ICIC Express Letters
Volume **17**, Number **5**, May **2023**

ICIC International ©2023 ISSN 1881-803X
pp. **595–604**

INTEGRATED TRAFFIC VIOLATION TYPE DETECTION AND RECOGNITION SYSTEM USING VIDEO PROCESSING BASED CONVOLUTIONAL NEURAL NETWORK

Ilham Fazri and Ika Candradewi*

Department of Computer Science and Electronics
Universitas Gadjah Mada
Bulaksumur, Yogyakarta 55281, Indonesia
ilhamfazri99@mail.ugm.ac.id; *Corresponding author: ika.candradewi@ugm.ac.id

Received May 2022; accepted August 2022

Abstract. *Based on data from the World Health Organization in 2018, the number of deaths worldwide due to road traffic accidents is 1.35 million people every year. One of the causes is the low level of driver discipline in driving, which is indicated by violating traffic regulations. The solution for this problem is implementing a traffic violation detection system based on computer vision and deep learning. The system designed in this study can detect traffic violations that include running a red light, not wearing a helmet, and being wrong-way. This study implements the YOLOv5 architecture as object detection has 74% mAP50 value performance and uses SORT as object tracking. Vehicle detectors and trackers are then integrated with methods designed to detect traffic violations. The test results using several sample video scenarios show that the running red light detector has an F1-Score value of 0.95. The helmet violation detector based on EfficientNet as a classifier has an F1-Score value of 0.88, and the wrong-way detector has an F1-Score value of 1.00.*

Keywords: Traffic violation, Computer vision, CNN, Deep learning, YOLO

1. Introduction.

Based on the World Health Organization (WHO) data, the number of deaths worldwide due to road traffic accidents is 1.35 million people. About 20-50 million people suffer severe and minor injuries [1]. In addition, based on data from Bappenas in 2018, 31,282 people died due to land traffic accidents in Indonesia [2]. The human factor is the most dominant in traffic accidents in Indonesia [3], marked by violating traffic rules. One solution to reduce this problem is implementing a traffic violation detection system.

Traffic violation detection systems generally implement two main processes, vehicle detection and violation detection. In-vehicle detection can use the object detection method approach. Currently, object detection has two categories, one-stage object detection and two-stage object detection. One-stage object detection system predicts object localization and classification simultaneously so that it is relatively faster than two-stage object detection. One of the famous architectures is YOLO (You Only Look Once) Family [4, 5, 6]. [7, 8] used YOLO for vehicle detection. Whereas the two-stage object detection system will look for a collection of ROI (Region of Interest), and each ROI will be classified. Meanwhile, the famous two-stage object detection architecture is the Region based Convolutional Neural Networks (RCNN) Family [9, 10, 11]. [12, 13, 14] used RCNN for vehicle detection.

The violation detection method varies according to the type of violation to be detected. There are several violation detection method weaknesses that we are trying to solve in this study. The system we proposed can detect and process three types of violations wrong-way violation, helmet violation (not wearing a helmet), and running a red light

DOI: 10.24507/icicel.17.05.595

simultaneously. For wrong-way violation, we proposed another approach [13]. Instead of classifying a motorcycle's front and rear view frame using CNN, we predict the movement based on object tracker data. For helmet violation, we use the same idea of [7]. Instead of classifying the upper part of the frame from the bounding box using CNN, we classify the entire frame from the bounding box using finetuned EfficientNet model, which can handle the situation when the vehicle is facing back and situation when one of the passengers or drivers is not wearing a helmet. For running a red light, we modified [14]. Instead of predicting based on crossing between the centroid of the vehicle bounding box and a protected area when the traffic light is red, we changed the protected area with a stop line. Lastly, the code and models are publicly available at https://github.com/ilhamfzri/traffic-violation-detection.git.

2. **Methodological Approach.** In this research, various computer vision and deep learning methods are implemented. The system's working is generally visualized by the diagram shown in Figure 1. Image sequence is a collection of frames obtained from video surveillance. Then the image data will be processed by the Vehicle Detection & Tracker (VDT) section to detect and track vehicle objects on the image. The system uses the output of VDT and manual calibration parameters to detect violations with different methods based on the type of violation. The output results in each violation detection section are used as input data for the violation recorder to create a violation annotation for each vehicle.

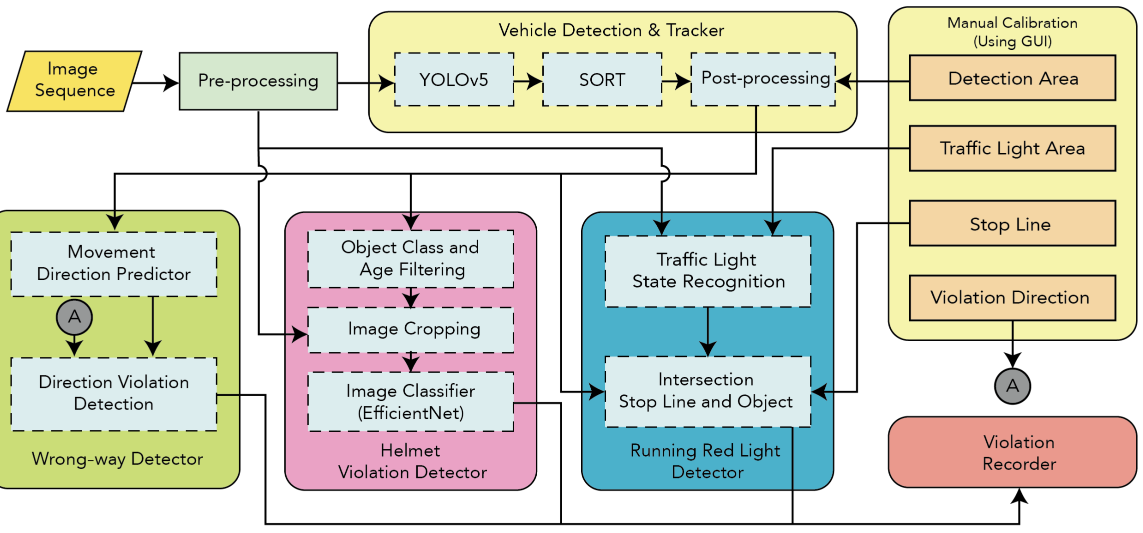


Image Sequence
Pre-processing
Vehicle Detection & Tracker
YOLOv5
SORT
Post-processing
Manual Calibration (Using GUI)
Detection Area
Traffic Light Area
Stop Line
Violation Direction
Movement Direction Predictor
A
Direction Violation Detection
Wrong-way Detector
Object Class and Age Filtering
Image Cropping
Image Classifier (EfficientNet)
Helmet Violation Detector
Traffic Light State Recognition
Intersection Stop Line and Object
Running Red Light Detector
A
Violation Recorder

FIGURE 1. System diagram

2.1. **Vehicle detection & tracking.** In detecting vehicles, this system implements the object detection method using the YOLOv5 architecture [15] shown in Figure 2. For the object tracking process, we use the Simple Online and Realtime Tracking (SORT) method [16]. Then we use a post-processing algorithm from the output of the object tracker to remove object data outside the detection area. The area is determined using GUI.

The vehicle detection detects five classes of vehicles: motorcycle, bus, truck, and bicycle. We used YOLOv5 as the baseline model. The reason is based on the consideration that YOLOv5 has good accuracy and is faster in the inference process. In the training phase, we combine the COCO [17], KITTI [18], and GTA V Vehicle Dataset with the purpose that the model can generalize better. GTA V Vehicle Dataset is a collection of vehicle object images taken from a height from the results of video game simulations collected by the authors. The total number of images we used is 25197 for training and 2825 for validation.

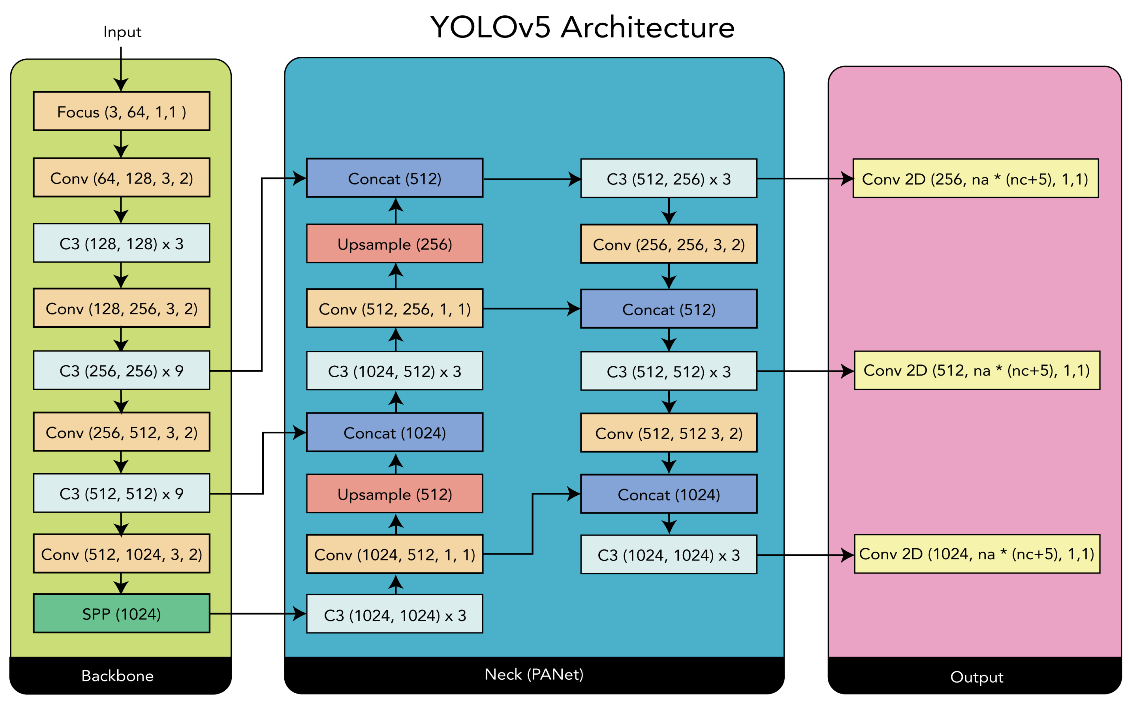


FIGURE 2. YOLOv5 architecture

Vehicle tracking has functions to track and provide ID for each vehicle. We use Simple Online and Realtime Tracking (SORT) because this method is good enough and very light in the computational process. However, the output of the vehicle tracking will detect all the vehicles in the frame, which can make the system difficult to detect running red light violation because vehicles in the next lane or across it will also be evaluated. Post-processing is added based on these problems so that the system will only evaluate vehicles detected in the specified area. For post-processing algorithm, we modified algorithm taken from [20]; the process is that first we calculate the vehicle centroid for each object and then check if the centroid is inside the polygon or the detection area.

2.2. **Wrong-way detector.** It is a subsystem to detect wrong-way violations. From Figure 1, this system consists of two inputs, data from VDT and violation direction parameters. Data from VDT consists of bounding boxes, id, and class. Then the violation direction parameter is the value that states the direction when the vehicle violates in the form of an angle. The value of the parameter is determined manually via the GUI. Movement direction predictor is a subsystem that predicts the direction of movement for each vehicle detected in the frame. In predicting, the algorithm used consists of several stages of the process. First, calculate the centroid of bounding box object VDT data using Formulae (1) and (2), where $x_{\min}$, $y_{\min}$, $x_{\max}$, and $y_{\max}$ are values from object bounding box.

$$x_{center} = x_{\min} + \frac{x_{\max} - x_{\min}}{2} \tag{1}$$

$$y_{center} = y_{\min} + \frac{y_{\max} - y_{\min}}{2} \tag{2}$$

Then, calculate the total number of changes in the centroid value of each occurrence in a particular frame range using Formulae (3) and (4), where n is the number of occurrences of the object in the frame, while i is the index.

$$\Delta x = \sum_{i=1}^{n-1} x_{center}^{i+1} - x_{center}^{i} \tag{3}$$

$$\Delta y = \sum_{i=1}^{n-1} y_{center}^{i+1} - y_{center}^{i} \quad (4)$$

After obtaining Δx and Δy, the next step is to convert the value of the change into an angle form. We used the concept of converting from Cartesian to polar, as shown in Formula (5). Lastly, the direction violation detection determines whether the vehicle commits a violation based on the estimated movement of the vehicle object and the violation direction parameter.

$$\theta = \arctan \frac{\Delta x}{\Delta y} \quad (5)$$

2.3. **Helmet violation detector.** Helmet violation detector or HVD is a subsystem to detect helmet violation which means two-wheeled (motorcycles and bicycles) vehicle driver or passenger that does not use helmets. As shown in Figure 3, this subsystem consists of three steps: object class and age filtering, image cropping, and image classifier. The input to the HVD data are preprocessing data (current frame) and VDT data consists of bounding boxes, object id, and class.

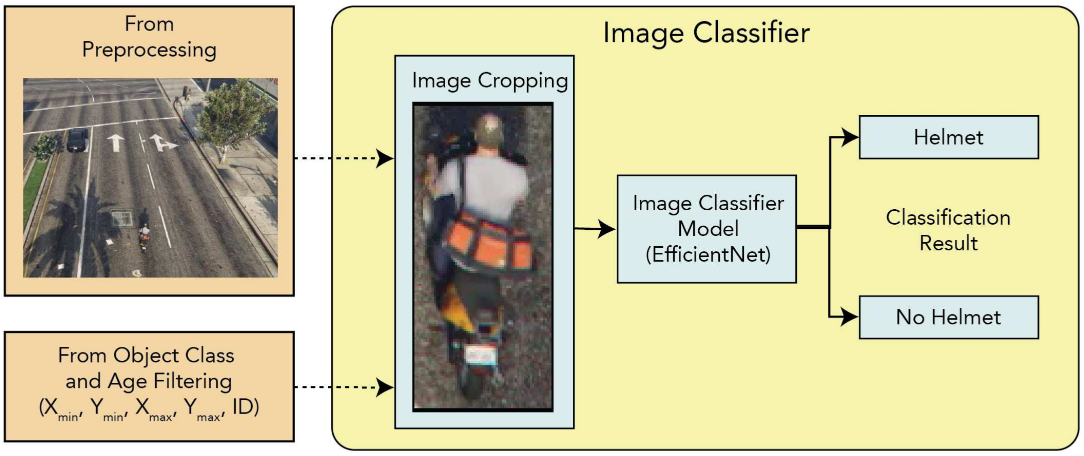

From Preprocessing
From Object Class and Age Filtering (X_{min}, Y_{min}, X_{max}, Y_{max}, ID)
Image Classifier
Image Cropping
Image Classifier Model (EfficientNet)
Helmet
Classification Result
No Helmet

FIGURE 3. Helmet violation detector diagram

In the object class and age filtering, the subsystem will remove object data besides motorcycles and bicycle class from VDT and calculate the age for each different object ID. The age represents the number of object occurrences in a set of frames. Then if the age of the object exceeded a minimum age threshold parameter, we cropped the vehicle frame based on the bounding box. Each cropped vehicle uses as input for the image classifier model to determine the violation. We use EfficientNet architecture [19] with the variant "efficientnet_b3" as a baseline model for the image classifier. The image classifier model is obtained through the finetuning process of the model that has been trained on ImageNet [21]. The dataset used for finetuning the model combines the modified Bikes Helmets Dataset [22] and 1KHDFW. We modified Bikes Helmets Dataset by changing the object detection dataset to the image classification dataset. For 1KHDFW, the dataset collected by the author contains two classes: no helmet, and helmet. The total number of images we used for finetuning the model is 1790 for training and 451 for validation.

2.4. **Running red light detector.** It is a subsystem to detect the running red light vehicle. As shown in Figure 1, this subsystem consists of two steps: traffic light state recognition and intersection stop line and object. In detecting the condition of traffic lights, the method used is color segmentation. The diagram for detecting traffic lights detection is shown in Figure 4. Based on the diagram above, the process that occurs is as follows, the data from the preprocessing (current frame) will be cropped based on the

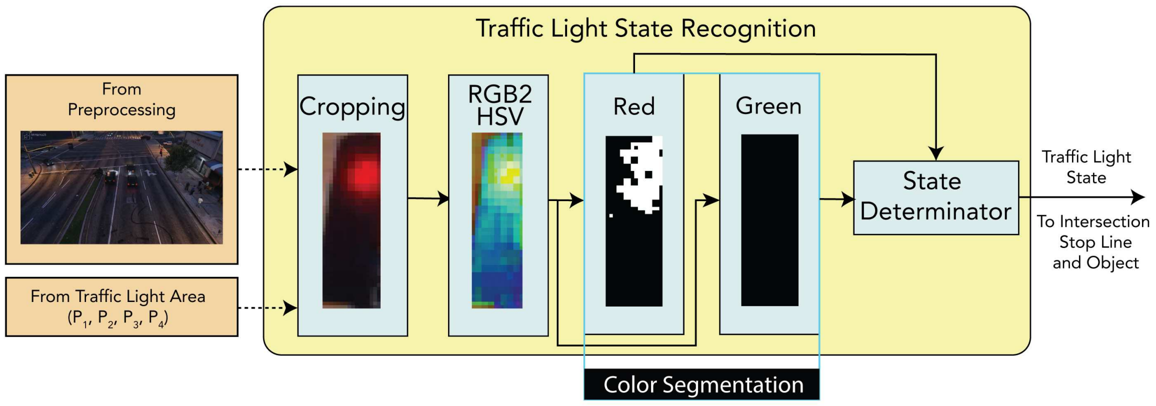

Traffic Light State Recognition
From Preprocessing
From Traffic Light Area (P1, P2, P3, P4)
Cropping
RGB2 HSV
Red
Green
State Determinator
Traffic Light State
To Intersection Stop Line and Object
Color Segmentation

FIGURE 4. Traffic light state recognition

parameters of the traffic light area to obtain a frame of the traffic light from the entire frame. The traffic light frame then converts from RGB colorspace to HSV colorspace. We converted it because HSV colorspace is better than RGB for color segmentation. The results of the HSV image are then thresholded to detect red and green colors using parameters $H_{\max}$, $S_{\min}$, $S_{\max}$, $V_{\min}$, and $V_{\max}$. To determine the state of the traffic light, we calculate the binary image area or the number of pixels with a value of 1 from the two-color segmentation result. For example, if the red color binary image area has the largest value, it will be compared with the $Area_{\min}$ value, if larger the state of the traffic light is red, and if smaller the state of a traffic light is yellow.

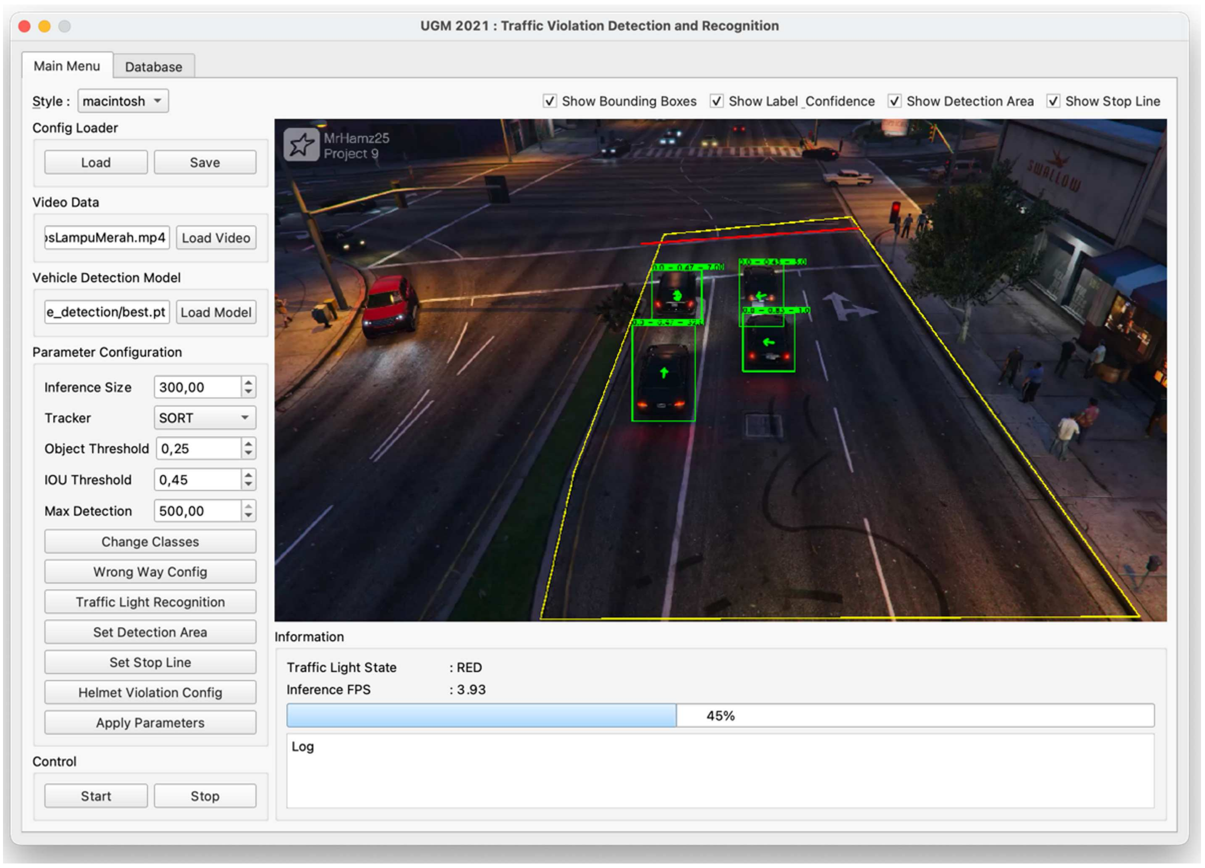
There are three inputs in the intersection of stop line and object: stop line parameter represented by the two points determined using GUI, traffic light state color obtained from traffic light state recognition, and VDT data consists of bounding boxes, object id, and class. Violation happens when the state of a traffic light is red and the bounding box of the vehicle intersects with the stop line. We implement the [23] algorithm for rectangle and line collision detection.

2.5. **Graphic user interface.** The Graphic User Interface (GUI) was designed using PySide2 in Python. The GUI facilitates setting configuration parameters such as traffic light area, stop line, wrong-way direction, or threshold value for the model inference process of vehicle detection. The appearance of the GUI is shown in Figure 5.

3. Results and Discussion.

3.1. **Vehicle detection model performance.** We train the vehicle detection model using the YOLOv5m variant type configuration, which applies compound scaling with a depth multiple of 0.67 and width multiples of 0.75. In the training process, we do not apply cross-validation when training process so that the validation set is used for evaluation. The model used in the system is selected based on the smallest loss value. The results of the model evaluation are shown in Table 1.

3.2. **Wrong-way detector performance.** In testing the wrong-way detector, we had difficulty finding videos of scenarios of this type of violation occurring in the real world. Therefore, we collected 25 videos obtained from the video game simulation process. The video is then grouped into two parts: high light and low light. Each video is in a different place, requiring configuration adjustments such as detection area and wrong-way parameters. Figure 6 shows that the wrong-way violation is indicated with red arrow. Based on Table 2, the designed method can detect violations well, indicated by the F1-Score value, which reaches a perfect score.



UGM 2021 : Traffic Violation Detection and Recognition

Main Menu | Database

Style : macintosh

☑ Show Bounding Boxes ☑ Show Label_Confidence ☑ Show Detection Area ☑ Show Stop Line

Config Loader: Load | Save

Video Data: ›sLampuMerah.mp4 | Load Video

Vehicle Detection Model: e_detection/best.pt | Load Model

Parameter Configuration

Inference Size	300,00
Tracker	SORT
Object Threshold	0,25
IOU Threshold	0,45
Max Detection	500,00

Change Classes | Wrong Way Config | Traffic Light Recognition | Set Detection Area | Set Stop Line | Helmet Violation Config | Apply Parameters

Control: Start | Stop

Information

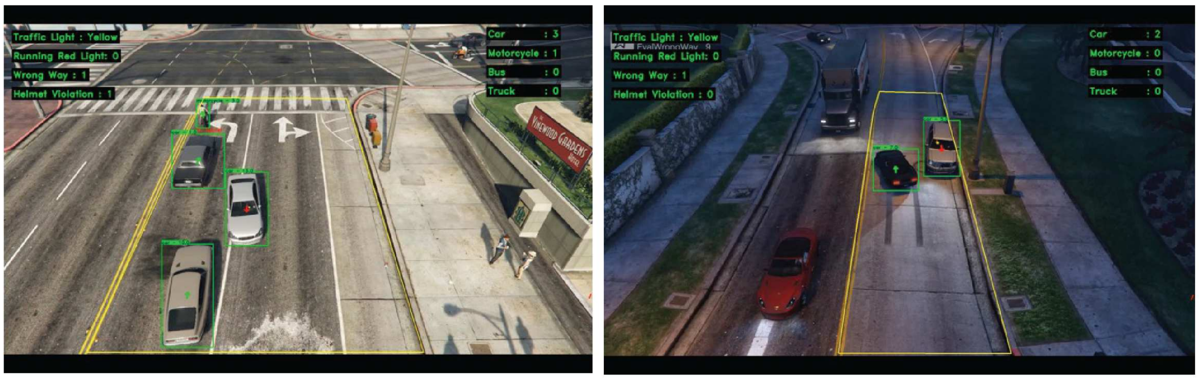
Traffic Light State : RED
Inference FPS : 3.93
45%

Log

FIGURE 5. Graphic user interface traffic violation detection

TABLE 1. Vehicle detection model validation result

Class	#Labels	P	R	F1	AP@0.5
Car	7941	0.79	0.75	0.77	0.82
Motorcycle	1081	0.78	0.67	0.72	0.72
Bus	667	0.83	0.78	0.80	0.85
Truck	1126	0.72	0.54	0.62	0.66
Bicycle	956	0.69	0.58	0.62	0.63
Average		0.76	0.66	0.71	0.74



Traffic Light : Yellow
Running Red Light: 0
Wrong Way : 1
Helmet Violation : 1

Car : 3
Motorcycle : 1
Bus : 0
Truck : 0

Traffic Light : Yellow
Running Red Light: 0
Wrong Way : 1
Helmet Violation : 0

Car : 2
Motorcycle : 0
Bus : 0
Truck : 0

FIGURE 6. Wrong-way detector inference result

Table 2. Wrong-way evaluation result

Location-based light	#Video	Duration	P	R	F1
Low light	15	193 s	1.00	1.00	1.00
High light	11	159 s	1.00	1.00	1.00
Average			1.00	1.00	1.00

3.3. **Helmet violation detector performance.** In testing the helmet violation detector, we used four videos from CCTV video clips around Gunungkidul, Yogyakarta, obtained from Live streaming from the Kominfo Gunungkidul channel. The location was chosen because more two-wheeled drivers were passing through the road. Figure 7 shows that the system can detect traffic violations for two-wheeled vehicles that do not use a helmet marked with a red "no helmet" under the vehicle's bounding box. The integrated system is designed to detect all three violations at once, which is indicated by a bounding box having an arrow. Based on the F1-Score value in Table 3, it can be stated that the system has a pretty good performance in detecting this violation.

3.4. **Running red light detector performance.** In testing the running red light detector, we used 81 video footage obtained from YouTube which contains a compilation of

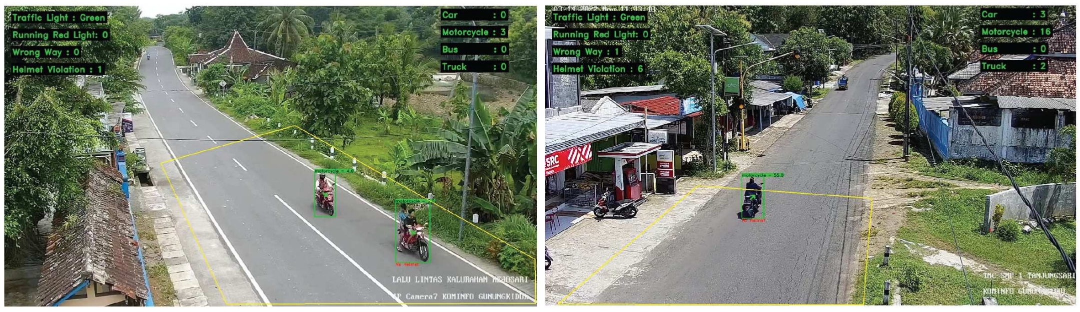


Figure 7. Helmet violation detector inference result

Table 3. Helmet violation detector evaluation result

Location	Duration	P	R	F1
Kelurahan Sambirejo	198 s	1.00	0.83	0.91
Perempatan Trowono	394 s	0.89	1.00	0.94
Rejosari	245 s	1.00	0.88	0.94
Tanjungsari	556 s	0.95	0.55	0.75
Average		0.96	0.81	0.88

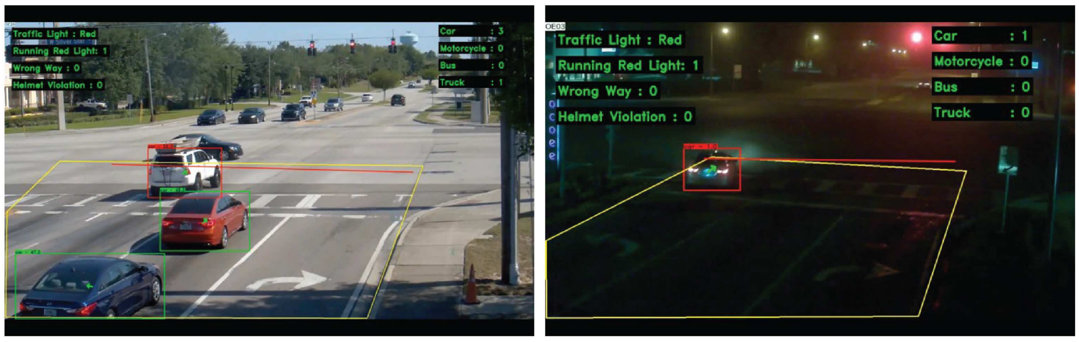


Figure 8. Running red light detector inference result

vehicles passing through red lights at various times ranging from day to night as shown in Figure 8. The video footage is then classified by location. In the test, each location has a different configuration according to the position of the traffic light, stop line and detection area. In Figure 8, shows that the violating vehicle is marked with a red bounding box line. Based on Table 4, the system for detecting this violation produces an F1-Score of 0.95.

Table 4. Running red light detector evaluation result

Location ID	#Video	Duration	P	R	F1
LOC_1 & LOC_2	15	82 s	0.94	0.94	0.94
LOC_3 & LOC_4	41	211 s	1.00	0.78	0.88
LOC_5 & LOC_6	7	42 s	1.00	1.00	1.00
LOC_7 & LOC_8	10	57 s	1.00	1.00	1.00
LOC_9	7	36 s	1.00	0.88	0.93
Average			0.98	0.92	0.95

Table 5. Comparision with the previous studies

Authors	Violation type	Method	Result
Rahman et al. [13]	Wrong-way	Classifying a motorcycle's front and rear view frame using Faster R-CNN	• The average accuracy value is 88%. • Violation detection only for motorcycles.
Tonge et al. [7]	Helmet violation	Classifying the upper part of the frame from the bounding box using a custom CNN model	• The precision, recall, and F1-Score values are 0.88, 0.89, and 0.88. • Not mentioned whether it can detect when the vehicle is facing behind and one of the drivers or the passengers is not using a helmet.
Spanhel et al. [14]	Running red light	Violation happens when the centroid of the bounding box inside the protected area and the traffic light state is red.	• The precision, recall, and F1-Score values are not mentioned.
Our	Wrong-way	Predict vehicle movement based on object tracker data.	• The precision, recall, and F1-Score values are 1.00, 1.00, and 1.00. • Violation detection for cars, buses, trucks, motorcycles, and bicycles.
	Helmet violation	Classifying the whole frame of the bounding box using finetuned EfficientNet model	• The precision, recall, and F1-Score values are 0.96, 0.81, and 0.88. • Can detect violations when the vehicle is facing behind the camera, and one of the drivers or passengers is not using a helmet.
	Running red light	Violation happens when the bounding box intersects with the stop line area, and the traffic light state is red.	• The precision, recall, and F1-Score values are 0.98, 0.92, and 0.95.

4. **Conclusions.** Table 5 shows that our proposed system can detect three types of violation, wrong-way, running a red light, and helmet violation in an integrated approach in which these methods run simultaneously for each frame of video. The result for each violation is acceptable and good enough with the F1-Score above 0.88. Besides that, our approach has several advantages compared to the previous method. For example, our wrong-way violation method can detect multiple vehicle types rather than just one and can detect helmet violation when the vehicle is facing behind. In the future study, we need to test the system on embedded hardware to find the possibility implemented on the output of CCTV directly on the spot.

Acknowledgment. This research was supported by Electronics and Instrumentation Research Laboratory, Universitas Gadjah Mada. We thank our colleagues from Universitas Gadjah Mada who provided insight and expertise that greatly assisted the research.

REFERENCES

[1] World Health Organization, Global status report on road safety 2018, *International Journal of Machine Tools and Manufacture*, vol.5, no.1, pp.86-96, https://www.who.int/publications/i/item/9789241565684, 2018.

[2] BPS RI, Land transportation statistics 2019, *Bps. Ri.*, vol.1, no.1, pp.1-8, https://www.bps.go.id/publication/2020/11/20/ddce434c92536777bf07605d/statistik-transportasi-darat-2019.html, 2020.

[3] A. D. Saputra, Study of road traffic accident rates in Indonesia based on NTSC data (National Transportation Safety Committee) from 2007-2016, *News of Transportation Research*, vol.29, https://ojs.balitbanghub.dephub.go.id/index.php/warlit/article/download/557/319, 2017.

[4] J. Redmon and A. Farhadi, YOLO9000: Better, faster, stronger, *Proc. of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR2017)*, vol.2017-Janua, pp.6517-6525, DOI: 10.1109/CVPR.2017.690, 2017.

[5] J. Redmon and A. Farhadi, YOLOv3: An incremental improvement, *arXiv.org*, arXiv: 1804.02767, 2018.

[6] A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, YOLOv4: Optimal speed and accuracy of object detection, *arXiv.org*, arXiv: 2004.10934, 2020.

[7] A. Tonge, S. Chandak, R. Khiste, U. Khan and L. A. Bewoor, Traffic rules violation detection using deep learning, *Proc. of the 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA2020)*, pp.1250-1257, DOI: 10.1109/ICECA49313.2020.9297495, 2020.

[8] Z. Liu, W. Chen and C. K. Yeo, Automatic detection of parking violation and capture of license plate, *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON2019)*, pp.495-500, DOI: 10.1109/IEMCON.2019.8936164, 2019.

[9] R. Girshick, J. Donahue, T. Darrell and J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp.580-587, DOI: 10.1109/CVPR.2014.81, 2014.

[10] R. Girshick, Fast R-CNN, *2015 IEEE International Conference on Computer Vision (ICCV)*, pp.1440-1448, DOI: 10.1109/ICCV.2015.169, 2015.

[11] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.39, no.6, pp.1137-1149, DOI: 10.1109/TPAMI.2016.2577031, 2017.

[12] S. Ibadov, R. Ibadov, B. Kalmukov and V. Krutov, Algorithm for detecting violations of traffic rules based on computer vision approaches, *MATEC Web of Conferences*, vol.132, DOI: 10.1051/matecconf/201713205005, 2017.

[13] I. W. Rahman, Z. Zainuddin, S. Syarif and A. Achmad, Recognition of the rearview image pattern of a motorcycle for a system to detect current violations on a one-way street, *2020 27th International Conference on Telecommunications (ICT)*, pp.1-5, DOI: 10.1109/ict49546.2020.9239442, 2020.

[14] J. Spanhel, J. Sochor and A. Makarov, Detection of traffic violations of road users based on convolutional neural networks, *2018 14th Symposium on Neural Networks and Applications (NEUREL2018)*, DOI: 10.1109/NEUREL.2018.8586996, 2018.

[15] G. Jocher et al., *ultralytics/yolov5: v6.1 – TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference*, Zenodo, DOI: 10.5281/ZENODO.6222936, 2022.

[16] A. Bewley, Z. Ge, L. Ott, F. Ramos and B. Upcroft, Simple online and realtime tracking, *2016 IEEE International Conference on Image Processing (ICIP)*, pp.3464-3468, DOI: 10.1109/ICIP.2016.7533003, 2016.

[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar and C. L. Zitnick, Microsoft COCO: Common objects in context, *European Conference on Computer Vision*, pp.740-755, 2014.

[18] A. Geiger, P. Lenz and R. Urtasun, Are we ready for autonomous driving? The KITTI vision benchmark suite, *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp.3354-3361, DOI: 10.1109/CVPR.2012.6248074, 2012.

[19] M. Tan and Q. Le, EfficientNet: Rethinking model scaling for convolutional neural networks, *Proc. of the 36th International Conference on Machine Learning*, vol.97, pp.6105-6114, 2019.

[20] W. R. Franklin, *PNPOLY – Point Inclusion in Polygon Test W. Randolph Franklin (WRF)*, https://wrf.ecse.rpi.edu/Research/Short_Notes/pnpoly.html, Accessed on Sep. 22, 2021.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and F.-F. Li, ImageNet: A large-scale hierarchical image database, *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp.248-255, DOI: 10.1109/CVPR.2009.5206848, 2009.

[22] Bikes Helmets Dataset, *MakeML*, https://makeml.app/datasets/helmets, Accessed on Sep. 22, 2021.

[23] J. Thompson, *Collision Detection*, http://www.jeffreythompson.org/collision-detection/line-rect.php, Accessed on Aug. 23, 2021.