Hindawi
Mobile Information Systems
Volume 2021, Article ID 2292128, 18 pages
https://doi.org/10.1155/2021/2292128

Research Article

Adaptive Polymorphic Fusion-Based Fast-Tracking Algorithm in Substations

Wenbin Shi,[1] **Jingsheng Lei**,[1] **Xingli Gan**,[1] **and Zhongguang Yang**[2]

[1]*Zhejiang University of Science and Technology, Hangzhou 310000, China*
[2]*Electric Power Research Institute of State Grid Shanghai Electric Power Company, Shanghai 200051, China*

Correspondence should be addressed to Zhongguang Yang; yzg_sgcc@qq.com

Received 1 October 2021; Accepted 13 November 2021; Published 29 November 2021

Academic Editor: Han Wang

Tracking multiple objects in a substation remains a challenging problem since pedestrians often overlap together and are occluded by infrastructures such as high-tension poles. In this paper, we propose an adaptive polymorphic fusion-based fast-tracking algorithm to address the problem. We first leverage the fast segmentation algorithm to obtain the fine masks of pedestrians and then combine the motion and performance information of pedestrians to realize the fast-tracking in substations. Our model is evaluated on the widely used MOT19 dataset and real-substation scenarios. Experimental results demonstrate that our model outperforms state-of-the-art models with a significant improvement in the MOT19 dataset and occlusion cases in substations.

1. Introduction

We have witnessed a significant change in the intelligent management of grid corporations promoted by the high demand for power grid construction. Substations are the bridges of the power transmission. However, due to the remote construction location and steep terrain, the maintenance and the safety precautions in substations become inevitably difficult. The rapid development of intelligent software and hardware technology achieves the extensive application of intelligent power equipment in grid construction [1]. Unattended substations based on video surveillance technology significantly improve the efficiency of the substation management and gradually replace the manned mode, becoming the main way forward in smart substations. Recently, most positioning systems in substations exploit hardware technologies such as ultrawide waves to localize and track targets [2] whereas the method cannot automatically track targets and can only collect videos in a fixed area. The appearance of machine learning introduces the traditional machine learning algorithms, such as support vector machines (SVMs), Kalman filter, and adaptive enhancement to locate and track targets [3]. With the emergence of the fifth-generation wireless communication technology [4], data communication with ultrahigh reliability and ultralow time delay [5, 6] becomes possible. It can achieve good results in areas with high requirements for time delay and reliability, such as real-time target detection, target tracking, target positioning, and automatic driving [7, 8]. Nevertheless, traditional methods cannot detect targets efficiently and deal with the scenarios such as occlusion and blurry shooting.

Lately, the introduction of deep learning models has promoted the robustness and accuracy of tracking in substations. The SORT (Simple Online and Real-time Tracking) algorithm [9] is an online real-time multiobject tracking algorithm, which uses the Kalman filter and the Hungary algorithm to achieve data association and determine whether the targets detected in different frames are the same object or not. Chen et al. [10] proposed the MOTDT algorithm, which formulated a standard pedestrian detection trajectory scoring mechanism through fusing the object classifiers and the tracker credibility, utilizing the generated standard credibility as nonmaximum suppression (NMS) input to obtain more accurate candidates. Wang et al. [11] proposed the JDE model, which uses the YOLOv3 algorithm for the purpose of combining the detection and the embedded model to improve time efficiency. Besides, the JDE model makes use of triplet

loss to train the network, which automatically learns the loss weight strategies so as to achieve the weighting of nonuniform loss. Zhang et al. [12] proposed a one-shot MOT-based fair algorithm, which takes advantage of an anchor-free tactic to reduce the impact of the detection frame on pedestrian recognition. By estimating the center of the object on a high-resolution feature map, the features of the pedestrian can be better aligned with the center of the object. Therefore, the mainstream tracking algorithms usually identify and track the target by combining the motion information with the performance information of the target. Based on the aforementioned methods, this paper realizes the rapid and accurate detection and tracking of pedestrians in the substation.

In the substation, the tracking targets are generally made up of working pedestrians and vehicles. Working pedestrians are uniformly dressed, which frequently causes the occlusion and loss of pedestrians during tracking. Meanwhile, the complex infrastructures coupled with the hindrance when obtaining video surveillance data make pedestrian detection and tracking in the substation become difficult. Consequently, we first enlarge the tracking dataset with a data expansion method based on existing substation surveillance video data. And we proposed an adaptive segmentation strategy to train the model so that the model can automatically update when different pedestrian information is input. Besides, aiming at the pedestrian loss and identity change caused by obstructions in the substation, we adopted the correlation algorithm and priority allocation strategy to address the problem. We verified the effectiveness of our model on the established test dataset, which consists of real-substation monitoring scenarios, and we compared our model with other state-of-the-art models on widely used datasets. As a result, our method is able to detect and track pedestrians quickly and robustly.

The contributions of the proposed model are mainly in the following four aspects:

(1) We proposed a lightweight real-time pedestrian detection network Light-YOLOv4. We took advantage of the segmentation algorithm to obtain the fine mask of pedestrians and the segmentation results. Then, we matched the cosine similarity of the segmentation results in multiple frames to obtain the results of tracking.

(2) We proposed a multimodal fusion substation pedestrian tracking method. Combining with the motion and performance information of pedestrians, our method tracks pedestrians in the substation on the basis of the detection results of the Light-YOLOv4 algorithm.

(3) We proposed a weight distribution method based on integrated metric learning, which adopts the results of segmentation branches to accurately express the performance information of pedestrians. Besides, aiming at solving the problems of pedestrian loss and identification change caused by dense obstructions in the substation, we adopted the correlation algorithm and priority allocation strategy.

(4) We proposed the algorithm to be in accord with the real scenarios in the substation. We expanded existing datasets using the data expansion method and adopted an adaptive segmentation training method to automatically update when various pedestrians enter the substation.

1.1. Methodology. Existing deep learning-based target detection methods are mainly divided into two categories: one is the candidate frame-based two-stage target detection algorithm represented by Fast RCNN network and Faster RCNN network, and the other is the end-to-end one-stage target detection algorithm represented by SSD and YOLO network. Because of the excellent real-time performance and flexibility of the detection algorithm, the detection-based tracking method has become one of the compelling research topics. In the field of tracking, how to confirm the trajectories of pedestrians has turned into a huge challenge. Nowadays, the classic strategies to solve the problem consist of the fluid network conception and the probabilistic graphic model. However, the abovementioned methods are unsuitable for online situations where a target exists in every frame. We combine the motion and the performance information to solve the aforementioned issues in the substation. We use the Kalman filter to update the trajectories of pedestrians so that tracking multiple objects in the substation can be realized.

1.2. YOLOv4. The YOLO series is an end-to-end one-stage target detection algorithm, which has a faster detection speed. The YOLOv4 algorithm is on the basis of the original YOLO detection architecture and adopts the best optimization strategies in five aspects, including data processing, backbone network, training, activation function, and loss function. The architecture of YOLOv4 is shown in Figure 1.

YOLOv4 is mainly composed of three components, comprising the feature extraction network, feature fusion network, and YOLO detection head. The feature extraction network is based on the CSPDarknet53 network, which contains 29 convolutional layers, $725 * 725$ receptive fields, and 27.6 M parameters. The CSPDarknet53 feature extraction network divides the feature into two parts. The first part is directly constructed to generate residual edges, and the second part ensures the accuracy of the model while reducing model calculation complexity by first convolving with the main branch and then concatenating with the input of the first part. In addition, YOLOv4 transforms feature maps of any size into fixed-size feature vectors by fusing with the SPP module. Meanwhile, we merge features in different levels together with the PANet.

1.3. Kalman Filter. Kalman filter is one of the classic recursive filters. The main idea of the Kalman filter comes from Bayesian Estimation Theory. Kalman filter is based on the optimal estimation value at the current moment and uses the error covariance matrix to calculate the predicted value of

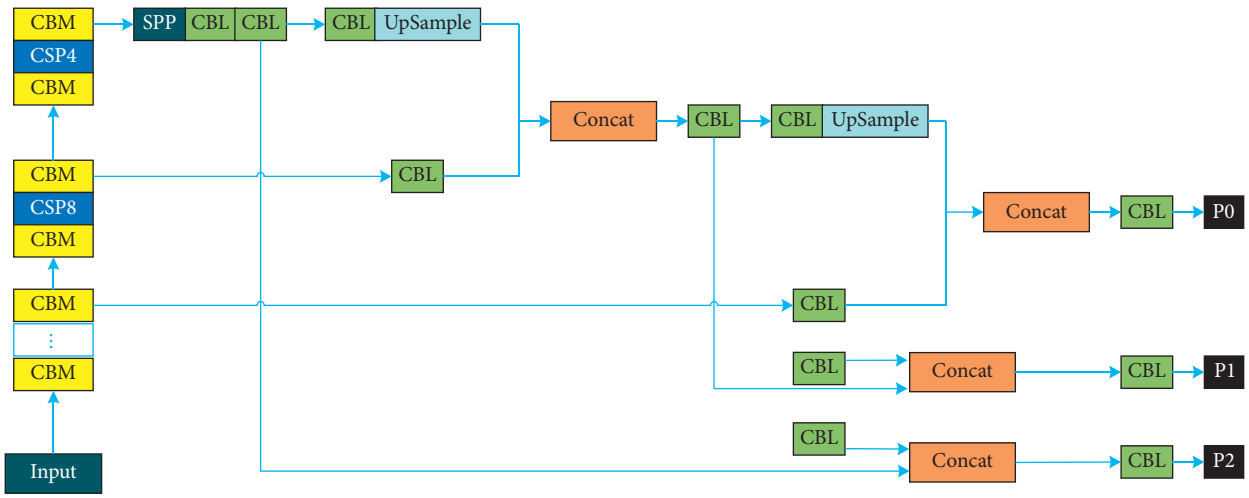


Figure 1: The architecture of the YOLOv4 network.

the state variable at the next moment. Meanwhile, the Kalman filter observes the state to obtain the observed variables and then uses the observed variables to correct the forecasts. Finally, we obtain the optimal state estimator at the next moment.

The Kalman filter prediction algorithm tracks the changes of the linear system quickly and accurately. The prediction model is mainly divided into two parts, which consist of the time update and the measurement update. The update equation assumes that the state at the time t evolves from the state at the time $(t-1)$, and the measurement equation calculates time based on the estimated state and noise at the time t.

1.4. Cosine Distance. The proposed model reidentifies pedestrians based on the performance information of the cosine distance. Specifically, the cosine similarity is used as a measurement function to calculate the cosine distance between all detected targets in different frames. The smaller the cosine distance is, the greater the probability that the two targets are the same target is. The proposed model reduces tracking errors caused by motion information and improves the accuracy of tracking.

2. The Framework of the Proposed Model

We proposed a lightweight Light-YOLOv4 real-time detection network, which combines motion and performance information of pedestrians, and proposed a pedestrian tracking method in substations using integrated metric learning. Inspired by Mask Region Proposal Network [13], we first utilize the detection algorithm to obtain the candidate box containing the pedestrian, then segment the pedestrian in the candidate box, and calculate the similarity of the candidate boxes in adjacent frames to confirm the identity of the pedestrian. Finally, we weigh the similarity of candidate boxes with the tracking boxes generated from the motion information of the pedestrian to obtain a more refined tracking box.

The framework of our model is shown in Figure 2. Concretely, (1) we preprocess the input real-time video frame signal and then detect pedestrians through the proposed lightweight Light-YOLOv4 algorithm and obtain the coordinates of the pedestrian's circumscribed rectangular boxes. (2) We crop the pedestrians according to the obtained coordinates, and the cropped pedestrian images are input to the segmentation branch, in order to obtain the fine mask of the pedestrians. In addition, input the segmentation results in the pretrained convolutional neural network. Moreover, we calculate the cosine similarity between the segmentation results of adjacent frames and determine the identity of the pedestrian through the performance information. (3) We use the Kalman filter algorithm to predict the trajectory of the detected pedestrian and make use of the Mahalanobis distance to indicate the predicted state and current state of the pedestrian. (4) Finally, we use the fusion metric learning method to calculate the degree of association between two independent objects in the front and rear frames in combination with motion and performance information. Furthermore, we introduce the weight coefficients and set a threshold to obtain a more refined tracking box. If the difference between the current frame time and the frame time that the pedestrian successfully matched last time is greater than the threshold, the pedestrian's trajectory is considered to be terminated and would be deleted in the subsequent tracking. Otherwise, it is considered that the trajectory is not lost. The above process is used to solve the problems of occlusion and crossing scenarios in the substation. In summary, we proposed a real-time pedestrian detection network called Light-YOLOv4. Light-YOLOv4 has the ability to track targets accurately and rapidly.

3. Light-YOLOv4 Detection Network and Segmentation Branch

In the substation, the high installation height of cameras and real-time requirement in surveillance demand tracking algorithms have the ability of real-time tracking and detecting accurately. The YOLO algorithm is a detection algorithm

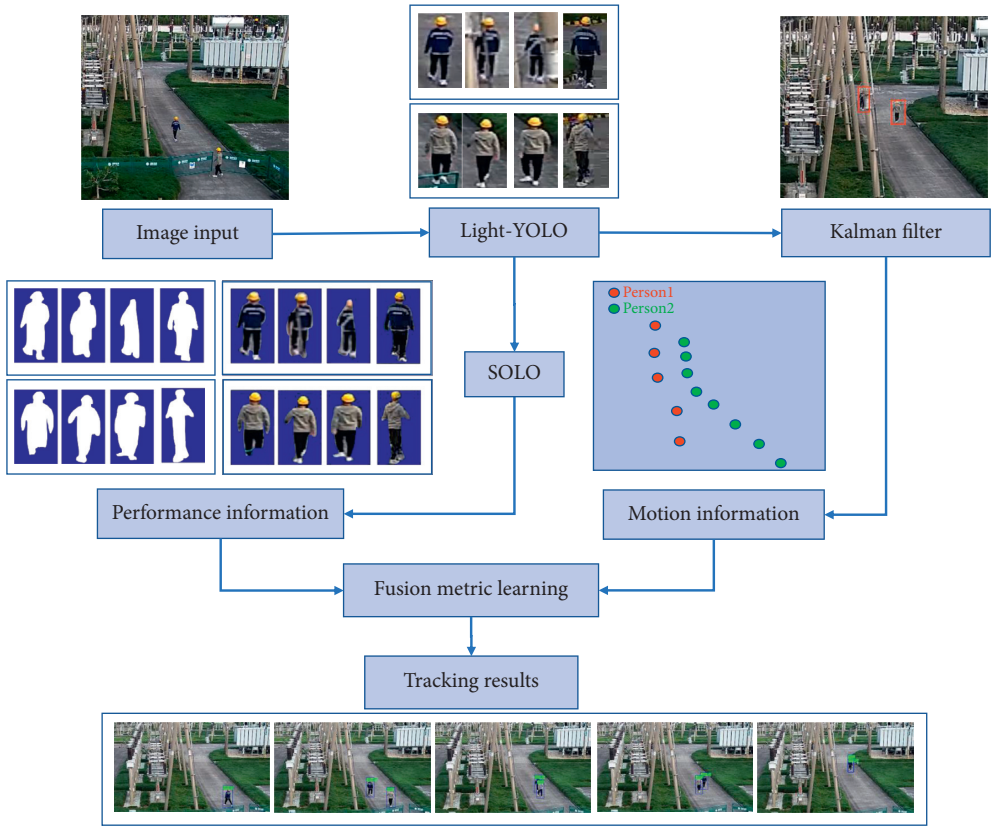


Figure 2: The framework of the proposed algorithm.

based on regression. The entire network is only composed of convolutional layers. And YOLO can complete the classification and location only once through the network. YOLOv4-Tiny is a lightweight version of YOLO, the running speed of which is faster than the YOLO network. However, YOLOv4-Tiny pursues the real-time requirement too much, thus giving up the accuracy.

Therefore, we proposed the Light-YOLOv4 real-time detection network, and the structure of Light-YOLOv4 is shown in Figure 3. The light-YOLOv4 network mainly contains three modules: multistream feature fusion module (MSFBlock), adaptive module (AdBlock), and hierarchical multiscale prediction module. The MSFBlock introduces an attention mechanism to obtain the different spatial and channel features and then merge the features of multistream branches to enhance the depth of features. In addition, we make use of the AdBlock to mine the connection between the points on the spatial and the channel feature maps. AdBlock can not only obtain rich global information but also improve detection accuracy and real-time performance. Finally, we optimize the multiscale prediction in the YOLO algorithm and combine multistage feature maps in the detection network to gain fine features and elevate the accuracy of the detection.

3.1. Multistream Feature Fusion Module. We adopt the MSFBlock to replace three CBL modules in the YOLOv4-Tiny network. CBL refers to the components made of conv, Batch Normalization, and Leaky ReLU function. The attention mechanism in the MSFBlock improves the ability of extraction, accuracy, and efficiency of detection.

As shown in Figure 4, the MSFBlock is made of two branches. The first branch is composed of a 3×3 convolutional layer connected with the output of the complete attention module (CAM) to extract features with different receptive fields. The second branch is shortcut by a 1×1 convolutional layer. The full convolution module in the first branch extracts semantic features. The first 1×1 convolution layer of the full convolution module reduces the dimensionality and the 3×3 convolution layer extracts features. The CAM in the first branch suppresses irrelevant noise. CAM uses a 1×1 convolutional layer to reduce dimensionality, then learns the weight of each feature group, and extracts important features through weights. Hence, the CAM filters out irrelevant features and extracts important features.

The structure of the CAM is depicted in Figure 5. By adding residual connections, the CAM makes better use of the previously extracted feature information. While introducing spatial feature information, as well as adopting the LeakyReLU activation function and a 1×1 convolutional layer, the CAM acquires the feature response.

3.2. Adaptive Module. The AdBlock is divided into spatial adaptation (SAd) and channel adaptation (CAd). The former module mainly obtains the spatial connections of features.

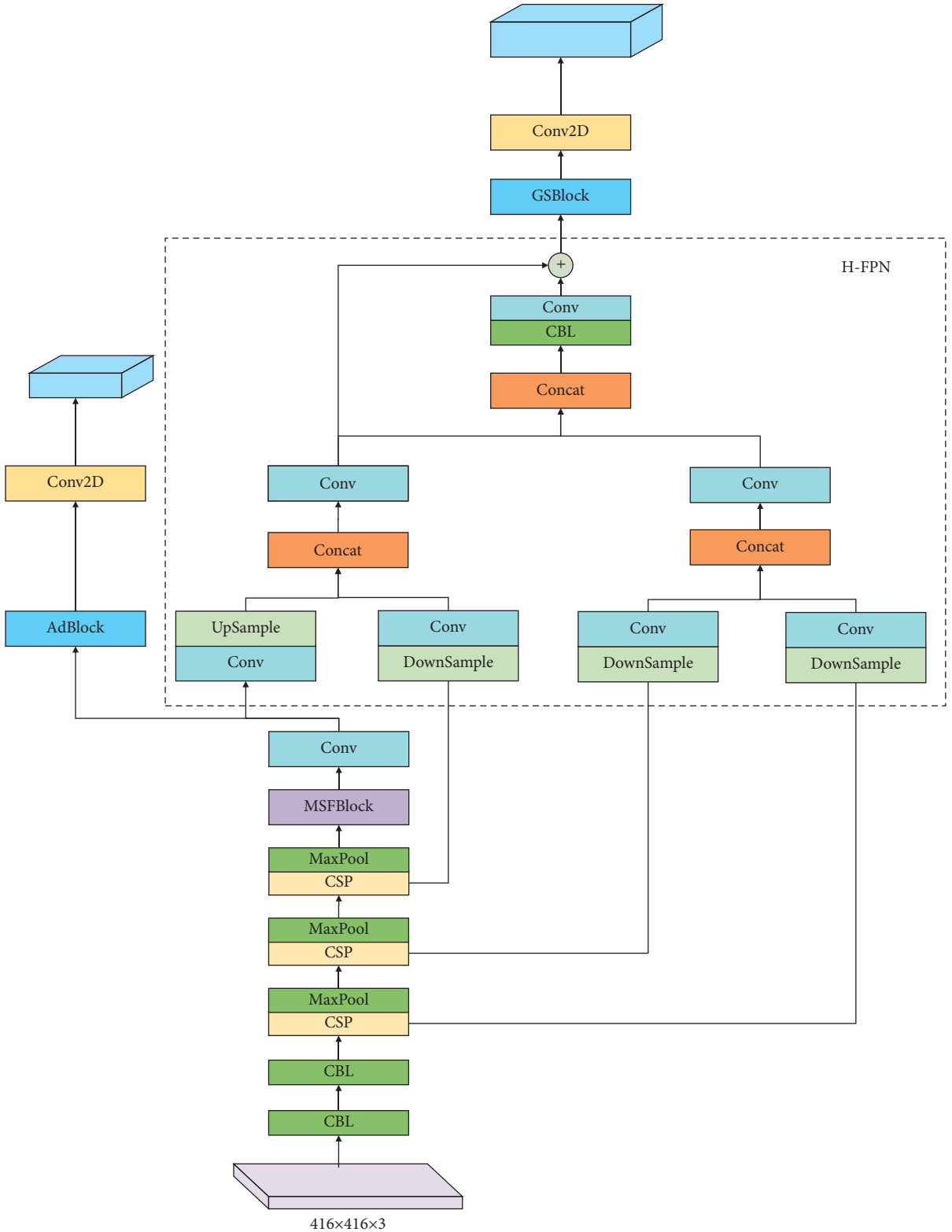


Figure 3: The structure of Light-YOLOv4.

Since high-level features in the channel are related to and share similarities with each other, the CAd is responsible for gaining feature information in the channel. Since the last MSFBlock removes redundant information and emphasizes important information but cannot obtain the global information of the target, the SAd is used to obtain the rich global

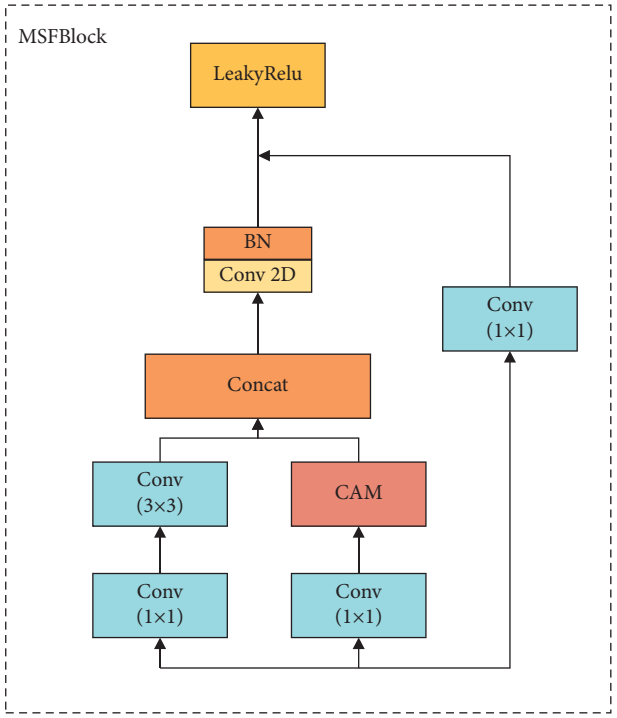


Figure 4: The structure of MSFBlock.

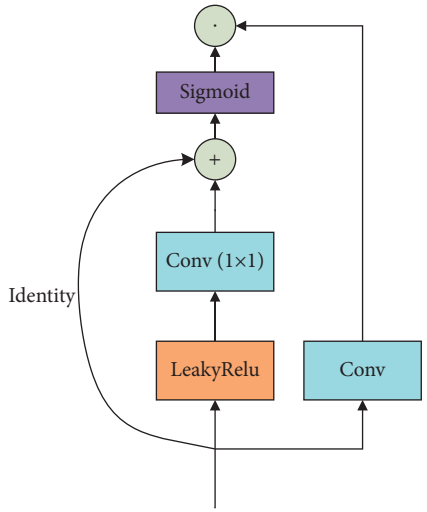


Figure 5: The structure of CAM.

feature information. The AdBlock mines the feature maps from the two dimensions of space and channel, which express the global information of targets comprehensively.

3.3. Hierarchical Multiscale Prediction. Feature Pyramid Network (FPN) improves the accuracy of pedestrian detection algorithms by extracting multiscale features. As shown in Figure 6, YOLOv4-Tiny first extracts and generates features at each level from the bottom up, which are denoted as {C2, C3, C4, C5}. Then, FPN obtains P5 from C5 with the convolution operation and uses top-down upsampling and horizontal connection operations to generate fusion feature P4. However, P4 fails to effectively leverage low-level feature

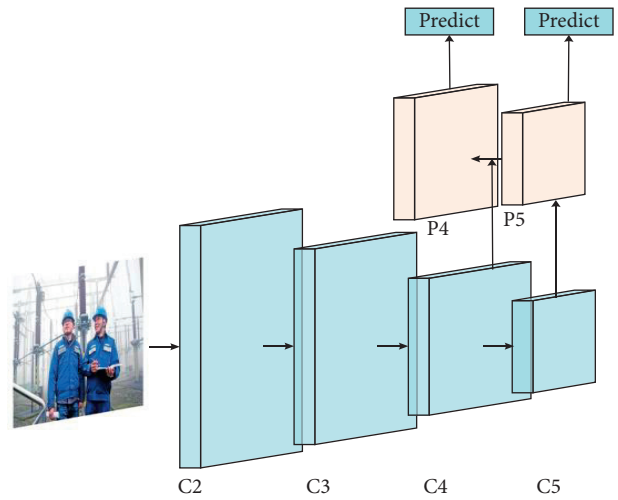


Figure 6: The structure of the FPN network in the YOLOv4-Tiny network.

maps; hence, FPN cannot accurately detect distant pedestrians. We remedy the multiscale prediction method. Specifically, we hierarchically fuse high-level semantic features with low-level features to make full use of all-scale feature maps, which improves the detection accuracy of distant pedestrians.

In order to fully utilize all-scale feature maps, we conduct the downsampling operation on C2 and C3, which are on a large scale, and upsampling C4 and C5, fourfold downsampling C2, and twofold downsampling C3. Then, we concatenate the results horizontally and reduce the dimensionality of the result to feature map N1 with the 1×1 convolutional layer. Similarly, we concatenate the results of C4 and C5 and reduce the dimensionality of the result to feature map N2 with the 1×1 convolutional layer. Finally, we concatenate N1 and N2 horizontally and reduce the dimensionality with a 1×1 convolutional layer to obtain a feature map {P4, P5} with more detailed information, which was shown in Figure 7. The structure in Figure 7 combines the low-level with the high-level semantic features, which can enrich the fine-grained information without increasing the amount of calculation and improve the detection of distant pedestrians without losing real-time performance.

3.4. Pedestrian Segmentation. Considering the requirements of the real-time performance in segmentation, we choose the SOLO [14] as the segmentation algorithm. SOLO introduces the concept of "instance category" to distinguish object instances. Specifically, the "instance category" refers to the quantized center position and the size of objects, which makes it possible to segment objects using their positions. SOLO is in an end-to-end manner and transforms coordinates into the problems of classification with discrete quantization. Hence, it has good real-time performance and is usually used with YOLO detectors.

SOLO directly distinguishes instances through the center position and object size, which are denoted as location and sizes, respectively. SOLO uses the location to allocate which

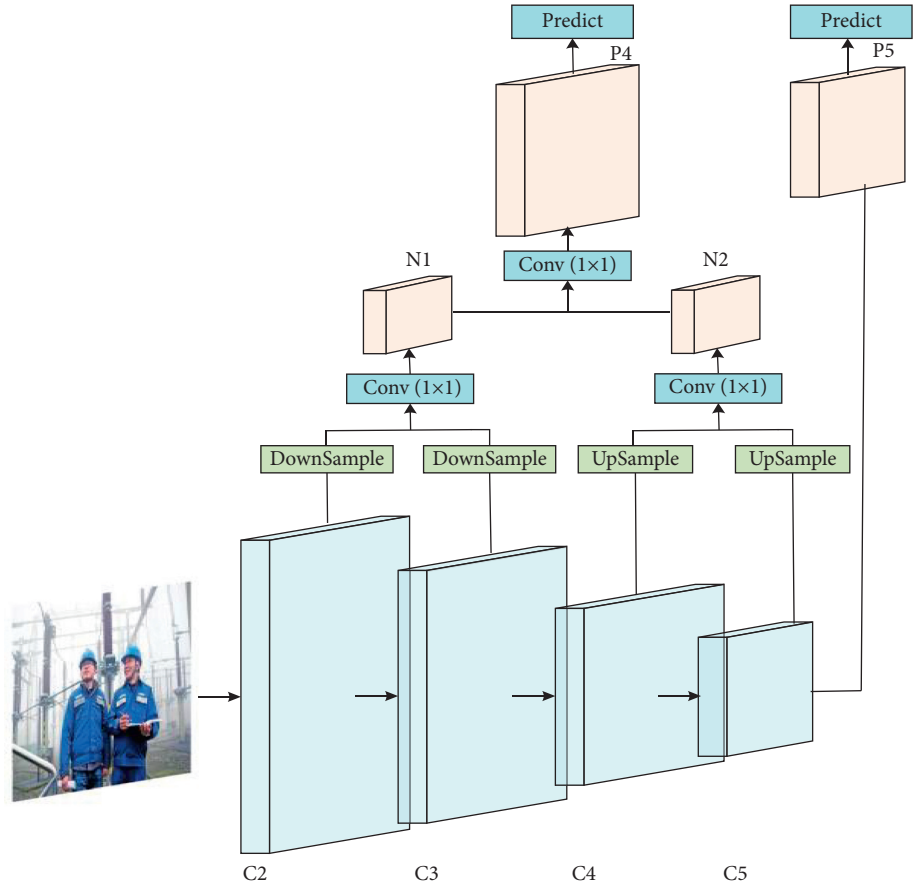


Figure 7: The network of Hierarchical FPN.

channel the instance should fall into and uses the FPN to solve the size problem. The specific process is shown in Figure 8.

SOLO is similar to the method in YOLO, which first separates the input image into $S \times S$ grids. The output is divided into two branches: category branch and mask branch. The size of the category branch is SSC, and C is the number of semantic categories. The mask branch size is HWS^2, and S^2 is the maximum number of predicted instances, which is corresponding to the position of the original image from top to bottom and from left to right. When the center of the pedestrian object falls into a grid, the corresponding position of the category branch and the corresponding channel of the mask branch are responsible for the prediction. We adopt the FPN to solve the problems of sizes. The large feature maps of FPN predict small pedestrians, and the small feature maps predict large pedestrians. FPN can also alleviate the problem of overlapping, and instances with different sizes will be assigned to different FPN output layers for prediction.

The SOLO algorithm also uses CoordConv to enhance the processing of location information. CoordConv concatenates two channels directly on the original tensor, storing the x and y coordinates, respectively, and normalizing them to $[-1, 1]$, which explicitly bring the position information into the next convolution operation. Meanwhile, CoordConv improves Decoupled head to solve the problem of too large output channels caused by too many grid cell settings. The final prediction process is that we first extract the predicted probability values of all category branches and use a threshold (such as 0.1) to filter the predicted values. Then, we obtain the i, j index corresponding to the remaining classification positions and divide the i channel in the X branch and the j channel in the Y branch using elementwise multiplication to obtain the mask of this category. Moreover, we use a threshold (such as 0.5) to filter the mask and conduct NMS on all masks. Finally, the mask is scaled to the size as original images to obtain the instance segmentation of the pedestrian, which is shown in Figure 9.

4. Pedestrian Tracking

In this section, we adopt the Kalman filter to predict the trajectory of the pedestrian based on the coordinate information detected by Light-YOLOv4 and the performance information (cosine similarity). Besides, we leverage the metric learning to match the predicted trajectory and the movement of the pedestrian, combining the performance information to realize the tracking.

4.1. Kalman Filter. The Kalman filter selects the pedestrian as the tracking object. The midpoint $(dx\ (k), dy\ (k))$ on the bottom edge of the detection box is taken as the tracking

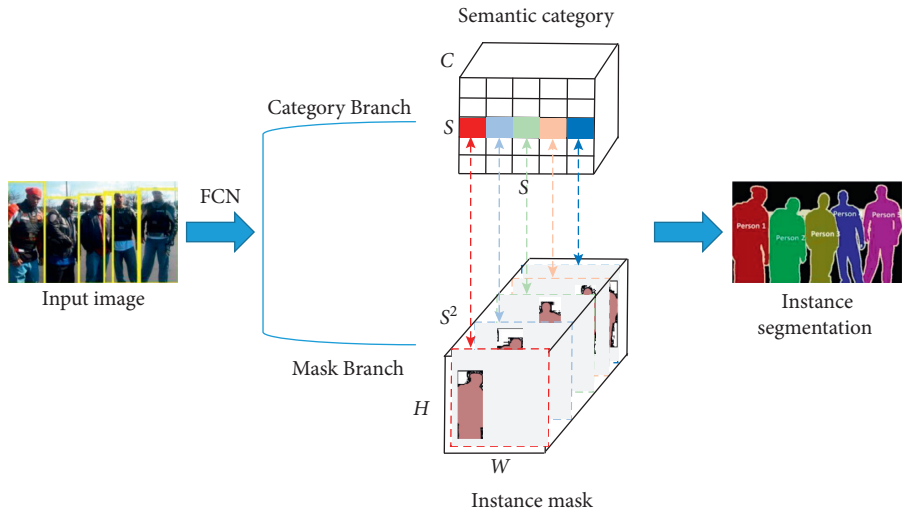


Figure 8: The processing of the SOLO algorithm.

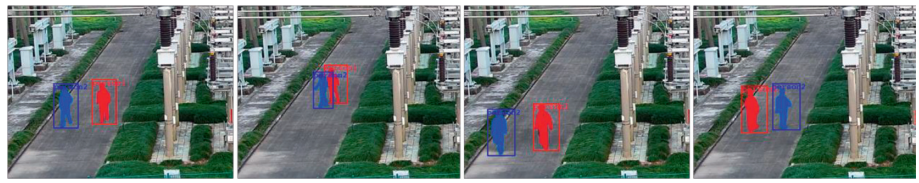


Figure 9: Visually results of the segmentation.

feature point, and the length w and height h of the detection box are selected as the other two feature variables. The above four variables form a four-dimensional state variable. Then, the workflow of the Kalman filter algorithm is used to predict the four-dimensional state variables of the detection box.

First, from the state at time k, the state function at time $k-1$ can be derived, as shown in

$$x_k = Fx_{k-1} + Bu_k + w_k, \tag{1}$$

where x_k refers to the state variable at time k, and F is a $n \times n$ gain matrix of the state variable. U_k is composed of c-dimensional vectors of input control. B is the $n \times c$ matrix related to input control and state change. The variable w_k represents the process noise.

Secondly, since the measured z_k may be equal to the state variable x_k, or may not to, the calculation of measured z_k is shown in

$$z_k = H_k x_k + v_k, \tag{2}$$

where H_k is an $m \times n$ observation matrix, and v_k represents the measurement noise.

Assume that the elements in w_k follow the Gaussian distribution $N(0, Q_k)$, where Q_K refers to the $n \times n$ covariance matrix. The elements in v_k follow the Gaussian distribution $N(0, R_k)$ and R_k represents the $m \times m$ covariance matrix.

First, we calculate the prior probability estimate of the current state $\hat{x}_k^-$ and use the superscript "–" to indicate "before the new measurement." The calculation of $\hat{x}_k^-$ is shown in

$$\hat{x}_k^- = F\hat{x}_{k-1} + Bu_{k-1} = w_k. \tag{3}$$

P_k^- denotes the error covariance, and P_k^-'s prior probability estimate at time k is obtained from its value at time $k-1$, the calculation of which is shown in

$$P_k^- = FP_{k-1}F^T + Q_{k-1}. \tag{4}$$

Equations (3) and (4) constitute the prediction part of the filter, from which the Kalman gain coefficient can be obtained, as shown in the following formulation:

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1}. \tag{5}$$

From equation (5), the optimal observation value and the corrected error covariance can be obtained, which is shown in equations (6) and (7).

$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - H_k\hat{x}_k^-\right), \tag{6}$$

$$P_k = P_k^- - P_k^- K_k H_k. \tag{7}$$

The workflow of the Kalman filter is generally sorted into two stages: one is the prediction stage and the other is the update stage of prediction results. The two stages are illustrated in Figure 10. In the prediction stage, the system state $\hat{x}_{k-1}$ at time $k-1$ is input into equations (2) and (3) to calculate a prior probability estimate $\hat{x}_k^-$ and measured value z_k. In the update stage of the prediction result, the Q_{k-1} at

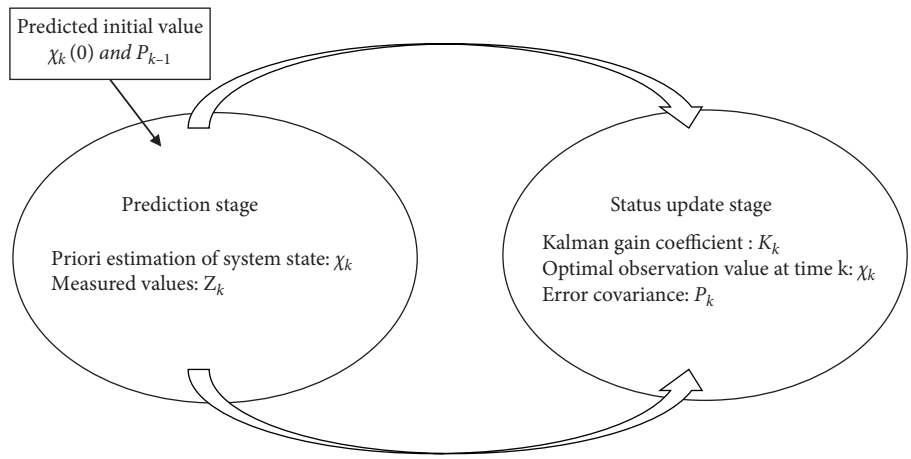


Figure 10: The workflow of the Kalman filter.

time $k-1$, the covariance P_{k-1}, and equation (4) are input to obtain a prior estimate P_k^- of the covariance at time k and then input P_k^- and RK at time k into equation (5) to obtain the Kalman gain coefficient K_k. Next, according to equations (6) and (7), the optimal observation value $\hat{x}_k$ and the corrected error covariance P_k are calculated. Then, we return to equation (2) and start the next pedestrian detection. Since the midpoint of the bottom edge $d_x(k)$ and $d_y(k)$ of the pedestrian detection frame and the length w and height h of the detection frame update every time, the initial position of $\hat{x}_k$ is the coordinates $(d_x(k), d_y(k))$, and the initial values of the length w and height h of the detection frame are the length and height of the pedestrian detection frame at the beginning, respectively. And the initial values of $\hat{x}_k(0)$, F, and H_k are shown in

$$x_k(0) = \begin{bmatrix} d_x(k) \\ d_y(k) \\ w \\ h \end{bmatrix},$$
$$F = \begin{bmatrix} 1 & 0 & \mathrm{d}t & 0 \\ 0 & 1 & 0 & \mathrm{d}t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (8)$$
$$H_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

4.2. Motion Information. Considering that the detection target in the current frame is composed of a four-dimensional vector, we adopt the Mahalanobis distance to measure the similarity between the current and historical trajectories of the target pedestrian. Mahalanobis distance represents the covariance distance of data, and it effectively calculates the similarity between two multidimensional pedestrians by taking into account the correlation between various features of the target.

Given a multivariable $t = (t_1, t_2, t_3, \ldots, t_p)^T$, the mean value and the covariant matrix are $\mu = (\mu_1, \mu_2, \mu_3, \ldots, \mu_p)^T$ and S, respectively. The formulation of t's Mahalanobis distance is shown as

$$D_M(t) = \sqrt{(t-\mu)^T S^{-1}(t-\mu)}. \quad (9)$$

In this paper, we calculate the Mahalanobis distance $M(i-1, i)$ of target personnel in i-th frame and in $(i-1)$-th frame, which is represented as

$$M(i-1, i) = \sqrt{(t_i - g_{i-1})^T S_{i-1}^{-1}(t_i - g_{i-1})}, \quad (10)$$

where t_i represents the state $(d_x(k), d_y(k), w, h)$ in #i frame. g_{i-1} denotes the prediction of i-th frame in $(i-1)$-th frame. S_{i-1} is the covariance matrix of target trajectory predicted by the Kalman filter algorithm in i-th frame. Since the motion in the video frame is continuous, we use $M(i-1, i)$ to filter the target and set 3.08 as the threshold of filtering. Equation (11) shows the filtering rule; filter represents the filtering function.

$$a(i-1, i) = \begin{cases} \text{filter}\{M(i-1, i) \le 3.08\}, & i \ge 2, \\ 0, & 0 < i \le 0. \end{cases} \quad (11)$$

4.3. Performance Information. When pedestrians are occluded or crossed, the identities of the pedestrians change frequently. Therefore, only using the motion information to track pedestrians will greatly reduce the accuracy. And thus, it is necessary to combine the segmentation to reidentify the performance information of the pedestrian, which remedies tracking errors caused by motion information and improves the accuracy of tracking.

In this paper, we adopt a deep convolutional neural network to extract the pedestrian masks segmented in each frame and utilize the cosine similarity as a metric function. The cosine distance is calculated for the feature vector of the object b extracted from the segmentation result of the i-th frame and the average value of the feature vector corresponding to the pedestrian segmentation result after f times

of successful tracking before the i-th frame. The cosine distance calculation is in

$$\cos \text{dis} = 1 - \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}. \tag{12}$$

Different from other distance algorithms that directly measure between two points, the cosine distance algorithm calculates the cosine value of the angle of two vectors and normalizes the result to [0, 1], which is not related to absolute value and can measure the similarity between vectors. Therefore, we use the cosine distance algorithm as the measurement function, in order to calculate the matching degree of the appearance of the pedestrians in the current frame and previous frame.

The feature extraction network used in this article is a deep convolutional neural network, consisting of three convolutional layers, a maximum pooling layer, and four residual layers. Since online training takes a long time, we pretrained the network on a large-scale pedestrian dataset in advance to obtain an appearance model. The features are suitable for tracking. Then, we use the pedestrian segmentation result as the input of the network. And the output feature is a 256-dimensional vector.

Let $r_i = \{r_1^{(b)}, r_2^{(b)}, \ldots, r_k^{(b)}\}$ denote feature vectors $r_k^{(b)}$ of object b before i-th frame. In continuous tracking, the performance information of pedestrians may change due to the long tracking duration. Therefore, we keep the average value of the feature vector corresponding to the pedestrian mask after f times of successful tracking before the i-th frame of the object b, which is shown in equation (13). And the cosine distance calculation is shown in equation (14):

$$\overline{r}_k^{(b)} = \frac{\sum_{k=1}^{f} r_k^{(b)}}{f}, \tag{13}$$

$$\cos \text{dis}\,(k, i) = 1 - r_i^T \overline{r}_k^{(b)}. \tag{14}$$

Similarly, the measurement of cosine distance needs a threshold th, which is obtained through training. When the cosine distance is less than a specific threshold th, two objects are associated. The calculation of the measurement is shown in

$$c\,(k, i) = \text{filter}\{\cos \text{dis}\,(k, i) \le \text{th}\}. \tag{15}$$

4.4. Integrated Metric Learning. We set a weight coefficient K and obtain a weighted average of Mahalanobis distance and cosine distance through:

$$u = Ka\,(i - 1, i) + (1 - K)c\,(k, i). \tag{16}$$

4.5. Solving the Occlusion. The occlusion of the trajectory for a long duration will lead to the uncertainty of Kalman filter prediction. If there are multiple trackers matching the detection results of the same target at the same time, then the position of the trajectory will not update for a long time, which will cause a relatively large deviation in the position predicted by the Kalman filter. It can be calculated from equations (5)–(7) that the covariance will become larger. Since the Mahalanobis distance is proportional to the reciprocal of the covariance, the Mahalanobis distance will be small, which makes the detection result more likely associated with the longer trajectory, causing a decrease in the performance of the tracking algorithm. We set a frame time threshold, which means that if the difference between the current frame time and the frame time that the target successfully matched last time is greater than the threshold, the target trajectory is considered as terminated. Furthermore, the target trajectory will be deleted in the subsequent tracking. Otherwise, it is considered that the target is not lost.

5. Datasets and Experimental Results

In order to verify the effectiveness and generalization of the algorithm, we train and test the proposed network on a widely used dataset. The training of this article includes 3 stages: first is pretraining the Light-YOLOv4 network in the widely used tracking dataset, second is training the segmentation branch on the video target segmentation dataset, and third is regulating the weight of the segmentation branch network by pretraining the appearance model on a large-scale pedestrian dataset.

5.1. The Training of Light-YOLOv4 Network. The images in the dataset are selected from large target tracking datasets such as OBT50, COCO [15], and ImageNet [16]. Since constantly updating the training samples is time-consuming, we choose a graphics processing server with high performance. The proposed method is implemented on a PC with Intel (R) Core (TM) i7-8086K CPU@4.00 GHz, x64-based processor, 64 G memory, RTX3090 ∗ 4 GPU, Linux operating system. The whole experiments are implemented on the Pytorch framework, using the stochastic gradient descent optimization algorithm, totally training 60 epochs, and the batch_size is set to 8, every epoch of which input 8000 photos for training. The training samples are shown in Figure 11.

5.2. The Training of Segmentation Branch. The segmentation branch only extracts the mask of the pedestrian. We distinguish the background and the foreground in the picture and set them to 0 and 1. In order to better distinguish people and objects, we train the network on a large number of different types of datasets, such as Open Images V4 and ImageNet. The Open Images V4 dataset contains 1.9 million images, 600 categories, and 15.4 million bounding-box annotations. It is currently the largest dataset with object location annotation information. The ImageNet dataset has more than 14 million pictures, covering more than 20,000 categories, which has more than one million pictures with clear category annotations and the location of objects. We train the network with the location of bounding boxes generated by the Light-YOLOv4. We uniformly adjust the image size to 64×64. Some training samples are shown in Figure 12.

Figure 11: The training samples of Light-YOLOv4.

Figure 12: Samples of the segmentation.

5.3. The Training of the Appearance Model. We select the Market-1501 [17] and MSMT17 [18] as the pedestrian reidentification datasets to guarantee the generalization capabilities of the proposed appearance model. The feature extraction network is a deep convolutional neural network. The Market-1501 dataset contains a large number of identities, each of which is taken by six disjoint cameras, and separated into a training set with 625 people and a test set with 636 people. The Market-1502 also includes 2793 false alarms from DPM, which are used as interference factors in simulated real scenes. Samples of Market-1502 are shown in Figure 13.

Figure 13: Samples of MSMT-1501 dataset.

MSMT17 is a large-scale reidentification dataset collected on a campus with 12 outdoor cameras and 3 indoor cameras. MSMT17 selects data from 4 days with different weather conditions in a month, and 3 hours of video is collected every day, covering three time periods: morning, noon, and afternoon. MSMT17 has similar points of view with Market-1501, but the scenarios are much more complicated. Sample images of MSMT17 are shown in Figure 14.

5.4. Experimental Results. We test the proposed model on MOT19, and the experimental results are shown in Figure 15, which demonstrates the tracking results of multiple moving objects. As we can see from the experimental results, the proposed algorithm has better accuracy when tracking multiple objects, and the accuracy will not be affected in the case of small objects. Besides, the proposed algorithm has satisfied the high requirement in terms of real-time. In complex situations such as occlusion and pedestrian crossing, the algorithm still maintains a high tracking accuracy without losing the target. We also compare the proposed model with several state-of-the-art tracking algorithms (such as: KCF [19], STRN [20], INARLA [21], Tracktor [22], STAM [23], and AMIR [24]).

5.4.1. Evaluation Criteria. We adopt several evaluation metrics to verify the effectiveness and availability of tracking algorithms. The calculation of the multiple objects tracking accuracy (MOTA) is shown in

$$\mathrm{MOTA} = 1 - \frac{\mathrm{FN} + \mathrm{FP} + \mathrm{IDSW}}{\mathrm{GT}} \in (-\infty, 1], \tag{17}$$

where FN represents false negatives, which is the sum of the number of false negatives in the entire video. FP refers to false positives, which is the sum of the number of false positives in the whole video. IDSW means the ID Switch. The smaller the value is, the greater the effectiveness is. GT is the sum of the number of ground truth objects in the video. The closer the MOTA is to 1, the better the performance of the tracker is. Due to the existence of the number of hops, there is a possibility that MOTA is less than 0. MOTA prevailingly indicates all matching errors of objects in tracking. MOTA intuitively measures the performance of the detection algorithm in detecting objects and keeping trajectories. Besides, MOTA is unrelated to detection accuracy.

The calculation of multiple objects tracking precision (MOTP) is shown in

$$\mathrm{MOTP} = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t}, \tag{18}$$

where c_t represents the number of matching in the t-th frame, and the matching error $d_{t,i}$ is the IOU of the detection frame in the t-th frame and the GT.

The calculations of Recall and Precision are formulated as $P = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$, $R = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$, where TP represents true positives, which is defined as the number of true answers in positive samples. FP represents false positives, which is defined as the number of false answers in positive samples. TN represents true negatives, which is defined as the number of true answers in negative samples. FN represents false negatives, which is defined as the number of false answers in negative samples.

The formulation of Identification Precision (IDP) is shown in equation (19), which refers to the accuracy of ID identification in each bounding box of the pedestrian.

$$\mathrm{IDP} = \frac{\mathrm{IDTP}}{\mathrm{IDTP} + \mathrm{IDFP}}, \tag{19}$$

where IDTP and IDFP represent the amount of ID's TP and FP, respectively.

Equation (20) illustrates the identification recall, which refers to the recall of ID identification in each bounding box of the pedestrian.

$$\mathrm{IDR} = \frac{\mathrm{IDTP}}{\mathrm{IDTP} + \mathrm{IDFN}}, \tag{20}$$

where IDFN represents the number of ID's FN.

IDF_1 means the identification F-score, which refers to the F-score of ID identification in each bounding box of the pedestrian.

$$\mathrm{IDF}_1 = \frac{2\mathrm{IDTP}}{2\mathrm{IDTP} + \mathrm{IDFP} + \mathrm{IDFN}}. \tag{21}$$

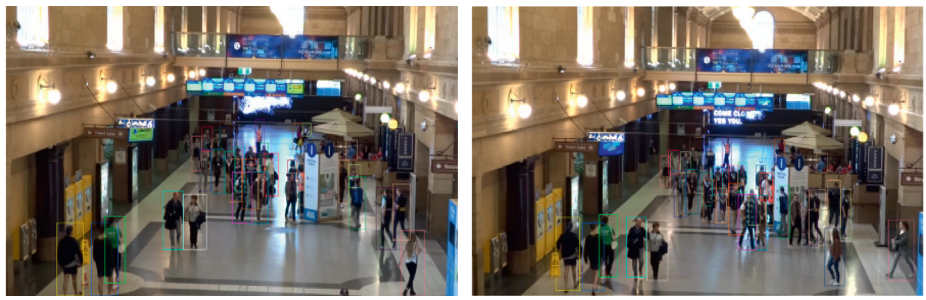
MT is the proportion of trajectories where most of the targets are tracked. A trajectory can be considered as MT if more than 80% of it is tracked. ML is the proportion of trajectories where most of the objects are lost. Only the trajectory tracked less than 20% can be regarded as ML.

IDS is the total number of identity switches. Frag, which is also called FM, refers to fragmentation. And the FPS means frames per second.

5.4.2. Comparison. The comparison of evaluation metrics on MOT19 is shown in Table 1. "↑" means the higher the score is, the better the performance is; "↓" means the opposite.

It can be seen that the MOTA of the proposed algorithm on MOT19 is 0.7% higher than the FFT algorithm, which has the best performance in 6 state-of-the-art methods. The

Figure 14: Samples of MSMT17 dataset.



Frame 7 tracking result | Frame 7 tracking result

Frame 115 tracking result | Frame 181 tracking result

Figure 15: Experimental results of the algorithm on mot19 data set.

Table 1: Comparison effect of experimental model on MOT19 dataset.

Method	MOTA↑	MOTP↑	IDF_1↑	IDP↑	IDR↑	MT↑ (%)	ML↓ (%)	FP↓	FN↓	IDS↓	Frag↓	FPS↑
KCF [19]	48.8	75.7	47.2	65.9	36.7	15.8	38.1	5875	86567	906	**1116**	106
STRN [20]	48.5	73.7	**53.9**	72.8	**42.8**	17.0	34.9	9038	84178	747	2919	90
FFT [21]	56.5	78.1	50.1	64.4	41.1	23.6	29.4	5831	71825	1635	1607	54
Tracktor [22]	54.4	78.2	52.5	71.3	41.6	19.0	36.9	3280	79149	682	1480	132
STAM [23]	46.0	74.9	48.1	62.7	38.2	14.6	43.6	6895	91117	473	1422	73
Sadeghian et al. [24]	47.2	75.8	42.6	52.1	36.6	14.0	41.6	**2681**	92856	774	1675	36
Ours	**57.2**	**78.3**	51.7	**75.2**	38.8	**26.1**	**23.6**	6913	**25753**	**384**	1431	152

results in Table 1 show that the proposed method improves the accuracy of object tracking. Meanwhile, we optimize the structure of the detection and segmentation algorithm. While elevating the accuracy, the improved structure can also reduce the amount of calculation. Therefore, the proposed tracking algorithm can not only work at a fast speed but meet the requirements of practical projects. The speed of the proposed model is 152 frames per second on a single graphics card, which can fully achieve real-time detection and tracking. Under the same conditions, the fastest tracking speed of the FFT algorithm is only 5 frames per second. The Tracktor algorithm has only 11 frames per second. Our model has the ability to accurately detect, segment, and accurately track objects. Moreover, our model has the basic characteristic of universality.

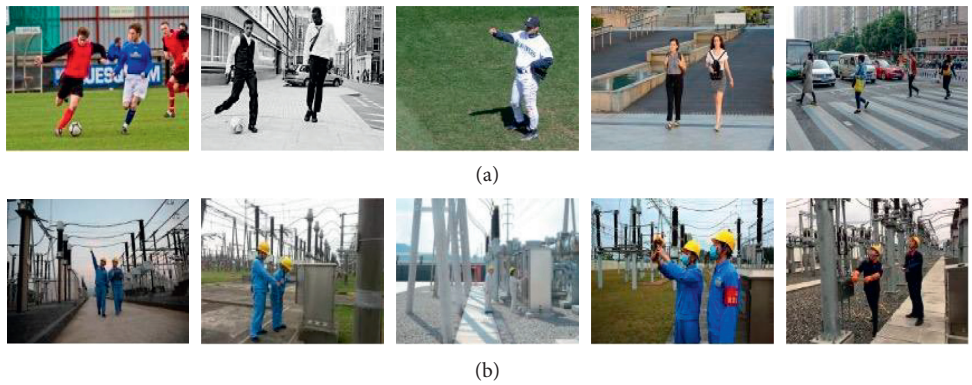
6. The Application in the Substation

Considering the requirements of real-substation situations, we expand the dataset. Test videos come from the real-world scenarios in the substation. Taking pedestrians in the substation as an example, the collected substation monitoring videos are divided into two groups with the same lighting but different occlusion conditions to evaluate the performance of the moving object tracking method. The first group of videos is used to evaluate the tracking performance when there are fewer obstructions. The second group of videos is used to test the robustness and accuracy of the tracking method in the case of serious obstructions. The detailed parameters are shown in Table 2. The entire experiment is implemented on the Pytorch framework, using the stochastic gradient descent optimization algorithm, a total of 60 epochs are trained, batch_size is set to 8, the initial learning rate is set to 0.001, and the number of images trained for each epoch is 15,073.

6.1. Dataset. Due to the complicated environment in the substation, we convert the surveillance video of the pedestrian in the substation into a picture frame by frame in order to more accurately detect the position of the pedestrian in the image and ensure the tracking effect. In addition,

Table 2: Statistics of substation scene data set.

Video	Video time (s)	Number of frames	Average number of staff members	Illumination, camera downward tilt angle	Occlusion
Group 1	41	1640	5	Strong illumination, 30° downward	There are few obstructions and the environment is relatively open
Group 2	52.8	2112	2	Strong illumination, 30° downward	There are few obstructions and the environment is relatively open



(a)

(b)

Figure 16: Data set example. (a) Common dataset example. (b) Example of the substation dataset.

in order to ensure the generalization of the model, we select the same number of pictures containing pedestrians from the public dataset and then mix the pictures to form the detection dataset, and the number of datasets is about 15,073. Next, we divide the dataset into a training set, validation set, and test set, the number of images in the training set is 9,011, the validation set is 3,041, and the test set is 3,021. We annotate the images according to the image annotation standard of the Pascal VOC dataset and output only the person category, which is also counted at the same time (for example, person1, person2, person3, person4, ...) as the pedestrian ID for the initial frame tracking. Examples of the dataset and the labeled dataset are shown in Figures 16 and 17.

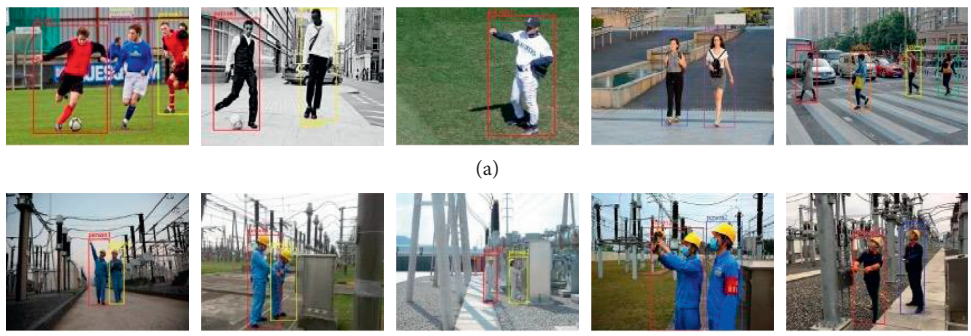
6.2. Results and Analysis. We test the proposed tracking model on two datasets containing substation surveillance videos, as shown in Figures 18 and 19. Figures 18 and 19 show the tracking results on the first and second groups of videos (when the occlusion is not serious), and the detection and tracking results are output every 2.5 s (100 frames). Experimental results prove that the proposed pedestrian tracking method in substations using integrated metric learning can effectively detect and track multiple pedestrians in the substation and has better robustness in the severely occluded environment (the second group of substation surveillance videos).

We can see from Figure 18 that under the circumstance that the occlusion is not serious, the proposed tracking method can accurately track multiple working pedestrians in the substation. When a small amount of pedestrians is crossed, overlapped, and occluded, the proposed tracking method still accurately reidentifies multiple working pedestrians.

There are more insulators, poles, and towers in the second group of surveillance videos of the substation, which causes dense obstruction. And the proposed algorithm still accurately tracks pedestrians in the case of severe occlusion. The visually tracking results are shown in Figure 19. Since our model can involve both the motion information and the performance information of pedestrians, it boosts the reidentification ability for the problem of pedestrians' loss. Tables 3 and 4 represent the extensive experiments under occlusion in two degrees.

It can be seen from the two tables that no matter there are light or heavy occlusions in the video, introducing more training data, adaptive segmentation training methods, and fusion measurement methods can significantly improve the tracking effect of pedestrians. The MOTA, MOTP, IDP, ML, and IDS results of our model are better than other algorithms. Moreover, in the light obstruction scenes, the MOTA of our model improves 1.6% compared to the FFT. In a scene with severe obstructions, the MOTA of our model increases to 4.7%, and the IDS of our model keeps the lowest.

In addition, since we adopt the lightweight Light-YOLOv4 real-time detection network framework and fast segmentation algorithm, the proposed algorithm guarantees a fast running speed. We can see from the experimental results that the running speed of the adaptive polymorphic fusion tracking algorithm is 134 FPS in a scene with light obstructions and 127 FPS in a scene with dense obstructions. Compared with other algorithms, the real-time performance of our method has been greatly improved. Experiments show that the adaptive polymorphic fusion tracking algorithm proposed in this paper can not only ensure tracking accuracy



(a)

(b)

Figure 17: Example of the annotated data set. (a) Example of the annotated common dataset. (b) Example of the annotated substation dataset.

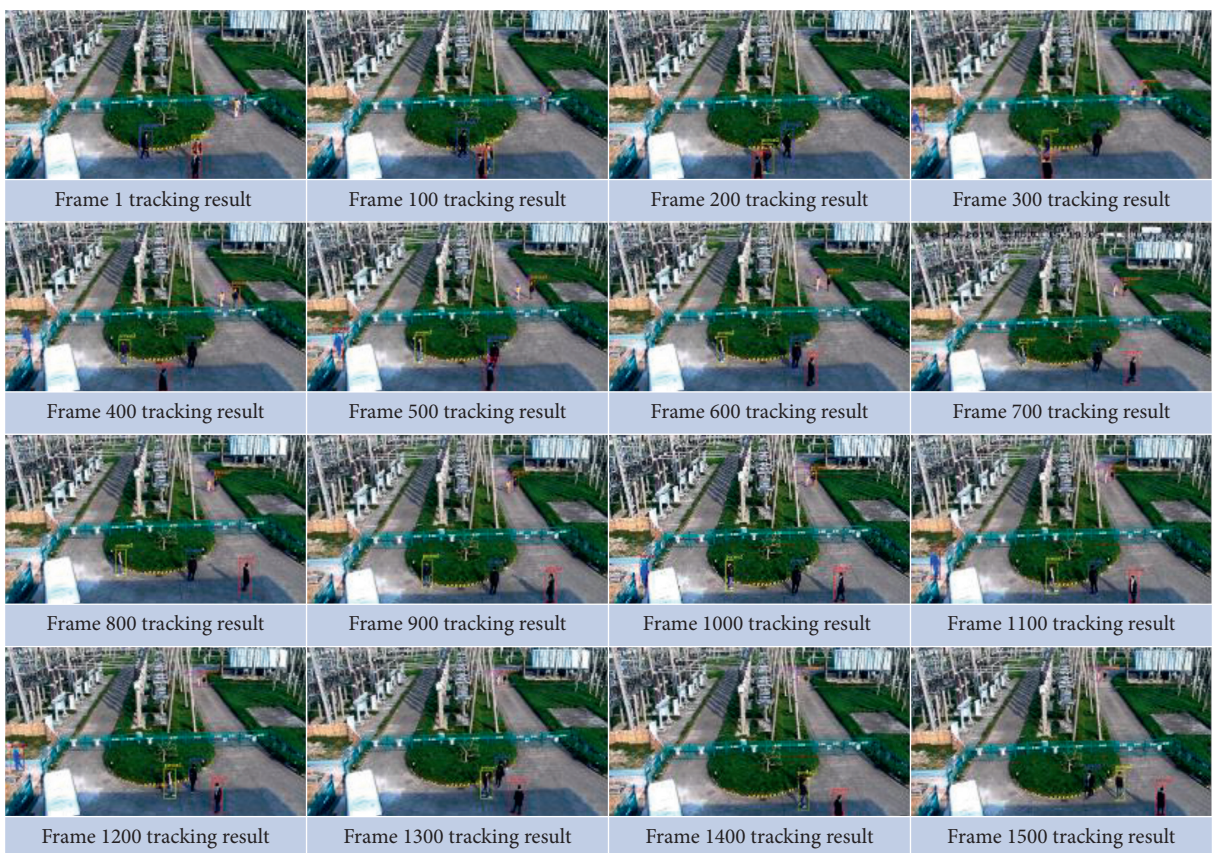


Figure 18: Tracking effect of this algorithm in the first group of videos (occlusion is not serious).

but also has extremely fast running speed. Our algorithms also solve the problems of dense occlusion and pedestrian loss by fusing pedestrian motion and performance information. Considering the actual scenarios and demands of substations, the proposed algorithm is more practical.

We also tested the proposed model on MOT19. The experimental results are shown in Figure 15 in Section 5.4. Under complex conditions such as occlusion and pedestrian crossing, the algorithm can still maintain high tracking accuracy without losing the target. Therefore, our algorithm

Figure 19: Tracking effect of this algorithm in the second group of videos (serious occlusion).

Table 3: Tracking performance of tracker under the first group of videos (less occlusion).

Method	MOTA↑	MOTP↑	IDF$_1$↑	IDP↑	IDR↑	MT↑ (%)	ML↓ (%)	FP↓	FN↓	IDS↓	Frag↓	FPS↑
KCF [19]	48.9	75.8	47.3	66.1	36.9	16.9	39.1	5865	86567	906	**1116**	97
STRN [20]	48.6	73.8	**54.1**	72.9	**43.9**	17.0	35.9	9038	84178	747	2919	74
FFT [21]	56.6	78.2	50.2	64.5	42.1	23.6	30.4	5931	69825	1635	1607	42
Tracktor [22]	54.5	78.3	52.6	71.4	42.6	20.0	37.9	3365	79249	682	1480	114
STAM [23]	46.1	74.4	48.2	62.9	39.2	15.6	45.6	6796	91137	473	1422	68
Sadeghian et al. [24]	47.3	75.9	42.7	52.2	37.6	15.0	42.6	**2693**	92856	774	1675	33
Ours	**57.2**	**78.3**	51.7	**75.3**	38.8	**26.1**	**25.6**	5913	**25753**	**384**	1431	**134**

Note. The data marked in bold indicates the optimal algorithm of each index corresponding to each video.

Table 4: Tracking performance of the tracker under the second group of videos (more obstructions).

Method	MOTA↑	MOTP↑	IDF$_1$↑	IDP↑	IDR↑	MT↑ (%)	ML↓ (%)	FP↓	FN↓	IDS↓	Frag↓	FPS↑
KCF [19]	43.7	64.7	42.2	48.9	34.7	14.8	38.1	5972	92352	1023	**1237**	92
STRN [20]	42.4	62.5	**50.9**	50.8	**42.5**	15.7	34.9	9934	87230	843	3326	73
FFT [21]	49.5	67.1	48.1	46.4	40.3	21.2	29.4	6027	75292	1843	1849	41
Tracktor [22]	50.4	63.2	49.5	49.3	40.3	17.4	36.9	4980	80422	704	1593	112
STAM [23]	41.0	63.9	45.1	41.7	35.5	13.4	43.6	6895	97739	536	1744	67
Sadeghian et al. [24]	42.2	62.8	39.6	39.1	32.4	14.2	41.6	**4581**	10745	832	1763	31
Ours	**51.2**	**68.3**	49.7	**54.2**	36.7	**22.3**	**28.6**	7613	**47762**	**523**	1631	**127**

Note. The data marked in bold indicates the optimal algorithm of each index corresponding to each video.

has general promotion potential, but the current research is based on a single camera. Cross-camera and multicamera scenes are the directions of our follow-up research.

7. Conclusion

In this paper, we have carefully designed an adaptive polymorphic fusion-based fast tracking algorithm, which integrates motion and performance information of pedestrians to achieve real-time tracking in substations. First, we proposed a lightweight real-time target detection method called Light-YOLOv4, the backbone network of which is based on CSPDarknet-Tiny network to enhance the performance of detection from three aspects, containing multibranch feature fusion, grouped self-attention, and hierarchical multiscale prediction. Second, we exploited the SOLO segmentation algorithm to gain the performance information by inputting the fine masks of a pedestrian into the deep convolutional neural network. Third, we not only adopted the Kalman filter to predict the trajectory of the detected pedestrian but also used the Mahalanobis distance to represent the matching degree between the current state and the previous state of pedestrians and denoted the motion information with the motion matching degree. Finally, we obtained the tracking results using the integrated metric learning method. Experimental results prove that the proposed algorithm has better real-time tracking performance, and it can solve the severe occlusions in the substation. However, there are still some problems in real-world scenarios, such as the cross-camera tracking scene, the night, and other scenes. For future work, we expect to investigate more in the above challenging scenarios [10].

Data Availability

Figures 8, 11–14, 16(a), and 17(a) were taken from public datasets: (1) Figure 8 was taken from the public dataset PASCAL VOC2012, which you can download from this website: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/. (2) Figure 11 was taken from the public dataset INRIA-Person, which you can download from this website: https://pascal.inrialpes.fr/data/human/. (3) Figure 12 was taken from the public dataset DAVIS 2016, which you can download from this website: https://davischallenge.org/. (4) Figure 13 was taken from the public dataset Market-1501, which you can download from this website: https://drive.google.com/file/d/0B8-rUzbwVRk0c054eEozWG9COHM/view?usp=sharingttps://www.pkuvmc.com/publications/msmt17.html. (5) Figure 14 was taken from the public dataset MSMT17, which you can download from this website: https://www.pkuvmc.com/publications/msmt17.html. (6) Figure 15 was taken from the public dataset MOT19, which you can download from this website: https://motchallenge.net. (7) Figure 16(a) was taken from the public dataset PASCAL VOC2012, which you can download from this website: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/. (8) Figure 17(a) was taken from the public dataset PASCAL VOC2012, which you can download from this website: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] S. Fang, X. Shu, and L. I. Dewu, "A method based on machine vision for opening-closing status recognition of substation disconnecting switches," *Journal of Hunan University of Technology*, vol. 31, 2017.

[2] J. Gabela, G. Retscher, S. Goel et al., "Experimental evaluation of a UWB-based cooperative positioning system for pedestrians in GNSS-denied environment," *Sensors*, vol. 19, no. 23, p. 5274, 2019.

[3] S. S. Blackman and S. Samuel, "Abstracts of previous tutorials in this series: multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 31, no. 3, pp. 90–96, 2016.

[4] L. Wan, K. Liu, Y.-C. Liang, and T. Zhu, "DOA and polarization estimation for non-circular signals in 3-D millimeter wave polarized massive MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 20, no. 5, pp. 3152–3167, 2021.

[5] H. Wang, L. Xu, Z. Yan, and T. A. Gulliver, "Low-complexity MIMO-FBMC sparse channel parameter estimation for industrial big data communications," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3422–3430, 2021.

[6] H. Wang, X. Li, R. H. Jhaveri et al., "Sparse Bayesian learning based channel estimation in FBMC/OQAM industrial IoT networks," *Computer Communications*, vol. 176, pp. 40–45, 2021.

[7] L. Wan, L. Sun, K. Liu, X. Wang, Q. Lin, and T. Zhu, "Autonomous vehicle source enumeration exploiting non-cooperative UAV in software defined internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3603–3615, 2021.

[8] L. Wan, Y. Sun, L. Sun, Z. Ning, and J. J. P. C. Rodrigues, "Deep learning based autonomous vehicle super resolution DOA estimation for safety driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4301–4315, 2021.

[9] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, 2016.

[10] L. Chen, H. Ai, Z. Zhuang, and C. Shang, "Real-time multiple people tracking with deeply learned candidate selection and person re-identification," in *Proceedings of the IEEE International Conference on Multimedia & Expo*, San Diego, CA, USA, 2018.

[11] Z. Wang, L. Zheng, Y. Liu, Y. Li, and S. Wang, *Towards Real-Time Multi-Object Tracking*, Springer, Cham, Switzerland, 2020.

[12] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "FairMOT: on the fairness of detection and re-identification in multiple object tracking," *International Journal of Computer Vision*, vol. 129, pp. 3069–3087, 2021.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 2017.

[14] X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "Solo: segmenting objects by locations," in *Proceedings of the European Conference on Computer Vision*, pp. 649–665, Glasgow, UK, 2020.

[15] T.-Y. Lin, M. Maire, S. Belongie et al., "Microsoft coco: common objects in context," in *Proceedings of the European Conference on Computer Vision*, pp. 740–755, Zurich, Switzerland, 2014.

[16] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009.

[17] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: a benchmark," in *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, 2015.

[18] L. Wei, S. Zhang, W. Gao, and Q. Tian, "Person transfer gan to bridge domain gap for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018.

[19] P. Chu, H. Fan, C. C. Tan, and H. Ling, "Online multi-object tracking with instance-aware tracker and dynamic model refreshment," in *Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, USA, 2019.

[20] J. Xu, Y. Cao, Z. Zhang, and H. Hu, "Spatial-temporal relation networks for multi-object tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Republic of Korea, 2019.

[21] J. Zhang, S. Zhou, X. Chang et al., "Multiple object tracking by flowing and fusing," 2020, https://arxiv.org/abs/2001.11180.

[22] P. Bergmann, T. Meinhardt, and L. Leal-Taixé, "Tracking without bells and whistles," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Republic of Korea, 2019.

[23] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu, "Online multi-object tracking using CNN-based single object tracker with spatial-temporal attention mechanism," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 2017.

[24] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking the untrackable: learning to track multiple cues with long-term dependencies," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 2017.