ISPRS Journal of Photogrammetry and Remote Sensing 177 (2021) 116–130

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Review Article

Cross-frame keypoint-based and spatial motion information-guided networks for moving vehicle detection and tracking in satellite videos

Jie Feng [a,*], Dening Zeng [a], Xiuping Jia [b], Xiangrong Zhang [a], Jie Li [c], Yuping Liang [a], Licheng Jiao [a]

[a] Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an, Shaanxi Province 710071, China
[b] School of Engineering and Information Technology, University of New South Wales, Canberra, ACT 2612, Australia
[c] Space Platform Business Division, Shanghai Aerospace Electronic Technology Institute, Shanghai 201109, China

ARTICLE INFO

Keywords:
Deep learning
Keypoint-based detection
Moving vehicle detection
Multi-object tracking
Satellite videos

ABSTRACT

Deep learning methods have achieved the state-of-the-art performance of object detection and tracking in natural images, such as keypoint-based detectors and appearance/motion-based trackers. However, for small and blurry moving vehicles in satellite videos, keypoint-based detectors cause the missed detection of keypoints and incorrect keypoint matching. In terms of multi-object tracking, it is difficult to track the crowded similar vehicles stably only by using the appearance or motion information. To address these problems, a novel deep learning framework is proposed for moving vehicle detection and tracking in the satellite videos. It is comprised of the cross-frame keypoint-based detection network (CKDNet) and spatial motion information-guided tracking network (SMTNet). In CKDNet, a customized cross-frame module is designed to assist the detection of keypoints by exploiting complementary information between frames. Furthermore, CKDNet improves keypoint matching by incorporating size prediction around the keypoints and defining the soft mismatch suppression for out-of-size keypoint pairs. Based on high-quality detection, SMTNet can track the densely-packed vehicles effectively by constructing two-branch long short-term memories. It extracts not only spatial information of the same frame by considering the relative spatial relationship of neighboring vehicles, but also motion information among consecutive frames by calculating the movement velocity. Especially, it regresses virtual positions for missed or occluded vehicles and keeps on tracking these vehicles while they reappear. Experimental results on Jilin-1 and SkySat satellite videos demonstrate the effectiveness of the proposed detection and tracking methods.

1. Introduction

As a new type of ground observation data, the satellite video (Ao et al., 2019) contains both static and dynamic information from the ground (Wang et al., 2020). It is a continuous image sequence and obtained by the optical sensors from video satellites gazing at specific areas for a certain time. Compared with the natural images, satellite videos contain more abundant temporal information and larger observation areas, which have been successfully used for disaster response, resource census, precision agriculture and dynamic traffic monitoring (Ao et al., 2019; Kopsiaftis and Karantzalos, 2015; Li et al., 2020a). Especially for dynamic traffic monitoring, satellite videos show obvious superiority and record the entire trajectory of the vehicles over a vast territory.

Recently, moving object detection and tracking have become research hotspots in the satellite video processing and analysis. In traditional moving object detection methods, moving object detection is considered as the foreground and background segmentation problem. The foreground represents the temporally changed pixels while background describes relatively stable pixels. These traditional methods are divided into three main categories: optical flow, background subtraction and frame-difference. Optical flow methods (Roy and Bhowmik, 2020) extract the moving object regions according to the distribution characteristics of optical flow fields. The estimation of optical flow fields generally needs a large amount of calculation, which limits the applications of optical flow methods in satellite videos. In comparison with optical flow methods, background subtraction and frame-difference methods are simpler and more efficient. A series of background subtraction methods extract the background from the videos firstly, and then obtain the moving objects by subtracting the background from the current frame, such as Gaussian mixture model (GMM) (Stauffer and

* Corresponding author.
E-mail address: jiefeng0109@163.com (J. Feng).

https://doi.org/10.1016/j.isprsjprs.2021.05.005
Received 19 August 2020; Received in revised form 18 March 2021; Accepted 7 May 2021
Available online 19 May 2021

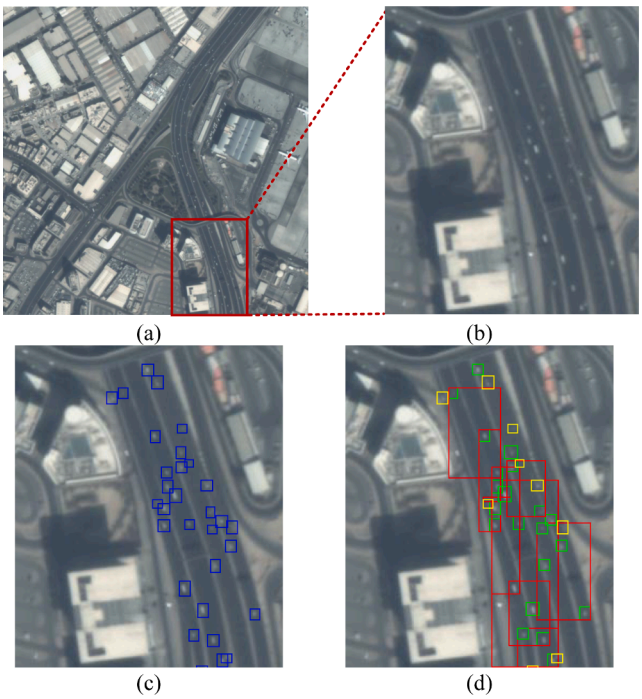


Fig. 1. An example of moving vehicle detection in Jilin-1 satellite video. (a) A local area of Dubai in a frame of the satellite video. (b) A partial enlarged image of (a). (c) The ground-truth of moving vehicle detection. (d) The detection result of CornerNet, where the missed detections are represented in yellow, the correct detections are in green and the false detections are in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Grimson, 1999; Zivkovic, 2004), visual background extraction (ViBe) (Barnich and Van Droogenbroeck, 2010), online low-rank and structured sparse decomposition (O-LSD) (Zhang et al., 2020a), and extended low-rank and structured sparse decomposition (E-LSD) (Zhang et al., 2019). Frame-difference methods (Roy and Bhowmik, 2020) detect the moving objects by seeking out the pixels with differences in adjacent frames. Unfortunately, most of background subtraction and frame-difference methods are easily affected by moving background (Ao et al., 2019) and brightness and contrast changes, which commonly exist in satellite videos.

Compared with traditional methods, deep neural networks significantly improve the performance of object detection in natural images because of powerful feature representation ability (Benenson et al., 2013). In previous works, most popular detection networks are anchor-based methods (Sharma and Mir, 2020). These methods design the anchor boxes with pre-defined sizes and aspect ratios, then classify and regress them to obtain the bounding boxes of objects. Some recent anchor-based methods (Li et al., 2020b) detect the objects in the remote sensing imagery. Among these methods, (Deng et al., 2017) proposed a new Faster R-CNN to improve the detection performance for small objects by combining hierarchical feature maps. YOLT (Van Etten, 2018) is a improved version of YOLO v2 (Redmon and Farhadi, 2017), which is specifically designed for object detection of the aerial remote sensing imagery.

Recently, keypoint-based methods have been proposed for object detection. Compared with many anchor-based detectors (Sharma and Mir, 2020), such as Faster R-CNN and YOLO v2, keypoint-based methods are anchor-free detectors that eliminate the need for designing anchor boxes. Besides, these methods alleviate the imbalance between positive and negative samples, and avoid the complicated computation caused by a large number of anchor boxes. The representative keypoint-based methods include CornerNet (Law and Deng, 2018), CenterNet (Duan et al., 2019), ExtremeNet (Zhou et al., 2019b). CornerNet predicts the heatmaps, embedding vectors and offsets for a pair of corners. These embeddings are used to determine whether top-left and bottom-right corners belong to the same object. The locations of the corners are slightly modified by the offsets. Based on CornerNet, CenterNet uses a triplet of keypoints (two corner keypoints and one center keypoint) to detect the objects, which improves both the precision and recall (Duan et al., 2019).

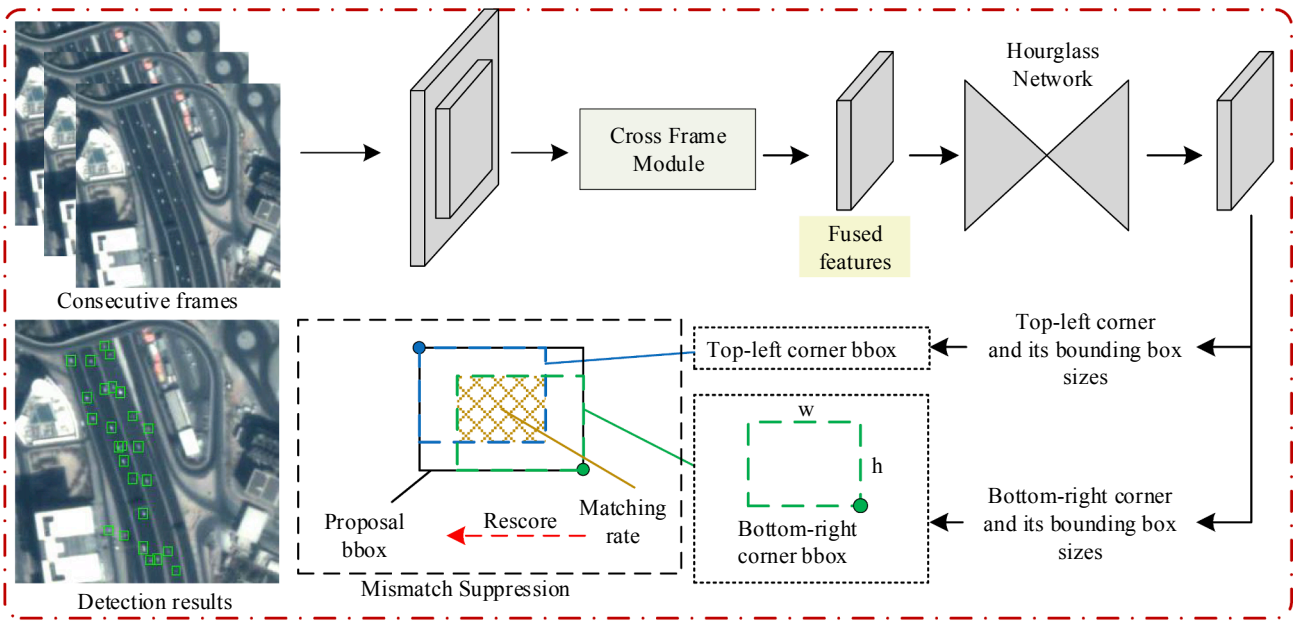
Unfortunately, there are some challenges in directly applying keypoint-based methods to detect moving vehicles in satellite videos. As shown in Fig. 1, there is an example of moving vehicle detection in the Jilin-1 satellite video. Fig. 1(a) and (b) show an area in a frame of the original satellite video and its partial enlarged image. Fig. 1(c) and (d) represent the ground-truth and the detection result obtained by CornerNet for moving vehicles. As shown in Fig. 1(b), vehicles almost lack texture, color and other appearance information caused by their small sizes in satellite videos. In Fig. 1(d), false detection and missed detection are represented in red and yellow, respectively. CornerNet predicts some similar embeddings for different vehicles. Thus, there are some false detections because of the incorrect keypoint matching caused by the similar embeddings, as shown in Fig. 1(d). Additionally, some vehicles are blurry in satellite videos, which greatly increases the difficulty of vehicle detection in satellite videos. The blurry vehicles blend into the background, which causes the missed detection of corners in CornerNet.

Based on deep neural network, some researchers have developed the methods for moving object detection of satellite videos. Li et al. (2019) proposed an improved Faster-RCNN based on frame difference for moving vehicle detection. In (Li et al., 2019), fused features are firstly obtained by extracting both the motion and pixel information, then these features are fed into the head of Faster-RCNN. However, the method in (Li et al., 2019) has the same problem as the anchor-based methods, and its detection performance would deteriorate due to the imbalance between positive and negative samples. In (LaLonde et al., 2018; Pflugfelder et al., 2020), FoveaNet was proposed to estimate a heatmap for the centroid location of all the moving objects via a spatio-temporal CNN. In FoveaNet, it is difficult to distinguish the objects that partially overlap with other objects by using the centroid heatmap.

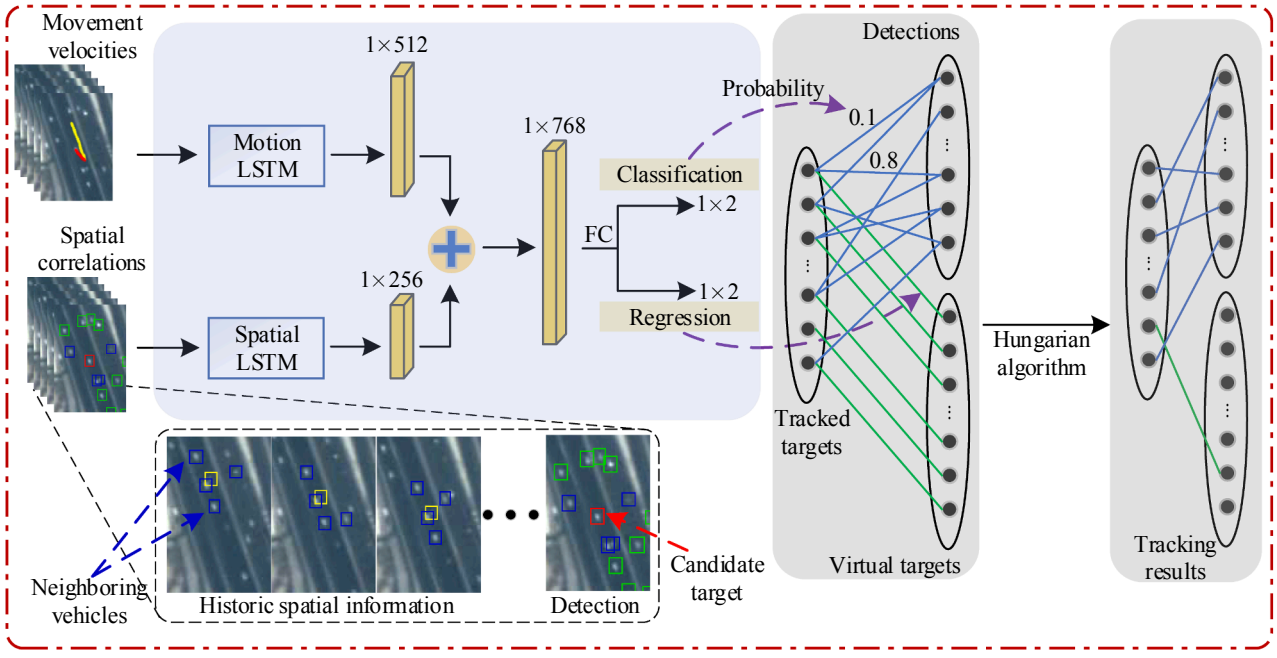
In terms of multi-object tracking, tracking-by-detection (TBD) is the most commonly-used method. TBD generates the trajectory by associating each measurement derived from the detection results (Bergmann et al., 2019). Most previous works on TBD fall into two categories: tracking based on existing detection results, and joint optimization of detection and tracking (Lu et al., 2020).

The first type of method accomplishes multi-object tracking via an association process based on existing detection results. The association process firstly learns the cost of the tracked trajectories and detection results, and then uses the graph (Kuhn, 1955) to achieve data association. Multiple hypothesis tracking (MHT) (Kim et al., 2015) is a data association algorithm, which uses Kalman filtering to model the motion information for tracking objects. However, the prior knowledge is required for the motion model in MHT, such as object dynamics and clutter distributions. In (Milan et al., 2017), recurrent neural network with the Hungarian algorithm (RNN_HA) was proposed for multi-object tracking, which achieves higher efficiency without any prior knowledge. Later, TT17 (Zhang et al., 2020b) uses long short-term memory network to learn long-term features, which can generate more complete trajectories. Graph neural network (GNN) (Scarselli et al., 2008) was proposed to optimize the graph structure instead of using Hungarian algorithm. Several related works (Brasó and Leal-Taixé, 2020; Jiang et al., 2019) designed end-to-end GNNs to solve the data association problem. These methods achieve the data association via optimizing the graph structure directly. Unfortunately, when the objects are similar and crowded in the satellite videos, these methods would encounter difficulties in linking the detection hypotheses into trajectories by only using the motion or appearance model.

Another type of method combines the object detection and tracking



(a) cross-frame keypoint-based detection network (CKDNet)



(b) spatial motion information-guided tracking network (SMTNet)

Fig. 2. A novel deep learning framework for moving vehicle detection and tracking in the satellite videos. Architecture of (a) CKDNet and (b) SMTNet.

into a single end-to-end network (Peng et al., 2020). Many researches proposed the deep neural network methods that simultaneously detect the objects, and learn the linking features, e.g. re-ID features, appearance features or motion features, for data association. These methods save the inference time for multi-object tracking. Additionally, the tracking results of these methods are not affected by the baseline detection results. CTracker (Peng et al., 2020) was proposed by using a joint multiple-object detection and tracking framework. It predicts the paired bounding boxes of the same object in adjacent frames, and combines ID verification to track multiple objects. Based on some anchor-free methods, FairMOT (Zhang et al., 2020c) achieves detection and tracking via training the object detection branch and re-ID branch. CenterTrack (Zhou et al., 2020) used CenterNet (Zhou et al., 2019a) as the detection module, and added an extra branch to regress the spatial distance of track-detection center points for per-frame data association. However, for the objects lacking the appearance information and occluded objects, it is difficult for these methods to detect and associate well under the conditions of large displacement.

In this paper, a novel deep learning framework is proposed for moving vehicle detection and tracking in the satellite videos. It consists of a cross-frame keypoint-based detection network (CKDNet) and a spatial motion information-guided tracking network (SMTNet). In CKDNet, several video frames are directly fed in the cross-frame module (CFM). CFM extracts the appearance features of the current frame and the temporal features between different frames simultaneously. Based on the fused features, the hourglass network is constructed to generate the heatmaps for a pair of keypoints (top-left corner and bottom-right corner). Unlike the CornerNet, the corresponding bounding box sizes for each corner are also predicted to calculate a matching rate. Besides, soft or hard mismatch suppression (MMS) strategy is devised to suppress

the unreliable bounding boxes under different conditions. In soft-MMS, a gaussian penalty function is defined by nonlinearly transforming the matching rate to improve the detection precision greatly while ensuring the recall. SMTNet designs two-branch long short-term memories (LSTMs). One LSTM is constructed to extract the spatial information between neighboring vehicles in the same frame. It is beneficial to stably track the densely-packed vehicles in the absence of appearance information. The other LSTM establishes the motion model of vehicles via the movement velocity in consecutive frames. SMTNet judges whether a hypothetical trajectory is true by combining the motion information with the spatial information. In addition, SMTNet regresses a new virtual position for the tracked vehicle simultaneously. The main contributions of this paper can be summarized as follows.

(1) To consider the special characteristic of moving vehicles in satellite videos, a novel deep learning framework is devised to guarantee the high-quality detection and stable and long-term tracking.
(2) CFM extracts fused features for moving vehicle detection by using both the temporal and appearance information. These extracted fused features are insensitive to moving background, brightness and contrast changes in the satellite videos.
(3) CKDNet devises the hard and soft mismatch suppression based on predicted sizes of bounding boxes for corners. It significantly improves the precision through eliminating the unreliable bounding boxes caused by incorrect keypoint matching in CornerNet.
(4) To deal with crowded and similar vehicles, SMTNet considers both the spatial and motion information. Even if some vehicles are lost or occluded in several frames, when these vehicles appear again, SMTNet can effectively associate the vehicles with the original trajectories.

The rest of this paper is organized as follows. The architecture of the novel deep learning framework for moving vehicle detection and tracking is described in Section 2. In Section 3, the experimental results and analysis of moving vehicle detection and multi-object tracking on the satellite video datasets are reported. Finally, Section 4 gives a few concluding observations and some suggestions for future work.

2. CKDNet and SMTNet

In this section, we firstly introduce a novel deep learning framework, which is consist of CKDNet and SMTNet. In CKDNet, CornerNet is selected as the baseline detection network because it avoids manually-designed anchors and is easy to implement in satellite videos. Based on CornerNet, cross-frame module (CFM) is devised to improve the recall by using the complementary information among frames. Moreover, soft or hard mismatch suppression is defined to improve the detection precision. SMTNet combines the spatial information of the same frame with the motion information among consecutive frames. It can associate each detection result to obtain the trajectories of vehicles effectively.

2.1. Overview of the proposed detection and tracking algorithms

Fig. 2 shows the proposed deep learning framework for moving vehicle detection and tracking in the satellite videos. In CKDNet, fused features are extracted by the CFM through the consecutive frames, and fed into next hourglass-based backbone network. After the hourglass network, some convolutional layers are used instead of the corner pooling in CornerNet to output the heatmaps, offsets and sizes. Then, CKDNet matches the top-left and bottom-right corners to generate the proposal bounding boxes. In order to obtain high-quality detection results, soft or hard mismatch suppression (MMS) strategy is devised to re-score the proposal bounding boxes via a matching rate. After rescoring the proposal bounding boxes, the bounding boxes of vehicles with high confidence are retained.

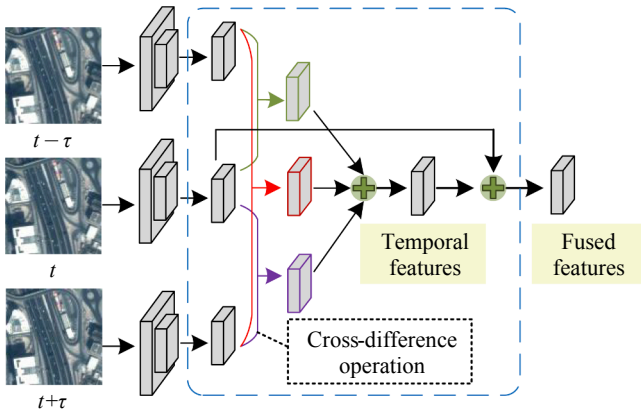


Fig. 3. In CFM, temporal information is extracted from several frames by using the cross-difference operation. The features extracted by CFM combine the temporal with appearance information, which is beneficial for the detection of blurry vehicles.

Two essential differences between CKDNet and CornerNet are listed as follows. 1) CFM in CKDNet can remove the background interference and enhance the features for blurry moving vehicles effectively. Compared with CKDNet, CornerNet is designed for object detection in images, which lacks the usage of the temporal information in the satellite videos. 2) CKDNet uses MMS to match the corners instead of embedding in CornerNet. CKDNet significantly improves the precision through using the MMS for out-of-size corner pairs.

In terms of tracking, SMTNet is constructed by using two-branch LSTMs. As shown in Fig. 2(b), given a hypothetical trajectory for a tracked vehicle, SMTNet would predict a probability and regress a new virtual position. After obtaining the predicted probabilities of all the hypothetical trajectories and the virtual positions of the tracked targets in current frame, a set of trajectories for all the tracked targets are generated via the Hungarian algorithm (Kuhn, 1955).

2.2. Cross-frame keypoint-based detection network (CKDNet)

In the satellite videos, the appearance information of moving vehicles is weak because vehicles are very small in the whole scene. Thus, it is necessary to combine temporal information among consecutive frames with appearance information for blurry and small moving vehicle detection in satellite videos. CKDNet based on CFM and MMS is proposed for moving vehicle detection in satellite videos.

2.2.1. Cross-frame module

To make full use of temporal and appearance information, CFM is designed as shown in Fig. 3. It is placed before the backbone network to obtain the fused features from the temporal and appearance information in the satellite videos. Let $I = \{I_1, I_2, \cdots, I_n\}$ denote the satellite video, where $I_t \in \mathbb{R}^{W\times H\times 3}$ $(1 \leqslant t \leqslant n)$ denotes the image of the t - th frame and n represents the number of frames in the satellite video. To obtain the fused features of the current frame I_t, consecutive frames $I_s = \{I_{t-\tau}, I_t, I_{t+\tau}\}$ are fed into CFM. In CFM, convolutional layers with a stride of 2 are stacked to firstly extract the appearance features $\{F_{t-\tau}, F_t, F_{t+\tau}\}$ for each frame. Then, the difference features $F_{(t,t-\tau)}$ between $F_{t-\tau}$ and F_t are obtained after a cross-difference operation as follows:

$$F_{(t,t-\tau)} = CD(F_t, F_{t-\tau}) = W_{cd}\cdot(F_t - F_{t-\tau}) \quad (1)$$

where $CD(\cdot)$ denotes the cross-difference operation, and W_{cd} is a convolution operation with 3×3 convolution filter, followed by a tanh activation function. After obtaining all the difference features, temporal features f_t are extracted by aggregating the temporal information in the

cross frame:

$$f_t = W_1 \cdot concat(F_{(t,t-\tau)}, F_{(t,t\ +\ \tau)}, F_{(t-\tau,t+\tau)}) \tag{2}$$

where $concat(\cdot)$ denotes a concatenation operation, and W_1 is a convolution operation with 3×3 convolution filter. The temporal features f_t can provide abundant temporal information from consecutive frames. Finally, the fused features are obtained via concatenating the temporal features f_t and the appearance features F_t followed by the convolution operation W_2. The convolution operations W_1 and W_2 are followed by a batch normalization strategy and a rectified linear unit (ReLU) (Nair and Hinton, 2010). The fused features of temporal and appearance information can significantly reduce the missed detection for small and blurry moving vehicles in the satellite videos.

2.2.2. Mismatch suppression

As CornerNet, the location of the corner and its score are obtained via the heatmap, and the location is adjusted through its corresponding offset in CKDNet. Let L_{tl} and L_{br} denote the top-left corners and bottom-right corners. L_{tl} and L_{br} are defined as:

$$\begin{aligned} L_{tl} &= \{(x_{tl}, y_{tl}, s_{tl})_i | i = 1, ..., n1\} \\ L_{br} &= \{(x_{br}, y_{br}, s_{br})_i | i = 1, ..., n2\} \end{aligned} \tag{3}$$

where $(x., y.)$ is the coordinate of the corner on the feature map and $s.$ is the confidence score. $n1$ and $n2$ are the number of top-left corners and bottom-right corners, respectively.

Different from CornerNet, CKDNet predicts the corresponding bounding box size for each corner. Given the sizes $Q_{tl} = \{(\mathrm{w_{tl}}, \mathrm{h_{tl}})_i | \mathrm{i} = 1, ..., \mathrm{n1}\}$ for top-left corners, the predicted bounding boxes of top-left corners are generated as

$$\begin{aligned} BB_{tl} &= \{\ \mathrm{BB_{tl_i}} | \mathrm{i} = 1, ..., \mathrm{n1}\} \\ &= \{(\mathrm{x_{tl}}, \mathrm{y_{tl}}, \mathrm{x^{'}_{br}}, \mathrm{y^{'}_{br}}, \mathrm{s_{tl}})_i | \mathrm{i} = 1, ..., \mathrm{n1}\} \\ &= \{(\mathrm{x_{tl}}, \mathrm{y_{tl}}, \mathrm{x_{tl}} + \mathrm{w_{tl}}, \mathrm{y_{tl}} + \mathrm{h_{tl}}, \mathrm{s_{tl}})_i | \mathrm{i} = 1, ..., \mathrm{n1}\} \end{aligned} \tag{4}$$

where $(x^{'}_{br}, y^{'}_{br})$ is the corresponding bottom-right corner generated by the top-left corner and its predicted size. Similarly, given the sizes $Q_{br} = \{(\mathrm{w_{br}}, \mathrm{h_{br}})_i | \mathrm{i} = 1, ..., \mathrm{n2}\}$ for bottom-right corners, the predicted bounding boxes of bottom-right corners are generated as

$$\begin{aligned} BB_{br} &= \{BB_{br_i} | i = 1, ..., n2\} \\ &= \{(x^{'}_{tl}, y^{'}_{tl}, x_{br}, y_{br}, s_{br})_i | i = 1, ..., n2\} \\ &= \{(x_{br} - w_{br}, y_{br} - h_{br}, x_{br}, y_{br}, s_{br})_i | i = 1, ..., n2\} \end{aligned} \tag{5}$$

The proposal bounding boxes BB are constructed by matching all the pairwise corners as follows:

$$BB = \{(x_{tl}, y_{tl}, x_{br}, y_{br}, s_{bb})_k | k = 1, ..., n1 \times n2\} \tag{6}$$

where (x_{tl}, y_{tl}) and $(x_{br}, y_{br})$are the coordinates of the proposal bounding boxes' top-left and bottom-right corners, respectively. s_{bb} denotes the confidence score of the proposal bounding box, and $s_{bb} = (s_{tl} + s_{br})/2$.

In this way, the number of proposal bounding boxes is excessive. Among these proposal bounding boxes, there are lots of mismatches of the top-left and bottom-right corners. For the mismatch bounding boxes, their scores are unreliable. In this case, the proposal bounding boxes are incorrect even if the scores are high. Thus, it is necessary to reset the scores of proposal bounding boxes and prune the incorrect proposal bounding boxes.

Inspired by soft non-maximum suppression (NMS) (Bodla et al., 2017), mismatch suppression (MMS) method is proposed to re-score the proposal bounding boxes by defining a hard or soft penalty function based on the matching rate. The matching rate mr_k is calculated as:

$$mr_k = IoU(BB_{tl_i}, BB_{br_j}) \ , \qquad (1 \leqslant i \leqslant n1, \quad 1 \leqslant j \leqslant n2) \tag{7}$$

where $IoU(\cdot)$ is the intersection over union (IoU) between predicted bounding boxes.

Based on the matching rate mr_k, two kinds of MMSs are defined according to different conditions.

Hard-MMS: In some conditions, it is more inclined to the detection results with high precision rather than high recall. In hard-MMS, the lower matching rate means the bounding box is more unreliable, even if its score is high. Under these conditions, hard-MMS removes the bounding boxes whose matching rates are lower than a threshold N_t. In Hard-MMS, a hard penalty function is defined as:

$$s_{bb_k} = \begin{cases} 0, & mr_k < N_t \\ s_{bb_k}, & mr_k \geqslant N_t \end{cases} \tag{8}$$

Soft-MMS: Hard-MMS may remove the correct bounding boxes that have lower matching rates, which would decrease the recall greatly. Thus, hard-MMS is unsuitable for the conditions that require the high recall. To adapt to these conditions, soft-MMS is designed. In soft-MMS, a linear penalty function is defined as:

$$s_{bb_k} = \begin{cases} s_{bb_k} mr_k, & mr_k < N_t \\ s_{bb_k}, & mr_k \geqslant N_t \end{cases} \tag{9}$$

Unfortunately, hard and linear penalty functions are not continuous. The non-continuous penalty functions may lead to the abrupt changes for the scores of the proposal bounding boxes (Bodla et al., 2017). Thus, another gaussian penalty function is defined in soft-MMS.

$$s_{bb_k} = s_{bb_k} e^{\frac{(1-mr_k)^2}{\sigma}} \tag{10}$$

where σ is a hyper-parameter to balance the recall and precision.

The Gaussian penalty function is suitable to make a good trade-off between the recall and precision. Compared with hard-MMS, soft-MMS re-scores the proposal bounding boxes that have lower matching rates instead of removing them. In this case, these proposal bounding boxes still have opportunity to be retained. The recall is not seriously influenced.

Compared with the embedding of CornerNet, MMS in CKDNet would lead to slightly large computation cost in both the training and testing stages. In the testing stage, the main time of MMS focuses on the IoU calculation. Although the whole testing time slightly increase, MMS in CKDNet has the potential to improve the detection performance.

Algorithm 1. The procedure of the mismatch suppression.

Input: the predicted bounding boxes of top-left and bottom-right corners BB_{tl} and BB_{br};N_t is the matching rate threshold; $f(\cdot)$ is the penalty function

Begin

$D^{'} \leftarrow \{\ \}$

for BB_{tl_i} in BB_{tl} do

for BB_{br_j} in BB_{br} do

$(x_{tl}, y_{tl}, x_{br}, y_{br}, s_{bb})_k \leftarrow BB_{tl_i}, BB_{br_j}$

$mr_k = IoU(BB_{tl_i}, BB_{br_j})$

if $f(\cdot)$ is hard or linear then

$s_{bb_k} = f(s_{bb_k}, mr_k, N_t)$

else

$s_{bb_k} = f(s_{bb_k}, mr_k)$

end if

$D^{'} \leftarrow D^{'} \cup (x_{tl}, y_{tl}, x_{br}, y_{br}, s_{bb})_k$

end for

end for

$D_t \leftarrow$ topk $D^{'}$

End

Output: the final bounding boxes D_t in the t - th frame

The detailed procedure of MMS is provided in Algorithm 1. The final bounding boxes D_t are obtained by rescoring the proposal bounding boxes via hard or soft MMS in the t - th frame.

2.2.3. The loss function of CKDNet

In CKDNet, the moving vehicles of the current frame are predicted through the current frame and its adjacent frames. In the training stage, the loss function is defined as

$$L_{CKDNet} = L_{det} + \alpha_1 L_{\text{off}} + \alpha_2 L_{\text{size}} \tag{11}$$

where α_1 and α_2 are the weights for the offset and size losses, respectively. In Eq. (11), L_{det} and L_{off} are the loss functions for the heatmap and offset as that in CornerNet (Law and Deng, 2018).

In Eq. (12), a new loss function L_{size} is defined for the predicted bounding box sizes of corners. It is calculated as follows:

$$L_{\text{size}} = \frac{1}{N}\sum_{l=1}^{N} L_1(r_l, \hat{r}_l) \tag{12}$$

where $L_1(\cdot)$ is L1 loss function, $\hat{r}_l$ is the predicted size for the l-th corner, and N is the number of corners. The ground-truth size is represented as $r_l = (w_l, h_l)$, where (w_l, h_l) is the width and height of corresponding bounding box for the l - th corner.

2.3. Spatial motion information-guided tracking network (SMTNet)

For the similar and crowded moving vehicles in the satellite videos, how to track multiple vehicles effectively is challenging because of the lack of appearance information. In satellite videos, some potential cues, e.g. the relative position of vehicles in the same frame, are obvious and available, which may be highly practical for tracking. However, existing tracking methods (Ao et al., 2019; Kim et al., 2015; Milan et al., 2017; Wang et al., 2020; Zhang et al., 2018) are unable to fully exploit the spatial information. Hence, how to make full use of the spatial information of vehicles to assist the motion information is worthy of consideration.

To jointly exploit the motion and spatial information, SMTNet is constructed by designing two-branch LSTMs—motion LSTM and spatial LSTM. It is used to predict a probability that a hypothetical trajectory is a true or not and regress a new virtual position for missed or occluded vehicles.

Let T_i^{t-1} denote the trajectory of the i - th tracked vehicle in previous $t-1$ frames, where $T_i^{t-1} = \{bb_i^1, ..., bb_i^{t-1}\}$. Here, $bb_i^{t-1} = (x_i^{t-1}, y_i^{t-1}, w_i^{t-1}, h_i^{t-1})$ denotes the center position, height and width of the i - th tracked vehicle in the $(t-1)$ - th frame, which is obtained from the detection result D_{t-1}. Then, a hypothetical trajectory H_{i_j} of the i - th tracked vehicle in previous t frames is represented as:

$$H_{i_j} = \{bb_i^1, ..., bb_i^{t-1}, bb_i^t\} = \{T_i^{t-1}, bb_{i_j}^t\} \tag{13}$$

where $bb_{i_j}^t$ denotes a bounding box in D_t that is picked as the j - th candidate target for the trajectory T_i^{t-1} in the t - th frame.

Motion LSTM: Motion LSTM is constructed to extract the motion information of the hypothetical trajectory H_{i_j} via its movement velocity. The movement velocity v_i^t from the $(t-1)$ - th frame to the t - th frame of the i - th vehicle is defined as:

Algorithm 2. The procedure of SMTNet for tracking.

Input: all the trajectories T^{t-1} in previous $t-1$ frame, all the detection results $D = \{D_1, ..., D_t\}$, a tracking threshold N_{track}

Begin

$HT \leftarrow \{\}$

for T_i^{t-1} in T^{t-1} do

$vp \leftarrow \{\}$ #a set of virtual positions

for $bb_{i_j}^t$ in D_t do

$H_{i_j} \leftarrow \{T_i^{t-1}, bb_{i_j}^t\}$

$v_i \leftarrow H_{i_j}; \{bb_{i_1}, \cdots, bb_{i_m}\} \leftarrow D$

$d_i \leftarrow H_{i_j}, \{bb_{i_1}, \cdots, bb_{i_m}\}$

$p_j, vp_j \leftarrow SMTNet(v_i, d_i)$

$HT \leftarrow HT \cup (H_{i_j}, p_j); vp \leftarrow vp \cup vp_j$

end for

$(x, y) \leftarrow$ mean $vp; (w, h) \leftarrow T_i^{t-1}$

$bb_i^t \leftarrow (x, y, w, h)$

$H_{i_{j+1}} \leftarrow \{T_i^{t-1}, bb_i^t\}$

$HT \leftarrow HT \cup (H_{i_{j+1}}, N_{track})$

(continued on next column)

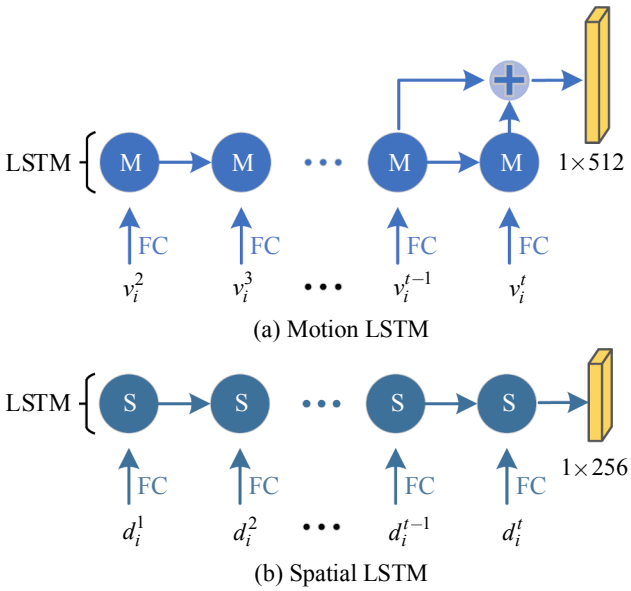


Fig. 4. (a) In motion LSTM, historic and hypothetical motion features are extracted and fused from the movement velocities. (b) In spatial LSTM, the spatial information is extracted from the relative spatial relationship between the tracked vehicle and its neighboring vehicles.

(continued)

end for

$T^t \leftarrow$ Hungarian HT

End

Output: all the trajectories T^t for all the tracked vehicles in previous t - th frame.

$$v_i^t = (vx_i^t, vy_i^t) = (x_i^t - x_i^{t-1}, y_i^t - y_i^{t-1}) \tag{14}$$

Fig. 4 shows the architecture diagrams of motion LSTM and spatial LSTM. As shown in Fig. 4(a), some fully connected layers are applied to extract high-dimensional features from the movement velocities. These high-dimensional features at the time steps $2, ..., t$ are fed into motion LSTM in sequence. At the $(t-1)$ - th moment, the hidden state $mh_{i_j}^{t-1} \in \mathbb{R}^{1\times 256}$ of motion LSTM is obtained, which is historic motion features extracted by previous frames. The historic motion features play an important role in generating accurate virtual positions, which is beneficial for long-term tracking of the missed or occluded vehicles. At the t - th moment, the hidden state $mh_{i_j}^t \in \mathbb{R}^{1\times 256}$ of motion LSTM is hypothetical motion features extracted by the hypothetical trajectory H_{i_j}. Then, $mh_{i_j}^{t-1}$ and $mh_{i_j}^t$ are concatenated to generate the output features for motion LSTM.

Spatial LSTM: To make full use of spatial information, spatial LSTM is constructed. In spatial LSTM, spatial information is extracted by calculating the relative spatial relationship between the tracked vehicle and its neighboring vehicles in the same frame. First, m-nearest neighbor vehicles $\{bb_{i_1}^t, \cdots, bb_{i_m}^t\}$are selected for the i - th tracked vehicle in the t - th frame. Then, the relative spatial relationship is defined by considering the difference between the tracked vehicle and its neighboring vehicles in terms of all the center position, height and width:

$$d_i^t = max(-\gamma, min(\gamma, (bb_i^t - bb_{i_1}^t; ...; bb_i^t - bb_{i_m}^t))) \tag{15}$$

where γ is a scale factor and it is used to suppress excessive difference values, and $(bb_i^t - bb_{i_m}^t) = (x_i^t - x_{i_m}^t, y_i^t - y_{i_m}^t, w_i^t - w_{i_m}^t, h_i^t - h_{i_m}^t)$. Here, d_i^t would be reshaped to $1 \times (4m)$.

Similar to motion LSTM, the features of the relative spatial relationship at the time steps $1, ..., t$ are fed into spatial LSTM after some fully connected layers, as illustrated in Fig. 4(b). From the 1 - th to the t - thmoments, the historic spatial information of moving neighboring

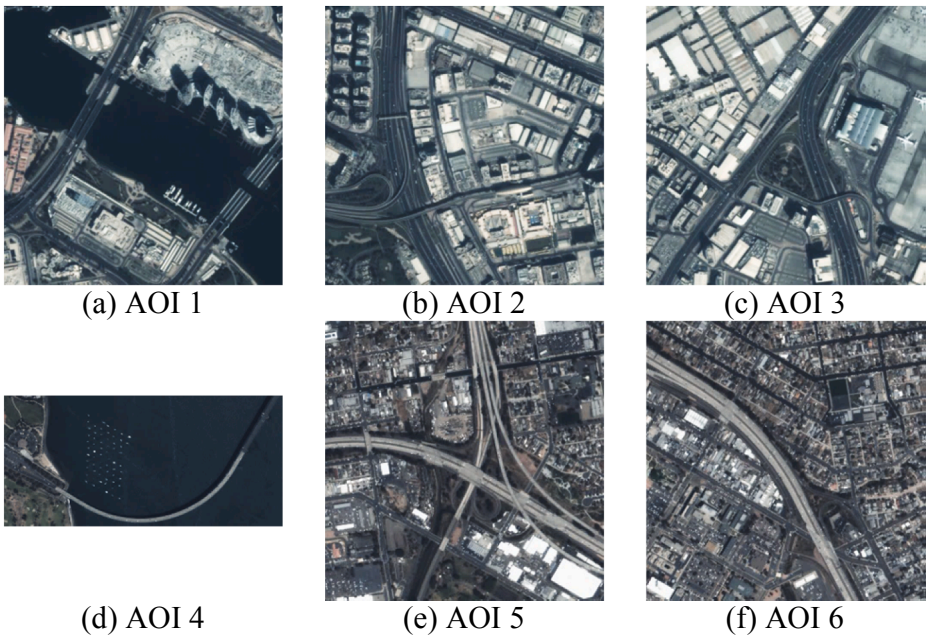


Fig. 5. An example of Jilin-1 satellite video dataset. (a), (b), (c) are cropped from the Jilin-1 satellite video in Dubai. (d), (e), (f) are cropped from the Jilin-1 satellite video in San Diego.

vehicles is extracted and learned frame by frame. At the t - th moment, the hidden state $dh_{ij}^{t} \in \mathbb{R}^{1\times 256}$ of spatial LSTM is obtained and outputted, which has the spatial information of the hypothetical trajectory H_{ij}. The extraction of spatial information greatly improves the discrimination of crowded and similar vehicles, which can track these vehicles more stably.

Output of SMTNet: As shown in Fig. 2. (b), the output of SMTNet is obtained by concatenating the features from motion LSTM and spatial LSTM, followed by some fully connected layers. In general, a tracked vehicle has some candidate targets in the t - th frame, which are selected in the area with its center at the tracked target and of radius r. Then, some hypothetical trajectories are generated by associating the tracked target with these candidate targets. Let p_j denotes the maximum predicted probability of all the hypothetical trajectories, and $bb_{i'}^{t}$ represents the generated virtual bounding box, where $bb_{i'}^{t}$ is obtained by using the size of the bounding box of the previous frame bb_{i}^{t-1} and averaging the predicted virtual positions of all the hypothetical trajectories. Then, a tracking threshold N_{track} is set to judge whether the hypothetical trajectory is true. If p_j is lower than the threshold, the tracked target is considered to be in a missed or occluded state in the t - th frame. In this condition, the tracked target would be associated with the virtual position. The final trajectory T_i^t of the i - th tracked vehicle in the t - th frame is generated by:

$$T_i^t = \begin{cases} H_{ij}, \text{ if } p_j > N_{track} \\ \{T_i^{t-1}, bb_{i'}^{t}\}, \text{ otherwise} \end{cases} \tag{16}$$

Here, Eq. (16) is achieved by using the Hungarian algorithm. To describe the proposed tracking network clearly, the procedure of SMTNet is listed in Algorithm 2.

In SMTNet, both motion and spatial LSTMs are used to predict the virtual positions for missed detections. The satellite video datasets are from complex traffic scenes. Under these traffic scenes, the directions and movement velocities of vehicles are different, and the relative positions of different vehicles are also varied. Thus, motion and spatial LSTMs can model diverse spatial relationship. In SMTNet, we used some strategies in both the training process and the network structure to alleviate the overfitting problem. In the training stage, we train the network with a relatively large dataset and statistical noise is added to the inputs. Besides, dropout is also used in SMTNet.

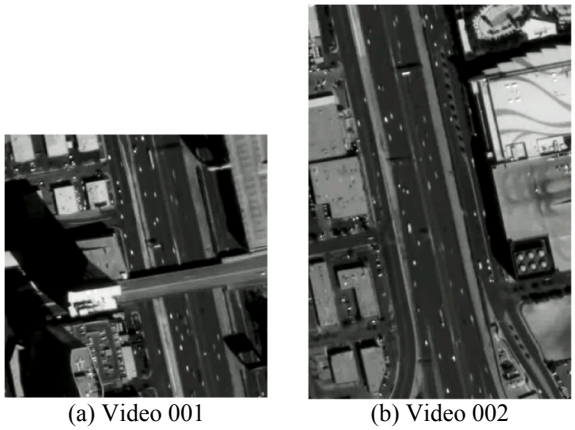


Fig. 6. An example of SkySat satellite video dataset. (a) and (b) are cropped from the SkySat satellite video in Las Vegas.

Loss function: The loss function of SMTNet is defined as

$$L_{SMTNet} = L_{cls} + \lambda L_{reg} \tag{17}$$

It consists of the classification and regression terms. λ is the weight of the regression term. The classification loss L_{cls} is computed by the cross entropy between the real probability and the output probability of the hypothetical trajectory. The regression loss L_{reg} is designed by using the Smooth L1 loss for the real position and virtual position of the tracked target.

3. Experimental results

3.1. Dataset description

In this paper, the Jilin-1 and SkySat satellite videos are used to validate the effectiveness of the proposed CKDNet and SMTNet.

Jinlin-1 satellite video dataset: The Jilin-1 satellite video dataset is

Table 1
Detection results of different methods in the Jilin-1 and SkySat satellite videos.

	Jilin-1				SkySat			
Method	Recall↑	Precision↑	F1↑	AP↑	Recall↑	Precision↑	F1↑	AP↑
background subtraction								
ViBe	37.09%	59.39%	45.67%	–	90.65%	27.60%	42.32%	–
FasteMCD	35.65%	63.25%	45.60%	–	69.58%	74.82%	72.11%	–
GMMv2	74.68%	35.86%	48.45%	–	51.93%	26.40%	35.00%	–
E-LSD	66.77%	65.14%	65.95%	–	85.00%	78.90%	81.84%	–
deep neural network								
CornerNet*	90.59%	71.83%	80.13%	71.91%	89.65%	56.24%	69.11%	65.17%
CenterNet*	89.27%	81.46%	85.19%	78.47%	85.99%	67.96%	75.92%	70.57%
FoveaNet	82.71%	90.07%	86.23%	–	83.75%	80.76%	82.23%	–
CKDNet	**92.99%**	**90.79%**	**91.88%**	**88.14%**	**92.19%**	**84.44%**	**88.14%**	**86.02%**

captured by Jilin-1 satellite, purchased from Changchun-based Chang Guang Satellite Technology. This dataset has six satellite video sequences, where each of them contains 320 frames with 10 fps and 0.92 m spatial resolution. As shown in Fig. 5, areas of interest (AOIs) 1–3 were cropped from the satellite video that was captured over Dubai, UAE, on November 9, 2018. AOIs 4–6 were cropped from the satellite video that was captured over San Diego, USA, on May 23, 2017. Specifically, the frame sizes of AOIs 1–3 and 5–6 are 1000×1000 pixels and the frame size of AOI 4 is 1500×700 pixels. The ground-truth of this dataset is composed of the bounding boxes and their identities of moving vehicles. In the experiments, AOIs 1, 2, 4 and 5 are used as the training set, and AOIs 3 and 6 are used for test.

SkySat satellite video dataset: As shown in Fig. 6, the SkySat satellite video dataset contains two satellite video sequences—Video 001 (400×400 pixels) and Video 002 (600×400 pixels) (Zhang et al., 2019). This satellite video dataset was captured by SkySat-1 satellite from Las Vegas, USA. Each video consists of 700 frames with 30 fps and the spatial resolution is 1.0 m. In the experiment, Video 001 is used to evaluate the proposed method and Video 002 is used for training.

3.2. Experimental setups

We implemented CKDNet and SMTNet in PyTorch 1.5.0. Both two networks are not pretrained with any dataset, and are initialized by PyTorch default setting. The networks are trained on the Ubuntu 18.04 system with a single Titan RTX GPU, Intel i7-7820X CPU. In CKDNet, we set $\tau = 5$. During the training, the Adam optimizer with a learning rate of 2.5×10^{-4} is used. The batch size is set as 16. In the loss of CKDNet, the weight $\alpha_1 = 0.1$, and $\alpha_2 = 1$. In the experiments, the training and test data are picked up every five frames. The training data is randomly cropped as 256×256 pixels per frame and augmented by flipping and rotation. During the test, the original frames are used without any data augment. As CornerNet, top 100 top-left and bottom-right corners are retained in each frame. After applying MMS to re-score the proposal bounding boxes, soft-NMS (Bodla et al., 2017) is used. In SMTNet, the batch size and learning rate are set to 256 and 2.5×10^{-4}, respectively. The scale factor $\gamma = 20$ and the weight of the loss $\lambda = 1$.

3.3. Detection performance and analysis

In this section, we first verify the performance of CKDNet by comparing with some background subtraction methods and deep learning methods. Then, CKDNet with different kinds of MMS is analyzed. Next, we analyze the sensitivity to the interval of consecutive frames. Finally, the effectiveness of CFM and MMS in CKDNet is verified through some ablation experiments.

3.3.1. Detection results of different methods

To verify the effectiveness of CKDNet, some representative background subtraction methods, ViBe (Barnich and Van Droogenbroeck, 2010), FasteMCD (Moo Yi et al., 2013), GMMv2 (Zivkovic, 2004) and E-LSD (Zhang et al., 2019), and deep learning methods, CornerNet (Law and Deng, 2018), CenterNet (Duan et al., 2019) and FoveaNet (Pflugfelder et al., 2020), are selected for comparison. Both CornerNet and CenterNet are designed for image object detection. For a fair comparison, CFM is added into CornerNet and CenterNet, which are indicated as CornerNet* and CenterNet*. In FoveaNet, the detection results falling within the radius of 8 pixels from the center coordinate of ground-truth are considered as true positive. In addition to FoveaNet, the detected bounding box whose IoU with ground-truth is greater than 0.3 as true positive during the evaluation. To evaluate the detection performance of all the methods, recall, precision, average precision (AP) and F1-score indexes are used.

Table 1 records the detection results of all the methods in the Jilin-1 and SkySat satellite video datasets. AP is calculated only in CornerNet*, CenterNet* and CKDNet because the bounding box of other detection methods is generated by connected regions from foreground segmentation. In Table 1, the bold represents the best result under the current index. It can be seen from Table 1 that the precision of ViBe and GMMv2 is lower than others due to moving background in satellite videos. Compared with background subtraction methods, deep learning-based methods obtain better detection performance. FoveaNet reduces the false detections of CornerNet* and CenterNet* caused by mismatching corners with similar embeddings. But, it leads to more missed detections. As a whole, FoveaNet has higher F1-score than CornerNet* and CenterNet*. Among deep learning-based methods, CKDNet achieves the best detection performance in both Jilin-1 and SkySat satellite video datasets. Compared with the best baseline (FoveaNet), CKDNet improves by 5.65% and 5.91% in terms of F1-score in Jilin-1 and SkySat satellite video datasets, respectively. It achieves state-of-the-art detection performance in both Jilin-1 and SkySat satellite video datasets.

Fig. 7 shows the detection results in Jilin-1 and SkySat satellite video datasets, where the true positive is represented in green, the false negative is in yellow and the false positive is in red. As shown in Fig. 7(a) and (c), ViBe and GMMv2 cause lots of false positives in the background. Although FasteMCD and E-LSD are relatively insensitive to moving background, these methods are hard to avoid the false positive caused by brightness and contrast changes in some areas, as shown in Fig. 7(b) and (d). Besides, blurry vehicles easily submerge into the background. In this case, it is difficult for background subtraction methods to detect the blurry vehicles. In deep learning-based methods, the detection performance has an obvious improvement compared with background subtraction methods. CornerNet* and CenterNet* increase the number of true positives via CFM, but there are still many inaccurate bounding boxes, as shown in Fig. 7(e) and (f). Although FoveaNet achieves better detection results than CornerNet* and CenterNet*, it would incorrectly detect some crowded vehicles as a single target, as shown in Fig. 7(g). Moreover, some blurry vehicles are also lost in FoveaNet. Compared with other methods, CKDNet reduces the false positive and false negative significantly. Furthermore, it can detect the blurry and crowded

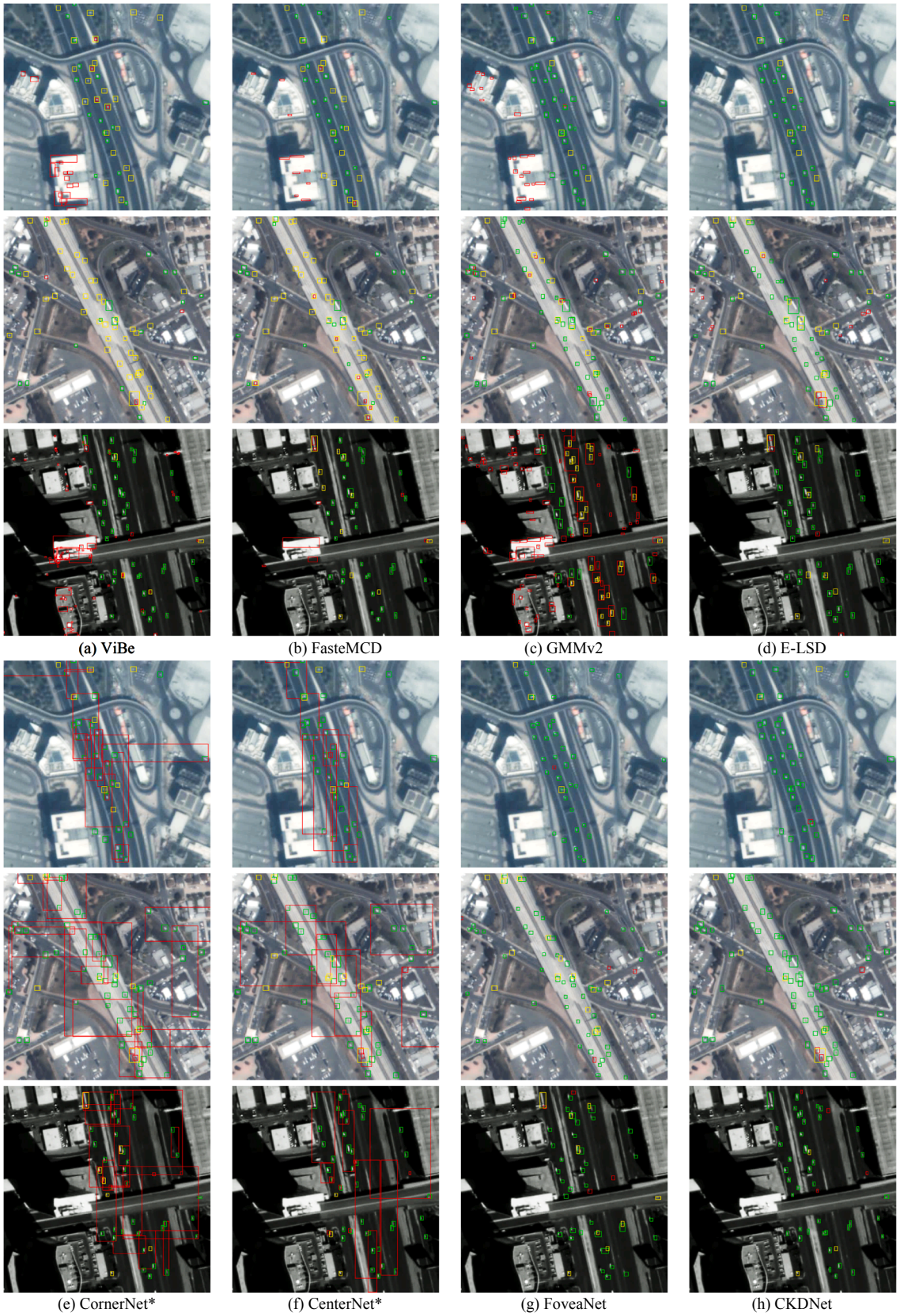


Fig. 7. Detection results after the application of (a) ViBe, (b) FasteMCD, (c) GMMv2, (d) E-LSD, (e) CornerNet*, (f) CenterNet*, (g) FoveaNet, and (h) CKDNet. The first two rows and last row represent the results on the Jinlin-1 and SkySat satellite video datasets, respectively. The true positive is represented in green, the false negative is in yellow, and the false positive is in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
Detection performance of three penalty functions in CKDNet. The bold number represents the best result in each row.

		Hard penalty function				Linear penalty function			Gaussian penalty function		
Dataset	Index	$N_t = 0.5$	$N_t = 0.6$	$N_t = 0.7$	$N_t = 0.8$	$N_t = 0.25$	$N_t = 0.35$	$N_t = 0.45$	$\sigma = 0.7$	$\sigma = 0.85$	$\sigma = 1.0$
Jilin-1	Recall↑	87.98%	79.26%	62.72%	41.05%	93.42%	**93.43%**	91.05%	92.36%	92.84%	93.05%
	Precision↑	90.13%	90.67%	91.16%	**91.74%**	88.85%	89.29%	89.78%	91.18%	90.96%	90.70%
	F1↑	89.04%	84.58%	74.31%	56.72%	91.08%	91.31%	90.41%	91.77%	**91.89%**	91.86%
	AP↑	83.73%	75.46%	59.73%	39.14%	88.85%	**88.88%**	86.64%	87.65%	88.14%	88.41%
SkySat	Recall↑	75.71%	60.26%	45.24%	26.65%	91.78%	87.94%	83.14%	89.68%	91.50%	**92.19%**
	Precision↑	83.48%	83.53%	85.19%	**87.23%**	81.87%	82.50%	83.43%	85.44%	84.85%	84.44%
	F1↑	79.41%	70.01%	59.10%	40.83%	86.54%	85.13%	83.28%	87.51%	88.05%	**88.14%**
	AP↑	71.51%	56.77%	42.97%	25.49%	86.35%	83.08%	78.35%	83.43%	85.24%	**86.02%**

Table 3
The detection performance of CKDNet to different hyper-parameter τ.

	Jilin-1				SkySat			
τ	Recall↑	Precision↑	F1↑	AP↑	Recall↑	Precision↑	F1↑	AP↑
0	83.98%	69.64%	76.14%	74.74%	89.22%	67.02%	76.54%	78.07%
3	**94.56%**	86.84%	90.53%	**90.34%**	92.60%	**82.63%**	**87.33%**	85.81%
5	93.40%	**89.95%**	**91.65%**	90.29%	**93.27%**	81.98%	87.26%	**86.96%**
10	93.68%	89.06%	91.31%	89.55%	91.93%	75.79%	83.09%	83.05%
20	93.53%	89.63%	91.53%	89.63%	89.43%	65.12%	75.37%	69.00%

vehicles more effectively.

3.3.2. Mismatch suppression analysis in CKDNet

The detection performance of CKDNet with different penalty functions in the Jilin-1 and SkySat satellite video datasets is listed in Table 2. In hard penalty function, the threshold N_t is set to 0.5, 0.6, 0.7 and 0.8. The precision of CKDNet with the hard penalty function is improved with the increasing threshold. The higher threshold setting would remove more unreliable bounding boxes. At the same time, the recall is decreased rapidly because the correct bounding boxes may be removed. In linear penalty function, the threshold N_t is set to 0.25, 0.35 and 0.45. As shown in Table 2, its recall is much higher than that of the hard penalty function because more bounding boxes are maintained. However, its precision is lower than that of the hard penalty function. In gaussian penalty function, it is used to make a good trade-off between the recall and precision. The parameter σ is set to 0.7, 0.85 and 1.0. In Table 2, the best detection performance is achieved by using the gaussian penalty function. Its F1-score is greater than others. CKDNet uses the gaussian penalty function to improve the precision without much decreasing the recall.

3.3.3. Sensitivity to the interval τ of consecutive frames in CKDNet

The detection performance of CKDNet is affected by the hyper-parameter τ. τ indicates the interval of consecutive frames extracted in CFM of CKDNet.

Table 3 records the detection performance of CKDNet under different τ in Jilin-1 and SkySat satellite video datasets. As shown in Table 3, when $\tau = 0$, CKDNet achieves poor detection performance in both two datasets. In this case, CKDNet cannot utilize the temporal information of videos. When the value of τ is too small, it is easy to cause the incomplete detection of vehicles by using the cross-difference operation. The detection performance of CKDNet improves with the increasing τ. But the value of τ is not the bigger the better. When the value of τ is too large, it is likely for CKDNet to cause false detection due to background differences between different frames. CKDNet is more sensitive to τ in SkySat dataset than Jilin-1 dataset. This is because the background difference between different frames is larger in SkySat dataset.

3.3.4. Ablation experiments in CKDNet

To investigate the effectiveness of CFM and MMS in CKDNet, the ablation experiment is implemented in this section. Specifically, CKDNet has two core contributions, including cross-frame module (CFM) and mismatch suppression (MMS). To analyze the contribution of each individual component, the detection results of CKDNet without different components are recorded in Table 4. To further verify the effectiveness of CFM, CFM is inserted into CornerNet and CenterNet. The corresponding detection results are recorded in Table 5. To further verify the effectiveness of MMS, MMS is added to CornerNet and CenterNet. The corresponding detection results are recorded in Table 6.

As shown in Table 4, the first row represents the detection results of CKDNet without CFM. CKDNet without CFM achieves poor detection results in terms of recall due to many missed detections. In the satellite video datasets, it is difficult to detect very small vehicle objects only with the appearance information. In the second row of Table 4, CKDNet without MMS causes many false detections because of incorrect corner matching. Compared with CKDNet without MMS, CKDNet improves the F1 by 48.62% (from 43.03% to 91.65%) in Jilin-1 satellite video dataset, and 46.95% (from 40.31% to 87.26%) in SkySat satellite video dataset. Compared with CKDNet without CFM, CKDNet improves the F1 by 12.96% (from 78.69% to 91.65%) in Jilin-1 satellite video dataset, and 1.5% (from 85.76% to 87.26%) in SkySat satellite video dataset. This indicates CFM and MMS are beneficial for vehicle detections in satellite videos.

In Table 5, CKDNet w/o CFM denotes CKDNet without CFM. As

Table 4
The ablation study of CKDNet in Jilin-1 and SkySat satellite video datasets.

		Jilin-1				SkySat			
CFM	MMS	Recall↑	Precision↑	F1↑	AP↑	Recall↑	Precision↑	F1↑	AP↑
	√	79.60%	77.80%	78.69%	74.39%	88.49%	**83.20%**	85.76%	83.54%
√		83.38%	29.00%	43.03%	40.92%	85.93%	26.33%	40.31%	37.71%
√	√	**93.40%**	**89.95%**	**91.65%**	**90.29%**	**93.27%**	81.98%	**87.26%**	**86.96%**

Table 5
The detection performance of different networks with/without CFM in Jilin-1 and SkySat satellite video datasets.

	Jilin-1				SkySat			
	Recall↑	Precision↑	F1↑	AP↑	Recall↑	Precision↑	F1↑	AP↑
CornerNet	76.21%	57.78%	65.73%	54.31%	85.37%	53.51%	65.79%	60.41%
CenterNet	76.07%	71.76%	73.85%	64.71%	80.89%	69.39%	74.70%	66.44%
CKDNet w/o CFM	79.60%	77.80%	78.69%	74.39%	88.49%	**83.20%**	85.76%	83.54%
CornerNet + CFM	90.59%	71.83%	80.13%	71.91%	89.65%	56.24%	69.11%	65.17%
CenterNet + CFM	89.27%	81.46%	85.19%	78.47%	85.99%	67.96%	75.92%	70.57%
CKDNet	**93.40%**	**89.95%**	**91.65%**	**90.29%**	**93.27%**	81.98%	**87.26%**	**86.96%**

Table 6
The detection performance of different networks with/without MMS in Jilin-1 and SkySat satellite video datasets.

	Jilin-1				SkySat			
	Recall↑	Precision↑	F1↑	AP↑	Recall↑	Precision↑	F1↑	AP↑
CornerNet	76.21%	57.78%	65.73%	54.31%	85.37%	53.51%	65.79%	60.41%
CenterNet	76.07%	71.76%	73.85%	64.71%	80.89%	69.39%	74.70%	66.44%
CKDNet w/o MMS	83.38%	29.00%	43.03%	40.92%	85.93%	26.33%	40.31%	37.71%
CornerNet + MMS	77.38%	81.93%	79.59%	72.08%	85.00%	81.01%	83.00%	79.49%
CenterNet + MMS	69.27%	89.18%	77.98%	67.37%	74.67%	**88.23%**	80.88%	71.84%
CKDNet	**93.40%**	**89.95%**	**91.65%**	**90.29%**	**93.27%**	81.98%	**87.26%**	**86.96%**

shown in Table 5, CFM significantly improves the recall of CornerNet and CenterNet by 14.38% (from 76.21% to 90.59%) and 13.20% (from 76.07% to 89.27%) in Jinlin-1 satellite video dataset. Compared with CornerNet + CFM and CenterNet + CFM, CKDNet has obvious improvement in terms of F1 and AP in two satellite video datasets.

In Table 6, although CenterNet adds the center point to reduce false detections, it still has poor precision in satellite videos. With the help of MMS, CornerNet and CenterNet improve the precision by 24.15% (from 57.78% to 81.93%) and 17.42% (from 71.76% to 89.18%) in Jilin-1 satellite video dataset. MMS is designed to match the corners, which significantly improves the precision in CornerNet and CenterNet through eliminating the incorrect corner matching. Although the recall of CenterNet decreases, its F1 and AP are improved in both two satellite video datasets. By adding the MMS, CKDNet also has an obvious improvement in the detection performance. This demonstrates the effectiveness of MMS in all the CornerNet and CenterNet, CKDNet methods.

3.4. Multi-target tracking performance and analysis

In this section, we first compare with some TBD methods to verify the tracking performance of the proposed method. Then, some ablation studies are executed to analyze the effectiveness of motion information, spatial information and regression of virtual position for the tracking performance of SMTNet.

MOT metrics (Milan et al., 2016) and ID measures (Ristani et al., 2016) are used to evaluate the tracking performance of all the methods. In MOT metrics, MOTA evaluates the accuracy of tracking, which combines false positives (FP), missed targets (FN) and identity switches (IDs) (Milan et al., 2016). MOTP measures the tracker's precision at estimating the target position. Recall and precision for the tracked targets are abbreviated as Rcll and Prcn. GT denotes the number of real targets, MT represents the successful tracked, and PT denotes partially tracked. In ID measures, IDF1 indicates the tracker's identification performance for tracking, which balances the identification precision (IDP) and recall (IDR) via their harmonic mean (Ristani et al., 2016). IDF1 is an important measure in the case of long-term trajectory tracking. Moreover, Hz measures the number of processed frames per second.

3.4.1. Tracking results of different methods

To verify the effectiveness of SMTNet, six representative tracking methods, MHT (Kim et al., 2015), Kalman filter (Bewley et al., 2016), particle filter(Okuma et al., 2004), RNN_HA (Milan et al., 2017), MPNTrack (Brasó and Leal-Taixé, 2020) and CenterTrack (Zhou et al., 2020) are used for comparison. Table 7 records the tracking results of different methods in various indexes.

As shown in Table 7, MHT performs worse than RNN_HA and SMTNet, especially in the Jilin-1 satellite video dataset. In SkySat satellite video dataset, a lot of targets are sparse and moving uniformly in a

Table 7
Tracking performance of different methods.

Datasets	Method	IDF1	IDP	IDR	Rcll	Prcn	GT	MT	PT	FP	FN	IDs	MOTA	MOTP	Hz
Jilin-1	MHT	39.7	46.9	34.4	54.2	73.9	570	135	300	3705	8869	1434	27.6	61.3	0.31
	Particle filter	42.5	73.1	30	37.5	91.5	570	99	200	705	12,634	890	29.6	45.9	23.5
	Kalman filter	41.4	74.9	28.6	35.4	92.8	570	109	154	**556**	13,066	661	29.4	**65.4**	**25.2**
	RNN_HA	72.7	74.6	70.8	80.8	85.1	570	362	179	2737	3711	1391	58.5	60.7	1.22
	MPNTrack	64.8	66.5	63.1	**90.9**	78.3	570	**492**	**74**	3196	**1830**	824	70.1	64.5	0.52
	CenterTrack	74.9	79.4	70.9	84.1	**94.2**	570	386	120	998	3074	**231**	77.8	65.1	0.66
	SMTNet	**83.3**	**85.3**	**81.4**	89.3	93.6	570	448	93	1189	2062	313	**81.6**	65.3	1.86
SkySat	MHT	65.6	63.7	67.5	85	80.2	193	145	34	1113	796	178	60.6	59.1	2.71
	Particle filter	75.9	88.9	66.2	67.9	91.2	193	77	89	354	1726	46	60.5	48.4	198.6
	Kalman filter	73.2	**89.1**	62.1	63.9	**91.7**	193	72	78	**313**	1944	51	57.1	**60.4**	**211.9**
	RNN_HA	72	72.9	71.2	73.7	75.5	193	94	88	1269	1390	46	48.9	55.9	6.58
	MPNTrack	66.3	61.2	72.3	86.4	73.2	193	**161**	26	1678	718	188	51.2	58.7	3.73
	CenterTrack	75.8	70.8	81.6	**88.9**	77.1	193	160	**20**	1398	**589**	50	61.5	59.3	4.1
	SMTNet	**84.1**	84.4	**83.8**	86.7	87.3	193	157	26	665	706	**37**	**73.4**	58.5	12.9

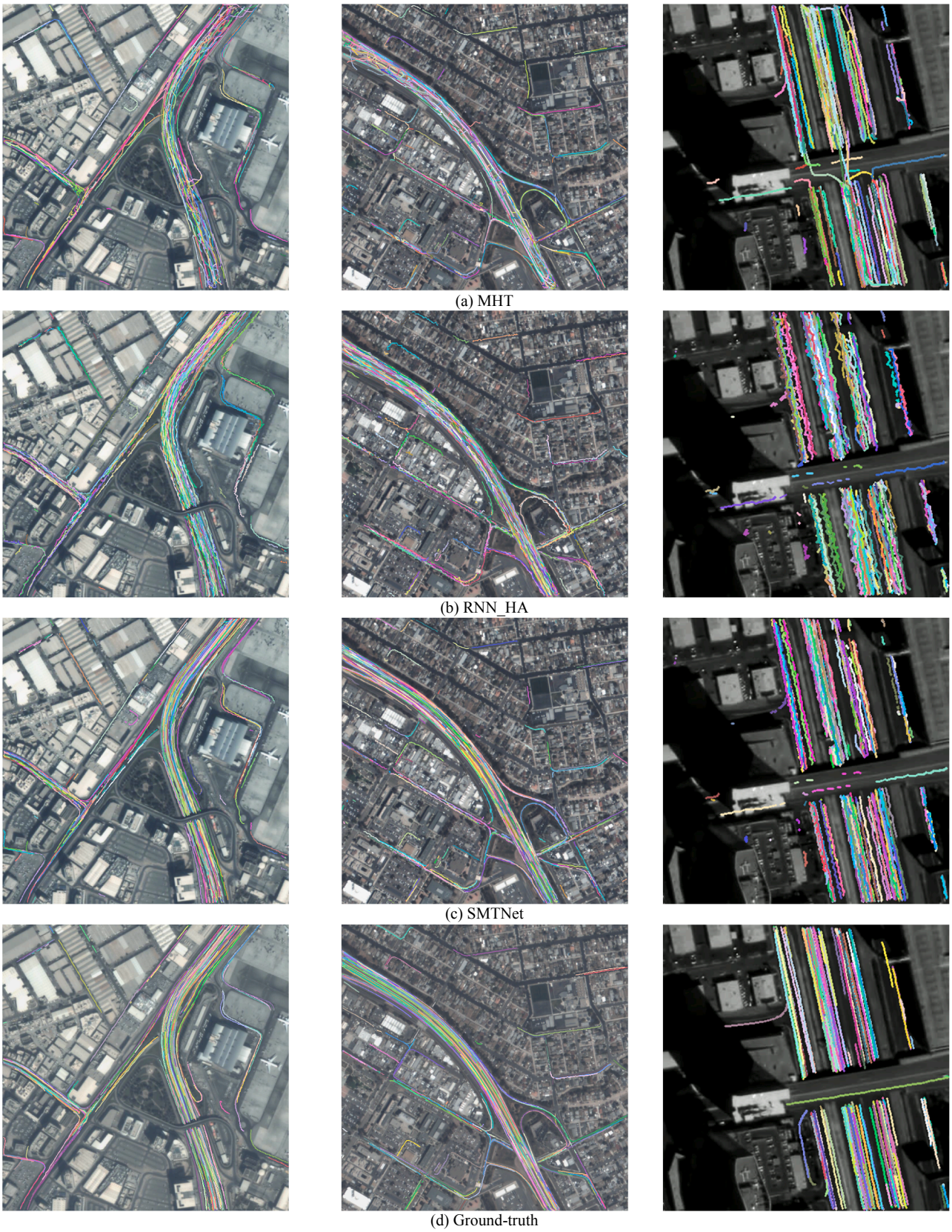
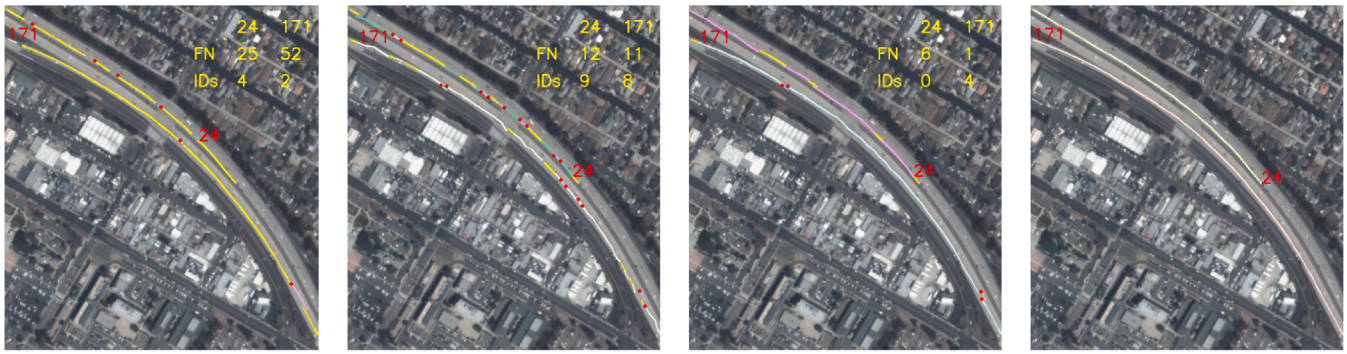


Fig. 8. The tracked trajectories of moving vehicles after the application of (a) MHT, (b) RNN_HA, (c) SMTNet and (d) ground-truth trajectories in the Jilin-1 (columns 1 and 2) and SkySat (column 3) satellite video datasets. Different colors represent different trajectories.



(a) MHT (b) RNN_HA (c) SMTNet (d) Ground-Truth

Fig. 9. An example of tracked trajectories of the 24 - th and 171 - th moving vehicles by (a) MHT, (b) RNN_HA and (c) SMTNet in Jilin-1 satellite video dataset. The red points represent IDs, the yellow lines represent FN, and different colors represent different tracked trajectories. (d) is the ground-truth. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

straight line, while the targets are crowded and moving complicatedly in Jilin-1 satellite video dataset. It is difficult for MHT to track the complex targets in Jilin-1 satellite video dataset. Compare with MHT, Kalman filter and particle filter are faster and better both in two satellite video datasets. However, both Kalman filter and particle filter have poor tracking performance in Jilin-1 satellite video dataset, where FN is too high. The traffic scene of Jilin-1 satellite video dataset is more complicated than that of SkySat satellite video dataset. In Jilin-1 satellite video dataset, the false tracking of Kalman filter and particle filter is caused by many crowded or lost vehicles.

RNN_HA is better than MHT in ID measures, but not as good as MHT in MOT metrics in the SkySat satellite video dataset. The sizes of moving vehicles are relatively small in the SkySat satellite video. It is easy to suffer from missed detections caused by adjusting the detected bounding boxes in RNN_HA. Besides, RNN_HA is hard to stably track the crowded vehicles, which causes the increase in FN, IDs and FM. Compared with RNN_HA, MPNTrack improves the MOTA in the two satellite video datasets. Although the appearance and geometry features are combined, MPNTrack is inferior to CenterTrack in terms of IDF1 and MOTA. Besides, MPNTrack performs simple bilinear interpolation for the frames where the vehicles are missing in tracking. This causes MPNTrack has higher FP and lower MOTA. CenterTrack simultaneously detects vehicles and tracks the vehicles only by using their offsets. Compared with CenterTrack, SMTNet improves by more than 8% in IDF1 due to the effective usage of spatial information. Among these tracking methods, SMTNet is the most outstanding in both MOTA and IDF1 indexes.

Fig. 8 shows the tracked trajectory of each moving vehicle in the Jilin-1 and SkySat satellite video datasets. Different trajectories are represented in different colors. In Fig. 8(a), MHT performs well for uncrowded targets. However, the trajectories of MHT are messy due to the false tracking for crowded targets. In Fig. 8(c), SMTNet obtains the closest tracking results to the ground-truth. It can not only track crowded similar targets, but also re-track the targets that reappear after being lost or occluded.

To provide a clearer display, Fig. 9 shows the tracked trajectories of three moving vehicles under the crowded condition in Jilin-1 satellite video dataset. Additionally, their FN and IDs are shown in the figure as well. It is difficult to track crowded vehicles for MHT. In MHT, most of the trajectories of these three tracked vehicles are lost. Compared with MHT, RNN_HA successfully tracks these vehicles. But RNN_HA has higher IDs and its trajectories are not stable in Fig. 9(b). Among these tracking methods, SMTNet has lowest FN and IDs because it can stably track these vehicles for a long time (see Fig. 9(c)).

3.4.2. Ablation study of SMTNet

In SMTNet, there are three main components, spatial LSTM, motion LSTM and regression of virtual position, abbreviated as S, M, and R. To analyze the contribution of each individual component in SMTNet, SMTNets without different components are used for comparison. Table 8 records the tracking results of SMTNets without different components.

Motion LSTM: The motion LSTM is designed to extract the motion information from the movement velocities. In SMTNet, the motion LSTM contributes to the key tracking performance under some ordinary conditions. Since some vehicles are well-detected and not crowded, these vehicles are easy to track only via the motion LSTM. As shown in Table 8, the first and seventh rows indicate the results of SMTNet without S and R. Its IDF1 and MOTA are not bad, which are 71.5% and 79.3% in Jinlin-1 satellite video dataset, and 75.7% and 70.4% in SkySat satellite video dataset.

Spatial LSTM: In the satellite video datasets, there are many crowed vehicles. In order to track the crowded vehicles, the spatial LSTM is designed to extract spatial information from relative positions. As shown in fourth and tenth rows of Table 8, SMTNet without M and R has poor tracking performance by merely training the network with spatial LSTM.

Table 8
The performance of ablation experiment in SMTNet. The abbreviations are as follows: motion LSTM (M), spatial LSTM (S), and regression of virtual position (R).

Datasets	M	S	R	IDF1	IDP	IDR	Rcll	Prcn	GT	MT	PT	FP	FN	IDs	MOTA	MOTP
Jilin-1	√			71.5	74.7	68.6	86.7	94.5	570	422	107	**972**	2569	463	79.3	65.3
	√		√	79.5	81.9	77.2	88.3	93.7	570	445	89	1156	2268	349	80.5	65.3
		√		14.9	19.8	12	54.2	89.8	570	31	493	1190	8859	4998	22.3	65.2
		√	√	16.4	20.1	13.9	62.7	90.5	570	86	455	1275	7218	6328	23.4	65.3
	√	√		74.5	77.3	71.8	87.8	**94.5**	570	428	114	993	2367	415	80.5	65.3
	√	√	√	**83.3**	**85.3**	**81.4**	**89.3**	93.6	570	**448**	**93**	1189	**2062**	**313**	**81.6**	65.3
SkySat	√			75.7	78.1	73.4	83.1	88.5	193	137	44	**571**	893	105	70.4	58.5
	√		√	82.9	84	81.7	84.9	87.3	193	146	35	684	798	51	71.6	58.5
		√		24.9	27.8	22.5	69.5	86.2	193	68	117	591	1615	1339	33.0	**58.7**
		√	√	30.9	33.2	28.8	74.2	85.3	193	94	90	676	1367	1578	31.6	58.4
	√	√		74.9	76.6	73.2	84.6	**88.5**	193	144	39	584	816	98	71.7	58.5
	√	√	√	**84.1**	**84.4**	**83.8**	**86.7**	87.3	193	**157**	**26**	665	**706**	**37**	**73.4**	58.5

However, compared with SMTNet without S and R, SMTNet without R improves the IDF1 by 3% (from 71.5% to 74.5%) in Jilin-1 satellite video dataset through adding the spatial LSTM.

Regression of virtual position: Because the regression of virtual position relies on the spatial LSTM or motion LSTM, we would not train the network by itself. As shown in Table 8, SMTNet without S performs better than SMTNet without S and R (see the first and second rows of Table 8), and SMTNet without M performs better than SMTNet without M and R (see the third and fourth rows of Table 8) by predicting the virtual positions for all the vehicles in the Jilin-1 satellite video dataset. A similar improvement can be observed in the SkySat satellite video dataset. It demonstrates the regression of virtual position is effective for tracking vehicles. SMTNet without S and SMTNet without M can re-track the lost or occluded vehicles. The sixth and twelfth rows of Table 8 record the results of SMTNet. Among all the comparison results, SMTNet obtains the best tracking results in terms of IDF1 and MOTA. This indicates spatial LSTM, motion LSTM and regression of virtual position are beneficial to the tracking task.

4. Conclusion

In this paper, a novel deep learning framework combining CKDNet and SMTNet was proposed for moving vehicle detection and tracking in the satellite videos. CKDNet improves the recall of blurry moving vehicles by extracting the fused temporal and appearance features from consecutive frames. Moreover, it also improves the precision by using hard or soft penalty function to suppress the incorrect corner matching effectively. Based on the detection results of CKDNet, SMTNet associates all the detection results with tracked targets via two-branch LSTMs. For tracking crowded and similar vehicles, spatial LSTM is designed to assist the motion LSTM by considering the relative spatial relationship with neighboring vehicles. According to the motion and spatial information, SMTNet predicts whether the hypothetical trajectory is true, and regresses a new virtual position for each tracked vehicle simultaneously. It is beneficial to re-identify the missed or occluded vehicles and stably track the vehicles. In the future, the tracking network will be merged into the detection network rather than working separately.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Ao, W., Fu, Y., Hou, X., Xu, F., 2019. Needles in a Haystack: Tracking city-scale moving vehicles from continuously moving satellite. IEEE Trans. Image Process. 29, 1944–1957.

Barnich, O., Van Droogenbroeck, M., 2010. ViBe: A universal background subtraction algorithm for video sequences. IEEE Trans. Image Process. 20, 1709–1724.

Benenson, R., Mathias, M., Tuytelaars, T., Van Gool, L., 2013. Seeking the strongest rigid detector. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3666-3673.

Bergmann, P., Meinhardt, T., Leal-Taixe, L., 2019. Tracking without bells and whistles. In: Proceedings of the IEEE international conference on computer vision, pp. 941–951.

Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., 2016. Simple online and realtime tracking, 2016 IEEE international conference on image processing (ICIP). IEEE 3464–3468.

Bodla, N., Singh, B., Chellappa, R., Davis, L.S., 2017. Soft-NMS–improving object detection with one line of code. In: Proceedings of the IEEE international conference on computer vision, pp. 5561-5569.

Brasó, G., Leal-Taixé, L., 2020. Learning a neural solver for multiple object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6247-6257.

Deng, Z., Sun, H., Zhou, S., Zhao, J., Zou, H., 2017. Toward fast and accurate vehicle detection in aerial images using coupled region-based convolutional neural networks. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10, 3652–3664.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q., 2019. Centernet: Keypoint triplets for object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 6569-6578.

Jiang, X., Li, P., Li, Y., Zhen, X.J.a.p.a., 2019. Graph Neural Based End-to-end Data Association Framework for Online Multiple-Object Tracking.

Kim, C., Li, F., Ciptadi, A., Rehg, J.M., 2015. Multiple hypothesis tracking revisited. In: Proceedings of the IEEE international conference on computer vision, pp. 4696–4704.

Kopsiaftis, G., Karantzalos, K., 2015. Vehicle detection and traffic density monitoring from very high resolution satellite video data, 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE 1881–1884.

Kuhn, H.W., 1955. The Hungarian method for the assignment problem. Naval Res. Logist. Quart. 2, 83–97.

LaLonde, R., Zhang, D., Shah, M., 2018. Clusternet: Detecting small objects in large scenes by exploiting spatio-temporal information. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4003-4012.

Law, H., Deng, J., 2018. Cornernet: Detecting objects as paired keypoints. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 734-750.

Li, K., Wan, G., Cheng, G., Meng, L., Han, J., 2020a. Object detection in optical remote sensing images: A survey and a new benchmark. ISPRS J. Photogramm. Remote Sens. 159, 296–307.

Li, K., Wan, G., Cheng, G., Meng, L., Han, J.J.I.J.o.P., Sensing, R., 2020b. Object detection in optical remote sensing images: A survey and a new benchmark. 159, 296-307.

Li, Y., Jiao, L., Tang, X., Zhang, X., Zhang, W., Gao, L., 2019. Weak Moving Object Detection In Optical Remote Sensing Video With Motion-Drive Fusion Network. In: IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium. IEEE, pp. 5476-5479.

Lu, Z., Rathod, V., Votel, R., Huang, J., 2020. RetinaTrack: Online Single Stage Joint Detection and Tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14668-14678.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K., 2016. MOT16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831.

Milan, A., Rezatofighi, S.H., Dick, A., Reid, I., Schindler, K., 2017. Online multi-target tracking using recurrent neural networks. In: Thirty-First AAAI Conference on Artificial Intelligence.

Moo Yi, K., Yun, K., Wan Kim, S., Jin Chang, H., Young Choi, J., 2013. Detection of moving objects with non-stationary cameras in 5.8 ms: Bringing motion detection to your mobile device. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 27-34.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines, ICML.

Okuma, K., Taleghani, A., De Freitas, N., Little, J.J., Lowe, D.G., 2004. A boosted particle filter: Multitarget detection and tracking. European conference on computer vision. Springer, pp. 28-39.

Peng, J., Wang, C., Wan, F., Wu, Y., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y., 2020. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking, European Conference on Computer Vision. Springer, pp. 145-161.

Pflugfelder, R., Weissenfeld, A., Wagner, J., 2020. On Learning Vehicle Detection in Satellite Video. arXiv preprint arXiv:2001.10900.

Redmon, J., Farhadi, A., 2017. YOLO9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263-7271.

Ristani, E., Solera, F., Zou, R., Cucchiara, R., Tomasi, C., 2016. Performance measures and a data set for multi-target, multi-camera tracking. In: European Conference on Computer Vision. Springer, pp. 17-35.

Roy, S.D., Bhowmik, M.K., 2020. A Comprehensive Survey on Computer Vision Based Approaches for Moving Object Detection. In: 2020 IEEE Region 10 Symposium (TENSYMP). IEEE, pp. 1531–1534.

Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.J.I.T.O.N.N., 2008. The graph neural network model. 20, 61-80.

Sharma, V., Mir, R.N.J.C.S.R., 2020. A comprehensive and systematic look up into deep learning based object detection techniques: A review. 38, 100301.

Stauffer, C., Grimson, W.E.L., 1999. Adaptive background mixture models for real-time tracking. In: Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149). IEEE, pp. 246-252.

Van Etten, A.J.A.P.A., 2018. You only look twice: Rapid multi-scale object detection in satellite imagery.

Wang, Y., Wang, T., Zhang, G., Cheng, Q., Wu, J.-Q., 2020. Small Target Tracking in Satellite Videos Using Background Compensation. IEEE Trans. Geosci. Remote Sens.

Zhang, J., Jia, X., Hu, J., 2019. Error bounded foreground and background modeling for moving object detection in satellite videos. IEEE Trans. Geosci. Remote Sens. 58, 2659–2669.

Zhang, J., Jia, X., Hu, J., Chanussot, J., 2020a. Online Structured Sparsity-Based Moving-Object Detection From Satellite Videos. IEEE Trans. Geosci. Remote Sens.

Zhang, J., Jia, X., Hu, J., Tan, K., 2018. Satellite multi-vehicle tracking under inconsistent detection conditions by bilevel K-shortest paths optimization, 2018 Digital Image Computing: Techniques and Applications (DICTA). IEEE 1–8.

Zhang, Y., Sheng, H., Wu, Y., Wang, S., Lyu, W., Ke, W., Xiong, Z.J.I.T.o.I.P., 2020b. Long-term tracking with deep tracklet association. 29, 6694-6706.

Zhang, Y., Wang, C., Wang, X., Zeng, W., Liu, W., 2020c. FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object Tracking.

Zhou, X., Koltun, V., Krähenbühl, P.J.A.P.A., 2020. Tracking Objects as Points.

Zhou, X., Wang, D., Krähenbühl, P.J.A.P.A., 2019a. Objects as points.
Zhou, X., Zhuo, J., Krahenbuhl, P., 2019b. Bottom-up object detection by grouping extreme and center points. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 850-859.
Zivkovic, Z., 2004. Improved adaptive Gaussian mixture model for background subtraction. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. IEEE, pp. 28–31.