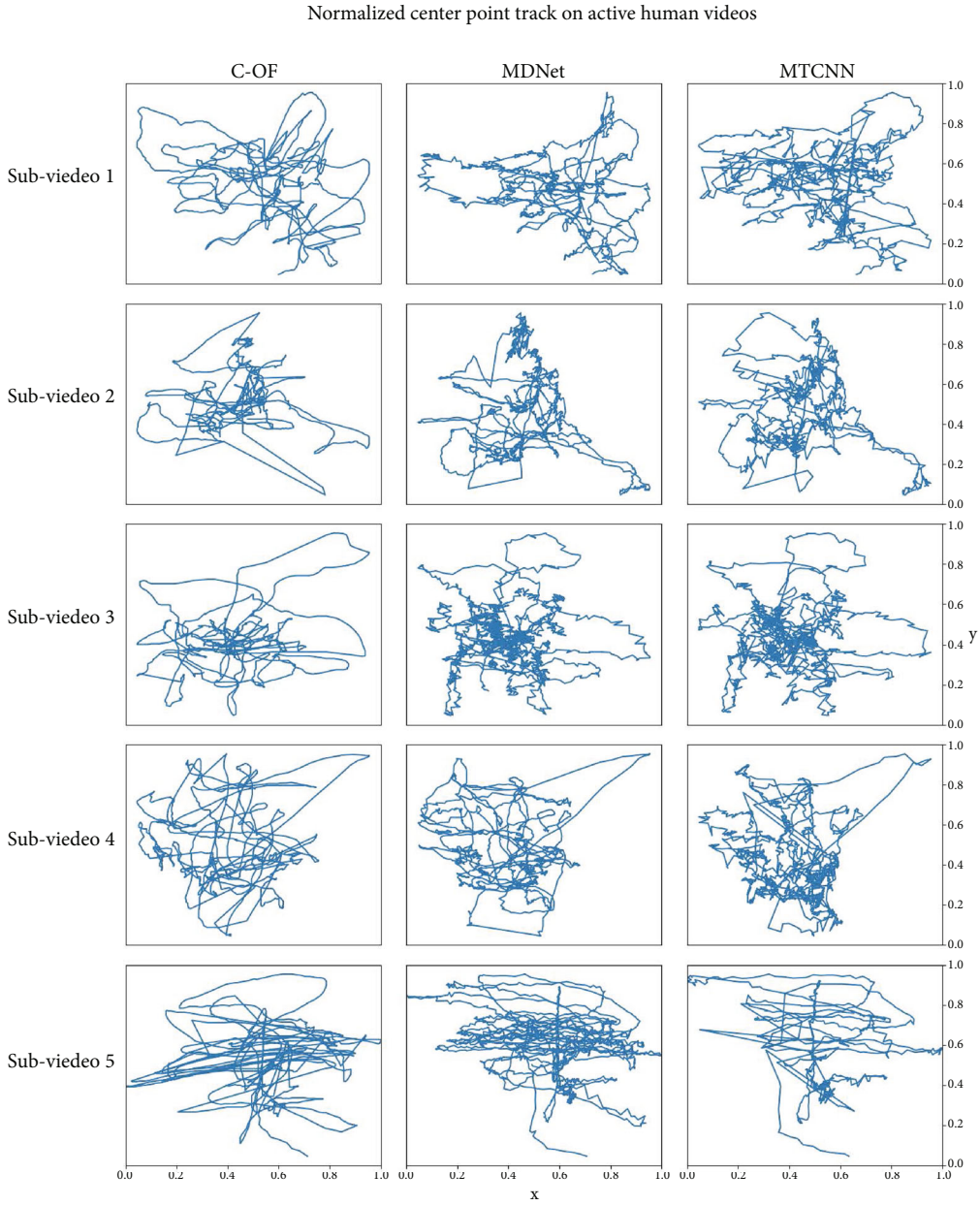

Figure 5: Center point motion tracking on active human videos.

platforms. The devices we used in the experiments are *Intel Core i7-8700K* and *Nvidia GTX TITAN X*. Details of running time using Python and C++ implementations are presented in Running Time.

In terms of the dataset, four long-term videos are recorded for the experiments, as there are no public benchmark aims for stable face tracking. The four videos are recorded via different conditions: active camera, active human, static human, and active illumination, respectively. As for the active camera and illumination, we let the actor be static and randomly move the camera and light source in front of the face. And for the active and static human, we first ask the actor to talk to the cameraman and act in a freestyle. Then, we ask the actor to stop acting and sit statically in front of the camera. We split each video into five clips, each clip is about one-minute long. The recording device is iPhone X with a 12-megapixel rear camera. We resize each frame to $1080 * 608$ resolution. The three different methods can detect and track faces all the time in each clip. So, the only difference is the scale and position changes of bounding boxes frame by frame. The download link of this testing dataset can be found in the GitHub repository as well.

4.2. Result Discussion

4.2.1. Stability. In the perspective of evaluating the stability and smoothness of face boxes from a frame sequence, comparing with ground truth is not the principal aspect. Besides, to the best of our knowledge, there is no general metric to

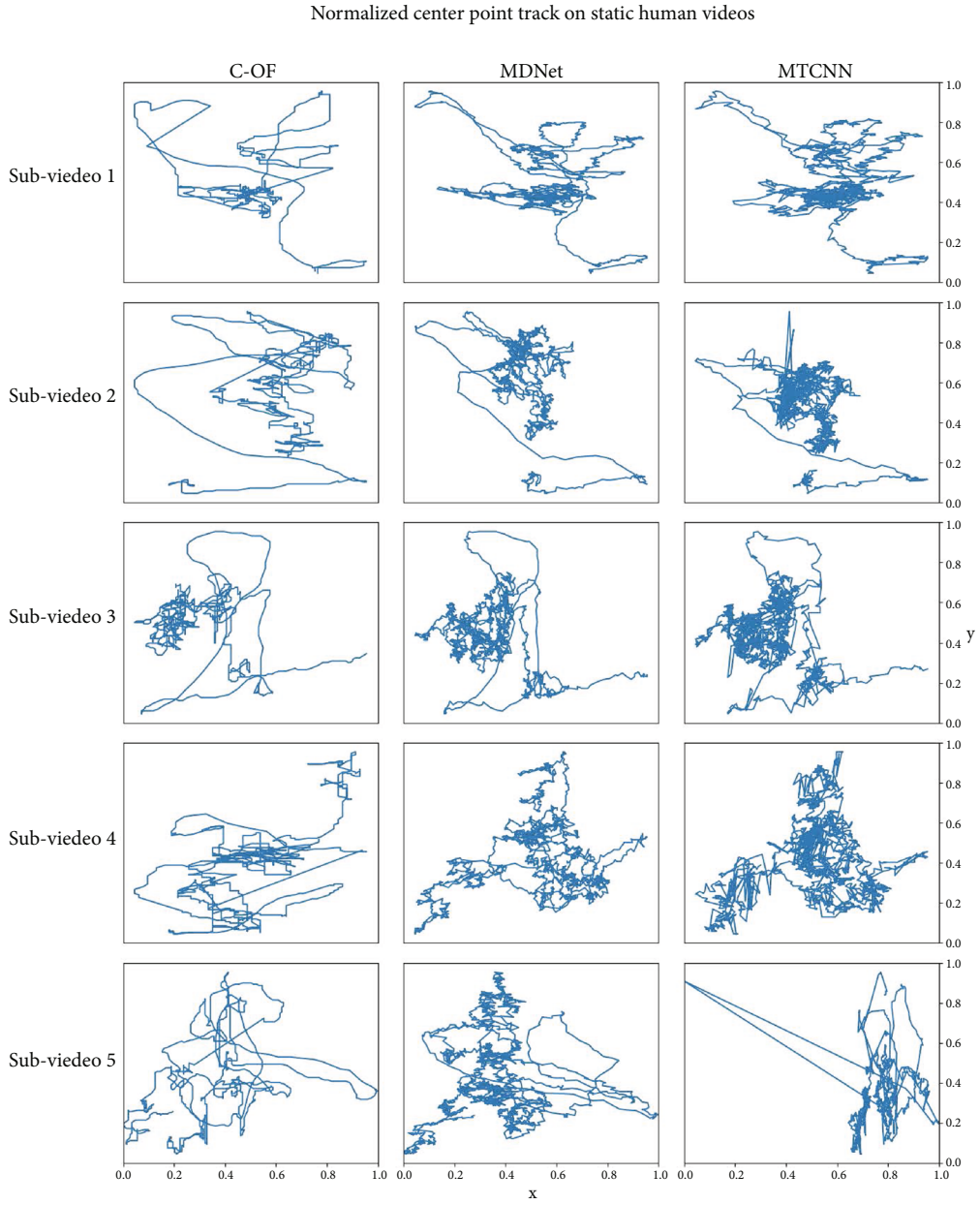

Figure 6: Center point motion tracking on static human videos.

quantificate the stability and smoothness of the bounding box. Hence, we naively consider the moving route of the bounding box's corner and center points to evaluate how the bounding box changes along with frames regarding scale and position. We first quantificate the stability by judging the change of width and height and position of the center point:

$$r_n^c = \sqrt{(x_n^c - x_{n-1}^c)^2 + (y_n^c - y_{n-1}^c)^2}, \tag{2}$$

$$\text{Stability} = \frac{1}{3N}\sum_{n=0}^{N}(|w_n - w_{n-1}| + |h_n - h_{n-1}| + r_n^c), \tag{3}$$

where x^c and y^c are the x, y coordinate of the center point, respectively, r^c is the absolute moving route of the center point, N is the total number of frames, and w and h are width and height of the bounding box, respectively. The summation offsets of width, height, and center point illustrate the change of scale and position of bounding box collaboratively. A smaller value means the box moves a short distance or has a little change of width or height. Table 3 gives all the results of stability for MDNet, MTCNN, and C-OF, respectively, on aforementioned four videos with different conditions. It can be viewed that our proposed C-OF significantly outperforms the other two methods in all experiments. Faces in static human videos are the more stable, where the minimum values take in place; for example, MDNet gains its minimum value of 1.425355, MTCNN's is 1.994141, and C-OF's is 0.245097 which is the global minimum value as well. Also, the maximum value of C-OF, which is 0.321575, on static human

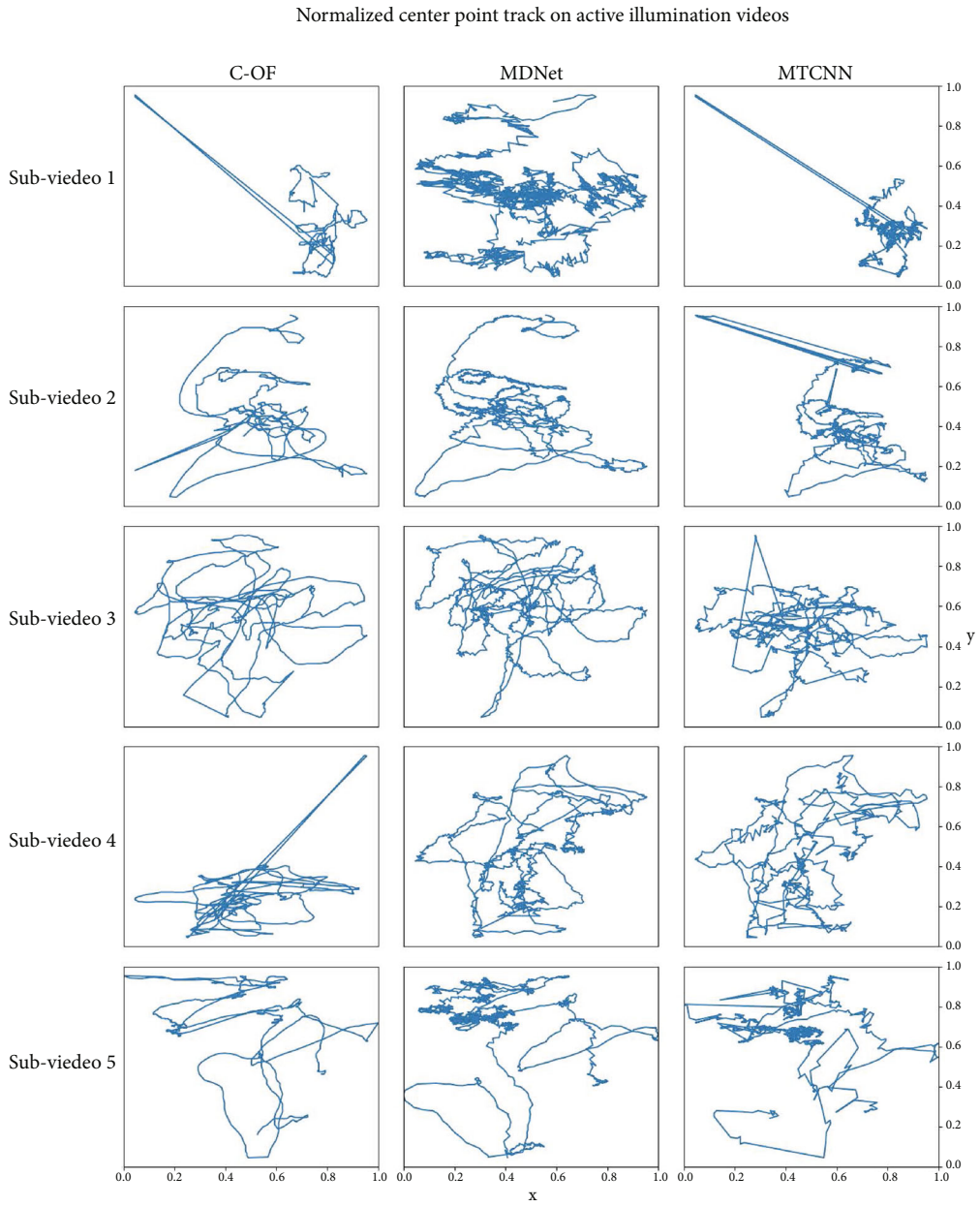

Figure 7: Center point motion tracking on active illumination videos.

videos is even smaller than the minimum values of MDNet and MTCNN, which are 1.425355 and 1.994141, respectively.

4.2.2. Smoothness. On the other hand, to further quantify the smoothness, we design another function, which mostly focuses on the scale change of the bounding box:

$$R_n = \frac{r_n^{tl} + r_n^{tr} + r_n^{br} + r_n^{bl}}{4}, \tag{4}$$

$$\text{Smoothness} = \frac{1}{N}\sum_{n=0}^{N} \ln\left(\frac{e^{R_n}}{e^{r_n^c}}\right), \tag{5}$$

where r^{tl}, r^{tr}, r^{br}, and r^{bl} are the absolute moving route of top left, top right, bottom right, and bottom left points, respectively. In this function, we empirically consider the ratio of corner and center points' absolute moving routes. By observation, we found that a bounding box that moves a long distance usually comes with a change of its scale. In Equations (4) and (5), e^r is larger than 1, as r is a nonnegative value. The mean route of corner point R_n is larger than the route of center point r_n^c, so logarithm of the ratio of e^{R_n} and $e^{r_n^c}$ is always a nonnegative value. In conclusion, we say that the box that moves a short distance without or with little scale change may gain a smaller value. We report the smoothness values in Table 4, where our proposed C-OF is superior to MDNet and MTCNN all the time.

4.2.3. Motion Tracking. The stability and smoothness can be observed clearly on the image sequence, but it is not

Table 5: Approximate running time for MDNet, MTCNN, and C-OF on different computing devices and resolutions. CPU refers to *Intel Core i7-8700K*; GPU is *Nvidia GTX TITAN X*.

Method	Language & framework	CPU		GPU	
		320 ∗ 240	640 ∗ 480	320 ∗ 240	640 ∗ 480
MDNet	Python & PyTorch	1362 ms	1334 ms	269 ms	270 ms
MTCNN		15 ms	32 ms	17 ms	22 ms
C-OF (ours)		10 ms	11 ms	11 ms	11 ms
MDNet	C++ & ncnn	—	—	—	—
MTCNN		10 ms	31 ms	—	—
C-OF (ours)		2 ms	3 ms	—	—

convenient to show out the image sequence in a paper. To this end, visualizing the motion tracking of any specific point from the bounding box is a feasible way. Figure 3 shows some visualization examples for the motion tracking of the center point in the bounding box from three methods. It is obvious that the proposed C-OF has far more smooth lines than MDNet and MTCNN, which illustrates that motion tracking of face boxes generated by C-OF is more smooth. In graphs (k) and (l), the outlier means a wrong face box is taken in place. All of the visualization examples are presented in Figures 4–7 for your reference.

4.2.4. Running Time. Running time is another principal aspect that commercial applications mostly take into account. Benefitting from lightweight models and optical flow method, our proposed C-OF is super real-time even on CPU, while the typical deep learning model commonly depends on GPU to attain a sufficient performance. Table 5 shows all the experimental results of approximate running time for MDNet, MTCNN, and C-OF on different computing devices and resolutions. The input image is resized to 320 ∗ 240 resolution. Regarding Python implementations, as MDNet needs to fine-tune the model online, its running time is far slower than the other two methods. Our proposed C-OF is super real-time of approximately 200 FPS and spends 5 ms and 6 ms less than MTCNN on CPU and GPU, respectively. Note that both MTCNN and C-OF have no massively parallel computing, in which case PyTorch using GPU performs worse than using CPU. That is why the running time of MTCNN and C-OF conducted on CPU (15 ms and 10 ms) is faster than GPU (17 ms and 11 ms). Typically, C++ implementation is more common than Python implementation in the industry field, so we also provide a C++ version of C-OF, which is also publicly available. We say C-OF is hyper real-time when using C++ and ncnn. Although benefitting from C++ and ncnn, MTCNN has a speedup from 15 ms to 10 ms on CPU; C-OF has a five times progress from 10 ms to 2 ms and achieves 500 FPS. If we change the input resolution from 320 ∗ 240 to 640 ∗ 480 , MDNet and C-OF have little running time increment or even a reduction. For MDNet, all the candidate face boxes are cut off, resized to a fixed resolution, and then fed into the model for fine-tuning. So the size of the input image does not matter, cause the number and the size of the candidate face boxes are predefined. In terms of C-OF, the resolution increment definitely may slow down the detector part; however, it only runs very limited times in one experiment. Other than the detector, optical flow is insensitive to the size of the image, and the identifier is too light to present out the performance loss. On the other hand, MTCNN has a normal running time increment from 15 ms to 32 ms on CPU and 17 ms to 22 ms on GPU as the input image's resolution changes. For the C++ version, MTCNN has a normal running time increment as well from 10 ms to 31 ms. Same as the Python version, C-OF runs 1 ms more than 320 ∗ 240 resolution (3 ms vs. 2 ms). Overall, no matter what implementation language or computing devices we use, our proposed C-OF is the fastest one, and the running time is more than sufficient for the commercial application.

5. Conclusion

In this paper, we proposed a stable, smooth, super real-time, and long-term face tracking system using lightweight CNN and optical flow, namely, C-OF, which consists of a face detector, face tracker, and face identifier. The method is aimed at solving the bounding box shaking problem, which commonly occurs in deep learning methods. We also optimize the system to make it run faster than most face detection and tracking methods. The experimental results show that C-OF can produce stable and smooth face boxes on a long-term face sequence with super even hyper real time. We design two functions to quantificate the stability and smoothness individually, and C-OF is superior to both MDNet and MTCNN. Meanwhile, we visualize the center point motion tracking of face boxes to observe the path the box goes and conclude that C-OF has a far more stable and smooth path line with a little crook. In the end, we make the Python and C++ implementations of C-OF public available for people who are interested in the work.

Data Availability

The testing data (20 mp4 files) used to support the findings of this study are included within the article, which also can be downloaded from https://www.dropbox.com/sh/fcks3k2l9xs36ze/AABlXm3FY3pMzStNrPktYKdRa?dl=0.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study is supported by the National Key Technology R&D Program of China (No. 2019YFC1606401), Beijing Natural Science Foundation (No. 4202014), and Humanity and Social Science Youth Foundation of Ministry of Education of China (No. 20YJCZH229).

References

[1] Z. Guo, S. Yu, A. K. Bashir, K. Yu, and J. C.-W. Lin, "Graph embedding-based intelligent industrial decision for complex sewage treatment processes," *International Journal of Intelligent Systems*, 2021.

[2] Z. Guo, K. Yu, A. Jolfaei, A. K. Bashir, A. O. Almagrabi, and N. Kumar, "A fuzzy detection system for rumors through explainable adaptive learning," *IEEE Transactions on Fuzzy Systems*, p. 1, 2021.

[3] Z. Guo, K. Yu, L. Yu, G. Srivastava, and J. C.-W. Lin, "Deep learning-embedded social internet of things for ambiguity-aware social recommendations," *IEEE Transactions on Network Science and Engineering*, 2021.

[4] S. K. Lakshmanaprabu, K. Shankar, A. Khanna et al., "Effective features to classify big data using social internet of things," *IEEE access*, vol. 6, pp. 24196–24204, 2018.

[5] Y. Gong, Z. Lin, R. Liu, K. Yu, and G. Srivastava, "Nonlinear mimo for industrial internet of things in cyber–physical systems," *IEEE TII*, vol. 17, no. 8, pp. 5533–5541, 2021.

[6] W.-L. Shang, J. Chen, H. Bi, Y. Sui, Y. Chen, and H. Yu, "Impacts of covid-19 pandemic on user behaviors and environmental benefits of bike sharing: a big-data analysis," *Applied Energy*, vol. 285, p. 116429, 2021.

[7] F. Ding, G. Zhu, M. Alazab, X. Li, and Y. Keping, "Deep-learning-empowered digital forensics for edge consumer electronics in 5g hetnets," *IEEE Consumer Electronics Magazine*, p. 1, 2020.

[8] K. Yu, L. Tan, L. Lin, X. Cheng, Z. Yi, and T. Sato, "Deep-learning-empowered breast cancer auxiliary diagnosis for 5gb remote e-health," *IEEE Wireless Communications*, vol. 28, no. 3, pp. 54–61, 2021.

[9] Y. Sun, J. Liu, K. Yu, M. Alazab, and K. Lin, "PMRSS: privacy-preserving medical record searching scheme for intelligent diagnosis in iot healthcare," *IEEE Transactions on Industrial Informatics*, p. 1, 2021.

[10] K. Yu, Z. Guo, Y. Shen, W. Wang, J. C.-W. Lin, and T. Sato, "Secure artificial intelligence of things for implicit group recommendations," *IEEE Internet of Things Journal*, 2021.

[11] L. Zhen, Y. Zhang, K. Yu, N. Kumar, A. Barnawi, and Y. Xie, "Early collision detection for massive random access in satellite based internet of things," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 5, pp. 5184–5189, 2021.

[12] K. Yu, M. Arifuzzaman, W. Zheng, D. Zhang, and T. Sato, "A key management scheme for secure communications of information centric advanced metering infrastructure in smart grid," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 8, pp. 2072–2085, 2015.

[13] K. Sharma, A. Aggarwal, T. Singhania, D. Gupta, and A. Khanna, "Hiding data in images using cryptography and deep neural network," 2019, arXiv preprint arXiv:1912.10413.

[14] T. Liang, K. Yu, F. Ming, X. Chen, and G. Srivastava, "Secure and resilient artificial intelligence of things: a honeynet approach for threat detection and situational awareness," *IEEE CEM*, p. 1, 2021.

[15] H. Li, K. Yu, B. Liu, C. Feng, Z. Qin, and G. Srivastava, "An efficient ciphertext-policy weighted attribute-based encryption for the internet of health things," *IEEE Journal of Biomedical and Health Informatics*, 2021.

[16] L. Tan, K. Yu, N. Shi, C. Yang, W. Wei, and H. Lu, "Towards secure and privacy-preserving data sharing for covid-19 medical records: a blockchain-empowered approach," *IEEE Transactions on Network Science and Engineering*, 2021.

[17] C. Feng, K. Yu, and M. Aloqaily, "Attribute-based encryption with parallel outsourced decryption for edge intelligent iov," *IEEE TVT*, vol. 69, no. 11, pp. 13784–13795, 2020.

[18] L. Liu, J. Feng, and Q. Pei, "Blockchain-enabled secure data sharing scheme in mobile-edge computing: an asynchronous advantage actor–critic learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2342–2353, 2021.

[19] C. Feng, B. Liu, Z. Guo, K. Yu, Z. Qin, and K.-K. R. Choo, "Blockchain-based cross-domain authentication for intelligent 5g-enabled internet of drones," *IEEE Internet of Things Journal*, 2021.

[20] J. L. Crowley and F. Berard, "Multi-modal tracking of faces for video communications," in *In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 640–645, San Juan, PR, USA, 1997.

[21] G. R. Bradski, "Real time face and object tracking as a component of a perceptual user interface," in *In Proceedings Fourth IEEE Workshop on Applications of Computer Vision. WACV'98 (Cat. No. 98EX201)*, pp. 214–219, Princeton, NJ, USA, 1998.

[22] R. J. Qian, M. I. Sezan, and K. E. Matthews, "A robust real-time face tracking algorithm," in *In Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No. 98CB36269)*, vol. 1, pp. 131–135, Chicago, IL, USA, 1998.

[23] K. Schwerdt and J. L. Crowley, "Robust face tracking using color," in *In Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, pp. 90–95, Grenoble, France, 2000.

[24] H. Stern and B. Efros, "Adaptive color space switching for face tracking in multi-colored lighting environments," in *In Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, pp. 249–254, IEEE, 2002.

[25] P. Vadakkepat, P. Lim, L. C. De Silva, L. Jing, and L. L. Ling, "Multimodal approach to human-face detection and tracking," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 3, pp. 1385–1393, 2008.

[26] Z. Kalal, K. Mikolajczyk, and J. Matas, "Face-tld: tracking-learning-detection applied to faces," in *In 2010 IEEE International Conference on Image Processing*, pp. 3789–3792, Hong Kong, China, 2010.

[27] Z. Kalal, J. Matas, and K. Mikolajczyk, "Online learning of robust object detectors during unstable tracking," in *In 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pp. 1417–1424, Kyoto, Japan, 2009.

[28] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[29] Z. Kalal, J. Matas, and K. Mikolajczyk, "Weighted sampling for large-scale boosting," *In BMVC*, 2008.

[30] A. Ranftl, F. Alonso-Fernandez, and S. Karlsson, "Face tracking using optical flow," in *In 2015 International Conference*

of the Biometrics Special Interest Group (BIOSIG), pp. 1–5, Darmstadt, Germany, 2015.

[31] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *In Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1, Kauai, HI, USA, 2001.

[32] Y. Lin, J. Shen, S. Cheng, and M. Pantic, "Ft-rcnn: real-time visual face tracking with region based convolutional neural networks," in *In 2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)(FG)*, pp. 267–274, Buenos Aires, Argentina, 2020.

[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[34] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *In European conference on computational learning theory*, pp. 23–37, Springer, 1995.

[35] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[36] S. Charles Brubaker, J. Wu, J. Sun, M. D. Mullin, and J. M. Rehg, "On the design of cascades of boosted ensembles for face detection," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 65–86, 2008.

[37] S. Liao, "A fast and accurate unconstrained face detector," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 211–223, 2016.

[38] M.-T. Pham and T.-J. Cham, "Fast training and selection of haar features using statistics in boosting based face detection," in *In 2007 IEEE 11th International Conference on Computer Vision*, pp. 1–7, IEEE, 2007.

[39] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[40] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *In 2012 IEEE conference on computer vision and pattern recognition*, pp. 2879–2886, Providence, RI, USA, 2012.

[41] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *In European conference on computer vision*, pp. 720–735, Springer, 2014.

[42] J. Yan, Z. Lei, L. Wen, and S. Z. Li, "The fastest deformable part model for object detection," in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2497–2504, Columbus, OH, USA, 2014.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[44] C. Zhu, Y. Zheng, K. Luu, and M. Savvides, "Cms-rcnn: contextual multi-scale regionbased cnn for unconstrained face detection," in *In Deep learning for biometrics*, pp. 57–79, Springer, 2017.

[45] H. Wang, Z. Li, X. Ji, and Y. Wang, "Face r-cnn," 2017, arXiv preprint arXiv:1706.01061.

[46] C. Zhang, X. Xu, and T. Dandan, "Face detection using improved faster rcnn," 2018, arXiv preprint.arXiv:1802.02142.

[47] W. Liu, D. Anguelov, and D. Erhan, "Ssd: single shot multibox detector," in *In European conference on computer vision*, pp. 21–37, Springer, 2016.

[48] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "Faceboxes: a cpu real-time face detector with high accuracy," in *In 2017 IEEE International Joint Conference on Biometrics (IJCB)*, pp. 1–9, Denver, CO, USA, 2017.

[49] J. Zhang, X. Wu, S. C. H. Hoi, and J. Zhu, "Feature agglomeration networks for single stage face detection," *Neurocomputing*, vol. 380, pp. 180–189, 2020.

[50] T.-Y. Lin, P. Dollár, and R. Girshick, "Feature pyramid networks for object detection," in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[51] K. Zhang, Z. Zhang, Z. Li, and Q. Yu, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[52] X. Shi, S. Shan, M. Kan, S. Wu, and X. Chen, "Real-time rotation-invariant face detection with progressive calibration networks," in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2295–2303, Salt Lake City, UT, USA, 2018.

[53] J. Deng and X. Xie, "Nested shallow cnn-cascade for face detection in the wild," in *In 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pp. 165–172, Washington, DC, USA, 2017.

[54] C. Li, X. Liang, Y. Lu, N. Zhao, and J. Tang, "Rgb-t object tracking: benchmark and baseline," *Pattern Recognition*, vol. 96, p. 106977, 2019.

[55] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: a benchmark for multi-object tracking," 2016, arXiv preprint arXiv:1603.00831.

[56] H. Fan, H. Ling, L. Lin et al., "Lasot: a high-quality benchmark for large-scale single object tracking," in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5374–5383, Long Beach, CA, USA, 2019.

[57] F. João, R. C. Henriques, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *In European conference on computer vision*, pp. 702–715, Springer, 2012.

[58] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *In 2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 2544–2550, San Francisco, CA, USA, 2010.

[59] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 8, pp. 1561–1575, 2017.

[60] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *In European Conference on Computer Vision*, pp. 749–765, Springer, 2016.

[61] Z. Zheng, Q. Wang, and B. Li, "Distractor-aware Siamese networks for visual object tracking," in *In Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 101–117, 2018.

[62] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4293–4302, Las Vegas, NV, USA, 2016.

[63] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, Boston, MA, USA, 2015.

[64] J.-Y. Bouguet, "Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm," *Intel corporation*, vol. 5, 2001.

[65] A. Paszke, S. Gross, and F. Massa, "Pytorch: an imperative style, high-performance deeplearning library," 2019, arXiv preprint arXiv:1912.01703.