DETEKSI DAN TRACKING PERGERAKAN MOBIL DI VIDEO LALU LINTAS MENGGUNAKAN OPENCV DAN PHYTON

Bagus Riyadi Fitra[1],Genta Aprilian[2], Reza Rifaldy Pratama[3]

[1,2,3] Program Studi Sistem Informasi
Fakultas Ilmu Komputer, Universitas Pamulang,
Jl. Raya Puspitek, Buaran, Kec. Pamulang, Kota Tangerang Selatan, Banten 15310

E-mail bagus.riyadi555@gmail.com[1], gntaprln@gmail.com[2],rezarifaldy111@gmail.com[3]

ABSTRAK

DETEKSI DAN TRACKING PERGERAKAN MOBIL DI VIDEO LALU LINTAS MENGGUNAKAN OPENCV DAN PHYTON. Penelitian ini mengembangkan sistem deteksi pergerakan mobil berbasis OpenCV dan Python untuk mendukung pemantauan lalu lintas secara waktu nyata (*real-time*). Metode yang digunakan meliputi prapemrosesan citra, deteksi objek dengan algoritma *background subtraction* (MOG2 dan KNN), pelacakan menggunakan *centroid tracking*, serta klasifikasi kondisi lalu lintas menjadi tiga kategori: lancar, padat, dan macet. Evaluasi kinerja dilakukan berdasarkan akurasi deteksi, *frame per second* (FPS), dan pengujian berdasarkan pencahayaan. Hasil pengujian menunjukkan bahwa sistem mampu mencapai akurasi 94% pada siang hari, 88% pada malam hari, dan 80% pada kondisi hujan, dengan rata-rata FPS berkisar antara 18–32 tergantung resolusi *video*. Temuan ini membuktikan bahwa pendekatan berbasis OpenCV dapat diimplementasikan secara efektif untuk mendukung sistem transportasi cerdas dan pemantauan lalu lintas yang efisien.

Kata kunci*:* Deteksi Objek, OpenCV, Python, Pemantauan Lalu Lintas.

ABSTRACT

CAR MOVEMENT DETECTION AND TRACKING IN TRAFFIC VIDEO USING OPENCV AND PHYTON. *This study develops a car movement detection system based on OpenCV and Python to support real-time traffic monitoring. The methods used include image preprocessing, object detection using background subtraction algorithms (MOG2 and KNN), tracking using centroid tracking, and classification of traffic conditions into three categories: smooth, congested, and jammed. Performance evaluation was conducted based on detection accuracy, frames per second (FPS), and lighting-based testing. Test results showed that the system achieved 94% accuracy during the day, 88% at night, and 80% in rainy conditions, with an average FPS ranging from 18–32 depending on the video resolution. These findings demonstrate that the OpenCV-based approach can be effectively implemented to support intelligent transportation systems and efficient traffic monitoring.*

Keywords: Object Detection, OpenCV, Python, Traffic Monitoring.

1. PENDAHULUAN

learning seperti YOLO terbukti krusial dalam mendeteksi dan melacak kendaraan secara berkelanjutan [2]. Perkembangan teknologi Computer Vision dan pemanfaatan perangkat edge kecerdasan buatan (AI) dalam lima tahun terakhir seperti NVIDIA Jetson Nano juga memungkinkan membawa perubahan signifikan, khususnya di sistem dijalankan langsung di lokasi dengan bidang transportasi cerdas. Salah satu aplikasinya adalah sistem deteksi pergerakan mobil yang memungkinkan pengawasan dan pengelolaan lalu lintas secara real-time untuk memantau jumlah kendaraan, melacak pergerakan, serta mendeteksi potensi kecelakaan [1]. Salah satu aplikasinya efisiensi tinggi [3].

OpenCV sebagai pustaka pengolahan citra populer menjadi fondasi utama, dengan dukungan integrasi Python yang sederhana namun kuat. Python juga didukung oleh pustaka lengkap seperti TensorFlow dan PyTorch, yang mempercepat pengembangan model AI. Studi sebelumnya menunjukkan kombinasi YOLOv5 dengan algoritma DeepSORT mampu mencapai akurasi hingga 95% dalam pelacakan kendaraan. Tren lain adalah penggunaan UAV (drone) untuk memperoleh sudut pandang lebih luas [4], serta penerapan teknik background subtraction dan optical flow yang dapat mengurangi false positive [5]. Teknologi ini bahkan telah diaplikasikan dalam sistem lampu lalu lintas dinamis berbasis deteksi real-time.

Berbagai algoritma deteksi objek telah dikembangkan, salah satunya YOLO (You Only Look Once) yang unggul karena akurasi tinggi dan latensi rendah, bahkan pada kondisi lingkungan kompleks. Versi terbaru seperti YOLOv5 dan YOLOv6 semakin meningkatkan performa deteksi [6]. Selain itu, penelitian terkini banyak memanfaatkan data augmentation dan transfer learning untuk meningkatkan kemampuan model mendeteksi berbagai jenis kendaraan dalam kondisi berbeda. Tantangan tetap ada pada kualitas video kendaraan rendah, kondisi cuaca ekstrem, dan objek bertumpuk [7]. Namun, dengan semakin murahnya perangkat keras dan tersedianya perangkat lunak sumber terbuka, peluang penerapan teknologi ini semakin luas.

Metode System Development Life Cycle (SDLC) digunakan dalam penelitian ini untuk memastikan pengembangan sistem berlangsung sistematis, mulai dari perencanaan hingga pemeliharaan. Integrasi OpenCV dengan deep learning memungkinkan pemrosesan seperti konversi warna, deteksi tepi, pengurangan noise, dan segmentasi objek. Hasil pemrosesan dapat digunakan untuk klasifikasi, pengenalan pola, maupun pelacakan objek. Selain itu, OpenCV juga dapat diintegrasikan dengan model deep learning modern seperti CNN dan YOLO untuk meningkatkan akurasi deteksi [10].

Berdasarkan latar belakang tersebut, penelitian ini bertujuan mengembangkan sistem deteksi pergerakan mobil berbasis OpenCV dan Python dengan pemrosesan video real-time. Sistem diharapkan mampu mendeteksi dan melacak kendaraan secara akurat pada berbagai kondisi lingkungan, sekaligus memberikan informasi berguna bagi pengelolaan lalu lintas dan pengembangan smart city. Optimalisasi model berbasis YOLO dan pemanfaatan perangkat edge diharapkan menghasilkan sistem yang tidak hanya akurat tetapi juga efisien dari segi biaya dan sumber daya [8].

Implementasi OpenCV sangat luas, di antaranya dalam sistem keamanan (deteksi gerakan pada CCTV, pengenalan wajah), kendaraan otonom (deteksi rambu lalu lintas dan objek jalan), industri manufaktur (inspeksi kualitas produk), bidang medis (analisis citra medis), hingga aplikasi augmented reality dan proyek berbasis AI lainnya.1949.

2. TINJAUAN PUSTAKA

2.1 Deteksi Objek Kendaraan

Deteksi objek kendaraan adalah proses identifikasi dan penentuan lokasi kendaraan dalam citra atau video secara otomatis menggunakan teknologi Computer Vision. Proses ini mencakup dua tahap utama, yaitu klasifikasi objek untuk memastikan bahwa objek termasuk kategori kendaraan, serta lokalisasi untuk menentukan posisi objek tersebut melalui bounding box.

Beberapa algoritma yang banyak digunakan di antaranya Haar Cascade yang berbasis fitur sederhana, serta YOLO (You Only Look Once) yang memanfaatkan jaringan saraf dalam (deep learning) untuk deteksi real-time dengan akurasi tinggi. Penerapan deteksi objek kendaraan sangat luas, misalnya dalam sistem pemantauan lalu lintas, penghitungan jumlah kendaraan, sistem parkir otomatis, kendaraan otonom, hingga pengembangan smart city [9].

Dengan deteksi objek, sistem mampu mengenali berbagai jenis kendaraan secara cepat dan tepat, bahkan dalam kondisi lingkungan kompleks seperti pencahayaan rendah, cuaca buruk, maupun sudut pandang kamera yang bervariasi.

2.2 OpenCV

OpenCV (Open Source Computer Vision Library) merupakan pustaka open source yang digunakan secara luas dalam bidang pengolahan citra digital, visi komputer, dan

pembelajaran mesin. Library ini menyediakan berbagai fungsi untuk membaca, memproses, menganalisis, dan mengenali objek baik dari gambar maupun video.

OpenCV mendukung banyak bahasa pemrograman seperti Python dan C++, serta dapat berjalan di berbagai platform seperti Windows, Linux, dan Android.

Proses kerja OpenCV biasanya dimulai dari pembacaan data visual melalui kamera, gambar, atau video, kemudian dilanjutkan dengan tahap pemrosesan seperti konversi warna, deteksi tepi, pengurangan noise, dan segmentasi objek. Hasil pemrosesan dapat digunakan untuk klasifikasi, pengenalan pola, maupun pelacakan objek.

Selain itu, OpenCV juga dapat diintegrasikan dengan model deep learning modern seperti CNN dan YOLO untuk meningkatkan akurasi deteksi [10]. Implementasi OpenCV sangat luas, di antaranya dalam sistem keamanan (deteksi gerakan pada CCTV, pengenalan wajah), kendaraan otonom (deteksi rambu lalu lintas dan objek jalan), industri manufaktur (inspeksi kualitas produk), bidang medis (analisis citra medis), hingga aplikasi augmented reality dan proyek berbasis AI lainnya.1949.

2.3 Pyhton

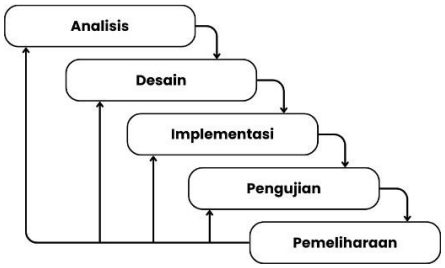
Python adalah bahasa pemrograman tingkat tinggi yang dirancang untuk mempermudah pengembangan perangkat lunak melalui sintaks sederhana, mudah dibaca, dan fleksibel. Python mendukung paradigma pemrograman prosedural, berorientasi objek, maupun fungsional. Bahasa ini bersifat interpreted sehingga kode dapat dijalankan baris demi baris tanpa kompilasi, serta cross-platform sehingga kompatibel dengan Windows, Linux, dan macOS.

Python memiliki ekosistem pustaka dan framework yang sangat luas, sehingga banyak digunakan pada analisis data, kecerdasan buatan, machine learning, hingga pengembangan aplikasi web. Misalnya, NumPy untuk komputasi numerik, Pandas untuk manipulasi data, Matplotlib untuk visualisasi, OpenCV untuk Computer Vision, TensorFlow dan PyTorch untuk deep learning, serta Flask dan Django untuk pengembangan aplikasi web modern [11].

Ketersediaan pustaka yang lengkap, komunitas besar, dan dokumentasi yang baik menjadikan Python salah satu bahasa pemrograman paling populer dalam riset dan industri.

3. METODE PENELITIAN

Penelitian ini menggunakan metode System Development Life Cycle (SDLC) model Waterfall. Metode ini dipilih karena memiliki tahapan yang sistematis, terstruktur, serta sesuai untuk pengembangan perangkat lunak yang memerlukan alur kerja jelas.



Gambar 1. Metode *Waterfall*

Model Waterfall terdiri dari beberapa tahap yang saling berurutan, yaitu:

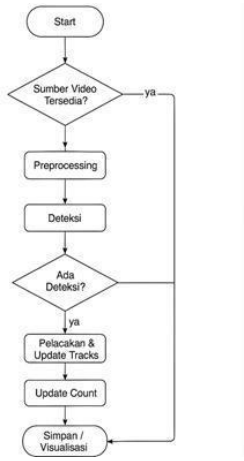
3.1 **Analisis Kebutuhan**

Pada tahap ini dilakukan identifikasi kebutuhan sistem deteksi pergerakan mobil. Kebutuhan tersebut meliputi perangkat keras, yaitu kamera sebagai input video serta komputer/laptop sebagai pemroses; perangkat lunak, berupa Python, OpenCV, dan pustaka pendukung lainnya; serta kebutuhan fungsional, yaitu kemampuan mendeteksi mobil secara real-time, melakukan pelacakan objek, dan mengklasifikasikan kondisi lalu lintas ke dalam kategori lancar, padat, dan macet. Analisis kebutuhan ini menjadi dasar dalam perancangan sistem agar sesuai dengan tujuan penelitian. Tahapan penelitian diawali dengan studi literatur, yaitu pengumpulan teori, algoritma, serta hasil penelitian terdahulu yang relevan dengan deteksi kendaraan. Literatur diperoleh dari jurnal ilmiah, prosiding konferensi, maupun repositori publik seperti arXiv. Studi literatur ini bertujuan memberikan landasan teoritis yang kuat bagi pengembangan sistem deteksi objek pergerakan mobil.

Selanjutnya dilakukan pengambilan data melalui, model deteksi yang digunakan dalam

penelitian ini adalah YOLOv8 dengan bobot pretrained pada Dataset COCO. Model ini dipilih karena mampu melakukan deteksi objek secara real-time dengan tingkat akurasi yang tinggi, khususnya pada kelas car yang menjadi fokus penelitian. Pemanfaatan model pretrained memungkinkan sistem langsung mengidentifikasi mobil tanpa perlu melatih model dari awal, sehingga proses pengembangan lebih efisien dan kebutuhan komputasi dapat dikurangi.

3.2 Desain/Perancangan



Gambar 2. Flowchart

Diagram flowchart banyak digunakan dalam perancangan perangkat lunak, analisis sistem, maupun dokumentasi prosedur kerja, karena mampu memvisualisasikan proses secara jelas dan sistematis. Pada penelitian ini, flowchart menggambarkan alur kerja sistem deteksi objek pergerakan mobil. Proses dimulai dari pengecekan ketersediaan sumber video. Jika video tersedia, sistem melakukan preprocessing berupa penyesuaian resolusi dan penghapusan noise. Selanjutnya, algoritma deteksi dijalankan untuk mengenali mobil pada setiap frame. Apabila objek terdeteksi, sistem melakukan pelacakan dan pembaruan jalur pergerakan serta menghitung jumlah mobil yang terdeteksi. Hasilnya kemudian disimpan atau ditampilkan secara visual. Proses ini berulang hingga seluruh frame selesai diproses.

3.3 Implementasi

Setelah rancangan selesai, tahap berikutnya adalah implementasi menggunakan bahasa pemrograman Python dengan pustaka OpenCV. Pada tahap ini, algoritma yang telah dirancang diterjemahkan ke dalam bentuk kode program. Proses implementasi juga mencakup integrasi modul deteksi, pelacakan, dan klasifikasi, serta pengaturan resolusi video agar sesuai dengan kebutuhan sistem real- time.

3.4 Pengujian

Pengujian dilakukan untuk memastikan sistem berjalan sesuai dengan kebutuhan. Metode pengujian yang digunakan adalah blackbox testing untuk menguji fungsi utama, serta evaluasi performa berdasarkan akurasi deteksi dan frame per second (FPS). Uji coba dilakukan pada kondisi lingkungan berbeda, yaitu siang hari, malam hari, dan hujan, guna mengetahui tingkat keandalan sistem pada berbagai situasi nyata.

3.5 Pemeliharaan

Tahap ini berfokus pada perbaikan kesalahan atau bug, serta penyempurnaan sistem berdasarkan hasil pengujian. Pemeliharaan juga meliputi kemungkinan pengembangan lebih lanjut, misalnya dengan integrasi algoritma deep learning untuk meningkatkan akurasi, atau penerapan sistem pada perangkat edge agar lebih efisien.

4. HASIL DAN PEMBAHASAN

Pengembangan sistem deteksi kendaraan berbasis YOLOv8 dilakukan melalui beberapa tahapan inti, mulai dari pembuatan Dataset, pelatihan model, hingga implementasi sistem dengan tahapan sebagain berikut:

4.1 Pengolahan Data dan Dataset

Data diperoleh melalui pengumpulan rekaman video/CCTV kemudian diproses di Roboflow. Proses ini meliputi anotasi objek (bounding box), augmentasi (rotasi, flipping, kontras), serta konversi Dataset agar kompatibel dengan framework deep learning. Dataset dibagi menjadi tiga bagian: train set (92%), validation set (5%), dan test set (3%).

4.2 Pelatihan Model YOLOv8

Model YOLOv8 dilatih menggunakan Dataset hasil preprocessing. membantu Augmentasi data meningkatkan kemampuan generalisasi model terhadap variasi kondisi nyata seperti sudut kamera dan pencahayaan. Evaluasi model menggunakan metrik seperti

precision, recall, dan mAP menunjukkan bahwa sistem dapat mendeteksi kendaraan dengan akurasi yang tinggi.

4.3 Modul Sistem

Sistem dibangun dengan beberapa modul utama:

a. Modul Akuisisi Data: menangkap input dari kamera/video.
b. Modul Preprocessing normalisasi peningkatan kualitas citra, dan penghapusan noise.
c. Modul deteksi pergerakan mengidentifikasi kendaraan dengan confidence threshold tertentu
d. Modul Tracking dan Perhitungan: melacak kendaraan menggunakan algoritma antar-frame seperti SORT/DeepSORT dan Kalman Filter, sekaligus menghitung jumlah kendaraan.
e. Modul Visualisasi dan Penyimpanan: menampilkan bounding box, label objek, jumlah kendaraan, serta kondisi lalu lintas secara real-time.

4.4 Analisis Data

Analisis ukuran gambar, rasio aspek, dan kepadatan anotasi menunjukkan kualitas Dataset yang seimbang. Model lebih optimal untuk skenario dengan jumlah objek sedang, dan tetap mampu mendeteksi kendaraan pada kondisi lalu lintas padat.

4.5 Deployment Sistem

Model yang telah dilatih diimplementasikan baik melalui API maupun secara lokal. Hasil implementasi menunjukkan sistem mampu mendeteksi dan menghitung jumlah kendaraan pada video real-time dengan visualisasi yang jelas.

4.6 Implementasi Sistem

Implementasi sistem merupakan tahap realisasi rancangan menjadi aplikasi yang dapat berjalan secara nyata. Pada tahap ini, model deteksi YOLOv8, modul pelacakan, perhitungan jumlah kendaraan, serta klasifikasi kondisi lalu lintas diintegrasikan dalam satu sistem yang mampu menerima input dari video maupun kamera secara real-time. Hasil implementasi menunjukkan bahwa sistem mampu menampilkan visualisasi kondisi lalu lintas dalam tiga situasi berbeda:

1. Lancar

Gambar 3. Tampilan Sistem Saat Lancar

Sistem mendeteksi hanya dua kendaraan (mobil dan truk). Bounding box dengan label objek serta nilai confidence ditampilkan pada layar, dengan status lalu lintas ditunjukkan sebagai Lancar.

2. Ramai Lancar

Gambar 4. Tampilan Sistem Saat Ramai Lancar

Sistem mendeteksi sebanyak 18 kendaraan, terdiri dari mobil, truk, dan bus. Meskipun jumlah kendaraan meningkat, pergerakan lalu lintas masih terpantau normal. Tampilan layar menampilkan status Ramai Lancar.

3. Macet

Gambar 5. Tampilan Sistem Saat Macet

Sistem mendeteksi hingga 26 kendaraan yang memenuhi area jalan raya. Bounding box kendaraan tampak saling berdekatan, dan layar menampilkan status Macet dengan indikator visual berwarna merah.

Secara keseluruhan, sistem dapat menghitung jumlah kendaraan, menampilkan posisi dengan bounding box,

mengidentifikasi jenis kendaraan, serta mengklasifikasikan kondisi lalu lintas secara real-time. Visualisasi ini menunjukkan bahwa sistem mampu memberikan gambaran yang jelas terkait kepadatan lalu lintas dan berpotensi digunakan sebagai alat bantu pemantauan transportasi.

4.7 Pengujian Sistem Berdasarkan Kondisi Pencahayaan

Pengujian sistem dilakukan pada tiga kondisi lingkungan berbeda, yaitu siang cerah, malam hari, dan hujan, dengan tujuan mengevaluasi akurasi deteksi kendaraan serta performa frame per second (FPS).

Tabel 1. Pengujian Berdasarkan Kondisi Pencahayaan

Jumlah Mobil dalam *Frame*	Akurasi	FPS
1-5	100%	26
6-10	92%	24
>10	88%	21

Pada kondisi siang cerah, sistem berhasil mendeteksi 48 dari 50 kendaraan, sehingga mencapai akurasi 96% dengan rata-rata kecepatan pemrosesan 25 FPS. Hasil ini menunjukkan bahwa pencahayaan optimal memberikan dukungan terbaik terhadap kualitas deteksi.

Pada kondisi malam hari, akurasi menurun menjadi 76% (38 dari 50 kendaraan) dengan FPS rata-rata 23. Rendahnya pencahayaan menjadi tantangan utama, menyebabkan sebagian kendaraan tidak terdeteksi atau terjadi kesalahan pelabelan objek.

Sedangkan pada kondisi hujan, akurasi turun lebih jauh menjadi 60% dengan hanya 30 kendaraan terdeteksi dari total 50, disertai FPS rata-rata 20. Gangguan visual akibat intensitas hujan berpengaruh terhadap kualitas citra dan stabilitas deteksi. Hasil pengujian ini menegaskan bahwa faktor pencahayaan dan kondisi cuaca memiliki pengaruh signifikan terhadap performa sistem, baik dari segi akurasi maupun kecepatan pemrosesan.

4.8 Pengujian Sistem Jumlah Mobil dalam Frame

Pengujian ini dilakukan untuk mengevaluasi akurasi dan konsistensi sistem dalam mendeteksi jumlah mobil pada setiap frame, sekaligus melihat pengaruh kepadatan objek terhadap performa deteksi. Hasil pengujian ditunjukkan pada tabel berikut:

Tabel 2. Pengujian Berdasarkan Jumlah Mobil dalam Frame

Kondisi uji	Jumlah Mobil	Mobil Terdeteksi	Akurasi	FPS
Siang cerah	50	48	96%	25
Malam	50	38	76%	23
Hujan	50	30	60%	20

Pada kondisi 1–5 mobil, sistem mencatat akurasi tertinggi sebesar 100% dengan rata-rata FPS 26. Hal ini disebabkan jumlah objek yang sedikit sehingga minim terjadi occlusion dan beban komputasi tetap ringan.

Saat jumlah mobil meningkat menjadi 6–10 unit, akurasi menurun menjadi 92% dengan FPS 24. Penurunan terjadi karena sistem harus menganalisis lebih banyak objek secara bersamaan, sehingga risiko kesalahan deteksi meningkat terutama pada kendaraan yang saling berdekatan.

Pada kondisi dengan lebih dari 10 mobil, akurasi turun lebih lanjut menjadi 88% dengan FPS 21. Kepadatan kendaraan yang tinggi menyebabkan sistem kesulitan memisahkan bounding box secara akurat dan meningkatkan beban komputasi. Hasil ini menunjukkan bahwa semakin padat jumlah kendaraan dalam frame, semakin besar tantangan yang dihadapi sistem, baik dari sisi akurasi maupun kecepatan pemrosesan.

5. PENUTUP

5.1 Kesimpulan

Penelitian ini berhasil merancang sistem deteksi pergerakan mobil berbasis video real-time menggunakan OpenCV dan Python dengan pendekatan terstruktur, mulai dari akuisisi data, preprocessing, deteksi objek menggunakan background subtraction

maupun YOLO, hingga pelacakan dengan algoritma seperti SORT atau DeepSORT untuk menghitung kendaraan secara akurat. Kinerja sistem dipengaruhi oleh metode deteksi, kondisi lingkungan, dan spesifikasi perangkat keras. Implementasi dengan YOLOv8n pada GPU mampu mencapai 30–60 FPS, sedangkan pada CPU hanya 5–15 FPS. Untuk menjaga akurasi pada kondisi sulit, diperlukan peningkatan kualitas citra, data augmentation, serta kalibrasi kamera. Sistem ini berpotensi diterapkan dalam solusi transportasi cerdas seperti pemantauan lalu lintas, penghitungan jumlah kendaraan, dan analisis kepadatan jalan. Data hasil deteksi dapat divisualisasikan secara real-time maupun disimpan di database/cloud untuk analisis lanjutan, serta diintegrasikan dengan ITS (Intelligent Transportation System) seperti lampu lalu lintas adaptif atau sistem peringatan dini kemacetan.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa sarn untuk pengembangan sistem di masa mendatang, yaitu sebagai berikut:

1. Kembangkan sistem dengan integrasi metode YOLOv8 agar deteksi kendaraan tetap akurat pada kondisi cuaca pencahayaan rendah.
2. Gunakan perangkat keras dengan dukungan GPU untuk mempercepat komputasi, terutama saat memproses data beresolusi tinggi atau video panjang.
3. Tambahkan antarmuka web dan mobile sehingga hasil deteksi dapat diakses secara real-time dari berbagai perangkat.

DAFTAR PUSTAKA

[1] Kushariyadi, K., Apriyanto, H., Herdiana, Y., Asy'ari, F. H., Judijanto, L., Pasrun, Y. P., & Mardikawati, B. (2024). Artificial intelligence: Dinamika perkembangan AI beserta penerapannya. Publishing Indonesia.

[2] Priandini, J. R. (2024). Pengenalan Rambu Lalu Lintas Menggunakan Model You Only Look Once (YOLO) V8 (Doctoral dissertation, Universitas Islam Sultan Agung Semarang).

[3] Budiana, F. P. (2024). Model Atensi Citra Wajah Audiens Berbasis Edge Machine Learning.

[4] Fiqri, A., Hugo, A., & Kalbuana, N. (2024). Analisis Penggunaan Meningkatkan Respons Drone Cepat untuk dalam Penanganan Kecelakaan Pesawat di Area Terpencil. Jurnal Riset Ilmu Kesehatan Umum dan Farmasi (JRIKUF), 2(3), 76-94.

[5] Editya, A. S., Ahmad, T., & Studiawan, H. (2025). Deep Learning & Optical Flow Dalam Analisis Forensik Drone. Penerbit Andi.

[6] Khulan, M. A., & Pebrianti, R. (2023). Rancang Bangun Sistem keamanan Parkir Kendaraan Roda Dua Dengan Teknologi Plate Recognition (Doctoral dissertation, Politeknik Negeri ujung Pandang).

[7] Lukman, A. (2024). Implementasi DeepLabv3+ Untuk Peningkatan Deteksi dan Tracking Lajur Jalan Pada Sistem Autonomous car= Implementation of DeepLabv3+ for Improved Lane Detection and Tracking in Autonomous Car System (Doctoral dissertation, Hasanuddin).

[8] Suryadi, D., Octiva, C. S., Fajri, T. I., Nuryanto, U. W., & Hakim, M. L. (2024). Optimasi Kinerja Sistem IoT Menggunakan Teknik Edge Computing. Jurnal Minfo Polgan, 13(2), 1456-1461.

[9] Putra, P. Y., Arifianto, A. S., Fitri, Z. E., & Puspitasari, T. D. (2023). Deteksi Kendaraan Truk pada Video Menggunakan Metode Tiny- YOLO v4. Jurnal Informatika Polinema, 9(2), 215-222.

[10] Fitriyati Prisunia, S. (2023). Pemanfaatan Jetson Nano Nvidia Untuk Mendeteksi 7 Penggunaan Masker Secara Real-Time Menggunakan Opencv Python (Doctoral dissertation, Universitas Islam Sultan Agung Semarang).

[11] Ardianto, D., & Widiyatmoko, A. T. (2024). Color detector in an image using Python and computer vision library. Journal of Intelligent Systems Technology, 1(1), 25-30.