An algorithm using YOLOv4 and DeepSORT for tracking vehicle speed on the highway

Phat Nguyen Huu[1] Manh Bui Duy[2]
[1,2]School of Electrical and Electronic Engineering, Hanoi University of Science and Technology (HUST), Vietnam

Article Info

Article history:

Received Oct 12, 2021
Revised Jan 29, 2022
Accepted Feb 21, 2022

Keywords:

YOLOv4
DeepSORT
Object tracking
Tracking vehicle
Highways

ABSTRACT

Currently, expressways are increasingly developed and expanded. Several highways of Vietnam allow vehicles to travel up to 120 kilometers per hour helping to transport goods quickly and bringing a lot of socio-economic benefits. Vehicle monitoring plays an important role in reducing traffic accidents helping to handle violations. The standard for vehicle speed estimation is radar or lidar speed signs which can be costly to buy and maintain. The paper proposes a model to identify and monitor car speed on highways. The proposed method uses YOLOv4 combined with DeepSORT for vehicle identification and tracking. We then calculate the speed of the car based on video recording and sending it back from the highway. The execution context is a highway where vehicles move very fast. The results show that the system improves 46% average accuracy compared with [27] and [28] and execution times for up to 70 frames per second that is suitable for real systems.

Corresponding Author:

Phat Nguyen Huu,
School of Electrical and Electronic Engineering,
Hanoi University of Science and Technology (HUST),
01 Daicoviet Road, Hai Ba Trung district, Hanoi, Vietnam.
Email: phat.nguyenhuu@hust.edu.vn

1. INTRODUCTION

Traffic monitoring with an intelligent traffic system provides solutions to various challenges such as vehicle counting, speed estimation, accident detection, and traffic monitoring assistance. Traffic monitoring systems act as a tool to detect vehicles appearing on video images and estimate their locations when they are at the scene. In complex cases with many types of vehicles and a high density of vehicles, it is very difficult to accurately locate and classify vehicles in traffic flows. Furthermore, due to the limitations that occur in vehicle detection due to environmental changes, the vehicle's various features and detection rates are relatively low. Therefore, an algorithm has to be developed for a real-time traffic monitoring system with accurate vehicle detection and calculation capabilities. Therefore, the accurate and rapid detection of traffic vehicles from traffic images or videos has great practical significance.

Currently, expressways are increasingly developed and expanded. Several highways of Vietnam allow vehicles to travel up to 120 kilometers per hour helping to transport goods quickly and bringing a lot of socio-economic benefits. Vehicle monitoring plays an important role in reducing traffic accidents helping to handle violations. However, investment in researching an automatic monitoring system on highways is facing many difficulties. As a result, surveillance systems are performed by manual monitoring from video capturing and sending back on the highway. Therefore, we will study the car speed detection and monitoring algorithm on a highway that makes monitoring work easier in the paper.

On highways, vehicles need to be tracked and speed calculated in [1], [2]. There are two problems for existing systems, namely tracking and calculating the velocity of objects [3],[4],[5].

Object tracking (OT) is the problem of tracking one or more moving objects of video. It is a higher-level problem than object detection when the object being processed is not simply a single image. OT can be divided into two main approaches, namely single object tracking (SOT) and multiple object tracking (MOT).

SOT focuses on tracking a single object throughout the video. To know which object to tracking, we need to have a bounding box. The details of this method are mentioned in [6],[7],[8]. MOT is geared towards more extensible applications. In the MOT method, they try to simultaneously detect and track all objects including new objects appearing in the video. Therefore, MOT is often more difficult problems than SOT and receives a lot of attention from researchers [9], [10]. Besides, the methods of solving this class are also divided and popularized, namely online tracking and offline tracking [11],[12]. In online tracking, this method only uses current and previous frames for tracking while processing video. The treatment may reduce the accuracy of the algorithm. However, it reflects how the problem is handled in practice when "online" is essential. Several effective approaches have been proposed including sort [11] and DeepSORT [13]. In offline tracking, these methods usually use an entire frame of video. Therefore, it often achieves much higher accuracy than online tracking. However, its computational cost is much higher. While monitoring vehicle speed, vehicles must be tracked from frame to others since object detection may not require continuous tracking. The Kalman filter has been a reliable tool for solving object tracking tasks in many fields. The authors [3] have shown that a simple tracker can be better than complex ones when objects are reliably detected. The authors [16] present an algorithm for the detection of 3D bounding boxes of vehicles and then tracking and estimating the speed. However, the error of the system is still high from 30% to 40%. In [17], the authors proposed the system using homography and YOLOv4 object detectors that are capable of vehicle speed estimation. However, the speed of vehicles is still from 10 to 40 km/h.

Many studies have been performed for field of vehicle speed detection with different methods [14],[15]. The authors [15] propose an approach involving vehicle position comparison between the current and previous frames to predict traffic speed from digital video capturing by fixing camera. Camera calibration is performed by applying geometric equations. The system is designed to be scalable to other application domains and has an average error of ±7 kilometers per hour for detecting vehicle speeds. The authors [18] uses motion parameters of an image in conjunction with projection information between ground and image planes to obtain various metrics including vehicle speed. The method uses real-time monitoring techniques. The authors [14] presented an approach based on detecting the moving target of video by mapping the relationship between a pixel and actual distances. In the algorithm, three-frame background difference is used to extract features from moving vehicles. The authors [18] combined modern deep learning models to improve an efficient way to predict vehicle speed. However, the maximum speed of vehicles is 70 km/h. In [19], the authors reviewed vision-based vehicle speed estimation. The authors showed that the accuracy of the system depended on the speed camera and status traffic.

In general, the above existing methods have low accuracy and do not meet actual requirements in the highways of Vietnam. To our knowledge, the proposal system with speed calculation and usage tracking using YOLOv4 combined with DeepSORT has not been mentioned.

The objective of the paper is to build a model to recognize and monitor vehicle speed on highways with video input recording on the highway. The scope of the model is a highway with only vehicles moving at a fast speed. In the proposed model, we change two blocks to determine correct velocity compared to the traditional model as follows:

- First, we use YOLOv4 to combine DeepSORT to identify and track vehicles. This is a high-precision identification algorithm for the context of express highways where objects move fast and need real-time processing.
- Secondly, we convert from pixel to distance (meter) and use the recycle track to solve the same vehicle problems that appear in consecutive frames.

Our main contributions to the paper include:

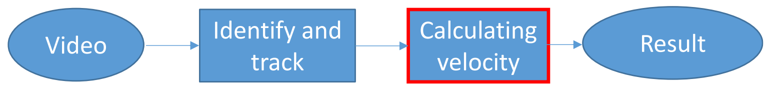
- Improving performance in tracking multiple vehicles by removing unnecessary vehicles from the tracklist.
- The velocity determination method has high accuracy and low processing cost by reducing the number of calculation parameters.

The rest of the paper includes five parts and is organized as follows. Section 1 presents an overview of estimating the distance to the camera system. Section 2 will evaluate the proposed model. In section 3, we analyze the results. In the final section, we give conclusions and future research directions.

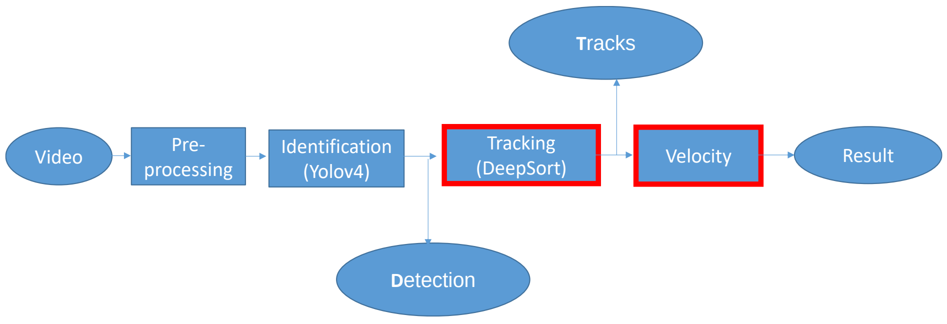
2. PROPOSED SYSTEM

2.1. Overview of proposal system

Figure 1 is an overview of the proposal system. Video through the pre-processor block obtains images that are input into the recognition module. The id will be assigned to track an object through the tracking block after predicting the label and location of an object. Vehicles after identifying and assigning Id will be included in the velocity block to calculate speed. Details of implementation blocks are as follows.



a)



b)

Figure 1. The proposal model (a) overview and (b) detailed implementation.

2.1.1 Identification block

In the context of highways where the vehicles move very fast, it is necessary to use a recognition algorithm with fast processing speed and suitable accuracy. Therefore, we use YOLOv4 for the identification module.

YOLOv4 is a recognition algorithm suitable for real-time recognition problems [20],[21],[22]. It strikes a balance between processing speed and accuracy. Besides, we can use YOLOv4-tiny, a stripped-down version of YOLOv4, with almost 10 times fewer parameters than the original version. Table 1 is a chart comparing the processing speed and accuracy of YOLO-v4 and YOLOv4-tiny with other recognition algorithms [23], [29]. In Tab. 1, AP and ASFF are average precision and adaptively spatial feature fusion, respectively. The details of this method are mentioned in [25] and [31]. They proposed a novel and data-driven strategy for pyramidal feature fusion that is called ASFF. It learns the way to spatially filter conflictive information to suppress the inconsistency that improves the scale-invariance of features and nearly free inference overhead.

Table 1. Comparison of average precision (AP) among YOLOv4 with other algorithms using MS COCO object detection [23].

FPS	Adaptively spatial feature fusion (ASFF)	EfficientDet	YOLOV3	YOLOV4
10	45	49	36	45.5
30	44	45	35	45
50	41	39	34	44
70	34	31	33	43.5
90	10	10	32.5	43
110	5	5	31.5	40
120	5	5	30	38

YOLO architecture includes the following blocks. The base network is convolutional and fully connected layers to extract features. YOLOv4 uses the backbone as cspdarknet35. In YOLOv4, the neck block has the function of getting rich information for typical maps. The next part is extra layers applied to detect objects on a characteristic map of the base network. YOLO architecture is also quite diverse and able to customize for different input shapes.

The YOLO architecture includes base networks that are convolution networks to perform feature extraction. The following are extra layers applied to detect objects on the feature map of the base network. The base network uses mainly convolutional and fully connected layers. YOLO architectures are also quite diverse and can be customized into versions for different input shapes.

Darknet architecture component (base network) is used to extract features. The output of the base network is a 7 × 7 × 1024 feature map that will be used as input for extra layers that predict label objects and bounding box coordinates.

In YOLO version 3, the authors apply network feature extractor Darknet-53. The network consists of 53 consecutively connected convolutional layers and each of them is followed by batch normalization and leaky Relu activation. To reduce output size after each convolution layer, the authors downsample with filters of size 2. This helps to reduce the number of parameters for the model.

The output of the YOLO model is a vector that will include the following components as

$$y = \left[p_o, \left(t_x, t_y, t_w, t_h \right), \left(p_1, p_2, \ldots, p_c \right) \right], \tag{1}$$

where p_0 is predicting the probability that an object will appear in a bounding box. (t_x, t_y, t_w, t_h) help to define the bounding box where t_x, t_y are coordinates of center and t_w, t_h are width and length dimensions of a bounding box. (p_1, p_2,... p_c) is predictive probability distribution vector of classes.

To evaluate the accuracy of Yolo, the loss function is used. It is divided into two parts: L_{loc} (localization loss) is used to measure the error of bounding box and L_{cls} (confidence loss) is used to measure the error of probability distribution of classes as

$$L_{loc} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [\left(x_i - \hat{x}_i \right)^2 + \left(y_i - \hat{y}_i \right)^2 + \left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2, \tag{2}$$

$$L_{cls} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} (1_{ij}^{obj} + \lambda_{noobj} \left(1 - 1_{ij}^{obj} \right)) \left(C_{ij} - \hat{C}_{ij} \right)^2 + \sum_{i=0}^{S^2} \sum_{c \in C} 1_i^{obj} \left(p_i(c) - \hat{p}_i(c) \right)^2, \tag{3}$$

$$L = L_{loc} + L_{cls}. \tag{4}$$

More detail of architecture is shown in Fig. 2.

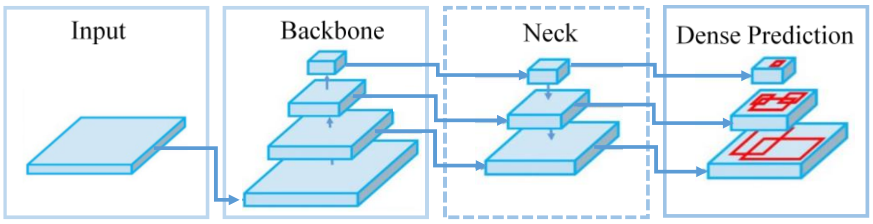


Figure 2. Yolov4 network structure [25, 26].

2.1.2 Tracking block

After identifying objects with the Yolo algorithm, we use the DeepSORT algorithm [17] to perform monitoring and assigning IDs for vehicles. The method of monitoring objects using DeepSORT includes the following steps:

(1) Step 1: Using Yolo to detect objects of the current frame.

(2) Step 2: Using Kalman filter to predict new tracks based on past ones. These statuses at the beginning will be assigned a tentative value. If the value is still guaranteed to be maintained in the next three frames, the status will move from probe to confirmation status. It will try to maintain the following 30 frames.

Otherwise, if losing signs when there are not enough three frames, the state will be deleted from the tracking process.

(3) Step 3: Using confirming tracks, we take into matching cascade to link detection objects based on distance and characteristics.

(4) Step 4: Tracks and detections that have not been linked will be taken to a subsequent filter layer. We use the Hungarian algorithm [27] to solve the assignment problem with the IOU cost matrix to the second link.

(5) Step 5: Handling, classifying the detection, and tracks

(6) Step 6: Using Kalman filter to correct the value of track from detections that are linked to it and initialize new ones.

2.1.3 Velocity block

Instead of determining the travel distance of each vehicle at multiple locations through each frame, we fixed two locations as landmarks to determine the travel distance of the vehicle. This method will have much higher accuracy than calculating the distance traveled for each vehicle since only one parameter needs to be determined that is the time to travel through that fixed area. In addition, the accuracy is no longer dependent on the viewing angle of the surveillance camera

In the block, we rely on the travel time of a vehicle for fixing distance to determine its speed. More details are shown in Fig. 3.

The distance *S* is determined using the main measurement method. It is the distance between l_1 and l_2 lines of the frame. The movement time between l_1 and l_2 is calculated by the formula:

$$t = \frac{N}{FPS}, \tag{5}$$

where t is time to move between l_1 and l_2 and N is the number of frames that move from l_1 and l_2, and FPS (frame per second) is the number of frames within one second.
Since we calculates vehicle speed v of formula:

$$v = \frac{s}{t}. \tag{6}$$

The most difficult of the block is that we must rely on the actual measurement and video to determine how many pixels on the image correspond to the actual one meter.

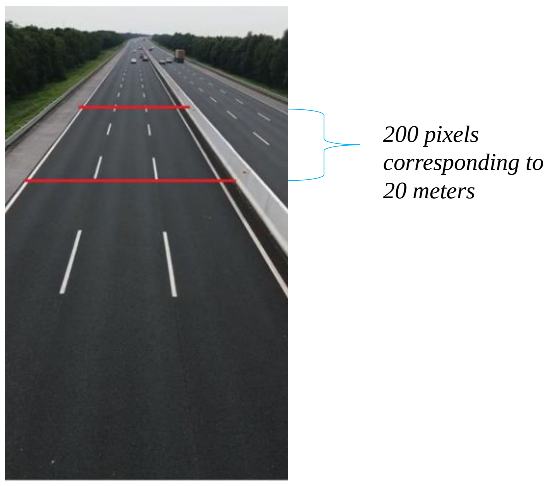


Figure 3. An example of the determination of velocity based on fixing distance.

2.2. Recycle track

In addition, we also added the Recycle track process to increase accuracy as well as minimize processing costs for the operation. Compared with the traditional YOLOv4 combined with DeepSORT or other Object tracking models, our object detection and tracking model has higher accuracy and processing performance for highway scenes since the vehicles only need to be tracked for a very short time while determining the velocity.

Deleting track after 30 frames in section 3.1 will result in vehicles of the same shape appearing in consecutive frames causing ID vehicles to be mistaken. Therefore, we use a solution called recycle track to discard tracks after being identified and display velocity to minimize errors and free up memory for them.

In [25], life-cycle management of track will include 3 states, namely probing, confirming, and deleting steps, as shown in Fig. 5. The process is as follows:

(1) These states are initially assigned an exploratory value (tentative).

(2) If this value is still guaranteed to be maintained for the next three frames, the state will change from the probe to confirmed.

(3) Tracks with confirmed status will try to stay track. Deep SORT will still maintain tracking for the next 30 frames even if they disappear.

(4) Otherwise, the status will be deleted from the tracker if the track is lost when less than three frames.

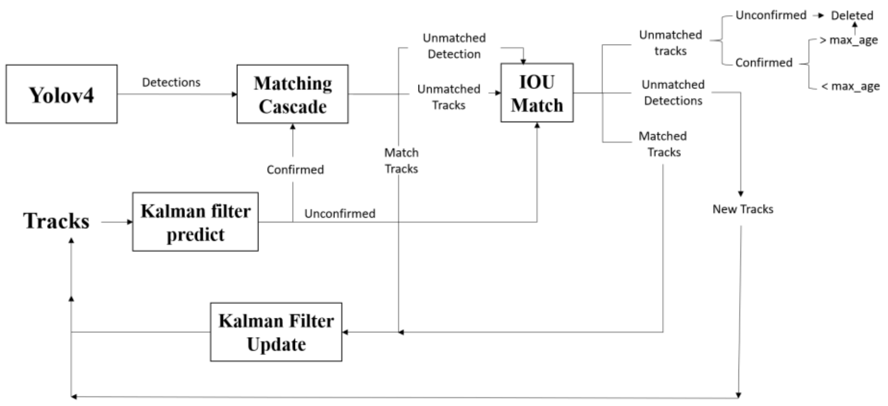


Figure 4. Process flow diagram of DeepSORT [11].

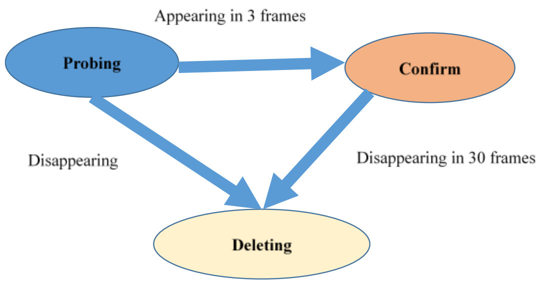


Figure 5. Managing track life-cycle in DeepSORT.

The processing flow of DeepSORT is performed sequentially as shown in Fig. 4 through the following steps:

(1) Step 1: Using faster region CNN (with backbone VGG16) to detect objects of the current frame,

(2) Step 2: DeepSORT uses Kalman filter to predict new track states based on past ones. These states are initially assigned a tentative value. If this value is still guaranteed to be maintained for the next three frames, it will change from the probe to confirmed and will try to stay tracked for the next 30 frames. Otherwise, it will be removed from the tracker if the track is lost when less than three frames are reached,

(3) Step 3: Using verified tracks and performing matching cascade to associate detections based on their distance and feature metrics,

(4) Step 4: Unlinking tracks and detections will be passed to the next filter layer. Using Hungarian algorithm to solve assignment problem with IOU cost matrix to link,

(5) Step 5: Processing and classifying detections and tracks,

(6) Step 6: Using Kalman filter to recalibrate value from detections associated with track and initialize new ones.

DeepSORT has improved the problem of SORT with association strategy as well as using appropriate metrics. The number of switches ID is reduced from 1423 to 781 which is a 45% reduction and also reduces errors relating to objects being obscured or disappearing. Although processing speed is slightly reduced, DeepSORT still ensures approximately real-time speed with GPU.

Details of the recycling track process are shown in Fig. 6.

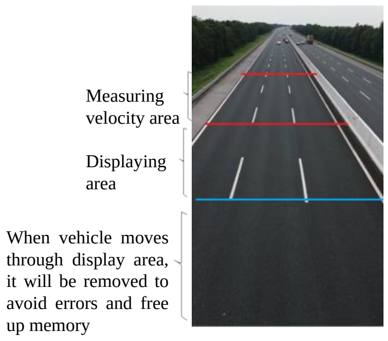

Measuring velocity area
Displaying area
When vehicle moves through display area, it will be removed to avoid errors and free up memory

Figure 6. Details of steps of recycling track process.

2.3. Prediction model
2.3.1. Preparing data
Data used in the paper consists of 3000 images cut from videos taken by us on the highway as shown in Fig. 7. This is the data that we built ourselves.

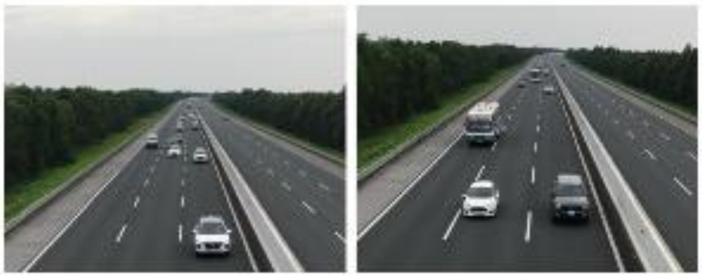


Figure 7. Images of vehicles on the highway are taken by ourselves.

2.3.2. Data labeling
In this step, we perform ROI block determination based on manual labeling. In the paper, we use an available tool called labeling. The process is drawing boxes around an object of the image. Figure 8 shows an example using the LabelImg tool that automatically creates a ".txt" file describing the location of an object in an image.

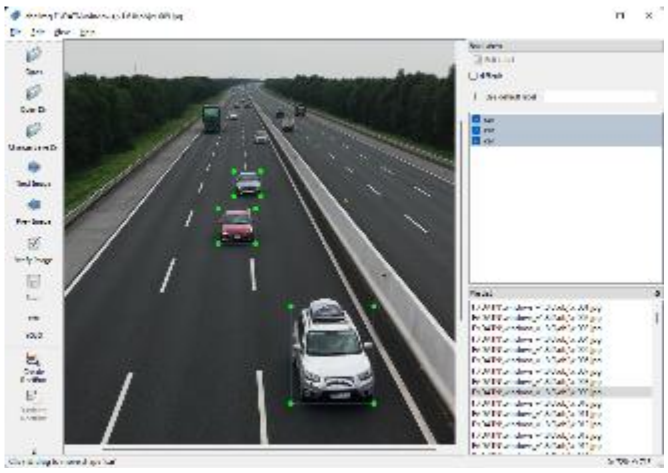


Figure 8. An example of data labeling.

The operations performing as shown in Fig. 9 are based on [28]. After assigning data labels, we divide them into train/test files and create a configuration file of parameters and upload it to drive. Model is next trained by Google collab. Finally, the values are put into the model for evaluation.

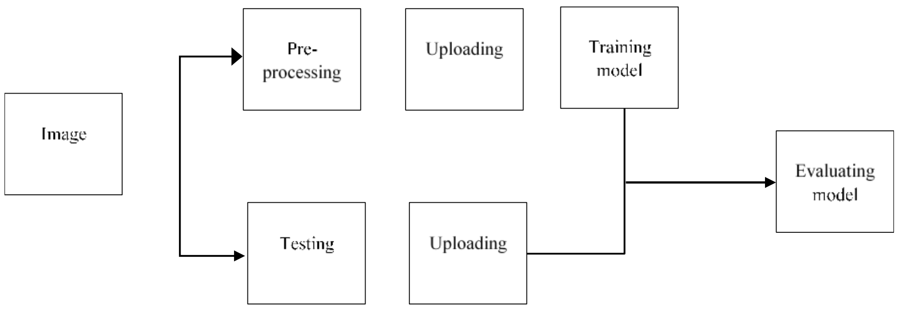


Figure 9. Detailed model of labeling implementation.

After training the model, we proceed to identify and track the vehicle. The output of the identification process includes Id, box, and frame. They are input data to the velocity module. The results of the training process are shown in Tab. 2.

Table 2. Result of training model.

Type	Filters	Size	Input	Output
0 conv	16	3 × 3/1	416 × 416 ×3	416 × 416 × 16 0.150 BF
1 max		2 × 2/2	416 × 416 × 16	208 × 208 × 16 0.003 BF
2 conv	32	3 × 3/1	208 × 208 × 16	208 × 208 × 32 0.399 BF
3 max		2 × 2/2	208 × 208 × 32	104 × 104 × 32 0.001 BF
4 conv	64	3 × 3/1	104 × 104 × 32	104 × 104 × 64 0.399 BF
5 max		2 × 2/2	104 × 104 × 64	52 × 52 × 64 0.001 BF
6 conv	128	3 × 3/1	52 × 52 × 64	52 × 52 × 128 0.399 BF
7 max		2 × 2/2	52 × 52 × 128	26 × 26 × 128 0.000 BF
8 conv	256	3 × 3/1	26 × 26 × 128	26 × 26 × 256 0.399 BF
9 max		2 × 2/2	26 × 26 × 256	13 × 13 × 256 0.000 BF
10 conv	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512 0.399 BF
11 max		2 × 2/1	13 × 13 × 512	13 × 13 × 512 0.000 BF
12 conv	1024	3 × 3/1	13 × 13 × 512	13 × 13 × 1024 1.595 BF
13 conv	256	1 × 1/1	13 × 13 × 1024	13 × 13 × 256 0.089 BF
14 conv	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512 0.399 BF
15 conv	18	1 × 1/1	13 × 13 × 512	13 × 13 × 18 0.003 BF
16 yolo				
17 route	13			
18 conv	128	1 × 1/1	13 × 13 × 256	13 × 13 × 128 0.011 BF
19 upsample		2 ×	13 × 13 × 128	26 × 26 × 128
20 route	19			
21 conv	256	3 × 3/1	26 × 26 × 384	26 × 26 × 256 1.196 BF
22 conv	18	1 × 1/1	26 × 26 × 256	26 × 26 × 18 0.006 BF
23 yolo				
Total BFLOPS = 5.448				
Learning rate =0.001				
Momentum = 0.9				
Decay = 0.0005				

3. SIMULATION AND RESULT

3.1. Setup

We use videos taken on any highway to evaluate the accuracy of the tracking model and calculate vehicle speed in the paper. For tracking identification module, we perform as following

(1) Using two modules yolov4 + DeepSORT, yolov4-tiny + DeepSORT,

(2) Evaluating of accuracy and performance of the model.

For the velocity calculation module, we evaluate the results through two methods as follows:

(1) Determine velocity for every tracking vehicle of frame [27][28],

(2) Determine vehicle speed to be tracked in specified location in the proposed method.

Figure 10 shows the detail of the testing system.

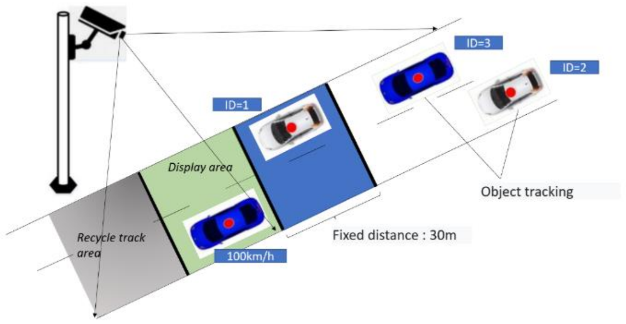


Figure 10. Illustrating of the system scenario

3.2. Result

3.2.1. Tracking module

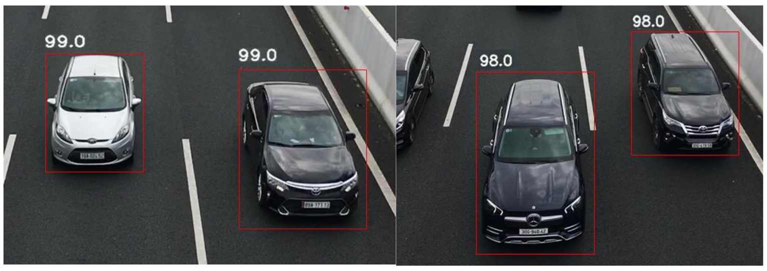
Firstly, we perform identification and tracking modules on the highway. The results are shown in Fig. 11 and Tabs. 3 and 4.

Table 3. Result of the frame rate of yolov4-tiny+DeepSORT.

Track ID	Class	BBox Coord (xmin, ymin, xmax, ymax)
10	car	(1123, 445, 1402, 660)
27	car	(603, 252, 762, 362)
29	car	(350, 232, 521, 364)
FPS =41.98		
Frame number = 262		
10	car	(1133, 452, 1404, 663)
27	car	(608, 252, 766, 361)
29	car	(358, 229, 534, 365)
FPS =40.94		
Frame number = 263		
10	car	(1140, 460, 1405, 665)
27	car	(611, 252, 768, 362)
29	car	(361, 227, 541, 367)
FPS =41.48		
Frame number = 264		

Table 4. Comparing the accuracy of Yolov4 and Yolov4-tiny.

Critical	Yolov4 + DeepSORT	Yolov4-tiny + DeepSORT
Scene	Good weather, bright enough	Good weather, bright enough
Training time	30 hours	3 ÷ 4 hours
Video length	5 minutes	5 minutes
Accuracy	93 ÷ 99%	90 ÷ 98%
Processing speed	7 ÷ 10 FPS	29 ÷ 70 FPS



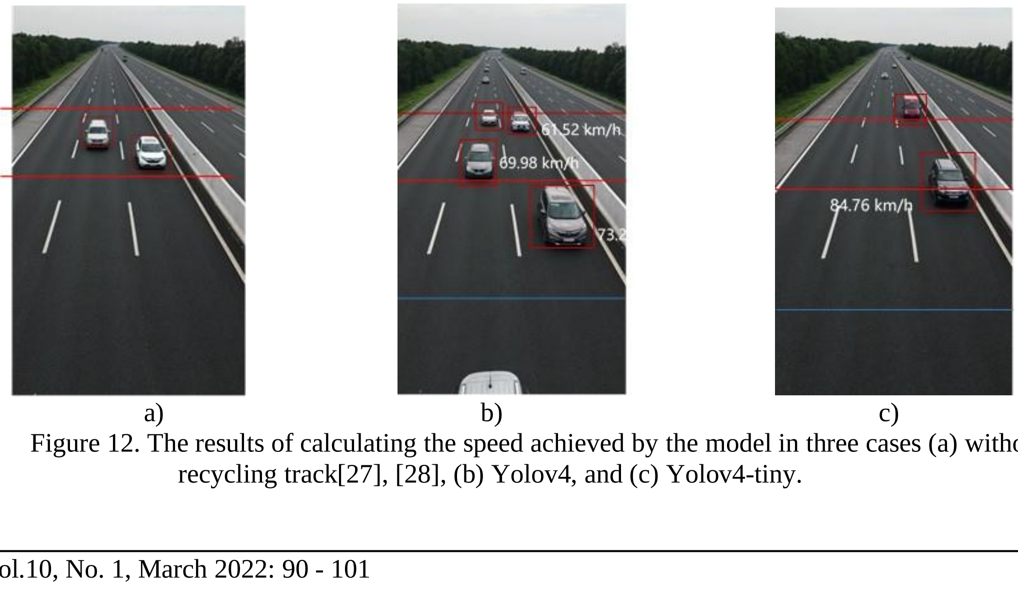
a) b)

Figure 11. Tracking results based on bounding box with (a) Yolov4 and (b) Yolov4-tiny.

From the results of Tab. 4, we see that Yolov4 + DeepSORT method has higher accuracy under the same conditions. However, computation time is large when the Yolov4 tiny + DeepSORT method has fast computation time that is suitable for real-time applications. Besides, its accuracy is not significantly lower.

3.2.2. Velocity module

After using DeepSORT to track vehicles to calculate their speed, we get the result as shown in Fig. 12. In Fig. 12 (a), we show that the results do not match reality. In several cases, vehicle ID is mistaken leading to wrong speed because of not recycling track. In Fig. 12 (b), the results of model yolov4 give high accuracy with actual processing speed from 7 to 12 FPS.



a) b) c)

Figure 12. The results of calculating the speed achieved by the model in three cases (a) without recycling track[27], [28], (b) Yolov4, and (c) Yolov4-tiny.

In Fig. 12, although the vehicles [27], [28] are updated with speed continuously for each frame, the processing time is greatly increased due to having to save the position of each vehicle of previous images. The second disadvantage is that distance of pixels is highly dependent on aspect ratio which leads to lower accuracy than selecting and measuring points. Besides, determining the time based on FPS for each frame leads to great dependence on processing speed. As a result, the same video is performed twice times and produces two different results. In our proposal, processing time increases slightly and accuracy is higher due to pre-measured distance. The only disadvantage is the limit on the number of velocity updating in the frame.

Table 5. Comparing accuracy with [27], [28]

Critical	Proposal	[27], [28]
Scene	Good weather, bright enough	Good weather, bright enough
Video length	5 minutes	5 minutes
Accuracy	93 ÷ 99%	<50%
Processing speed	30 ÷ 70 FPS	<7 FPS

We evaluate the performance and processing time of the proposed system with [27], [28] on computers with core I5 configuration and 16 GB RAM. The results are shown in Tab. 5. We see that the system achieves an accuracy of over 90% with execution time up to 70 FPS which is suitable for speed monitoring applications.

Besides, we also evaluated the system with real speeds between 70 and 120 km/h. The results are shown in Tab. 6. The system will achieve accuracy at speeds below 140 kilometers per hour which is suitable for highways in Vietnam. Table 6 shows the accuracy of the speed calculation. In the Table, an average error is 10%.

Table 6. Evaluating of actual and calculated speed.

ID vehicle	Actual speed	Calculated speed	Error (%)
1	93	90	3.23
2	91	92	1.1
3	86	90	4.65
4	92	92	0
5	90	92	2.22
6	105	108	2.86
7	87	90	3.45
8	75	72	4
9	75	73	2.67
10	73	71	2.74
11	100	96	4
12	98	103	5.1
13	90	83	7.78
14	78	73	6.41
15	86	80	6.98
16	82	79	3.66
17	80	72	10

4. CONCLUSION

The paper focuses on using neural networks for recognizing and tracking the speed of vehicles moving on highways. In the paper, we have identified vehicles on highways with an accuracy of over 90%. The determining speed is consistent with reality and the processing time of the model is low (20 Fps without VGA and up to 70 Fps for I5 configuration 9400 ram 16Gb using VGA 1050Ti). However, speed tracking and monitoring only achieve high results in bright enough daytime weather conditions.

Therefore, we will improve the model in bad weather conditions at night to increase the accuracy of the model in all cases in the next direction. In addition, the future will aim to build a model capable of calculating vehicle speed at every position of the frame.

ACKNOWLEDGMENTS

This research is carried out in the framework of the project funded by the Ministry of Education and Training (MOET), Vietnam under the grant B2020-BKA-06. The authors would like to thank the MOET for their financial support.

REFERENCES

[1] S. Hua, M. Kapoor, and D. C. Anastasiu, “Vehicle tracking and speed estimation from traffic videos,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 153–1537.

[2] L. Lou, Q. Zhang, C. Liu, M. Sheng, Y. Zheng, and X. Liu, “Vehicles detection of traffic flow video using deep learning,” in 2019 IEEE 8th Data-Driven Control and Learning Systems Conference (DDCLS), 2019, pp. 1012–1017.

[3] E. Bochinski, V. Eiselein, and T. Sikora, “High-speed tracking-by-detection without using image information,” in 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2017, pp. 1–6.

[4] Z. Soleimanitaleb, M. A. Keyvanrad, and A. Jafari, “Object tracking methods: a review,” in 9th International Conference on Computer and Knowledge Engineering (ICCKE), 2019, pp. 282–288.

[5] L. Fan, Z. Wang, B. Cail, C. Tao, Z. Zhang, Y. Wang, S. Li, F. Huang, S. Fu, and F. Zhang, “A survey on multiple object tracking algorithm,” in IEEE International Conference on Information and Automation (ICIA), 2016, pp. 1855–1862.

[6] A. Ellouze, M. Ksantini, F. Delmotte, and M. Karray, “Single object tracking applied to an aircraft,” in 15th International Multi-Conference on Systems, Signals Devices (SSD), 2018, pp. 1441–1446.

[7] R. Seth, S. Kumar Swain, and S. Kumar Mishra, “Single object tracking using estimation algorithms,” in 2nd International Conference on Power, Energy and Environment: Towards Smart Technology (ICEPE), 2018, pp. 1–6.

[8] B. Mocanu, R. Tapu, and T. Zaharia, “Single object tracking using offline trained deep regression networks,” in 7th International Conference on Image Processing Theory, Tools and Applications (IPTA), 2017, pp. 1–6.

[9] S. Sun, N. Akhtar, H. Song, A. Mian, and M. Shah, “Deep affinity network for multiple object tracking,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 1, pp. 104–119, 2021.

[10] W. Li, J. Mu, and G. Liu, “Multiple object tracking with motion and appearance cues,” in IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 161–169.

[11] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in IEEE International Conference on Image Processing (ICIP), 2016, pp. 3464–3468.

[12] J. Tang, X. Xiong, C. Xie, Y. Zhang, P. Wang, F. Wang, F. Du, L. Han, Y. Zheng, P. Pan, and H. Li, “Min-cost network flow and trajectory fix for multiple objects tracking,” in In Conference on Computer Vision and Pattern Recognition Workshop, 2020, pp. 1–4.

[13] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in IEEE International Conference on Image Processing (ICIP), 2017, pp. 3645–3649.

[14] Z. Tang, G. Wang, T. Liu, Y.-G. Lee, A. Jahn, X. Liu, X. He, and J.-N. Hwang, “Multiple-kernel based vehicle tracking using 3d deformable model and camera self-calibration,” 2017.

[15] A. Gholami, A. Dehghani, and M. Karim, “Vehicle speed detection in video image sequences using cvs method,” International Journal of Physical Sciences, vol. 5, pp. 2555–2563, Dec. 2010.

[16] V. Kocur, M. Ftáčnik, “Detection of 3D bounding boxes of vehicles using perspective transformation for accurate speed measurement,” Machine Vision and Applications, vol. 31, no. 62, pp. 1-15, 2020

[17] H. Mejia, E. Palomo, E. L´opez-Rubio, I. Pineda, and R. Fonseca, “Vehicle speed estimation using computer vision and evolutionary camera calibration,” in NeurIPS Workshop LatinX in AI, 2021

[18] S. Hua, M. Kapoor, and D.C. Anastasiu, “Vehicle tracking and speed estimation from traffic videos,” IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 153-1537

[19] D. Fernández-Llorca, A. Hernandez Martinez, and I Garcia Daza, “Vision-based vehicle speed estimation: A survey,” IET Intelligent Transport Systems, vol. 15, no. 3, pp. 1-16, 2021.

[20] M. Maity, S. Banerjee, and S. Sinha Chaudhuri, “Faster r-cnn and yolo based vehicle detection: A survey,” in 5th International Conference on Computing Methodologies and Communication (ICCMC), 2021, pp. 1442–1447.

[21] A. K. Shetty, I. Saha, R. M. Sanghvi, S. A. Save, and Y. J. Patel, “A review: Object detection models,” in 6th International Conference for Convergence in Technology (I2CT), 2021, pp. 1–8.

[22] K. Li and L. Cao, “A review of object detection techniques,” in 5th International Conference on Electromechanical Control Technology and Transportation (ICECTT), 2020, pp. 385–390.

[23] A. Bochkovskiy, C.-Y. Wang, and H.-y. Liao, “Yolov4: Optimal speed and accuracy of object detection,” pp. 1–17, April 2020.

[24] P. Adarsh, P. Rathi, and M. Kumar, “Yolo v3-tiny: Object detection and recognition using one stage improved model,” in 6th International Conference on Advanced Computing and Communication Systems (ICACCS), 2020, pp. 687–694.

[25] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, “Online multi-target tracking with strong and weak detections,” vol. 9914, Oct. 2016, pp. 84–99.

[26] R. Phadnis, J. Mishra, and S. Bendale, “Objects talk - object detection and pattern tracking using tensorflow,” in Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018, pp. 1216–1219.

[27] J. Lmer, D. Cymbalak, and F. Jakab, “Computer vision based object recognition principles in education,” in IEEE 11th International Conference on Emerging eLearning Technologies and Applications (ICETA), 2013, pp. 253–257.

[28] A. Rosebrock, “Simple object tracking with opencv,” July 23, 2018, retrieved from official website: https://www.pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/.

[29] Songtao Liu, Di Huang, and Yunhong Wang. Learning spatial fusion for single-shot object detection. arXiv preprint arXiv:1911.09516, 2019

BIOGRAPHY OF AUTHORS

Phat Nguyen Huu received his B.E. (2003), M.S. (2005) degrees in Electronics and Telecommunications at Hanoi University of Science and Technology (HUST), Vietnam, and Ph.D. degree (2012) in Computer Science at Shibaura Institute of Technology, Japan. Currently, he lecturer at Electrical and Electronic Engineering, HUST Vietnam. His research interests include digital image and video processing, wireless networks, ad hoc and sensor networks, intelligent traffic systems (ITS), and the internet of things (IoT). He received the best conference paper award in SoftCOM (2011), best student grant award in APNOMS (2011), hisayoshi yanai honorary award by Shibaura Institute of Technology, Japan in 2012.

Manh Bui Duy is a student of Electronics and Telecommunications at Hanoi University of Science and Technology (HUST), Vietnam. Currently, he is working in the Future Network lab. at HUST. His main duty is developing smart products which relate to digital image, video processing, machine learning, and the internet of things (IoT).