Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.5, October 2025, pp. 2310~2320
e-ISSN: 2548-6861

Evaluation of YOLOv8 and Centroid Tracking in Vehicle Detection, Classification, and Counting System

Kevin Dicky Dharmasaputra [1]*, Bambang Pilu Hartato [2]
* Informatika, Universitas Amikom Yogyakarta
kevindicky11@students.amikom.ac.id [1], bambang.pilu@amikom.ac.id [2]

Article Info

Article history:

Received 2025-08-22
Revised 2025-09-09
Accepted 2025-09-19

Keyword:

YOLOv8,
Centroid Tracking,
Vehicle Detection,
Vehicle Counting,
Computer Vision.

ABSTRACT

An automatic vehicle detection and counting system is essential for Intelligent Transportation Systems (ITS) to monitor and manage traffic effectively. This study evaluates the performance of the lightweight YOLOv8n (nano) model for vehicle detection and classification, combined with a Centroid Tracking algorithm to improve vehicle counting accuracy. YOLOv8n was selected for its balance between computational efficiency and detection accuracy, making it suitable for devices with limited resources. The research involved collecting a dataset of seven vehicle classes (bus_l, bus_s, car, truck_l, truck_m, truck_s, truck_xl), followed by data preprocessing and training the YOLOv8n model for 40 epochs. Data augmentation techniques were applied to enhance data variability and improve model robustness. The Centroid Tracking algorithm was integrated to maintain vehicle identity across frames and prevent double counting. Model evaluation used precision, recall, F1-score, and mean Average Precision (mAP). Results show YOLOv8n achieved an overall mAP@0.5 of 0.820 and mAP@0.5-0.95 of 0.755. The "car" class attained the highest mAP of 0.963, while "truck_s" had the lowest at 0.665, mainly due to imbalanced data distribution. The Centroid Tracking effectively maintained object identities and provided consistent vehicle counts during testing. This combination offers a reliable and efficient system suitable for real-time traffic monitoring, parking management, and enhancing road safety. The YOLOv8n and Centroid Tracking-based system demonstrates strong potential for practical ITS applications, especially on devices with limited computational resources. Future work should focus on expanding the dataset and improving class balance to further enhance detection accuracy and system robustness.

I. PENDAHULUAN

Pertumbuhan volume kendaraan di jalan tol menuntut adanya sistem monitoring lalu lintas yang cerdas dan efisien. Sistem deteksi dan penghitungan kendaraan otomatis menjadi elemen kunci dalam Pemantauan kendaraan secara *real-time* di jalan tol masih menjadi tantangan yang kompleks. Beberapa kendala utama yang dihadapi mencakup kemacetan akibat volume kendaraan yang meningkat, keterbatasan sistem manual dalam memantau arus lalu lintas secara akurat, kurangnya data *real-time* untuk mendukung pengambilan keputusan cepat, serta keterbatasan teknologi konvensional dalam mendeteksi dan menghitung kendaraan secara otomatis, terutama saat terjadi *occlusion*, variasi pencahayaan, dan kondisi cuaca buruk [1].

Sebagai solusi atas tantangan tersebut, teknologi *Intelligent Transportation Systems* (ITS) mulai banyak diterapkan. ITS merupakan sistem terintegrasi yang menggunakan teknologi informasi dan komunikasi untuk meningkatkan efisiensi, keamanan, dan manajemen lalu lintas secara cerdas [2]. Salah satu komponen penting dalam ITS adalah sistem pengawasan otomatis berbasis kamera dan sensor, yang memungkinkan pemantauan lalu lintas secara real-time tanpa intervensi manusia secara langsung. Teknologi sensor seperti LiDAR juga telah terbukti efektif untuk pelacakan kendaraan dan estimasi kecepatan [3].

Salah satu bidang dalam ITS yang semakin berkembang adalah pemanfaatan *Computer Vision* (CV) untuk deteksi, klasifikasi, dan pelacakan kendaraan. Teknologi ini memanfaatkan kamera video dan algoritma pembelajaran mesin untuk mengenali objek secara visual. Dalam beberapa tahun terakhir, pendekatan *deep learning* dalam CV telah menunjukkan kemajuan signifikan dalam akurasi dan efisiensi [4]. Algoritma deteksi objek seperti YOLO *(You Only Look Once)* telah banyak digunakan dalam penelitian-penelitian terkait pemantauan lalu lintas, seperti klasifikasi kendaraan dari citra udara dengan akurasi 95,6% [5], deteksi kendaraan di jalan tol dengan performa mencapai mAP hingga 98.8% menggunakan YOLOv8 [6], pelacakan lalu lintas berbasis drone [7], deteksi objek pada kondisi malam dan hujan [8], estimasi kepadatan lalu lintas di persimpangan jalan [9], serta deteksi objek pada berbagai kondisi cuaca untuk kendaraan otonom [10].

YOLO adalah algoritma deteksi objek berbasis CNN *(Convolutional Neural Network)* yang dikenal karena kecepatan dan ketepatannya. Versi terbarunya, YOLOv8, membawa sejumlah peningkatan signifikan dalam hal struktur arsitektur, efisiensi komputasi, dan akurasi deteksi [11]. YOLOv8 juga mendukung format model yang ringan seperti YOLOv8n (nano), yang cocok untuk perangkat rendah dengan sumber daya terbatas [12]. Beberapa kelebihan YOLOv8 mencakup kecepatan inferensi tinggi, kemampuan *real-time detection*, serta dukungan multi-backbone. Namun, tantangan tetap ada, seperti sensitivitas terhadap objek yang tumpang tindih (*occlusion*), ketidakseimbangan kelas, serta performa yang menurun pada kondisi ekstrem.

Potensi YOLOv8 tidak hanya terbatas pada deteksi, tetapi juga integrasi dengan sistem pelacakan seperti DeepSORT, yang dirancang untuk pelacakan *multi-objek* secara akurat dan tangguh [13], serta efektif dalam sistem monitoring lalu lintas perkotaan berbasis drone [14], dan juga *Centroid Tracking*, yang efektif dalam memantau arah pergerakan kendaraan [15]. Kombinasi ini telah digunakan untuk menganalisis pelanggaran arah lalu lintas [16], pelacakan pejalan kaki pada kendaraan otonom dengan akurasi yang cukup baik [17], deteksi perilaku pengemudi yang tidak fokus dengan akurasi 91.9% ketika siang hari dan 90.3% ketika malam hari [18], pengenalan kecelakaan kendaraan secara real-time dari video surveilans lalu lintas mencapai presisi hingga 93.8% dengan mAP 96.1% [19], serta deteksi dan pelacakan kendaraan pada berbagai kondisi lingkungan [20]. Penerapan sistem deteksi dan pelacakan secara bersamaan memungkinkan sistem untuk menghitung kendaraan tanpa penghitungan ganda atau kehilangan jejak [21]. Berbagai penelitian terbaru juga menunjukkan pengembangan YOLOv8 untuk aplikasi khusus, seperti deteksi objek kecil pada citra satelit [22], deteksi rambu lalu lintas [23], deteksi kerusakan jalan [24].

Dalam penelitian ini, kami berkontribusi dengan melakukan eksperimen terhadap YOLOv8n dalam mendeteksi, mengklasifikasikan dan menghitung kendaraan di jalan tol Indonesia berdasarkan dataset yang telah dikonsolidasikan ke dalam tujuh kelas utama: *car, bus_s, bus_l, truck_s, truck_m, truck_l*, dan *truck_xl*.

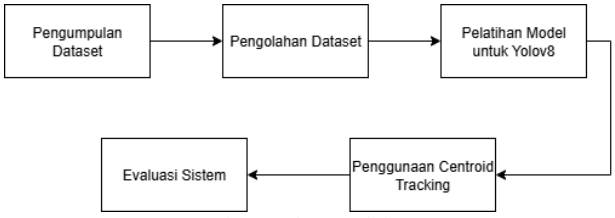
Penelitian ini secara spesifik fokus mengevaluasi performa kombinasi YOLOv8n dan *Centroid Tracking* untuk monitoring lalu lintas, tanpa melakukan perbandingan dengan metode tracking lain. Penelitian sebelumnya, masih menggunakan YOLOv3 yang relatif lebih lama. Sistem tersebut mampu mendeteksi kendaraan yang melawan arus dengan akurasi hampir 100% pada uji coba video lalu lintas, namun kinerjanya terbatas pada 2,5 FPS di CPU sehingga belum optimal untuk aplikasi *real-time*[16]. Sementara itu, penelitian terkini dengan YOLOv8 umumnya memanfaatkan perangkat GPU kelas tinggi dengan sumber daya komputasi yang melimpah. Namun, riset yang memanfaatkan YOLOv8n (*nano*) sebagai model *lightweight* dan dikombinasikan dengan algoritma pelacakan sederhana seperti *Centroid Tracking* pada perangkat dengan keterbatasan sumber daya masih relatif jarang dilakukan. Penelitian ini mengisi celah tersebut dengan mengevaluasi efektivitas model ringan untuk aplikasi ITS pada kondisi komputasi terbatas.

Penelitian ini bertujuan untuk mengevaluasi sejauh mana YOLOv8n, sebagai varian ringan dari YOLOv8, mampu digunakan secara efektif pada skenario lalu lintas jalan tol Indonesia, baik dari sisi akurasi klasifikasi maupun ketepatan penghitungan kendaraan.

Model dilatih menggunakan teknik augmentasi untuk mengatasi ketidakseimbangan data dan dievaluasi dengan metrik *precision, recall, F1-score*, dan mean *Average Precision* (mAP). Selain itu, metode *Centroid Tracking* diterapkan untuk memastikan penghitungan kendaraan yang akurat dan efisien.

II. METODE

Penelitian ini mengadopsi pendekatan kuantitatif eksperimental untuk mengembangkan dan mengevaluasi sistem deteksi, klasifikasi, dan penghitungan kendaraan. Alur metodologi penelitian ini digambarkan pada Gambar 1. Yang secara garis besar meliputi Pengumpulan Dataset, Pengolahan Dataset, Pelatihan Model untuk YOLOv8, penggunaan *Centroid Tracking*, dan Evaluasi Sistem.



Gambar 1. Alur Penelitian

A. Pengumpulan Data

Dataset kendaraan yang kami gunakan yaitu diperoleh dari platform Roboflow https://universe.roboflow.com/ilham-winar/venom/analytics. Dataset ini merupakan kumpulan

gambar kendaraan yang akan digunakan sebagai dasar untuk pelatihan dan pengujian model deteksi objek. Pada dataset ini terdiri dari 12 kelas kendaraan, yaitu *car, small truck, mid truck, big truck, truck(l), truck(m), truck(s), truck(xl), small bus, big bus, bus(s), dan bus(l)*

B. Pengolahan Dataset

Tahap pengolahan dataset melibatkan modifikasi dan konsolidasi kelas, serta augmentasi data. Dataset asli yang memiliki 12 kelas kendaraan *car, small truck, mid truck, big truck, truck(l), truck(m), truck(s), truck(xl), small bus, big bus, bus(s), dan bus(l)* dimodifikasi dan dikonsolidasi menjadi 7 kelas yang lebih spesifik dan relevan untuk konteks jalan tol. Konsolidasi kelas dilakukan untuk mengurangi kompleksitas dan meningkatkan fokus pada jenis kendaraan utama di jalan tol. Tujuh kelas yang digunakan adalah: *bus_l (large bus), bus_s (small bus), car, truck_l (large truck), truck_m (medium truck), truck_s (small truck), dan truck_xl (extra large truck)*.

Untuk mengatasi keterbatasan jumlah data awal (4058 gambar) dan meningkatkan model terhadap variasi kondisi pencahayaan dan posisi, augmentasi data ekstensif diterapkan. Teknik augmentasi yang digunakan meliputi, *Horizontal flip, Brightness & Contrast adjustment, Hue/Saturation/Value adjustment, Gaussian Blur, dan Motion Blur*. Setelah augmentasi, jumlah gambar meningkat secara signifikan dari 4058 menjadi 8081 gambar. Dataset yang telah diaugmentasi kemudian dibagi menjadi tiga set, seperti yang dapat dilihat di Tabel 1. 70% (5.662 Gambar) untuk data pelatihan (*train set*), 20% (1.579 Gambar) untuk data validasi (*validation set*), dan 10% (840 Gambar) untuk data pengujian (*test set*). Pembagian ini mengikuti praktik umum dalam pelatihan model pembelajaran mesin, di mana 70% dialokasikan untuk melatih model secara optimal, 20% digunakan untuk melihat performa model saat proses pelatihan serta mencegah *overfitting*, dan 10% terakhir digunakan sebagai evaluasi generalisasi model terhadap data yang belum pernah dilihat pada proses sebelumnya. Proporsi ini dianggap seimbang untuk memberikan cukup data pada setiap tahap, khususnya saat jumlah data terbatas namun tetap mengutamakan validitas evaluasi.

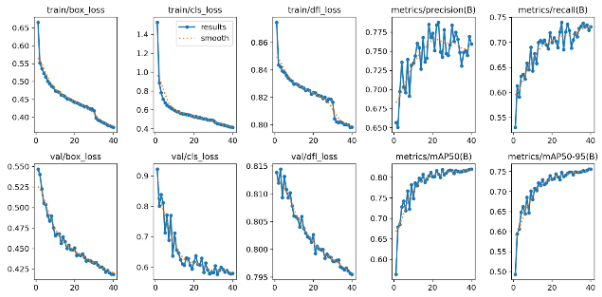
TABEL I
PEMBAGIAN DATA

No	Jenis data	Jumlah data	rasio
1	Train	5.662	70%
2	Valid	1.579	20%
3	Test	840	10%
4	**Total**	**8081**	**100%**

C. Pelatihan Model

Model deteksi objek yang digunakan dalam penelitian ini adalah YOLOv8n (nano), versi terkecil dan paling efisien dari keluarga YOLOv8. YOLOv8n dipilih karena keseimbangan optimal antara akurasi dan kecepatan, menjadikannya ideal untuk aplikasi seperti monitoring lalu lintas, mengikuti tren penggunaan model ringan untuk edge computing [25]. Model dilatih selama 40 epoch dengan parameter default YOLOv8n. *Loss function* yang digunakan dalam pelatihan YOLOv8n meliputi mAP50, mAP50-95, *box loss, classification loss, dan DFL loss*.

Proses pelatihan model dapat dievaluas melalui grafik *loss function* yang ditampilkan pada Gambar 2. Grafik ini menunjukkan bagaimana model belajar dan memperbaiki performanya selama proses training.



Gambar 2. Proses Pelatihan

Dari grafik training loss, dapat diamati bahwa *Box Loss* menurun dari 0.65 menjadi sekitar 0.37 pada epoch ke-40, yang menunjukkan konvergensi yang baik dalam prediksi *bounding box*. Fakta tersebut menunjukkan bahwa model mengalami peningkatan ketepatan dalam memperkirakan letak objek di dalam gambar. *Classification Loss* menurun dari 1.5 menjadi sekitar 0.4, yang menunjukkan peningkatan kemampuan klasifikasi yang signifikan. Penurunan yang stabil ini mengindikasikan bahwa model berhasil mempelajari pola-pola yang membedakan antar kelas objek.

DFL Loss menurun dari 0.86 menjadi sekitar 0.8, yang menunjukkan perbaikan dalam distribution *focal loss*. Penurunan ini mengindikasikan bahwa model semakin baik dalam menangani ketidakseimbangan kelas dan fokus pada contoh-contoh yang sulit. Konvergensi yang stabil pada semua jenis loss menunjukkan bahwa model telah mencapai kondisi yang optimal dan tidak mengalami *overfitting.*

Dari segi metrics performa, *Precision* meningkat dari 0.65 menjadi sekitar 0.77 pada epoch ke-40, yang menunjukkan peningkatan akurasi prediksi yang konsisten. *Recall* meningkat dari 0.55 menjadi sekitar 0.73 pada epoch ke-40, yang mengindikasikan bahwa model semakin baik dalam mendeteksi objek yang ada dalam dataset. mAP@0.5 meningkat dari 0.65 menjadi sekitar 0.8, yang menunjukkan peningkatan performa yang signifikan dalam deteksi objek dengan IoU *threshold* 0.5.

mAP@0.5:0.95 meningkat dari 0.5 menjadi sekitar 0.75, yang menunjukkan robustness model pada berbagai IoU *threshold* yang berbeda. Hal ini mengindikasikan bahwa model tidak hanya akurat dalam mendeteksi objek tetapi juga presisi dalam memprediksi lokasi objek yang tepat.

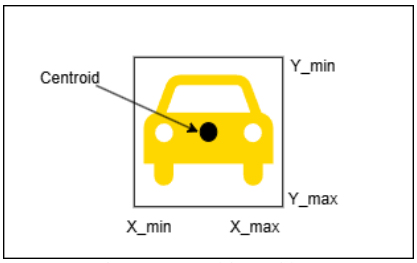
D. Penggunaan Centroid Tracking

Setelah model YOLOv8n mendeteksi objek kendaraan dalam setiap *frame* video, metode *Centroid Tracking* diterapkan untuk melacak pergerakan kendaraan

menggunakan ID dan melakukan penghitungan. Sistem *tracking* ini bekerja berdasarkan perhitungan jarak *Euclidean* antar *centroid* pada *frame* berturut-turut dengan *threshold* jarak 50 piksel untuk mengasosiasikan objek yang sama. *Euclidean distance* dipilih karena sederhana, cepat secara komputasi, dan efektif untuk objek yang bergerak relatif halus di kamera statis (seperti CCTV), sehingga cocok untuk aplikasi real-time pada perangkat dengan sumber daya terbatas. Alur kerja *Centroid Tracking* dimulai dengan ekstraksi centroid, dimana untuk setiap *bounding box* yang terdeteksi oleh YOLOv8n, titik pusat (*centroid*) dihitung sebagai koordinat referensi objek tersebut. Seperti yang di ilustrasikan pada gambar 3.

$$X_{Centroid} = \frac{x_{Min} + x_{Max}}{2}$$

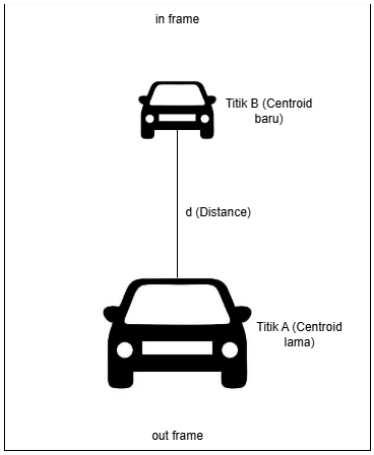
$$Y_{Centroid} = \frac{y_{Min} + y_{Max}}{2}$$



Gambar 3. Centroid Tracking

Proses selanjutnya adalah asosiasi objek, dimana centroid dari *frame* saat ini dibandingkan dengan centroid objek yang sudah ada dari *frame* sebelumnya. Objek baru diasosiasikan dengan objek yang sudah ada jika jarak *Euclidean* antar *centroid* berada di bawah ambang batas tertentu, sehingga sistem dapat mempertahankan identitas objek yang sama sepanjang sekuens video, seperti pada gambar 4 yang menampilkan ilustrasi jarak *Euclidean* kendaraan dari arah depan atas.

$$d = \sqrt{\{(x_1 - x_2)^2 + (y_1 - y_2)^2\}}$$



Gambar 4. Euclidean Distance

Untuk penghitungan kendaraan, ketika suatu objek muncul didalam *frame*, sistem akan mencatatnya sebagai satu hitungan kendaraan baru pada setiap kelas. Mekanisme ini membantu dalam mendapatkan data tentang volume lalu lintas secara akurat. Algoritma ini juga menangani kasus di mana objek hilang sementara atau objek baru muncul dengan mempertahankan riwayat centroid selama beberapa *frame*, sehingga dapat mengatasi masalah oklusi atau gangguan sementara pada proses deteksi.

E. Evaluasi Sistem

Performa sistem dievaluasi menggunakan metrik standar dalam deteksi objek dan analisis hasil *tracking*. *Mean Average Precision* (mAP) digunakan sebagai metrik utama yang menghitung rata-rata *Average Precision* (AP) dari semua kelas. mAP@0.5 menunjukkan mAP pada IoU *threshold* 0.5, sedangkan mAP@0.5-0.95 menunjukkan rata-rata mAP pada berbagai IoU *threshold* dari 0.5 hingga 0.95, memberikan evaluasi yang lebih komprehensif terhadap akurasi deteksi pada berbagai tingkat presisi.

$$mAP = \frac{1}{c} \sum_{\{c=1\}}^{\{C\}} AveragePrecision_c$$

Confusion Matrix juga digunakan untuk menampilan prediksi yang benar guna mengevaluasi kinerja model, baik dalam bentuk *raw count* maupun *normalized.*

III. HASIL DAN PEMBAHASAN

Pada Tabel 2 menunjukkan distribusi jumlah anotasi per kelas kendaraan yang digunakan dalam proses pelatihan (*training*), validasi (*validation*), dan pengujian (*testing*). Walaupun telah dilakukan Augmentasi dikelas minoritas, tetapi belum bisa mendekati anotasi dari kelas *car*. Dapat dilihat bahwa kelas *car* mendominasi dataset dengan jumlah anotasi yang jauh lebih tinggi dibandingkan kelas lain, terutama dibandingkan *bus_s* dan *truck_xl*, yang menunjukkan adanya ketidakseimbangan kelas.

TABEL II
ANOTASI PERKELAS

No	Kelas	Data Training	Data Validation	Data Testing
1	Bus_l	1.184	468	261
2	Bus_s	438	128	97
3	Car	25.335	9.944	5.579
4	Truck_s	4.797	1.478	864
5	Truck_m	6.108	2.287	1.345
6	Truck_l	4.189	1.468	1.017
7	Truck_xl	1.278	351	211

Pada Tabel 3 berisi hyperparameter utama yang digunakan dalam proses pelatihan model YOLOv8n, sesuai dengan konfigurasi standar dari pustaka Ultralytics tanpa adanya perubahan. Proses pelatihan menggunakan optimizer AdamW, sebuah varian dari algoritma *Adam* dengan epoch

penambahan weight decay untuk regularisasi. Learning rate ditetapkan pada nilai 0.000909, sementara momentum disetel pada 0.9, yang membantu mempercepat konvergensi. Pelatihan dilakukan selama 40 epoch, yang cukup untuk mencapai stabilitas tanpa mengalami *overfitting*. Semua nilai *hyperparameter* tersebut digunakan secara *default*, tanpa penyesuaian manual, sebagaimana disediakan oleh konfigurasi resmi YOLOv8n dari Ultralytics.

TABEL III
HYPERPARAMETER

No	Hyperparameter	Value
1	Optimizer	AdamW
2	Learning Rate	0.000909
3	Momentum	0.9
4	Epoch	40

Pada Tabel 4 menyajikan perkembangan metrik pelatihan model YOLOv8n selama 40 epoch. Selama proses pelatihan, tiga jenis *loss* yang diamati meliputi *box_loss*, *cls_loss*, dan *dfl_loss*. *Box_loss* mencerminkan tingkat kesalahan dalam memprediksi posisi *bounding box*, sedangkan *cls_loss* menunjukkan kesalahan dalam klasifikasi objek ke dalam kelas yang benar. Sementara itu, *dfl_loss* atau *Distribution Focal Loss* digunakan untuk mengukur akurasi distribusi prediksi *bounding box* terhadap lokasi sebenarnya. Nilai mAP@0.5 menunjukkan rata-rata akurasi deteksi model ketika *Intersection over Union* (IoU) diatur pada ambang batas 0.5, sedangkan mAP@0.5–0.95 merupakan indikator yang lebih ketat dan menyeluruh karena menghitung rata-rata akurasi dari berbagai ambang batas IoU.

TABEL IV
PELATIHAN MODEL

Epoch	Box_ loss	Cls_ loss	Dfl_ loss	mAP 50	mAP 50-95
1	0.6655	1.529	0.8748	0.564	0.493
2	0.5518	0.888	0.8436	0.679	0.595
3	0.5363	0.783	0.8422	0.679	0.595
…	…	…	…	…	…
38	0.3763	0.422	0.7999	0.818	0.753
39	0.3739	0.418	0.7981	0.818	0.753
40	0.3714	0.413	0.7983	0.820	0.755

Pada Tabel 5 ini ditunjukkan performa model YOLOv8n perkelas berdasarkan hasil validasi terhadap data uji. Evaluasi dilakukan menggunakan beberapa metrik utama, yaitu *Precision* (P), *Recall* (R), *F1-Score*, mAP@0.5, dan mAP@0.5–0.95. *Precision* menggambarkan tingkat ketepatan model dalam melakukan prediksi *positif* yang benar, atau seberapa banyak dari semua deteksi kendaraan yang benar-benar akurat, sedangkan *Recall* mengukur seberapa besar proporsi kendaraan yang berhasil terdeteksi dari keseluruhan objek yang seharusnya terdeteksi dalam data. Sedangkan *F1-Score*, yang menggabungkan *Precision* dan *Recall*, menilai keseimbangan antara ketepatan dan kemampuan deteksi. Dari tabel terlihat bahwa kelas *Car* memiliki nilai *Precision*, *Recall*, dan *F1-Score* tertinggi, menandakan deteksi yang stabil dan akurat, sementara kelas *Truck_s* memiliki nilai terendah pada ketiga metrik, menunjukkan tantangan dalam mendeteksi kendaraan kecil atau jarang muncul. Kelas lainnya berada pada kisaran menengah, sehingga variasi performa model antar kelas kendaraan menjadi jelas terlihat.

TABEL V
HASIL PENGUJIAN

Kelas	Precision	Recall	F1-Score	mAP 50	mAP5 0-95
Bus_l	0.896	0.700	0.786	0.852	0.782
Bus_s	0.732	0.724	0.728	0.754	0.690
Car	0.940	0.774	0.849	0.963	0.893
Truck_l	0.759	0.757	0.758	0.850	0.796
Truck_m	0.726	0.699	0.712	0.806	0.741
Truck_s	0.563	0.602	0.582	0.665	0.603
Truck_xl	0.774	0.818	0.795	0.846	0.782

Pada Tabel 6 ditunjukkan performa model YOLOv8n untuk setiap kelas kendaraan berdasarkan hasil pengujian dengan menggunakan data test. Evaluasi dilakukan menggunakan lima metrik utama, yaitu *Precision (P), Recall (R), F1-Score*, *mAP*@0.5, dan *mAP*@0.5–0.95.

Dari tabel terlihat bahwa kelas *Car* memiliki performa terbaik dengan nilai *Precision* 0.970, *Recall* 0.817, dan *F1-Score* 0.887, sekaligus mencapai *mAP*@0.5 sebesar 0.978. Hal ini menunjukkan model mampu mendeteksi mobil dengan stabil dan akurat berkat dominasi jumlah data pada kelas ini. Sebaliknya, kelas *Truck_s* memiliki performa terendah dengan *Precision* 0.616, *Recall* 0.650, dan *F1-Score* 0.633, menandakan adanya kesulitan model dalam mengidentifikasi truk kecil, terutama karena kemiripan visual dengan kelas truk lain serta distribusi data yang terbatas.

Kelas lainnya, seperti *Bus_l*, *Truck_l*, dan *Truck_xl*, berada pada kisaran nilai menengah dengan *F1-Score* antara 0.74 hingga 0.78. Hal ini menunjukkan bahwa model cukup konsisten dalam mendeteksi kendaraan berukuran besar meskipun tetap ada kasus overcounting atau undercounting. Sementara itu, *Bus_s* dan *Truck_m* menunjukkan performa yang moderat dengan *F1-Score* masing-masing 0.654 dan 0.704, yang masih perlu ditingkatkan melalui penyeimbangan data dan variasi kondisi pelatihan.

TABEL VI
HASIL PENGUJIAN DENGAN DATA TEST

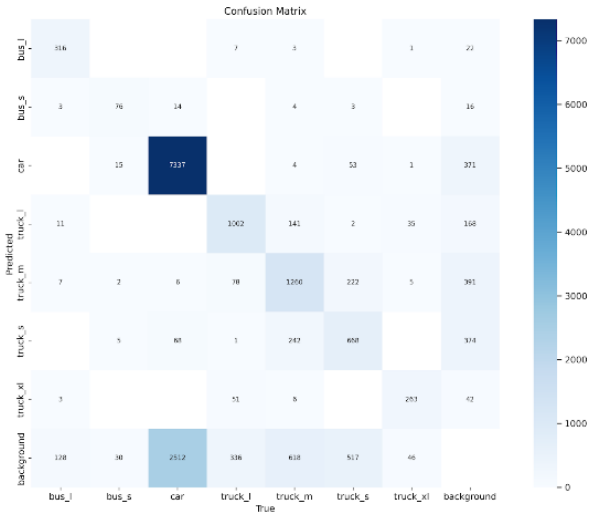
Kelas	Precision	Recall	F1-Score	mAP 50	mAP5 0-95
Bus_l	0.879	0.659	0.753	0.816	0.731
Bus_s	0.585	0.742	0.654	0.720	0.665
Car	0.970	0.817	0.887	0.978	0.903
Truck_l	0.702	0.793	0.745	0.845	0.792
Truck_m	0.699	0.710	0.704	0.776	0.711
Truck_s	0.616	0.650	0.633	0.692	0.622
Truck_xl	0.735	0.834	0.781	0.850	0.789

Secara keseluruhan, analisis ini menegaskan bahwa performa model sangat dipengaruhi oleh keseimbangan data

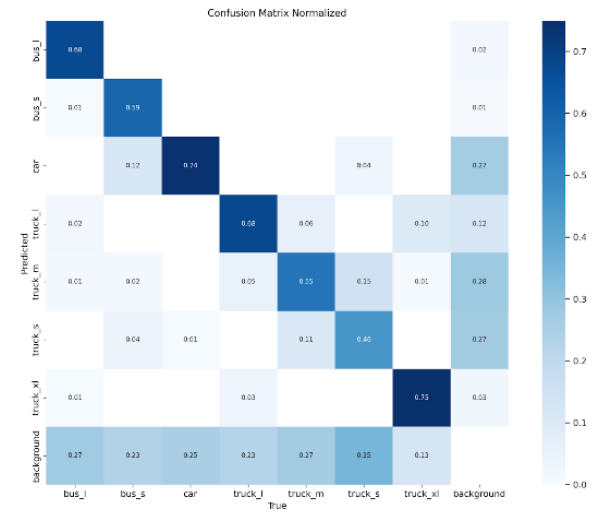
dan kemiripan visual antar kelas kendaraan, di mana kelas dominan seperti mobil terdeteksi dengan baik, sementara kelas minoritas atau dengan bentuk yang mirip menunjukkan tantangan yang lebih besar.

A. Hasil Evaluasi Model

Evaluasi performa model dilakukan menggunakan *confusion matrix* yang menunjukkan distribusi prediksi untuk setiap kelas objek. Gambar 5 menampilkan *confusion matrix* dalam bentuk nilai *absolut*, sedangkan Gambar 6 menunjukkan versi yang telah dinormalisasi.



Gambar 5. Confusion Matrix



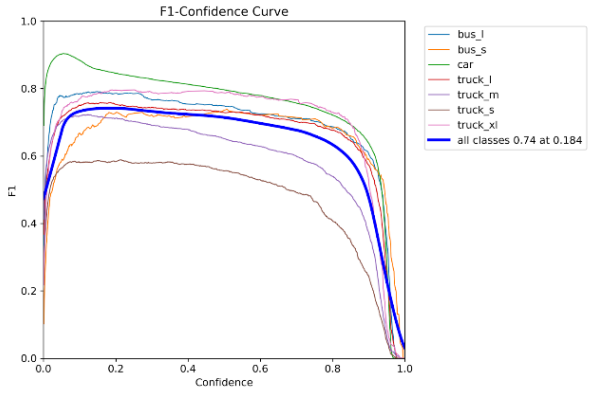
Gambar 6. Confusion Matrix Normalized

Dari hasil *confusion matrix* yang ditampilkan pada Gambar 5 dan 6, dapat diamati bahwa kelas "*car*" menunjukkan performa terbaik dengan 7.337 prediksi yang benar dan akurasi 74 persen. Hal ini menunjukkan bahwa model mampu mengidentifikasi mobil dengan tingkat akurasi yang tinggi dibandingkan kelas lainnya. Kelas *"bus_l"* memiliki akurasi 68 persen dengan 316 prediksi yang benar, menunjukkan performa yang cukup baik untuk kategori bis berukuran besar. Sementara itu, kelas *"truck_xl"* menunjukkan akurasi 75 persen dengan 263 prediksi yang benar, yang merupakan performa terbaik kedua setelah mobil.

Performa yang lebih rendah ditunjukkan oleh kelas *"truck_m"* dan *"truck_l"* dengan akurasi masing-masing 55 persen dan 68 persen. Kelas *"truck_s"* menunjukkan performa terendah dengan akurasi hanya 46 persen, yang mengindikasikan bahwa model mengalami kesulitan dalam mengidentifikasi truk berukuran kecil. Kesalahan klasifikasi yang paling sering terjadi adalah antara kelas-kelas truk yang berbeda ukuran, yang menunjukkan bahwa model masih kesulitan membedakan variasi ukuran truk yang memiliki karakteristik visual yang serupa.

B. Analisis Kurva F1-Confidence

Kurva *F1-Confidence* yang ditampilkan pada Gambar 7 menunjukkan hubungan antara nilai *F1-score* dan *confidence threshold* untuk setiap kelas objek. Analisis ini digunakan untuk memahami bagaimana performa model berubah seiring dengan perubahan tingkat keyakinan prediksi.



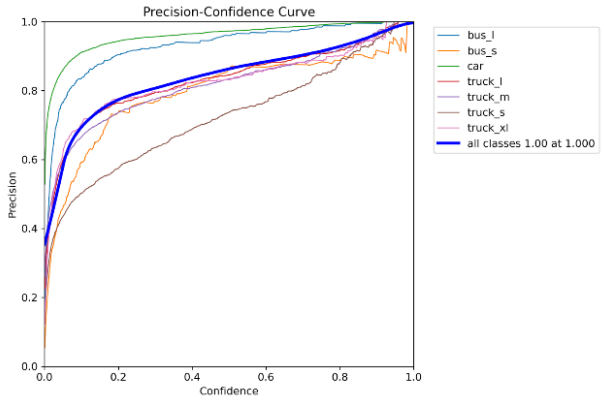
Gambar 7. F1-Confidence Curve

Hasil analisis menunjukkan bahwa kelas "*car*" yang ditampilkan dengan warna hijau memiliki performa terbaik dengan *F1-score* tertinggi mencapai sekitar 0.9 pada *confidence* yang rendah. Hal ini menunjukkan bahwa model memiliki kemampuan yang sangat baik dalam mendeteksi mobil dengan tingkat akurasi dan *recall* yang seimbang. Kelas "*bus_l*" dan "*truck_l*" menunjukkan performa yang stabil dengan *F1-score* sekitar 0.8, yang mengindikasikan bahwa model cukup konsisten dalam mendeteksi objek-objek berukuran besar.

Sebaliknya, kelas "*truck_s*" yang ditampilkan dengan warna coklat menunjukkan performa terburuk dengan *F1-score* maksimal hanya sekitar 0.6. Hal ini konsisten dengan hasil *confusion matrix* yang menunjukkan bahwa truk kecil merupakan kelas yang paling sulit untuk diidentifikasi. Performa keseluruhan yang ditampilkan dengan garis biru tebal mencapai *F1-score* 0.74 pada *confidence threshold* 0.184, yang menunjukkan bahwa model memiliki performa yang baik secara umum.

C. *Analisis Kurva Precision-Confidence*

Kurva *Precision-Confidence* yang ditampilkan pada Gambar 8 menunjukkan hubungan antara *precision* dan *confidence threshold* untuk memahami seberapa akurat prediksi model ketika tingkat keyakinan bervariasi.



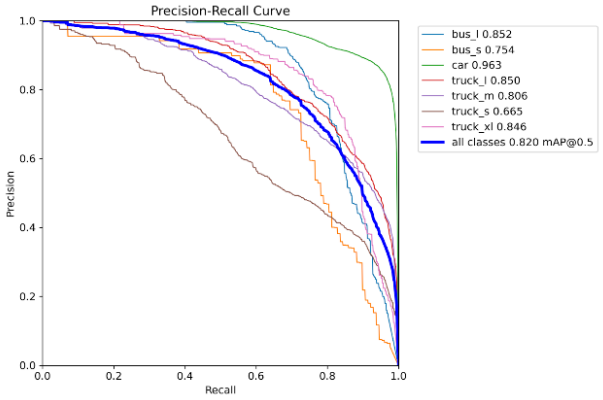
Gambar 8. Precision-Confidence Curve

Hasil menunjukkan bahwa kelas "*car*" mempertahankan *precision* yang tinggi di atas 0.9 pada berbagai tingkat *confidence*, yang mengindikasikan bahwa ketika model memprediksi sebuah objek sebagai mobil, prediksi tersebut sangat mungkin benar. Kelas "*bus_l*" dan "*truck_l*" menunjukkan *precision* yang stabil di atas 0.8, yang menunjukkan bahwa model juga cukup andal dalam mengidentifikasi objek-objek berukuran besar ini. Namun, kelas "*truck_s*" menunjukkan *precision* yang paling rendah dan fluktuatif, yang mengindikasikan bahwa model sering melakukan kesalahan ketika memprediksi truck berukuran kecil.

Performa keseluruhan mencapai *precision* 1.0 pada *confidence threshold* 1.0, yang menunjukkan bahwa ketika model sangat yakin dengan prediksinya, prediksi tersebut hampir selalu benar. Hal ini mengindikasikan bahwa model memiliki kemampuan yang baik dalam menilai tingkat keyakinan prediksinya sendiri.

D. *Analisis Kurva Precision-Recall*

Kurva *Precision-Recall* yang ditampilkan pada Gambar 9 menunjukkan *trade-off* antara *precision* dan *recall* untuk setiap kelas objek. Analisis ini memberikan gambaran menyeluruh tentang performa model untuk setiap kelas.



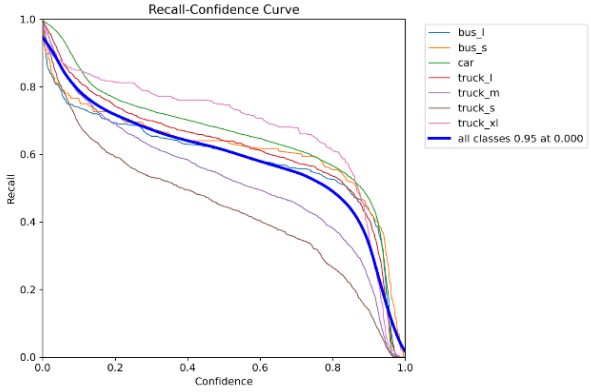
Gambar 9. Precision-Recall Curve

Hasil menunjukkan nilai *Average Precision* (AP) yang bervariasi untuk setiap kelas. Kelas "*car*" memiliki AP 0.963, yang merupakan performa terbaik dan menunjukkan bahwa model sangat efektif dalam mendeteksi mobil dengan tingkat *precision* dan *recall* yang tinggi. Kelas "*bus_l*" dan "*truck_l*" menunjukkan AP yang hampir sama, yaitu 0.852 dan 0.850, yang mengindikasikan bahwa model memiliki performa yang konsisten untuk objek-objek berukuran besar.

Kelas "*truck_xl*" memiliki AP 0.846, yang menunjukkan performa yang baik untuk kategori truck berukuran sangat besar. Kelas "*truck_m*" dan "*bus_s*" menunjukkan AP masing-masing 0.806 dan 0.754, yang masih dalam kategori performa yang baik namun lebih rendah dari kelas-kelas sebelumnya. Kelas "*truck_s*" menunjukkan AP terendah yaitu 0.665, yang mengkonfirmasi bahwa ini adalah kelas yang paling sulit untuk dideteksi. Nilai mAP@0.5 keseluruhan mencapai 0.820, yang menunjukkan bahwa model memiliki performa yang baik secara umum

E. *Analisis Kurva Recall-Confidence*

Kurva *Recall-Confidence* yang ditampilkan pada Gambar 10 menunjukkan hubungan antara *recall* dan *confidence threshold*. Analisis ini penting untuk memahami kemampuan model dalam mendeteksi semua objek yang ada dalam dataset.



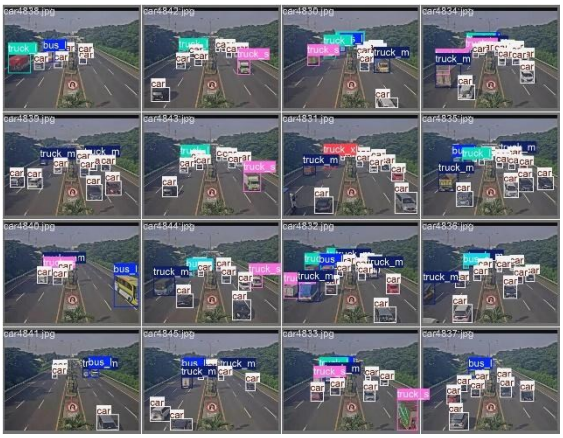
Gambar 10. Recall-Confidence Curve

Hasil menunjukkan bahwa semua kelas menunjukkan *recall* yang tinggi mendekati 1.0 pada *confidence threshold* rendah, yang mengindikasikan bahwa model mampu mendeteksi sebagian besar objek ketika menggunakan *threshold* yang rendah.

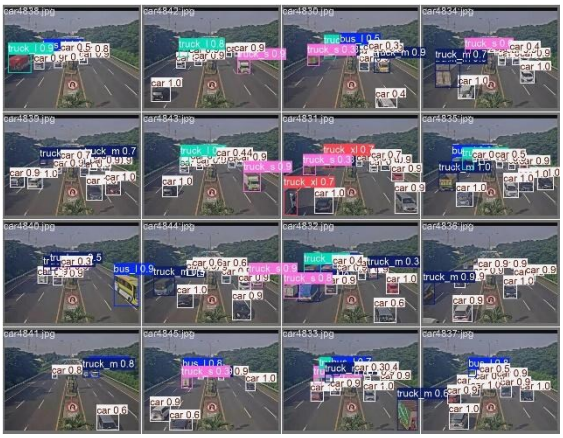
Performa keseluruhan mencapai *recall* 0.95 pada *confidence threshold* 0.0, yang menunjukkan bahwa model memiliki kemampuan deteksi yang sangat baik ketika menggunakan *threshold* yang rendah.

F. Hasil Pengujian

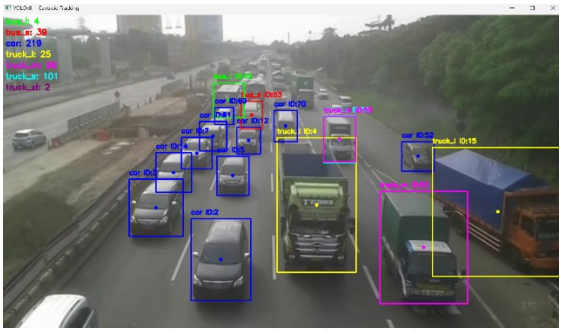
Dapat dilihat pada gambar 11 yaitu hasil pengujian dengan menampilkan hasil kelasnya, sedangkan untuk gambar 12 menampilkan hasil dengan titik koordinat. Dan pada gambar 13 menampilkan hasil pengujian/test dengan video lalu lintas jalan tol.



Gambar 11. Hasil dengan kelas



Gambar 12. Hasil dengan koordinat



Gambar 13. Hasil test dengan Video

Dari sisi performa komputasi, model YOLOv8n menunjukkan kecepatan inferensi yang baik dengan rata-rata sekitar 15 FPS ketika dijalankan pada laptop HP Pavilion Aero 13 dengan spesifikasi: Processor AMD Ryzen 7 5825U with Radeon Graphics 2.00 GHz, RAM 16GB, GPU Integrated AMD Radeon Graphics, Storage 477GB. Inferensi dilakukan menggunakan CPU dengan dukungan GPU terintegrasi. Kecepatan ini menunjukkan bahwa model mampu melakukan deteksi dan tracking secara near real-time, yang merupakan persyaratan penting untuk aplikasi monitoring lalu lintas

TABEL VII
GROUND TRUTH

Kelas	Hitungan Manual	Hitungan Model	Selisih Hitungan
Bus_l	2	4	2
Bus_s	30	39	9
Car	216	219	3
Truck_l	21	25	4
Truck_m	87	96	9
Truck_s	80	101	21
Truck_xl	3	2	1

Pada Tabel 7 menunjukkan hasil *ground Truth*, untuk kategori mobil (*car*), yang merupakan kelas dominan dalam dataset, model menunjukkan performa deteksi yang sangat baik. Dari total 216 kendaraan yang terdeteksi secara manual, model berhasil menghitung 219 kendaraan, dengan selisih hanya 3. Hal ini menunjukkan bahwa model mampu mengenali kendaraan jenis mobil dengan akurasi tinggi, didukung oleh banyaknya sampel kelas ini dalam proses pelatihan sehingga model dapat melakukan generalisasi dengan baik.

Sementara itu, pada kategori bis, baik bis kecil (*bus_s*) maupun bis besar (*bus_l*), terjadi kelebihan hitungan atau *overcounting*. Untuk bis kecil, model mendeteksi 39 kendaraan dari hitungan manual sebanyak 30, sedangkan pada bis besar terdapat dua deteksi lebih banyak dari seharusnya. Kelebihan ini dapat disebabkan oleh beberapa faktor, seperti ketidakseimbangan jumlah data antar kelas dalam pelatihan, kemiripan visual antara bis dan truk besar dalam beberapa *frame*, serta potensi terjadinya deteksi ganda saat kendaraan melambat atau berhenti dalam jangkauan kamera.

Kategori truk juga menunjukkan tren *overcounting* serupa. Pada truk kecil (*truck_s*), model menghitung 101 kendaraan dari hitungan manual sebanyak 80. Pada truk sedang (*truck_m*) dan truk besar (*truck_l*), masing-masing terjadi kelebihan sebanyak 9 dan 4 kendaraan. Hal ini kemungkinan besar disebabkan oleh pergerakan lambat yang menyebabkan objek terdeteksi dan dihitung lebih dari sekali oleh sistem pelacakan. Selain itu, kesamaan bentuk dan ukuran antar tipe truk juga menambah kompleksitas dalam proses klasifikasi, sehingga memengaruhi akurasi hasil deteksi dan hitungan.

Sebaliknya, pada kelas truk ekstra besar *(truck_xl)*, justru terjadi kekurangan hitungan (*undercounting*). Dari tiga kendaraan yang ada, hanya dua yang berhasil dideteksi oleh model. Hal ini dapat disebabkan oleh sebagian tubuh kendaraan yang tertutup (*occlusion*) dalam video uji, sehingga sistem gagal mengenali objek secara utuh. Faktor lain yang mungkin memengaruhi adalah minimnya data pelatihan khusus untuk kelas ini, yang membuat model belum cukup terlatih untuk mengenali truk jenis ini secara optimal.

Meskipun sistem *Centroid Tracking* dengan pendekataan *Euclidean* menunjukkan performa yang baik secara umum, namun masih terdapat tantangan ketika menghadapi kondisi *occlusion* yang berat. Hal ini terlihat pada kelas "truck_xl" yang mengalami *undercounting*, dimana dari tiga kendaraan yang ada hanya dua yang berhasil dideteksi. *Occlusion* yang terjadi ketika kendaraan besar saling menutupi atau tertutup oleh infrastruktur jalan menyebabkan sistem kesulitan mempertahankan tracking identitas objek secara konsisten. *Centroid Tracking* yang berbasis pada jarak *Euclidean* sederhana belum dilengkapi dengan mekanisme prediksi pergerakan yang lebih *sophisticated* untuk mengatasi *occlusion temporary*.

Secara keseluruhan, hasil ini menunjukkan bahwa model memiliki performa sangat baik pada kelas dominan, namun masih memerlukan perbaikan dalam menghadapi tantangan pada kelas minoritas, kemiripan visual antar kendaraan, serta kondisi lingkungan seperti *occlusion* dan pergerakan lambat yang memicu duplikasi identifikasi.

TABEL VIII
EVALUASI SEBELUM DAN SESUDAH INTEGRASI CENTROID TRACKING

Kelas	Hitungan Manual	Hitungan Yolo+Centroid Tracking	Hitungan Yolo Only
Bus_l	2	4	61
Bus_s	30	39	1046
Car	216	219	10729
Truck_l	21	25	671
Truck_m	87	96	2529
Truck_s	80	101	903
Truck_xl	3	2	48

Pada tabel 8 menunjukkan perbandingan hasil penghitungan kendaraan menggunakan tiga metode berbeda: hitungan manual sebagai ground truth, sistem YOLOv8n dengan *Centroid Tracking*, dan YOLOv8n tanpa tracking (*YOLO only*). Dari data tersebut terlihat bahwa *YOLO only* menghasilkan overcounting yang sangat signifikan pada hampir semua kelas kendaraan.

Untuk kelas *Bus_l*, hitungan manual menunjukkan 2 kendaraan, sistem *YOLO+Centroid Tracking* mendeteksi 4 kendaraan, sementara *YOLO only* menghasilkan 61 deteksi. Kelas *Bus_s* menampilkan pola serupa dengan hitungan manual 30, *YOLO+tracking* 39, dan *YOLO only* mencapai 1046 deteksi. Kelas *Car* yang merupakan kategori dominan menunjukkan hitungan manual 216, *YOLO+tracking* 219, namun *YOLO only* menghasilkan 10729 deteksi.

Pada kategori *truck*, kelas *Truck_l* menunjukkan hitungan manual 21, *YOLO+tracking* 25, dan *YOLO only* 671. *Truck_m* memperlihatkan hitungan manual 87, YOLO+tracking 96, sementara *YOLO only* mencapai 2529 deteksi. Kelas *Truck_s* menampilkan hitungan manual 80, *YOLO+tracking* 101, dan *YOLO only* 903 deteksi. Untuk *Truck_xl*, hitungan manual menunjukkan 3, *YOLO+tracking* 2, dan *YOLO only* 48 deteksi.

Hasil ini mengindikasikan bahwa tanpa sistem tracking, setiap deteksi objek pada *frame* yang berbeda dianggap sebagai objek baru, menyebabkan penghitungan berulang untuk kendaraan yang sama. Sistem *YOLO+Centroid Tracking* menunjukkan hasil yang jauh lebih mendekati hitungan manual, dengan selisih yang relatif kecil pada setiap kelas. Perbedaan ekstrem antara *YOLO only* dan kedua metode lainnya mendemonstrasikan pentingnya implementasi algoritma tracking untuk aplikasi penghitungan kendaraan yang akurat.

IV. KESIMPULAN

Berdasarkan hasil penelitian, model YOLOv8n menunjukkan performa yang baik untuk aplikasi monitoring lalu lintas jalan tol, dengan *F1-score* sebesar 0.74 dan mAP@0.5 mencapai 0.820. Kelas "*car*" mencatat AP tertinggi (0.963) berkat dominasi jumlah sampel, sedangkan kelas "*truck_s*" memiliki AP terendah (0.665) akibat distribusi data yang timpang. Ketidakseimbangan kelas ini berdampak langsung pada akurasi klasifikasi, terutama pada kelas minoritas seperti "*bus_s*" dan "*truck_s*". Walaupun augmentasi telah diaplikasikan, tetapi masih tidak cukup untuk menambah variasi dari kelas tersebut.

Konsolidasi menjadi tujuh kelas kendaraan terbukti relevan dengan kebutuhan klasifikasi kendaraan jalan tol, meskipun tantangan tetap muncul pada kelas yang mirip secara visual. Pengumpulan data tambahan dari berbagai kondisi lokasi, waktu, dan cuaca sangat disarankan guna memperkaya variasi dan meningkatkan generalisasi model.

Centroid Tracking yang telah diterapkan berhasil menjaga konsistensi identitas kendaraan di setiap *frame*, sehingga mencegah penghitungan ganda dan meningkatkan akurasi sistem penghitungan kendaraan. Pengujian pada kondisi lalu lintas padat menunjukkan efektivitas metode ini dalam menjaga ketepatan deteksi.

Dari sisi performa sistem penghitungan kendaraan, didapatkan total hitungan manual sebanyak 439 kendaraan, sementara model mendeteksi 486 kendaraan, dengan total

selisih absolut sebesar 47 kendaraan. Dengan demikian, tingkat keakuratan penghitungan secara keseluruhan mencapai sekitar 89,3%, yang menunjukkan bahwa sistem ini cukup andal untuk estimasi volume lalu lintas secara otomatis.

Pengujian performa sistem pada perangkat edge computing seperti Jetson Nano atau Raspberry Pi belum dilakukan, dan ini menjadi rencana penelitian selanjutnya. Selain itu, validasi sistem pada data nyata dengan berbagai variasi lingkungan, pemerataan distribusi dataset terutama untuk kelas minoritas, serta penentuan posisi kamera yang optimal sangat disarankan untuk penelitian berikutnya. Dengan dataset yang lebih seimbang dan representatif, sistem berbasis YOLOv8n dan Centroid Tracking berpotensi menjadi solusi efektif untuk monitoring lalu lintas jalan tol di Indonesia.

DAFTAR PUSTAKA

[1] A. P. H. Telaumbanua, T. P. Larosa, P. D. Pratama, R. H. Fauza, and A. M. Husein, "Vehicle Detection and Identification Using Computer Vision Technology with the Utilization of the YOLOv8 Deep Learning Method," *Sinkron*, vol. 8, no. 4, pp. 2150–2157, 2023, doi: 10.33395/sinkron.v8i4.12787.

[2] L. J. Zhang, J. J. Fang, Y. X. Liu, H. Feng Le, Z. Q. Rao, and J. X. Zhao, "CR-YOLOv8: Multiscale Object Detection in Traffic Sign Images," *IEEE Access*, vol. 12, 2024, doi: 10.1109/ACCESS.2023.3347352.

[3] J. Zhang, W. Xiao, B. Coifman, and J. P. Mills, "Vehicle Tracking and Speed Estimation from Roadside Lidar," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 13, pp. 5597–5608, 2020, doi: 10.1109/JSTARS.2020.3024921.

[4] Q. Liu, H. Ye, S. Wang, and Z. Xu, "YOLOv8-CB: Dense Pedestrian Detection Algorithm Based on In-Vehicle Camera," *Electron.*, vol. 13, no. 1, 2024, doi: 10.3390/electronics13010236.

[5] N. Al Mudawi *et al.*, "Vehicle Detection and Classification via YOLOv8 and Deep Belief Network over Aerial Image Sequences," *Sustain.*, vol. 15, no. 19, 2023, doi: 10.3390/su151914597.

[6] H. A. W. A. S. Sutikno, R. Kusumaningrum, "Improved car detection performance on highways based on YOLOv8 _ Sutikno _ Bulletin of Electrical Engineering and Informatics.pdf," *Inst. Adv. Eng. Sci. Publ. Bull. Electr. Eng. Informatics*, vol. 9, 2024, doi: https://doi.org/10.11591/eei.v13i5.8031.

[7] N. U. A. Tahir, Z. Long, Z. Zhang, M. Asim, and M. ELAffendi, "PVswin-YOLOv8s: UAV-Based Pedestrian and Vehicle Detection for Traffic Management in Smart Cities Using Improved YOLOv8," *Drones*, vol. 8, no. 3, pp. 1–20, 2024, doi: 10.3390/drones8030084.

[8] K. Gaur, J. Dhakar, S. Singh, and A. K. Khosla, "Nighttime Rainy Season Traffic Analysis: Vehicle Detection, Tracking, and Counting with YOLOv8 and DeepSORT," *J. Innov. Image Process.*, vol. 5, no. 3, pp. 214–228, 2023, doi: 10.36548/jiip.2023.3.001.

[9] H.-H. Ngo, "Vehicle-detection-based traffic density estimation at road intersections," *Int. J. Open Inf. Technol.*, vol. 11, no. 7, 2023.

[10] T. Sharma *et al.*, "Deep Learning-Based Object Detection and Classification for Autonomous Vehicles in Different Weather Scenarios of Quebec, Canada," *IEEE Access*, vol. 12, no. September 2023, pp. 13648–13662, 2024, doi: 10.1109/ACCESS.2024.3354076.

[11] H. Wang, C. Liu, Y. Cai, L. Chen, and Y. Li, "YOLOv8-QSD: An Improved Small Object Detection Algorithm for Autonomous Vehicles Based on YOLOv8," *IEEE Trans. Instrum. Meas.*, vol. 73, 2024, doi: 10.1109/TIM.2024.3379090.

[12] Y. Li, Q. Fan, H. Huang, Z. Han, and Q. Gu, "A Modified YOLOv8 Detection Network for UAV Aerial Image Recognition," *Drones*, vol. 7, no. 5, 2023, doi: 10.3390/drones7050304.

[13] P. Azevedo and V. Santos, "Comparative analysis of multiple YOLO-based target detectors and trackers for ADAS in edge devices," *Rob. Auton. Syst.*, vol. 171, 2024, doi: 10.1016/j.robot.2023.104558.

[14] H. Dou, S. Chen, F. Xu, Y. Liu, and H. Zhao, "Analysis of vehicle and pedestrian detection effects of improved YOLOv8 model in drone-assisted urban traffic monitoring system," *PLoS One*, vol. 20, no. 3 March, pp. 1–22, 2025, doi: 10.1371/journal.pone.0314817.

[15] R. Bharadwaj, A. Billade, S. Chenna, A. Chandrapatle, and G. Chinchalpalle, "Wrong Way Vehicle Detection in Single and Double Lane," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 11, no. 6s, pp. 457–462, 2023, doi: 10.17762/ijritcc.v11i6s.6953.

[16] Z. Rahman, A. M. Ami, and M. A. Ullah, "A Real-Time Wrong-Way Vehicle Detection Based on YOLO and Centroid Tracking," in *2020 IEEE Region 10 Symposium, TENSYMP 2020*, 2020. doi: 10.1109/TENSYMP50017.2020.9230463.

[17] M. Sukkar, M. Shukla, D. Kumar, V. C. Gerogiannis, A. Kanavos, and B. Acharya, "Enhancing Pedestrian Tracking in Autonomous Vehicles by Using Advanced Deep Learning Techniques," *Inf.*, vol. 15, no. 2, pp. 1–16, 2024, doi: 10.3390/info15020104.

[18] Y. Du, X. Liu, Y. Yi, and K. Wei, "Optimizing Road Safety: Advancements in Lightweight YOLOv8 Models and GhostC2f Design for Real-Time Distracted Driving Detection," *Sensors (Basel).*, vol. 23, no. 21, 2023, doi: 10.3390/s23218844.

[19] D. T. Mane, S. Sangve, S. Kandhare, S. Mohole, S. Sonar, and S. Tupare, "Real-Time Vehicle Accident Recognition from Traffic Video Surveillance using YOLOV8 and OpenCV," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 11, no. May, pp. 250–258, 2023, doi: 10.17762/ijritcc.v11i5s.6651.

[20] D. yuan Ge, X. fan Yao, W. jiang Xiang, and Y. ping Chen, "Vehicle detection and tracking based on video image processing in intelligent transportation system," *Neural Comput. Appl.*, vol. 35, no. 3, 2023, doi: 10.1007/s00521-022-06979-y.

[21] N. Sharma, S. Baral, M. P. Paing, and R. Chawuthai, "Parking Time Violation Tracking Using YOLOv8 and Tracking Algorithms," *Sensors*, vol. 23, no. 13, 2023, doi: 10.3390/s23135843.

[22] H. Yi, B. Liu, B. Zhao, and E. Liu, "Small Object Detection Algorithm Based on Improved YOLOv8 for Remote Sensing," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 17, 2024, doi: 10.1109/JSTARS.2023.3339235.

[23] E. Soylu and T. Soylu, "A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition," *Multimed. Tools Appl.*, vol. 83, no. 8, 2024, doi: 10.1007/s11042-023-16451-1.

[24] X. Wang, H. Gao, Z. Jia, and Z. Li, "BL-YOLOv8: An Improved Road Defect Detection Model Based on YOLOv8," *Sensors (Basel).*, vol. 23, no. 20, 2023, doi: 10.3390/s23208361.

[25] Y. Wu, T. Liao, F. Chen, H. Zeng, S. Ouyang, and J. Guan, "Overhead Power Line Damage Detection: An Innovative Approach Using Enhanced YOLOv8," *Electron.*, vol. 13, no. 4, 2024, doi: 10.3390/electronics13040739.