Multiple People Tracking Based on Improved SiameseFC Combined with Lightweight YOLO-V4

Lu Shen[1(✉)], Zhiwen Chen[2], Boliang Zhang[1], Su-Kit Tang[1], and Silvia Mirri[3]

[1] Faculty of Applied Sciences, Macao Polytechnic University, Macao S.A.R. 999078, China
{lu.shen,sktang}@mpu.edu.mo
[2] Eastern Communications Company Limited, Hangzhou 310053, China
[3] Department of Computer Science and Engineering, University of Bologna, 40126 Bologna, Italy

Abstract. Multi-object tracking (MOT) is an active area of research in computer vision that is extensively applied in various domains, including but not limited to video surveillance, security, and intelligent transportation. There are two types of tracking algorithms: standard visual tracking techniques and deep learning tracking methods. Deep learning methods are becoming more common, but current tracking algorithms still need to overcome the challenge of false detection due to occlusion, similar backgrounds, and also the problem of slow speed. In response to the existing difficulties in multi-object tracking, this paper improves the fully convolutional Siamese (SiameseFC) network and integrates the Kalman filter to enhance the performance of the tracker. The lightweight network is used to improve the YOLO-V4 structure. The multi-people tracking network designed in this paper combines both networks, enabling objects to be detected and re-tracked after they reappear. By comparing with the performance of the network before improvement and other high-performance multi-object tracking algorithms, our proposed method can improve the processing speed of images while almost not losing too much precision, significantly reducing the model size.

Keywords: Multi-object tracking · Object detection · Object tracking

1 Introduction

Over the past 30 years, multiple object tracking has emerged as a prominent research area in computer vision and has experienced significant advancements. It is widely used in motion-based recognition, automatic monitoring, human-computer interaction, vehicle navigation, guidance systems, and other applications [34]. However, creating a tracker that is both robust and effective is a challenging task due to several factors. These factors include partial or complete

V. C. M. Leung et al. (Eds.): Qshine 2023, LNICST 574, pp. 291–305, 2024.
https://doi.org/10.1007/978-3-031-65123-6_21

occlusion of the target, significant alterations in target area illumination, interference objects in the background that share the target's color or texture, and changes in the target's scale [38,39].

Conventional detection-based approaches, such as Background Subtraction [1] and Optical Flow [33], do not prove sufficiently effective in precisely determining the position of targets. This is particularly true in highly congested scenes where alterations in target scale or lighting can result in identity switching between targets, ultimately leading to inadequate tracking accuracy. Subsequent research improved object tracking algorithms, such as Ristic et al. [30] introduced particle filtering to model target motion trajectories. Particle filtering [7] approximates probability density functions by finding a set of random samples that propagate in the state space, analyzing the tracked target state, and obtaining the optimal solution. It and other classic generative models such as Meanshift [8] and Camshift [11] have real-time solid performance. However, they can accumulate errors easily in the case of rapid target movement, resulting in target drift and seriously affecting tracking accuracy.

With the fast growth of deep learning in recent years, numerous deep learning-based trackers have appeared, and their performance has substantially increased [5]. Bertinetto et al. [2] proposed a method for tracking arbitrary objects using a SiameseFC network for similarity learning. This offline end-to-end training-based tracking method achieves high tracking accuracy while maintaining fast tracking speeds. However, this method's tracking boxes are not flexible enough, and the algorithm may fail due to occlusion. Moreover, it is incapable of tracking multiple targets. In multi-object tracking, for the SORT algorithm [3], the tracking part uses a Kalman filter [36] for state prediction. It combines IoU to construct a cost matrix. Then it uses the Hungarian algorithm [24] to associate detection boxes and trajectories, achieving fast tracking speed without considering target features inside the box, which can easily cause identity switching. JDE method [35] extracts features directly from the detection network and then combines Kalman filtering [36] and data association techniques for matching. However, in the case of pedestrians occluding each other, the detection accuracy is reduced, and the model is difficult to train end-to-end. The DeepSORT algorithm [37] adds a shallow deep appearance feature extraction network based on SORT [3] to improve tracking accuracy. However, relying solely on feature similarity comparison is often not entirely reliable. Factors such as lighting, angle, and occlusion can cause the similarity of two features of the same object not to reach the threshold and be considered a new object. Therefore, the complexity of the multi-object tracking process, combined with the potential for interference from factors such as lighting changes, target occlusion, and target deformation, make it challenging for multi-object tracking tasks to address issues such as handling occlusion, accurately associating trajectories, and improving real-time performance [28]. These challenges remain the primary obstacles facing the field of multi-object tracking currently.

In response to the challenges above, this paper builds upon the idea of designing tracking algorithms that incorporate object detection networks and proposes

a multi-person tracking algorithm based on an improved SiameseFC (denoted as SiameseFC*) network combined with lightweight YOLO-V4 (LW-YOLO-V4). The specific contributions of this paper are as follows:

- To address the issue of decreased tracking precision and failure of the SiameseFC algorithm in complex environments, we modified its backbone network and incorporated the Kalman filter to correct the results and improve tracking performance.
- We replaced the backbone network of YOLO-V4 with MobileNet, and retained only one detection class, significantly reducing the number of parameters in the network while improving processing speed without sacrificing too much mean average precision.
- We combined the improved SiameseFC network with lightweight YOLO-V4 to enhance tracking performance by re-detecting and tracking objects that reappear after being lost due to occlusion.

The rest of this paper is organized as follows. Section 2 details the proposed tracking network, which consists of the improved SiameseFC and lightweight YOLO-V4. Section 3 presents the experiments conducted in this study and analyzes the results. Finally, Sect. 4 concludes the paper.

2 Proposed Method

This section presents the proposed multiple people tracking network and describes in detail the improvements made to the performance of SiameseFC and the lightweight of YOLO-V4.

2.1 Framework Overview

The core concept of the proposed methodology is to merge the enhanced SiameseFC network with the lightweight YOLO-V4 model to perform multi-people detection and tracking. YOLO is a target detection network that can identify the categories within a set of objects. SiameseFC is a fast, single-object tracking network with outstanding tracking ability. In this study, the detected targets are updated using the SiameseFC* tracker based on LW-YOLO-V4 detection. The initial image is sent to the LW-YOLO-V4 detection network. LW-YOLO-V4 is used to identify and locate all targets in the initial frame of the supplied video, and all discovered target boxes are added to a list. Each item in the list is given a tracker. Following that, the SiameseFC* tracker updates the following frames.

One drawback of utilizing SiameseFC for updating is that it is susceptible to misjudgment when the tracked target is similar to the background. Furthermore, if tracking fails, it may be unable to resume tracking. Therefore, this study incorporates an object updating mechanism. When tracking people, two Intersection over Union (IoU) values are updated and compared, one is the IoU value between the box in the first frame and the current tracking box, and the other is the IoU value between the Kalman filter forecast box and the present

tracking box. These two IoU values are used to calculate the weighted fusion coefficient. Furthermore, the updating method can be used to update and follow fresh objectives. The proposed methodology's framework is illustrated in Fig. 1.

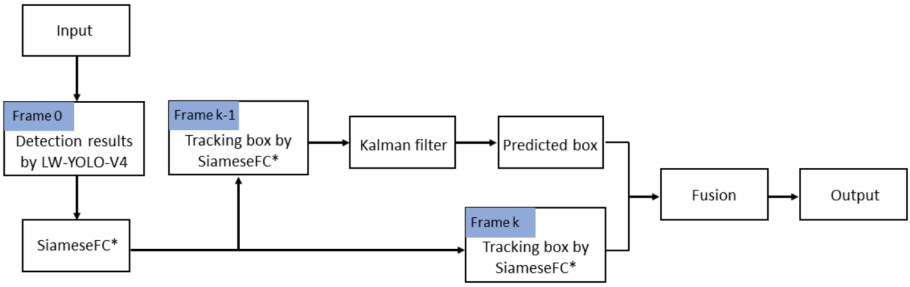

Fig. 1. The proposed multi-people tracking network combines SiameseFC* and LW-YOLO-V4.

2.2 Improvement to SiameseFC Network

SiameseFC is the first to use a Siamese network in object tracking, dramatically increasing the tracking speed of deep learning-based trackers. However, there are still some issues. A basic test of the SiameseFC algorithm's tracking performance, as depicted in Fig. 2, demonstrates that the tracking accuracy of the SiameseFC algorithm decreases in complex backgrounds and may even fail.

In this study, the improvements made to SiameseFC include replacing the backbone network and integrating a Kalman filter into the tracking network structure.

Replacement of the SiameseFC Backbone. The original SiameseFC network used AlexNet [23] as its backbone network. While the network structure was relatively simple and fast, its accuracy in ImageNet was not as good as the later VGG [32] and ResNet [14]. VGG and ResNet have deeper network structures and stronger feature extraction capabilities than AlexNet.

SiamFC is a similarity-based object tracker. Therefore, this method has limitations in translational invariance and symmetry. The ResNet-50 structure is commonly used as the benchmark backbone network for object detection and semantic segmentation models. Padding structures are used in practically all current backbone networks to assure network resolution accuracy and neatness. However, this design results in a lack of strict translational invariance in ResNet-50. Specifically, the padding layer introduces position-dependent perception in network response output, which poses a challenge for applying deep networks in the tracking field.

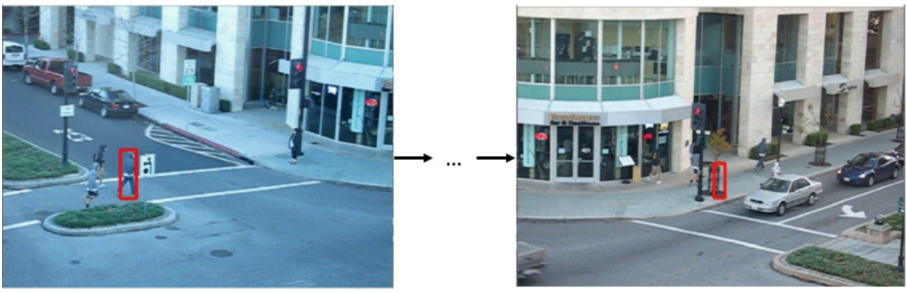

Fig. 2. An example of the SiameseFC algorithm failing to track a pedestrian in the natural scene. The arrow pointing from left to right indicates different frames of the video in chronological order. The red rectangular box on the left is the tracked target, and the red rectangular box on the right is the incorrectly tracked object. (Color figure online)

Therefore, this paper chooses the VGG network, specifically using VGG-16 as the backbone. VGG-16 consists of 13 convolutional layers and 3 fully connected layers. The network's architecture exclusively employs 3×3 convolutional and pooling layers to extract features. The final part of the network consists of three fully connected layers, with the output of the last fully connected layer serving as the classification prediction value. VGG incorporates ReLU as the activation function for every convolution layer, and dropout is implemented after the fully connected layer to prevent overfitting. Using small convolution kernels, VGG-16 can stack more convolutional layers to increase network depth, facilitating better learning of image features.

Integration with Kalman Filter. The Kalman filter is an optimal regression algorithm that utilizes mean squared error minimization to achieve linear filtering. It is an iterative strategy that uses state and observation equations to forecast and update the situation of the monitored target. However, the tracker may fail in the event of occlusions and other obstacles during object tracking. To address this issue, the Kalman filter can be employed to predict the target state, ensuring that the tracker can continue to track the target seamlessly. Consequently, this study integrates the Kalman filter into the tracking network, enhancing the algorithm's robustness.

The Kalman filter can work when applied to an evolving system with unclear information and create accurate forecasts for the system's future trends. When using the Kalman filter to predict the state, the prior state can be set to $x_{k-1} = (p_{k-1}, v_{k-1})$. The current moment is denoted as k, and the position and velocity of the target are represented by p and v, respectively. Two factors are provided to anticipate the present state with a Gaussian distribution: the mean $\hat{x}_{k-1}$ and the covariance matrix P_{k-1}, with external control input and external noise interference also taken into account. The Kalman filter's state forecast formula can be produced by using the kinematic calculation with associated computations in mathematics, as shown in Eq. (1).

$$\begin{cases} \hat{x}_k = F_k \cdot \hat{x}_{k-1} + B_k u_k \\ P_k = F_k \cdot P_{k-1} \cdot F_k^T + Q_k \end{cases} \tag{1}$$

Equation (1) involves several variables, including F_k, the kinematic coefficient matrix; B_k, the external control matrix; u_k, the external control quantity; and Q_k, the covariance matrix of external noise. Using Eq. (1), the current optimal estimate value can be calculated from the previous optimal estimation value and the known external control quantity, and the resulting uncertainty can be calculated from the prior uncertainty and external environmental interference.

The Kalman filter serves as a motion model to predict the target's position and correct the tracked object based on the predictions. Therefore, some parameters in the Kalman filter model require continuous updates to ensure real-time tracking and correctness. Specifically, by combining the previous prediction results with the detection output Z_k and its covariance matrix R_k from SiameseFC, two Gaussian distributions are generated: one is for prediction, denoted as $(\mu_0, \varepsilon_0) = \left(H_k \hat{x}_k, H_k P_k H_k^T\right)$, with H_k as the expected Gaussian distribution's covariance matrix; the other is for detection, represented by $(\mu_1, \varepsilon_1) = (Z_k, R_k)$. By performing mathematical operations, the state update Eq. (2) can be derived.

$$\begin{cases} \hat{x}_k^{'} = \hat{x}_k + K \cdot (Z_k - H_k \hat{x}_k) \\ P_k^{'} = P_k - K H_k P_k \\ K = P_k H_k^T \cdot \left(H_k P_k H_k^T + R_k\right)^{-1} \end{cases} \tag{2}$$

In Eq. (2), K is the Kalman gain. $\hat{x}_k^{'}$ is the new optimal estimation value, which can be iterated by using $P_k^{'}$ in the subsequent prediction and update equation.

2.3 Optimization for YOLO-V4

YOLO-V4 is a powerful and efficient detection model that combines speed and accuracy. The overall detection approach of YOLO-V4 is similar to that of the previous version, YOLO-V3, which uses three feature layers for classification and regression prediction. Based on this, we replace the YOLO-V4 backbone network with a lighter network and reduce detection classes to ensure the detection performance of the network while reducing the model size.

Replacement of the YOLO-V4 Backbone. MobileNet [18] is a lightweight deep neural network proposed by Google for embedded devices such as smartphones. In this study, we replace CSP-DarkNet53 in YOLOv4 for feature extraction with the MobileNet network and enhance feature extraction by using three initial effective feature layers with identical shapes, thus incorporating MobileNet into YOLOv4. MobileNet is widely recognized for its low parameter count, fast processing speed, and efficient memory utilization for video applications [29]. By serving as the backbone of an object detection network, MobileNet can perform effective feature extraction on input features while simultaneously reducing the number of network parameters, thereby achieving superior feature extraction performance.

Reduction of Detected Classes. The objective of this paper is to perform people tracking, which requires only one specific class detection using LW-YOLO-V4. To streamline the training process and reduce computation, images containing people and respective ground truth data are selected as the dataset. For instance, the COCO dataset consists of 80 classes. Thus, in detecting 80 classes, an image is divided into 19×19 grids, with each grid cell predicting three anchor points, and each anchor point consisting of 85 parameters (80 object-specific confidences, 1 detection confidence, and 4 coordinates). The total number of parameters is $19 \times 19 \times 3 \times 85 = 92{,}055$. Restricting the detection to a single class results in a reduction of parameters for each anchor box to six. These parameters comprise one object-specific confidence, one detection confidence, and four coordinates. This approach reduces the total number of parameters to approximately 1/14 of the original, which is $19 \times 19 \times 3 \times 6 = 6{,}498$.

3 Experiment and Result Analysis

This section experimentally verifies the tracking effectiveness of the network proposed in Sect. 2. The experimental environment, relevant datasets, network training, evaluation methods, and result analysis are presented.

3.1 Experiment Environment

For the experiments conducted in this paper, the server operating system utilized is Linux, and the GPU employed is NVIDIA Tesla V100. Meanwhile, the computer device runs on Windows 10, and the GPU in use is NVIDIA GTX 1060. The experiments involve the use of PyTorch as the deep learning framework, with Python as the primary programming language.

3.2 Dataset

In this experiment, the GOT-10K [19] dataset is used to train the SiameseFC* network and the VOC2007 [10], VOC2012 [9], and COCO [26] datasets are used to train the LW-YOLO-V4 network. The dataset OTB2015 [22] and the test set of VOC 2007 are used to test the improvement effect of the SiameseFC* and LW-YOLO-V4 networks, respectively. The dataset MOT16 [27] is used to test the tracking effect of the final combined network for multiple people.

3.3 Network Training

In this paper, the proposed SiameseFC* and LW-YOLO-V4 networks are trained separately.

Training of the SiameseFC*. To accommodate the changes that a target undergoes between two consecutive frames, it is necessary to expand the target region. The transformation method is illustrated in Eq. (3).

$$A = s\,(w + 2p) \times s\,(h + 2p) \tag{3}$$

In this context, w and h indicate the target region's width and height, respectively. The expansion side's length is p, which is calculated using $p = (w + h)/4$. $A = 127^2$. The image transformation, denoted by s, involves expanding the target region by $(w + 2p) \times (h + 2p)$ and then resizing it to a size of 127×127.

In generating training samples, positive samples are obtained from two distinct frames present within the same video. As the training data only comprises positive samples, SiameseFC* can only differentiate between foreground and non-semantic background, while objects carrying semantic information are regarded as interference. In scenarios where the interference is the background, SiameseFC* may mistakenly track the incorrect target. Hence, it becomes necessary to incorporate negative samples into the training set to enable the network to acquire the ability to recognize and handle interference.

In the experiment, the pretrained model utilized is trained on the ImageNet dataset. SGD is employed to regulate the learning rate, with a momentum value of 0.9. The learning rate decay mode is set to exponential decay, beginning at 10^{-2} to 10^{-8}. The decay weight is established at 0.0005. Training of the model is conducted for 50 epochs, with a batch size of 8.

Training of the LW-YOLO-V4. First, the dataset is processed to select images containing people and their corresponding ground truth information. A total of 5011 images are selected. Then, the processed dataset is subjected to mosaic data augmentation [4]. As shown in Fig. 3, mosaic data augmentation involves concatenating four images together, each with its corresponding ground truth bounding boxes. Finally, this new image is passed through the neural network, which effectively involves learning from four images at once. The mosaic approach significantly improves the backdrop of detected items and can be calculated in the BN layer by combining the four images.

The training procedure for the LW-YOLO-V4 model employs the Cosine Annealing method to control the learning rate. This method involves utilizing a Cosine function with a period that restarts the learning rate at the maximum value for each cycle. Initially, the maximum learning rate is defined as the initial learning rate, with the cycle duration set at $2 * Tmax$. During each cycle, the learning rate is gradually decreased and then increased. For this experiment, $Tmax$ is defined as 5, indicating that the learning rate is reset every 5 epochs. Within a period, the minimum learning rate is established at 10^{-5}. The training process entails 100 epochs using a batch size of 2.

3.4 Evaluation Method

The tracker's evaluation is executed by employing the One-Pass Evaluation (OPE) technique. This method commences by initializing the first frame of the

Fig. 3. An example of mosaic data augmentation connecting four images.

video sequence with the actual target position and subsequently running the target tracking algorithm to obtain the average precision and success rate of the tracking. The evaluation of the enhanced SiameseFC* tracker involves the use of two metrics, namely Precision Plot and Success Plot, to assess performance improvement. Additionally, the Multiple Object Tracking Accuracy (MOTA) metric is utilized to evaluate the model's tracking performance in multiple person scenarios.

Precision Plot. This metric refers to the percentage of frames where the estimated position is within a given threshold distance from the ground truth center. The Center Location Error (CLE) is a metric that calculates the average Euclidean distance between the predicted bounding box's center (x_e, y_e) and the manually labeled ground truth bounding box's center (x_{gt}, y_{gt}). The calculation formula is as follows:

$$CLE = \sqrt{(x_{gt} - x_e)^2 + (y_{gt} - y_e)^2} \tag{4}$$

Success Plot. This metric is synonymous with IoU. IoU denotes the proportion of the intersection between the predicted bounding box B_e and the ground truth bounding box B_{gt} to their union. The calculation formula is as follows:

$$IoU = \frac{|B_e \cap B_{gt}|}{|B_e \cup B_{gt}|} \tag{5}$$

By adjusting the threshold, we can obtain a plot of the average IoU as a function of the threshold, which is referred to as Success Plot.

Apart from detection accuracy, speed is a critical performance metric of the algorithm that plays a significant role in accomplishing real-time tracking. Frames per second (FPS) is a frequently used metric for measuring speed, indicating the quantity of images that can be processed per second.

MOTA. This metric represents the ratio of correctly predicted samples to all samples, excluding false alarms, lost targets, and ID switch errors. It measures

the performance of the tracker in detecting targets and maintaining trajectories, and is independent of the accuracy of target position estimation. The computation formula is expressed as follows:

$$MOTA = 1 - \frac{\sum_t (n_t + p_t + d_t)}{\sum_t g_t} \tag{6}$$

In Eq. (6), n_t is the count of false positives in the entire video sequence, p_t is the count of false negatives, d_t is the count of identity switches during tracking, and g_t is the count of ground truth objects in the entire video sequence.

3.5 Analysis of Experimental Results

In this paper, to validate the improvement of the proposed SiameseFC* tracker, Precision Plot and Success Plot are compared with the original SiameseFC network, and some classic trackers are also compared. LW-YOLO-V4 is compared with the original YOLO-V4. Finally, the combined network is compared with existing state-of-the-art multi-object tracking methods.

Evaluation of SiameseFC*. On the OTB 2015 dataset, we compared the performance of the proposed tracker and the tracker before improvement using the Overlap Success Rate (OSR) at an overlap threshold of 0.5 and the Distance Precision Rate (DPR) at a pixel distance threshold of 20, as shown in Table 1. The Precision and Success Rate of the improved model are both higher than those of the model before complete improvement on the OTB 2015 dataset, which demonstrates the effectiveness of our improvement on the SiameseFC network.

Table 1. Performance comparison of SiameseFC* with SiameseFC and SiameseFC combined with Kalman filter or VGG network on OTB2015 dataset.

Method	Precision (%)	Success rate (%)
SiameseFC	62	46.32
SiameseFC + Kalman filter	65	49.42
SiameseFC + VGG	75	56.66
SiameseFC* (Our proposed)	76	57.81

Figure 4 and Fig. 5 show the performance comparison of the proposed SiameseFC* network with some classic trackers MUSTER [17], MEEM [40], SAMF [25], KCF [16], DSST [6], Struck [13], ASLA [20], TLD [21], CSK [15], OAB [12], and IVT [31]. According to the comparison results, the Success Rate of the proposed SiameseFC* on the test set is higher than that of the classic trackers. Although the Precision is slightly lower than that of MUSTER and MEEM, the number of images that SiameseFC* can process per second is 82.8599, which

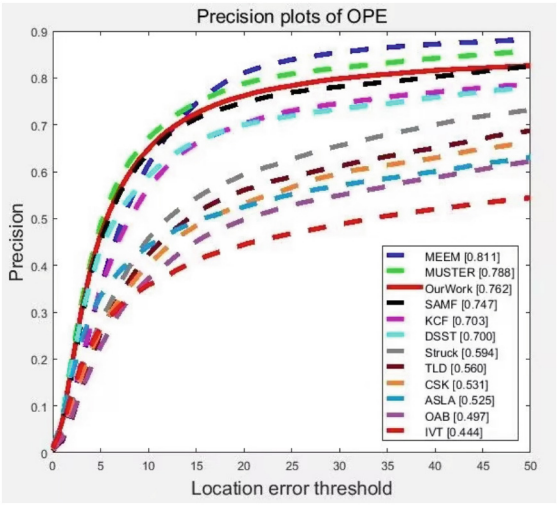

Fig. 4. Comparison of Precision between SiameseFC* and classic trackers on the OTB 2015 dataset.

is much higher than the FPS values of MEEM and MUSTER, 10.288 and 4.0116, respectively. The improvement in the performance of the single-object tracker is due to our improvement of the SiameseFC network structure while integrating the Kalman filter.

Evaluation of LW-YOLO-V4. Table 2 compares the performance of the proposed LW-YOLO-V4 with the original YOLO-V4. It is apparent that the network's speed has been enhanced, and the model size is reduced to about one-fifth of the original model without losing too much mean Average Precision (mAP). This is due to our improvement of the YOLO-V4 structure by using a lightweight network.

Table 2. Performance comparison of LW-YOLO-V4 and original YOLO-V4 on VOC 2007 test set.

Method	mAP	FPS	Model size (MB)
YOLO-V4	89	68	244
LW-YOLO-V4 (Our proposed)	79.72	89	51.4

Evaluation of Combined Network. Table 3 compares the performance of the final designed network that combines LW-YOLO-V4 and SiameseFC* with other high-performance multi-object tracking networks. It is apparent that the

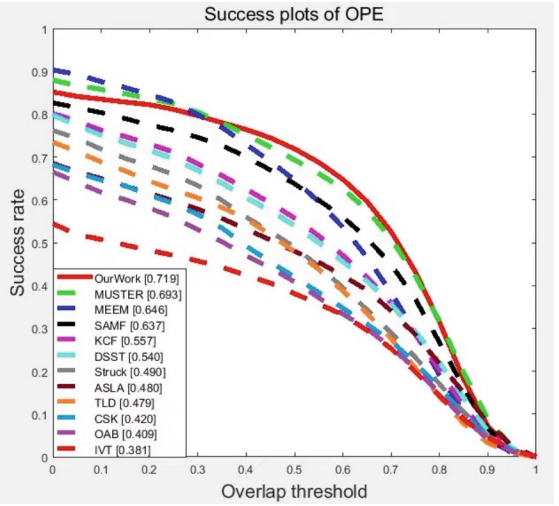

Fig. 5. Comparison of Success rate between SiameseFC* and classic trackers on the OTB 2015 dataset.

MOTA of our proposed method is higher than almost all the compared methods. Although the MOTA is slightly lower than the YOLO-V4 combined with DeepSORT, it is worth emphasizing that our approach relies on a lightweight network architecture, so a small loss in performance is acceptable.

Table 3. Performance comparison of LW-YOLO-V4 + SiameseFC* with other multi-object tracking methods on the MOT16 dataset.

Method	MOTA (%)
Faster-RCNN + SORT	59.8
Faster-RCNN + DeepSORT	61.6
Faster-RCNN + JDE	62.2
YOLO-V3 + SORT	62.3
YOLO-V3 + DeepSORT	63.7
YOLO-V4 + SORT	64.1
YOLO-V4 + DeepSORT	65.4
LW-YOLO-V4 + SiameseFC* (Our proposed)	64.7

4 Conclusion

This paper addresses the problems of existing multi-object tracking algorithms, such as being easily affected by complex backgrounds and resulting in

tracking failures, having large models, and slow processing speeds. We combine the SiameseFC network improved with the Kalman filter and the YOLO-V4 model improved with a lightweight network to significantly reduce the number of parameters and improve speed while maintaining precision. The proposed method can also detect and track objects when they reappear after a tracking failure. Our proposed method shows competitive tracking performance while improving towards a lightweight model by comparing it with original networks, classic trackers, and high-performance multi-object tracking algorithms. In the future, we will endeavor to address the challenge of treating re-detected targets as new objects during multi-object tracking, building upon the existing foundation of our work.

References

1. Amri, S., Barhoumi, W., Zagrouba, E.: A robust framework for joint background/foreground segmentation of complex video scenes filmed with freely moving camera. Multimedia Tools Appl. **46**(2–3), 175–205 (2010)
2. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: Fully-convolutional siamese networks for object tracking. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9914, pp. 850–865. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48881-3_56
3. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3464–3468. IEEE (2016)
4. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
5. Chou, K.S., et al.: Taxi demand and fare prediction with hybrid models: enhancing efficiency and user experience in city transportation. Appl. Sci. **13**(18) (2023). https://doi.org/10.3390/app131810192. https://www.mdpi.com/2076-3417/13/18/10192
6. Danelljan, M., Häger, G., Khan, F.S., Felsberg, M.: Discriminative scale space tracking. IEEE Trans. Pattern Anal. Mach. Intell. **39**(8), 1561–1575 (2016)
7. Doucet, A., Gordon, N.J., Krishnamurthy, V.: Particle filters for state estimation of jump Markov linear systems. IEEE Trans. Signal Process. **49**(3), 613–624 (2001)
8. Du, K., Ju, Y., Jin, Y., Li, G., Li, Y., Qian, S.: Object tracking based on improved meanshift and sift. In: 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), pp. 2716–2719. IEEE (2012)
9. Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge: a retrospective. Int. J. Comput. Vision **111**, 98–136 (2015)
10. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vision **88**, 303–338 (2010)
11. Exner, D., Bruns, E., Kurz, D., Grundhöfer, A., Bimber, O.: Fast and robust camshift tracking. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops, pp. 9–16. IEEE (2010)
12. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: BMVC, vol. 1, p. 6. Citeseer (2006)
13. Hare, S., et al.: Struck: structured output tracking with kernels. IEEE Trans. Pattern Anal. Mach. Intell. **38**(10), 2096–2109 (2015)

14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
15. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33765-9_50
16. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2014)
17. Hong, Z., Chen, Z., Wang, C., Mei, X., Prokhorov, D., Tao, D.: Multi-store tracker (muster): a cognitive psychology inspired approach to object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 749–758 (2015)
18. Howard, A.G., et al.: Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
19. Huang, L., Zhao, X., Huang, K.: Got-10k: a large high-diversity benchmark for generic object tracking in the wild. IEEE Trans. Pattern Anal. Mach. Intell. **43**(5), 1562–1577 (2019)
20. Jia, X., Lu, H., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1822–1829. IEEE (2012)
21. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1409–1422 (2011)
22. Kiani Galoogahi, H., Fagg, A., Huang, C., Ramanan, D., Lucey, S.: Need for speed: a benchmark for higher frame rate object tracking. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1125–1134 (2017)
23. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. **25** (2012)
24. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Res. Logist. Q. **2**(1–2), 83–97 (1955)
25. Li, Y., Zhu, J.: A scale adaptive kernel correlation filter tracker with feature integration. In: Agapito, L., Bronstein, M.M., Rother, C. (eds.) ECCV 2014. LNCS, vol. 8926, pp. 254–265. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16181-5_18
26. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
27. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K.: Mot16: a benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831 (2016)
28. Qian, Y., Yang, X., Tang, S.K.: Dual-space aggregation learning and random erasure for visible infrared person re-identification. IEEE Access (2023)
29. Qiu, J., Yan, X., Wang, W., Wei, W., Fang, K.: Skeleton-based abnormal behavior detection using secure partitioned convolutional neural network model. IEEE J. Biomed. Health Inf. **26**(12), 5829–5840 (2021)
30. Ristic, B., Arulampalam, M.S.: Tracking a manoeuvring target using angle-only measurements: algorithms and performance. Signal Process. **83**(6), 1223–1238 (2003)
31. Ross, D.A., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. Int. J. Comput. Vision **77**, 125–141 (2008)

32. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
33. Sun, D., Roth, S., Black, M.J.: Secrets of optical flow estimation and their principles. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2432–2439. IEEE (2010)
34. Wang, T., Li, J., Wei, W., Wang, W., Fang, K.: Deep-learning-based weak electromagnetic intrusion detection method for zero touch networks on industrial IoT. IEEE Netw. **36**(6), 236–242 (2022)
35. Wang, Z., Zheng, L., Liu, Y., Li, Y., Wang, S.: Towards real-time multi-object tracking. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12356, pp. 107–122. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58621-8_7
36. Welch, G., Bishop, G., et al.: An introduction to the kalman filter (1995)
37. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3645–3649. IEEE (2017)
38. Wong, T.L., Chou, K.S., Wong, K.L., Tang, S.K.: Dataset of public objects in uncontrolled environment for navigation aiding. Data **8**(2) (2023). https://doi.org/10.3390/data8020042. https://www.mdpi.com/2306-5729/8/2/42
39. Zhang, B., Shen, L., Yao, J., Tang, S.K., Mirri, S.: Uwb hybrid filtering-based mobile IoT device tracking. In: Proceedings of the 2023 ACM Conference on Information Technology for Social Good, pp. 471–476 (2023)
40. Zhang, J., Ma, S., Sclaroff, S.: MEEM: robust tracking via multiple experts using entropy minimization. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 188–203. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10599-4_13