A Simple Vehicle Re-Identification for Unlabeled Drone Flight Images

Youlkyeong Lee, Qing Tang, Jehwan Choi, Kanghyun Jo
Dept. of Electrical, Electronic and Computer Engineering
University of Ulsan, Ulsan, Korea
{yklee, tangqing, jehwan}@islab.ulsan.ac.kr, acejo@ulsan.ac.kr

***Abstract*—Recently advanced vehicle re-identification frameworks are mainly based on convolutional neural networks (CNN) and labeled information. Previous frameworks face two difficulties. First CNN includes complicated architectures, which require expensive GPU devices to perform computation. The second difficulty is that annotating vehicle identities for every frame is expensive and time-consuming. To tackle these two difficulties, this study proposes a simple but effective method to perform re-ID without CNN and labeled identities. The proposed method has two streams of vehicle re-identification. The object detector takes charge of detecting vehicles on the road. With the position of vehicles in the image, the condition module extracts the vehicle movement information and sets the condition to match the same vehicle between current and subsequent frames. To train the object detector and test the proposed algorithm, a set of drone flight images collect and annotate for studying the traffic road. It contains 9,776 train images and 2,200 test images for object detection. In the experiments, three different traffic video clips were applied for testing the proposed method. The full re-identification results on the video clips are provided in https://drive.google.com/drive/folders/1N_dbv41w1a9Xi_Obs2x_WNLjP2gMXhEW?usp=sharing.**

***Index Terms*—Drone flight image, object detection, re-identification**

I. Introduction

Smart drones can be used in many areas of life, such as delivering, disaster monitoring, facility safety diagnosis, patrol, and leisure sports. In particular, in the transportation field, drones have worked in various fields to collect and analyze information on the road, such as traffic volume analysis, facility inspection, and obstacle detection. Traditionally, a group of CCTV cameras acquires traffic information from a fixed location. However, drone flight images with a wide field of view are possible to collect traffic information without the time and place restrictions such as speed, vehicle type, and density of vehicles on the road. In a future transportation system like Fig. 1 is possible to collect each vehicle's information in real-time.

Recently, in computer vision, Convolutional Neural Network (CNN) has shown outstanding performance in object classification [1], [2], object detection [3]–[5], and object re-identification [6]–[8]. These techniques are applied in various ways and showed reliable performance. Training the CNN requires several large-scale datasets, such as MS-COCO [9], Pascal VOC [10], ImageNet [11] that are commonly used to create a base learning model. Continually acquiring vehicle information (position, color, type, etc) is an important role in analyzing a vehicle's activities. However, the object detection algorithm can only detect the vehicle in a single road image with the labeled vehicle location and class. The detector cannot identify vehicles between sequential frames. In other words, object detection can not consecutively acquire specific vehicle information. Therefore, this paper proposes a simple and light re-ID method that performs vehicle re-identification after detection to match vehicle across frames. The proposed method re-identify vehicles across frames by utilizing the relative and change information instead of using a heavy CNN architecture in previous studies [3], [4]. The process flow of the proposed method is shown in Fig. 2.

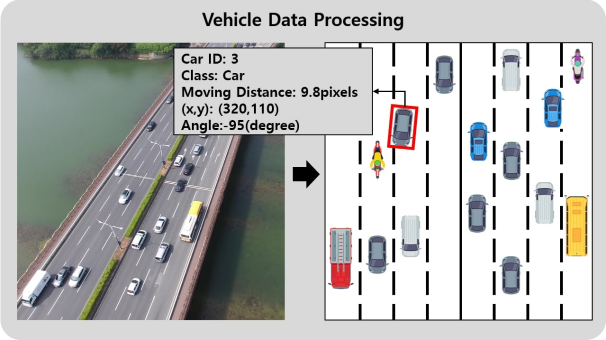


Fig. 1: The illustration of the application of a drone in intelligent transportation system.

This study proposes a simple framework that combines object detection and vehicle re-identification. Given as input a drone flight image, the object detector first is to find the vehicle on the image. With the detection result, re-identification contains the moving distance and direction of the vehicle as strategies. In a series of images, the moving distance computes from the center of vehicles in the current and next frame. The cropped area extracts the object's edge for analyzing the vehicle's direction. The proposed algorithm identifies the vehicle ID for every frame. The sequential frame tracks the vehicle ID between the current and subsequent frames. Finally, it generates the re-identification for the vehicle ID.

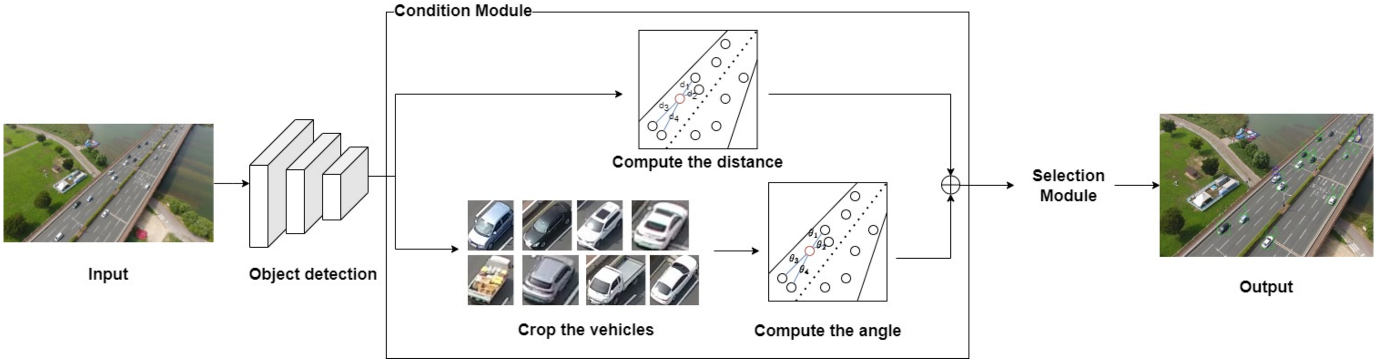


Fig. 2: The framework of the proposed method. Object detection generates bounding boxes of vehicles. Then, the condition module re-identifies and matches the vehicle ID in sequential images. In the condition module, the moving distance between the current and the next frame of each vehicle is computed, and the direction of each vehicle is computed. Subsequently, the selection module decides to remove or keep the detection result by considering the position of each vehicle.

II. Related work

A. Object Detection

Currently, the performance of object detection has rapidly improved in various fields. YOLO [12] algorithm is widely used in real-time applications. The architecture was designed for the single-stage for handling bounding boxes (x, y, width, height, classes) for each grid cell and the probability of every class. It brought the high speed(45 FPS) on the predicting object. However, there was still low performance for the result (63.4 mAP($AP_{.5}$), VOC2007 (20 classes)). YOLOv3 [5] achieved 61.1 mAP ($AP_{.5}$) based on MS-COCO (80 classes) and 171 FPS (Darknet-19 [13]). YOLOv5 [14] is the recent object detection algorithm in a series of YOLO [5], [12] architecture. This model achieved the high accuracy (YOLOv5l, 67.3 mAP($AP_{.5}$)) and high speed (YOLOv5l, 370fps) on MS-COCO [9] dataset. In the case of the depth of model and channel of layer, YOLOv5 contains YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, respectively. Furthermore, these days are required to develop edge devices that these architectures are highly suited for edge devices. That is why many mobile developers have to work with real-time embedding applications with the YOLOv5 model.

B. Object Re-Identification

Object re-ID, including person re-ID [15], [16] and vehicle re-ID [17], is an important application in intelligent surveillance systems and intelligent transportation. The conventional approaches to vehicle re-id came from the method of the person re-id. The algorithm of person re-id has challenge tasks to directly apply for vehicle re-id. Between person and vehicle has big differences in appearance, color, and texture. As a person, the vertical structure of the body separates the feature but, the vehicle is not different many horizontal changes [18]. In the state of the art, there are some methods to extract the feature map from neural networks [19], [20] to re-identify the vehicle. CNN architecture brings plenty of parameters that are affected by complex structure and expensive computation costs.

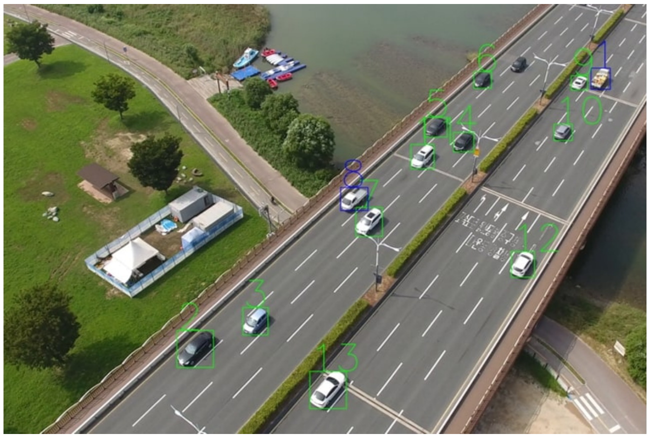


Fig. 3: Vehicle ID from the result of object detection

III. Proposed Algorithm

A. Object Detection

In this paper, YOLOv5 [14] is adopted as the detector. The size of the input image is resized from 3840×2160 to 960×960. Among the series of YOLOv5, YOLOv5l (large version). With the drone dataset, the detector is trained for detecting the vehicle.

B. Vehicle-Information and Condition Module

Vehicle-Information As shown in Fig. 3, it follows the vehicle ID randomly generated after detection in the first frame that is started using the object detection module. The vehicle-Information module generates vehicle movement information. The detected information is as follows:

- Vehicle-ID
- Class
- (x,y), coordinate
- Vehicle of width and height

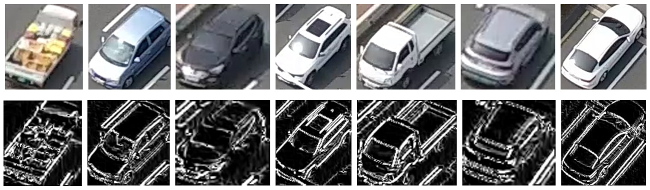


Fig. 4: Generated edge image for cropped vehicle area; first row: cropped image in the original, second row: edge image from a cropped image.

- Moving distance, d_k
- Angle

Vehicle ID has updated information in every frame, and class is the type of detected vehicle. (x,y), coordinate information is the center of the detected box center position, and the box size of the vehicle is expressed in width and height. Moving distance measures the moving distance as a pixel-level between the detected vehicle $V_{k,i}$, in i-th frame and $V_{k,i+1}$, in $i+1$-th frame. k is the index of the detected vehicle. Eq.(1) calculates the moving distance of the vehicle both current and next frames. $V_{k,i} = (x_{k,i}, y_{k,i})$ is the center coordinates of $V_{k,i}$. Therefore, the moving distance between $V_{k,i}$ and $V_{k,i+1}$ as follows:

$$D(V_{k,i}, V_{k,i+1}) = \sqrt{(x_{k,i} - x_{k,i+1})^2 + (y_{k,i} - y_{k,i+1})^2} \quad (1)$$

Among the vehicle movement information, the angle is converted to a degree by extracting a group of edges within cropped vehicle area and calculating the gradient by most edges. Edge detection extracts edges by applying a Sobel filter to the detected vehicle area. In Eq.(2), G_x is an output for the x-direction edge of the Sobel filter, and G_y is an output for the y-direction edge of the Sobel filter. I is an input image.

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * I,\ G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I \quad (2)$$

Fig. 4 shows the extracted image for the detected area as follows. The edge direction of the extracted area is $radian = arctan(G_y/G_x)$, and it is converted to $degree = radian * 180/\pi$. And the representative angle, Eq.(3) for each vehicle is selected as the average angle value for the angles within the area, $\mathbb{A} \in w \times h$.

$$\text{Average angle} = \frac{1}{w * h} \sum_{i=1}^{w} \sum_{j=1}^{h} \mathbb{A}_{ij} \quad (3)$$

Condition The condition module uses the moving distance and direction of the vehicle. First of all, a set of $V_{k,i}$ computes the direction of the vehicle. Fig. 5 shows how to measure the direction of the vehicle. For example, the center point of the 1st-frame car_vehicle, $V_{1,1}$ calculates the direction of the vehicle with the center point of the 2nd-frame car_vehicle, $V_{1,2}$. If the vehicle is close to the high probability of a similar vehicle, it satisfies $\theta_1 \leq 90°$. If the direction is $\theta_2 > 90°$, the vehicle is not similar to the current vehicle. Secondly, the moving distance is applied to remove the far distance from the current vehicle. Both conditions need to consider the similarity of vehicles.

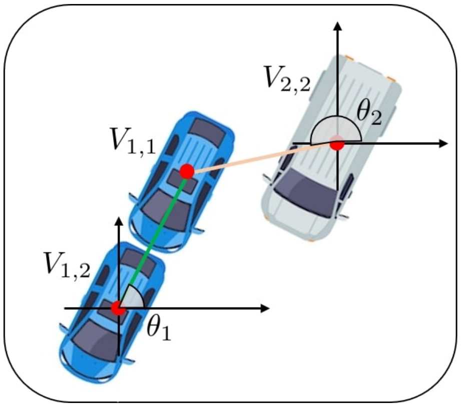


Fig. 5: Consideration of similar with angle. θ_1 is within a constrain for the same vehicle between the current and next frame. θ_2 is out of the constrain for a different vehicle between the current and next frame.

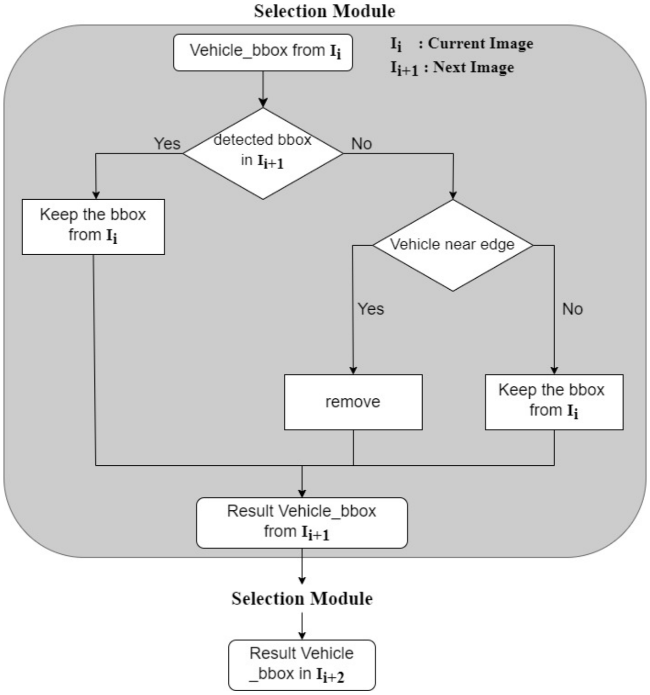
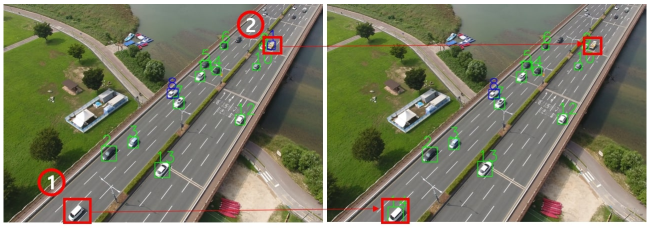

Selection Module
Vehicle_bbox from $\mathbf{I_i}$
$\mathbf{I_i}$: Current Image
$\mathbf{I_{i+1}}$: Next Image
detected bbox in $\mathbf{I_{i+1}}$
Yes
No
Keep the bbox from $\mathbf{I_i}$
Vehicle near edge
Yes
No
remove
Keep the bbox from $\mathbf{I_i}$
Result Vehicle_bbox from $\mathbf{I_{i+1}}$
Selection Module
Result Vehicle _bbox in $\mathbf{I_{i+2}}$

Fig. 6: The process of selection module

C. re-ID Module

The overall re-ID module shows the flow chart as shown in Fig. 6. This re-ID method proposes a method that is applied to drone images. The proposed method does not use CNN and can re-ID of vehicles in the simple method.

1) Edge vehicle selection: The ideal sequential vehicle detection focuses on the center position of the vehicle. Around the edge part of the image, some of the vehicles are going to disappear in the frame. As a result, when the algorithm first



(a) Current Image, I_i (b) Next Image, I_{i+1}

Fig. 7: Different detection result between current image, I_i and next image, I_{i+1}

searches over the frame, there is a margin, M(M=30 pixels). M indicates the distance from the edge of the image. It helps to prevent continuously showing vehicle detection.

Algorithm 1 Updating detected bbox

```
Data: current bbox: $V_{k,i}$, next bbox: $V_{k,i+1}$
Result: updated bbox: $V'_{k,i} \to V_{k,i+1}$
$L_{R,i} = \text{width} * \lambda$
 for $V_{k,i}$ do
    for $V_{k,i+1}$ Step $n$ do
        if $D(V_{k,i}, V_{k,i+1}) \leq L_{R,i}$ then
            if $\theta \leq 90°$ then
                List($V'_{k,i}$) $= n$
            end
        end
    end
end
```

2) Updating detected bbox: As shown in I_i and I_{i+1}, there is a difference between the two detected vehicles according to the performance of the vehicle detection model as shown in Fig. 7.

In order to generate updated re-ID information in the next frame I_{i+1}, the detected vehicle position is compared with I_i and I_{i+1}. The moving distance of $V_{k,i}$ and $V_{k,i+1}$ is a standard for a candidate that is the same vehicle between two frames. In this case, the restricted movement radius is as follows:

$$L_{R,i} = \text{width} \times \lambda \tag{4}$$

In Eq. (4), $L_{R,i}$ is the limiting radius for searching nearby the vehicle which is calculated as the width of $V_{k,i}$, $w_{k,i}$. Because, from the drone flight image dataset, each vehicle does not have the distance from the drone. The λ handles the flexible setting from the vehicle around the vehicle to figure the vehicle out. In the experiment, λ is an adaptive distance ratio, 0.4. $V^{'}_{k,i}$ is a list of vehicles from updating $V_{k,i+1}$. After comparing with $V_{k,i}$ and $V_{k,i+1}$, if $D(V_{k,i}, V_{k,i+1}) < L_{R,i}$ is satisfied, $V_{k,i+1}$ is included in the list of $V^{'}_{k,i}$ like 1 in Fig. 7. If $D(V_{k,i}, V_{k,i+1}) > L_{R,i}$ is satisfied, it is not included in $V^{'}_{k,i}$ in 2.

IV. Experiment

Drone AI Dataset: The dataset was created for the drone images from the drone view. The characteristics of this drone image dataset are data suitable for the proposed method because the upper surface of the vehicle is mainly photographed and the overlap phenomenon between the vehicles is low. A set of images are collected in consideration of the flight location, altitude, shooting angle, and device type. The dataset explains as shown in Table 1.

TABLE 1: Overview of drone flight information

	Contents
location	City
altitude	60m
angle	45°
device	4k Camera
# of images	9,776(train) / 2,200(test)
size of image	3820 × 2160
annotations	points[left top(x_{lt}, y_{lt}), right bottom(x_{rb}, y_{rb})], class

Implementation Details: In the configuration, the original image is resized to 960×960 and provides 16 images at each batch. The learning rate is 0.01 and uses OneCycleLR [21] for learning schedule. Object detection is based on Pytorch [22]. The implementation uses RTX 3090 GPU with 32G memory. The evaluation metrics comes from MS-COCO [9] that provides the mean average precision AP_{50} and $AP_{50:95}$(called by mAP).

Object Detection: This paper adopts YOLOv5 [14] as object detection. In the train and test process, this work focuses on two classes(car and truck). Table 2, shows the performance of the train and test dataset for object detection. With 9,776 images for train data, the results are 95.75 mAP(AP_{50}) and 83.8 mAP($AP_{50:95}$). Using 2,200 test images, it obtains 91.8 mAP(AP_{50}) and 80.3 mAP($AP_{50:95}$). With the detector, other video clips on traffic roads apply to detect vehicles and extract information about vehicles (position and class).

TABLE 2: The mAP performance of YOLOv5 on drone train and test dataset

Class	Images	Instance	mAP@50	mAP@50:95
all_train	**9,776**	**309,470**	**95.75**	**83.8**
car_vehicle	**9,776**	**277,263**	**97.2**	**86.0**
truck_vehicle	**9,776**	**32,207**	**94.3**	**81.6**
all_test	**2,200**	**85,398**	**91.8**	**80.3**
car_vehicle	**2,200**	**78,765**	**96.1**	**85.3**
truck_vehicle	**2,200**	**6,633**	**87.5**	**75.4**

Vehicle re-identification: In the experiment, three video clips adapt for vehicle re-identification. In Fig. 8, each row of images is the different kinds of video clips. At the same time, each column shows the resulting image per 10 frames. The second row of the first row had appeared No.15 vehicle until the next 10 frames. The third column of the second row did not contain the No.12 vehicle that failed to detect the vehicle from the learning model. As several frames go by, the number of vehicle id is getting increases. Because the detector has sometimes lost the vehicle. In the re-ID module, the performance to allocate the vehicle ID is affected by detecting vehicles in every frame. This algorithm does not have the estimation of vehicle position. Therefore it has a problem to denote the same vehicle ID within a $D(V_{k,i}, V_{k,i+1})$. The full experimental video

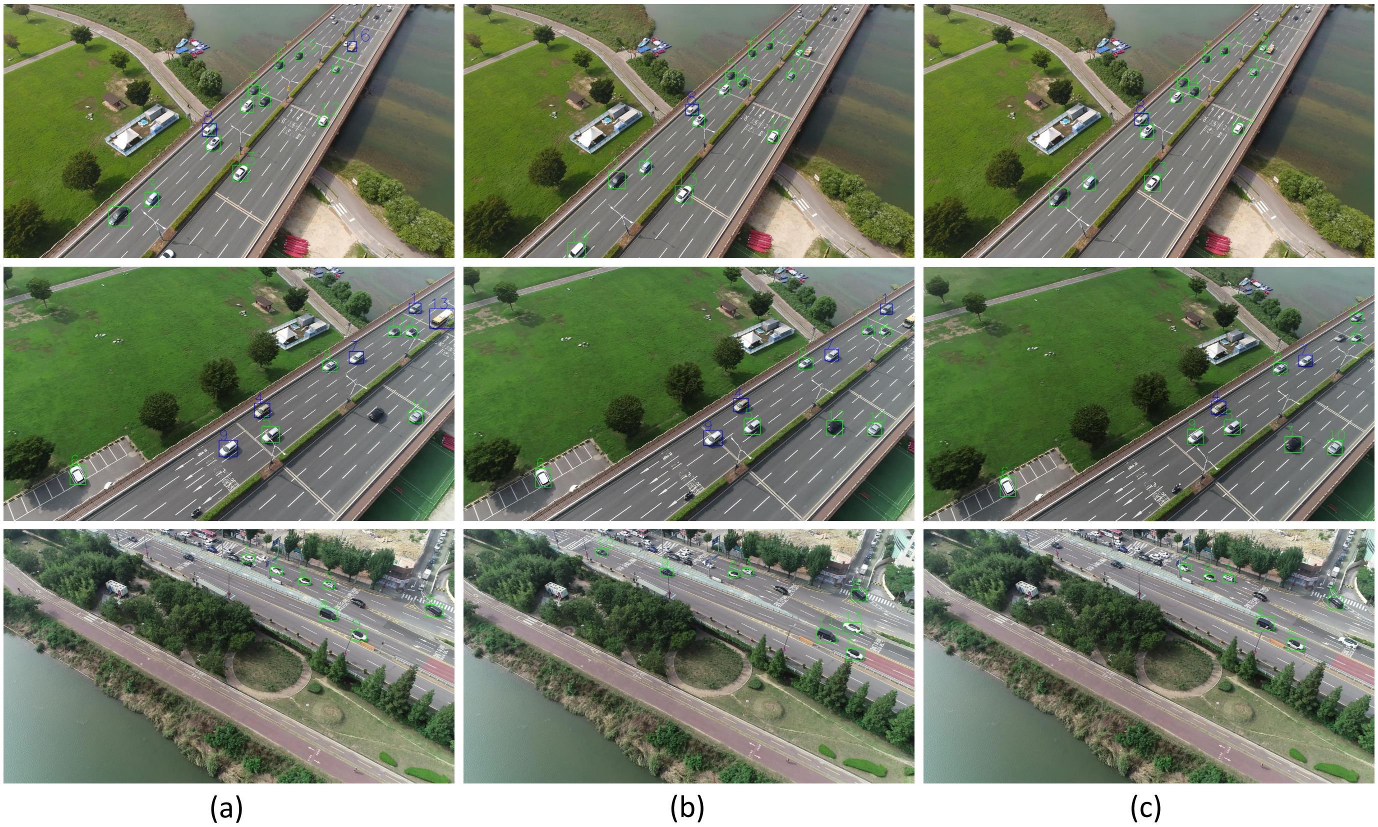


Fig. 8: The visualization of experimental results of vehicle re-identification from three video clips on the traffic road. Each image shows every 10 frames.

results are provided in https://drive.google.com/drive/folders/1N_dbv41w1a9Xi_Obs2x_WNLjP2gMXhEW?usp=sharing

V. Conclusion

This study proposed a simple vehicle re-ID algorithm for traffic road scenes shot by drone flight. Originally, the drone flight images are collected and annotated for the vehicle detection tasks rather than the vehicle re-ID tasks, and therefore the vehicle IDs are unknown here. To tackle the difficulty, this paper re-identifies vehicles by analyzing vehicles moving without using annotated identity information. The detector YOLOv5 is adopted in this paper for achieving high detection accuracy and speed. This algorithm does not need the high-cost GPU devices for CNN architecture. In future work, the proposed method is going to test on the edge devices like the NVIDIA jetson nano and Raspberry Pi 4 for real-time applications.

References

[1] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2015.

[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[3] S. Ren, K. He, R. B. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1137–1149, 2015.

[4] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, “Mask r-cnn,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 386–397, 2020.

[5] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *ArXiv*, vol. abs/1804.02767, 2018.

[6] C. Liang, Z. Zhang, Y. Lu, X. Zhou, B. Li, X. Ye, and J. Zou, “Rethinking the competition between detection and reid in multiobject tracking,” *IEEE Transactions on Image Processing*, vol. 31, pp. 3182–3196, 2022.

[7] Z. Wang, L. Zheng, Y. Liu, and S. Wang, “Towards real-time multi-object tracking,” *ArXiv*, vol. abs/1909.12605, 2020.

[8] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, “Fairmot: On the fairness of detection and re-identification in multiple object tracking,” *Int. J. Comput. Vis.*, vol. 129, pp. 3069–3087, 2021.

[9] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *ECCV*, 2014.

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei, “Imagenet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, pp. 211–252, 2015.

[11] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, pp. 303–338, 2009.

[12] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016.

[13] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6525, 2017.

[14] G. R. Jocher, A. Stoken, J. Borovec, NanoCode, A. Chaurasia, TaoXie, L. Changyu, Abhiram, Laughing, tkianai, yxNONG, A. Hogan, lorenzomammana, AlexWang, J. Hájek, L. Diaconu, Marc, Y. Kwon, Oleg, wanghaoyang, Y. Defretin, A. Lohia, ml ah, B. Milanko, B. Fineran, D. P. Khromov, D. Yiwei, Doug, Durgesh, and F. Ingham, “ultralytics/yolov5: v5.0 - yolov5-p6 1280 models, aws, supervise.ly and youtube integrations,” 2021.

[15] Q. Tang and K.-H. Jo, “Unsupervised person re-identification via nearest neighbor collaborative training strategy,” *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 1139–1143, 2021.

[16] Q. Tang, G. Cao, and K.-H. Jo, “Fully unsupervised person re-identification via multiple pseudo labels joint training,” *IEEE Access*, 2021.

[17] Y. Ge, D. Chen, F. Zhu, R. Zhao, and H. Li, “Self-paced contrastive learning with hybrid memory for domain adaptive object re-id,” *ArXiv*, vol. abs/2006.02713, 2020.

[18] Zakria, J. Deng, M. S. Khokhar, M. U. Aftab, J. Cai, R. Kumar, and J. Kumar, “Trends in vehicle re-identification past, present, and future: A comprehensive review,” *ArXiv*, vol. abs/2102.09744, 2021.

[19] S. V. Huynh, N.-H. Nguyen, N.-T. Nguyen, V. Nguyen, C. Huynh, and C. H. Nguyen, “A strong baseline for vehicle re-identification,” *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4142–4149, 2021.

[20] X. Zhu, Z. Luo, P. Fu, and X. Ji, “Voc-reld: Vehicle re-identification based on vehicle-orientation-camera,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2566–2573, 2020.

[21] L. N. Smith and N. Topin, “Super-convergence: very fast training of neural networks using large learning rates,” in *Defense + Commercial Sensing*, 2019.

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *NeurIPS*, 2019.