JSCE (Journal of System and Computer Engineering)
Vol.7, No.4, October 2026, pp. 398-407
ISSN (online): 2723-1240
DOI : https://doi.org/10.61628/jsce.v7i4.2848

Research Article Open Access (CC–BY-SA) ■

Vehicle Counting System on Ro-Ro Ferry Bengkalis Route Using YOLO26n and ByteTrack Algorithm

Muhamad Nasir [1,a,*]; Tengku Musri [2,b]; Eva Kurniawaty [3,c]; Thomas Brian [4,d]; Muhammad Rizki Kurniawan [5,e]

[1,2,3,5] *Politeknik Negeri Bengkalis, Jl. Bathin Alam Sungai Alam, Bengkalis 28711, Indonesia*
[4] *Politeknik Perkapalan Negeri Surabaya, Jl. Teknik Kimia Kampus ITS Keputih Sukolilo Surabaya 60111, Indonesia*
[a] *nasir@polbeng.ac.id;* [b] *musri@polbeng.ac.id;* [c] *evakurni125@gmail.com;* [d] *thomasbrian@ppns.ac.id;*
[e] *ogensasek2@gmail.com*
** Corresponding author*

Abstract

Vehicle transportation management on Ro-Ro ferries requires accurate information regarding the number and type of vehicles transported during the loading process. Conventional counting methods are still performed manually, which requires time and has the potential to produce human errors. This research proposes an automatic vehicle counting system on the Bengkalis Ro-Ro ferry route using YOLO26n object detection combined with ByteTrack multi-object tracking. The system processes video recordings from Ro-Ro ferry activities and detects three vehicle categories, namely cars, motorcycles, and trucks. The detection process uses YOLO26n with a confidence threshold of 0.25 and an image size of 640 pixels. ByteTrack is implemented to maintain vehicle identities and prevent duplicate counting when vehicles cross the counting line. The experimental results show that the proposed system successfully detected and counted 20 vehicles consisting of 5 cars, 10 motorcycles, and 5 trucks. The system achieved an average processing speed of 15 frames per second, indicating that the proposed approach can support real-time vehicle monitoring on Ro-Ro ferry transportation systems.

Keywords*—YOLO26n, ByteTrack, Vehicle Counting, Computer Vision, Ro-Ro*

1. Introduction

Ro-Ro (Roll-on/Roll-off) ferry transportation plays an important role in connecting regions separated by water and supporting the mobility of people, goods, and vehicles. In ferry transportation, the number and type of vehicles entering the ferry deck are important operational information because they are directly related to deck capacity utilization, loading efficiency, departure scheduling, and transportation safety. Accurate vehicle recording is therefore essential for supporting effective ferry management. However, vehicle counting at ferry terminals is still commonly performed manually by operators. Manual counting requires continuous human involvement and may become less reliable when vehicle volume increases, different vehicle types enter simultaneously, and vehicles move closely together during the loading process. These conditions can lead to missed counts, duplicate records, or incorrect vehicle classification. An automatic vehicle counting system based on computer vision can therefore provide an alternative approach to improve the efficiency and consistency of vehicle monitoring during Ro-Ro loading operations.

The development of deep learning-based object detection has significantly improved the capability of computer vision systems for automatic vehicle monitoring. The You Only Look Once (YOLO) family has been widely used because of its ability to perform object localization and classification in a single inference process, making it suitable for real-time applications. Redmon et al. (2016) introduced YOLO as a real-time object detection framework, while

subsequent versions have continuously improved detection accuracy and computational efficiency. In addition to object detection, vehicle counting requires an object tracking mechanism because the same vehicle may appear in multiple consecutive video frames. Without tracking, a detection-based system may count the same vehicle repeatedly. ByteTrack, proposed by Zhang et al. (2022), addresses this problem by associating detected objects across consecutive frames and maintaining their identities, including detections with relatively low confidence scores. The combination of YOLO-based object detection and ByteTrack has consequently become an effective approach for vehicle detection, tracking, and counting.

Several recent studies have investigated the combination of YOLO-based detectors and ByteTrack for vehicle monitoring and counting. Zhang (2023) applied YOLOv7 and ByteTrack for real-time vehicle detection and tracking, demonstrating the capability of the combination to maintain vehicle identities in traffic scenes. Ng et al. (2023) developed a vehicle detection, classification, tracking, and counting system using YOLOv5 and ByteTrack for traffic impact assessment. The study demonstrated the potential of combining object detection and multi-object tracking for automated traffic monitoring. Furthermore, Fahrezi and Laksana (2024) employed YOLOv8 and ByteTrack for vehicle detection, tracking, traffic flow analysis, and congestion detection using CCTV imagery. Similar implementations on surveillance cameras were conducted by Cahyono et al. (2025), who utilized YOLOv8 and ByteTrack to achieve stable real-time vehicle counting. Additionally, performance evaluations of various YOLO architectures on complex traffic datasets, such as those by Alamgir et al. (2022), have demonstrated the adaptability of YOLO-based models in dense traffic environments. These studies indicate that the YOLO–ByteTrack framework is capable of supporting real-time vehicle monitoring and automated counting.

Table 1. Comparison of Related Studies on Vehicle Detection and Tracking

Study	Detector	Environment	Vehicle Classes	Main Focus	Limitation
Zhang (2023)	YOLOv7	Road traffic	Vehicles	Detection and tracking	No Ro-Ro environment
Ng et al. (2023)	YOLOv5	Road junction	Multiple classes	Detection, tracking, counting	Road-based environment
Fahrezi & Laksana (2024)	YOLOv8	Urban CCTV	Multiple classes	Traffic flow and congestion	No ferry loading condition
Cahyono et al. (2025)	YOLOv8	Surveillance cameras	Multiple classes	Detection and counting	General surveillance view, no Ro-Ro specifics
Alamgir et al. (2022)	YOLO-based	Dense traffic images	Multiple classes	Performance analysis	Static images, no tracking/ferry conditions
This study	YOLO26n	Ro-Ro ferry Bengkalis	Car, motorcycle, truck	Vehicle counting during loading	Application-specific Ro-Ro environment

Despite these developments, most existing YOLO–ByteTrack vehicle counting studies have been conducted in conventional road traffic environments, including urban roads, intersections, highways, and CCTV-based traffic monitoring locations. The characteristics of these environments differ substantially from those of Ro-Ro ferry loading areas. In a road environment, vehicles generally move along relatively predictable trajectories with sufficient spacing between objects. In contrast, vehicles entering a ferry may move within a confined loading deck where maneuvering space is limited and vehicles are positioned very close to one another. The camera may also be installed at an elevated or oblique angle, producing substantial perspective variation and differences in vehicle scale. Furthermore, the simultaneous presence of motorcycles, cars,

and trucks creates a mixed-class detection problem, while vehicle queues and close interactions may produce partial or complete occlusion. These conditions can negatively affect both object detection and tracking performance and increase the possibility of missed detections, identity switches, and inaccurate vehicle counts.

The differences between conventional road traffic and Ro-Ro loading environments indicate an important research gap. First, previous YOLO–ByteTrack studies have primarily evaluated vehicle counting under road-based traffic conditions, while limited attention has been given to confined ferry loading environments. Second, the effects of perspective variation and high vehicle density on vehicle tracking and counting remain insufficiently explored in Ro-Ro operations. Third, mixed vehicle classes and frequent partial occlusion during ferry loading may create additional challenges for maintaining object identities and preventing duplicate or missed counts. Therefore, the performance of a YOLO–ByteTrack framework developed for conventional traffic cannot necessarily be assumed to be directly applicable to Ro-Ro ferry operations. A specialized application-oriented evaluation using actual Ro-Ro loading video is needed to determine whether the detection and tracking approach can reliably count vehicles under these operational conditions.

To address this research gap, this study develops an automatic vehicle detection, tracking, and counting system for the Ro-Ro ferry loading area on the Bengkalis route using YOLO26n and ByteTrack. YOLO26n is employed to detect and classify vehicles, while ByteTrack is used to associate vehicle detections across consecutive frames and maintain consistent object identities. The proposed system focuses on three vehicle categories, namely cars, motorcycles, and trucks, which represent the main vehicle types observed during the ferry loading process. Unlike previous studies that primarily evaluate vehicle counting in conventional road traffic environments, this research uses video data from an actual Ro-Ro ferry loading environment in Bengkalis. The system is therefore evaluated under practical conditions involving different camera perspectives, varying vehicle density, mixed vehicle classes, close vehicle interactions, and partial occlusion.

The main contributions of this study are as follows:

1. A YOLO26n–ByteTrack-based vehicle counting framework is developed for Ro-Ro ferry loading operations, integrating vehicle detection, classification, multi-object tracking, and automatic counting within a single monitoring system.
2. An application-specific evaluation is conducted using Ro-Ro ferry loading video from the Bengkalis route, addressing operational characteristics that are rarely considered in conventional road-based vehicle counting studies, including constrained loading space, perspective variation, high vehicle density, mixed vehicle classes, and vehicle occlusion.
3. The detection, tracking, and counting performance of the proposed system is evaluated for cars, motorcycles, and trucks, providing empirical evidence regarding the applicability of YOLO26n and ByteTrack for automated vehicle recording in Ro-Ro ferry transportation.

Through these contributions, this study extends existing YOLO-based vehicle counting research from conventional road traffic environments to the more complex operational setting of Ro-Ro ferry loading. The proposed system is expected to reduce dependence on manual vehicle recording, improve the consistency of vehicle counting, and provide useful information to support more efficient vehicle loading and operational monitoring of Ro-Ro ferry transportation.

2. Methods

This research applies a computer vision-based approach to automatically detect, track, and count vehicles transported by Ro-Ro ferry on the Bengkalis route. The proposed system consists of four main stages, namely video acquisition, vehicle detection using YOLO26n, vehicle tracking using ByteTrack, and vehicle counting based on object movement across a predefined counting line.

The research workflow begins with collecting video data from Ro-Ro ferry activities. The video contains vehicle movement during the loading process on the ferry deck. The recorded video is processed frame by frame using OpenCV. Each frame is analyzed by the YOLO26n

model to identify vehicle objects. The detected objects are then tracked using ByteTrack to maintain vehicle identity across consecutive frames.

The general workflow of the proposed system is illustrated as follows:

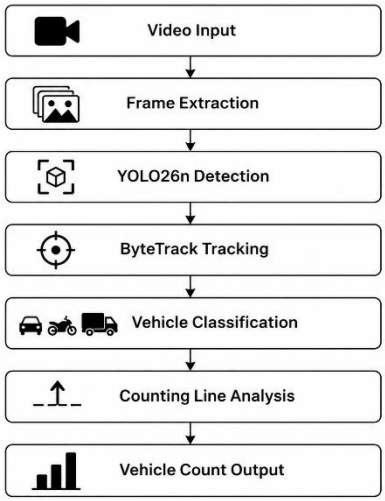


Figure 1. Proposed vehicle counting system workflow

2.1 Dataset and Experimental Setup

The dataset used in this research is a video recording of vehicle loading activities on the Ro-Ro ferry route Bengkalis. The video file used in the experiment is processed as the input source for vehicle detection and counting.

The system configuration used in the experiment is shown in Table 2.

Table 2. Experimental Configuration.

Parameter	Value
Object detection model	YOLO26n
Tracking algorithm	ByteTrack
Video source	Ro-Ro ferry Bengkalis route
Input image size	640 pixels
Confidence threshold	0.25
Processing device	CPU
Average FPS	15 frames per second
Vehicle classes	Car, Motorcycle, Truck

The YOLO26n model is selected because it provides a lightweight architecture suitable for real-time object detection applications. The model can perform tracking operations by combining detection results with multi-object tracking algorithms such as ByteTrack.

2.2 Vehicle Detection Using YOLO26n

YOLO26n is implemented as the primary object detection model in this research. The model receives video frames as input and generates bounding boxes, class labels, and confidence scores for detected vehicles. The detection process uses a confidence threshold of 0.25. Objects with confidence scores below this threshold are excluded from the detection results.

The proposed system uses vehicle classes from the COCO dataset. Since this research focuses on vehicle counting in Ro-Ro ferry transportation, the detected classes are simplified into three categories: motorcycle, car, and truck. The bus and truck classes are combined into a single truck category to simplify the classification process. The vehicle class mapping is presented in Table 3.

Table 3. Vehicle Classification Mapping

COCO Class	Research Class	Description
Car	Car	Passenger cars detected by YOLO26n
Motorcycle	Motorcycle	Two-wheeled vehicles detected by YOLO26n
Bus	Truck	Combined into truck category
Truck	Truck	Heavy vehicles detected by YOLO26n

The classification mapping allows the system to adapt the general COCO vehicle categories to the specific requirements of Ro-Ro ferry vehicle monitoring. The vehicle class configuration follows the implementation where COCO class IDs 2, 3, 5, and 7 are selected as vehicle objects, representing car, motorcycle, bus, and truck categories. In this research, the bus and truck categories are combined into a single truck class to simplify vehicle classification. This approach reduces class complexity and enables the system to provide vehicle counts according to operational categories used in Ro-Ro ferry transportation.

2.3 Multi-Object Tracking Using ByteTrack

After the detection process, each detected vehicle is tracked using ByteTrack. The purpose of tracking is to maintain the identity of each vehicle while moving between video frames.

The tracking process generates a unique identification number for each detected vehicle. This ID prevents duplicate counting when the same vehicle appears in multiple frames. The implementation uses persistent tracking through ByteTrack configuration to maintain object trajectories.

The tracking method is implemented using the YOLO tracking function with ByteTrack configuration:

- Persistent object ID
- Vehicle position tracking
- Frame-to-frame object association

The use of ByteTrack allows detected objects to maintain consistent identities during video processing. This approach is commonly applied in vehicle counting systems because stable object IDs allow counting based on unique tracked objects rather than individual frame detections.

2.4 Vehicle Counting Mechanism

The counting mechanism is based on vehicle movement through a predefined horizontal counting line. The system calculates the center point of each detected vehicle bounding box.

The center point is calculated using Equation (1):

$$\begin{cases} C_x = \dfrac{x_1 + x_2}{2} \\ C_y = \dfrac{y_1 + y_2}{2} \end{cases} \quad (1)$$

where:
(C_x) represents the horizontal center coordinate.
(C_y) represents the vertical center coordinate.
(x_1,y_1) represent the upper-left bounding box coordinates.
(x_2,y_2) represent the lower-right bounding box coordinates.

A vehicle is counted when the center point moves from above the counting line to below the counting line. The system stores previously counted vehicle IDs to ensure each vehicle is counted only once.

The counting condition is implemented as:
1. Detect vehicle position in the current frame.
2. Compare current position with previous frame position.
3. Check whether the vehicle crosses the counting line.

4. Add vehicle ID into the counting database.
5. Update total vehicle count based on vehicle category.

The program implementation stores previous vehicle center positions and counted IDs before increasing the total vehicle counter. The counting process increases the total value when a tracked vehicle moves across the counting line and its ID has not been counted previously.

2.5 System Output

The final output of the system consists of:
1. Total number of detected vehicles.
2. Number of vehicles based on category.
3. Real-time visualization containing:
   - Bounding box.
   - Vehicle ID.
   - Vehicle category.
   - Confidence score.
   - Counting line.
   - FPS information.

The system displays three main vehicle categories: Car, Motorcycle, and Truck. The final counting result is stored and displayed after the video processing is completed.

3. Results and Discussion

This section presents the experimental results of the proposed vehicle counting system using YOLO26n and ByteTrack on Ro-Ro ferry transportation activities in the Bengkalis route. The evaluation focuses on vehicle detection results, classification performance, counting accuracy, and processing speed.

3.1 Vehicle Detection and Counting Results

The proposed system successfully processed the Ro-Ro ferry video and detected vehicles entering the ferry area. The system classified detected objects into three categories: cars, motorcycles, and trucks. Each vehicle was counted based on its unique tracking ID when crossing the predefined counting line.
The detection and counting results are presented in Table 4.

Table 4. Vehicle Counting Results on Ro-Ro Ferry Bengkalis Route

No	Vehicle Category	Number of Vehicles	Percentage
1.	Car	5	25%
2.	Motorcycle	10	50%
3.	Truck	5	25%
	Total	20	100%

Based on Table 4, the system detected 20 vehicles in total. Motorcycles represent the highest number of detected vehicles with 10 units or 50% of the total vehicles. Cars and trucks each contributed 5 units or 25%. The results indicate that the proposed system can differentiate vehicle categories based on the YOLO26n detection output. The classification process follows the vehicle class mapping where car, motorcycle, bus, and truck classes are identified from the COCO dataset. The bus and truck classes are combined into the truck category to simplify the analysis.

3.2 Visualization and Tracking Performance

During the testing process, each detected vehicle received a unique tracking identity generated by ByteTrack. The tracking ID allows the system to monitor vehicle movement between consecutive frames and prevents multiple counting of the same vehicle.
The system successfully applied the counting mechanism by detecting vehicle movement across the counting line. A vehicle was counted only when its center point changed from the upper region of the frame to the lower region after crossing the counting line.

This approach reduces counting errors caused by repeated object detection in multiple frames. The combination of object detection and tracking provides more reliable counting compared with image-based detection without object tracking.

3.3 Processing Speed Analysis

The real-time capability of the system was evaluated based on frame processing speed. The experiment achieved an average processing speed of 15 frames per second.

Table 5. System Performance Evaluation

Parameter	Result
Detection model	YOLO26n
Tracking method	ByteTrack
Average FPS	15 FPS
Total vehicles counted	20 vehicles
Vehicle categories	3 categories

An average speed of 15 FPS indicates that the system can process video streams efficiently. Although the processing speed depends on hardware specifications and video resolution, the obtained FPS demonstrates that YOLO26n provides lightweight computation suitable for transportation monitoring applications.

The implementation uses an input image size of 640 pixels and a confidence threshold of 0.25. These parameters provide a balance between detection accuracy and computational efficiency. Lower confidence thresholds may increase detected objects but can introduce false detections, while higher thresholds may reduce detection sensitivity.

3.4 Discussion

The experimental results demonstrate that YOLO26n combined with ByteTrack can be implemented as an automatic vehicle counting system for Ro-Ro ferry transportation. The system successfully identifies different vehicle categories and counts vehicles based on object trajectories. Manual vehicle recording methods require direct observation from operators and may experience inconsistencies during high traffic conditions. The proposed computer vision approach provides automated counting capability and reduces dependency on manual observation.

The implementation of ByteTrack improves counting reliability because the system uses vehicle identity information instead of only relying on frame-based detection. Each vehicle receives a unique ID, allowing the system to recognize whether a vehicle has already been counted.

However, this research still has limitations. The experiment uses a single video source with a limited number of vehicles. Environmental conditions such as lighting variation, vehicle occlusion, camera angle, and vehicle density may affect detection performance. Future research can evaluate the system using larger datasets and different Ro-Ro operational conditions. Overall, the results show that the proposed YOLO26n and ByteTrack-based approach has potential for supporting automatic vehicle monitoring systems in ferry transportation management.

3.5 Experimental Testing Results

Additional experiments were conducted to evaluate the accuracy performance of the proposed vehicle counting system. The evaluation was performed by comparing the manual vehicle count as ground truth with the counting results generated by the YOLO26n and ByteTrack-based system. Three vehicle categories were evaluated, namely motorcycle, truck, and car. Accuracy was calculated using the ratio between correctly detected vehicles and manual counting results.

Table 6. Accuracy Evaluation of Vehicle Counting System in Test Scenario 1

Vehicle Category	Motorcycle	Truck	Car
Manual Calculation	2	5	6
System Calculation	2	8	6
Accuracy	100%	62.50%	100%
Average Accuracy	87.50%		

Table 7. Accuracy Evaluation of Vehicle Counting System in Test Scenario 2

Vehicle Category	Motorcycle	Truck	Car
Manual Calculation	2	1	10
System Calculation	2	1	14
Accuracy	100%	100%	71%
Average Accuracy	90.50%		

Analysis of Experimental Results

The first testing scenario achieved an average accuracy of 87.50%. The motorcycle and car categories reached 100% accuracy, while the truck category obtained 62.50% accuracy because the system detected additional objects compared with manual counting results. In the second testing scenario, the average accuracy increased to 90.50%. Motorcycle and truck detection achieved 100% accuracy, while car detection reached 71% accuracy due to overcounting during the detection process.

The experimental results indicate that the integration of YOLO26n and ByteTrack can provide reliable automatic vehicle counting performance. The tracking mechanism helps maintain vehicle identity and reduces duplicate counting. However, several factors such as vehicle similarity, object overlap, camera perspective, and tracking conditions can still influence detection accuracy.

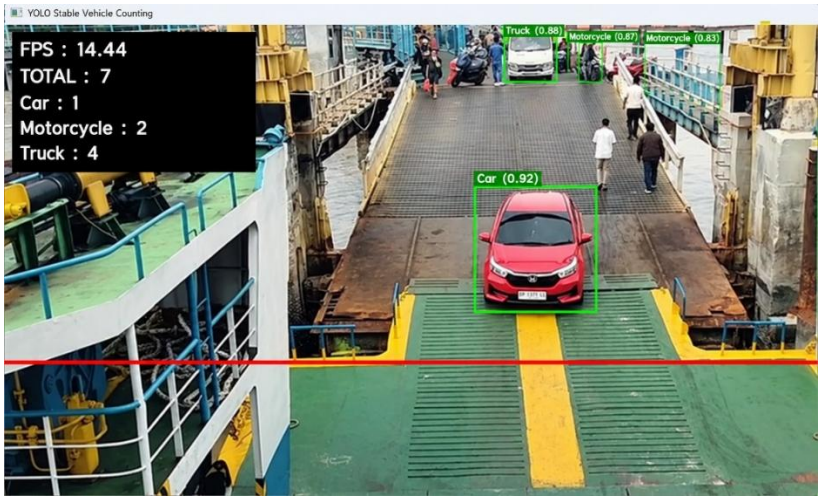


Figure 2. Vehicle Counting Application Interface

The application visualization presents the real-time vehicle detection and counting process. The displayed information includes video input, vehicle bounding boxes, tracking IDs, vehicle classification labels, counting line position, confidence scores, and final counting results.

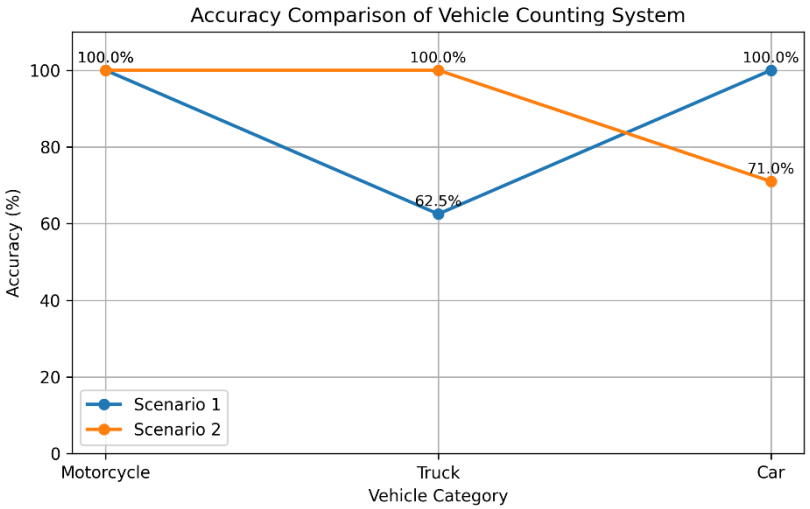


Figure 3. Accuracy Comparison of Vehicle Counting System

4. Conclusions

This research developed an automatic vehicle counting system for Ro-Ro ferry transportation on the Bengkalis route using YOLO26n object detection and ByteTrack tracking algorithm. The proposed system successfully detected, classified, and counted vehicles from video recordings of Ro-Ro ferry activities.

The experimental results show that the system detected 20 vehicles consisting of 5 cars, 10 motorcycles, and 5 trucks. The system achieved an average processing speed of 15 frames per second, demonstrating that the proposed approach can support real-time vehicle monitoring applications.

The combination of YOLO26n and ByteTrack provides advantages in maintaining vehicle identity and preventing duplicate counting. The tracking mechanism based on vehicle IDs improves counting reliability because each vehicle is counted only once when crossing the counting line.

However, this research still has limitations related to the number of testing videos and environmental conditions. Future research should evaluate the system using larger datasets, various camera positions, different lighting conditions, and higher vehicle densities to improve system robustness.

Acknowledgements

The research was financially supported by the Center for Research and Community Service (Pusat Penelitian dan Pengabdian kepada Masyarakat, P3M), Politeknik Negeri Bengkalis. The author gratefully acknowledges P3M Politeknik Negeri Bengkalis for funding this research and supporting the development of the vehicle counting system for Ro-Ro ferry transportation.

References

Alamgir, R. M., Shuvro, A. A., Mushabbir, M. A., et al. (2022). Performance Analysis of YOLO-based Architectures for Vehicle Detection from Traffic Images in Bangladesh. arXiv preprint arXiv:2212.09144.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple Online and Realtime Tracking. Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 3464–3468.

Bochkovskiy, A., Wang, C. Y., and Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

Cahyono, F. Y. A., Setiawan, R. A., and Razak, A. A. (2025). Real-Time Vehicle Detection and Counting Using YOLOv8 and ByteTrack Multi-Object Tracking on Surveillance Cameras. Journal Zetroem.

Diwan, T., Anirudh, G., and Tembhurne, J. V. (2023). Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 82, 9243–9275.

Fahrezi, M. M., and Laksana, E. A. (2024). Traffic flow and congestion detection with YOLOv8 and ByteTrack-based multi object tracking. Jurnal Teknik Informatika (JUTIF), 5(4), 253–261.

Gomaa, A., Minematsu, T., Abdelwahab, M. M., Abo-Zahhad, M., and Taniguchi, R. (2022). Faster CNN-based vehicle detection and counting strategy for fixed camera scenes. Multimedia Tools and Applications, 81, 25443–25471.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning. Cambridge, MA: MIT Press.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2961–2969.

Jocher, G., Chaurasia, A., and Qiu, J. (2023). Ultralytics YOLO by Ultralytics. GitHub Repository.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., and Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision (ECCV), pp. 21–37.

Ng, J. J., Michael Goh, K. O., and Tee, C. (2023). Traffic Impact Assessment System using Yolov5 and ByteTrack. Journal of Informatics and Web Engineering, 2(2), 168–188.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788.

Redmon, J., and Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems, 28.

Szeliski, R. (2022). Computer Vision: Algorithms and Applications. 2nd Edition. Springer.

Wang, C. Y., Bochkovskiy, A., and Liao, H. Y. M. (2023). YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7464–7475.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple Online and Realtime Tracking with a Deep Association Metric. Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 3645–3649.

Wu, D., Liao, M. W., Zhang, W. T., et al. (2022). YOLOP: You Only Look Once for Panoptic Driving Perception. Machine Intelligence Research, 19, 550–562.

Zhang, Y. (2023). Real-time vehicle detection and tracking based on the combination of YOLOv7 and ByteTrack. Applied and Computational Engineering, 4(1), 267–271.

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., and Wang, X. (2022). ByteTrack: Multi-Object Tracking by Associating Every Detection Box. European Conference on Computer Vision (ECCV), pp. 1–17.