Jurnal Teknik Mesin S-1, Vol. 12, No. 3, Tahun 2024
Online: https://ejournal3.undip.ac.id/index.php/jtm

PENGEMBANGAN MODEL *TIMING* LAMPU LALU LINTAS PADA PEREMPATAN JALAN MENGGUNAKAN *DEEP LEARNING* DAN *COMPUTER VISION*

***Nachel Hanif Syauqi[1], M. Munadi[2], Joga Dharma Setiawan[2]**
[1]Mahasiswa Jurusan Teknik Mesin, Fakultas Teknik, Universitas Diponegoro
[2]Dosen Jurusan Teknik Mesin, Fakultas Teknik, Universitas Diponegoro
Jl. Prof. Sudharto, SH., Tembalang-Semarang 50275, Telp. +62247460059
*E-mail: nachelhanif@gmail.com

Abstrak

Kemacetan merupakan masalah yang menjadi penghambat masyarakat dalam melakukan mobilitas yang cepat dan efektif, yang disebabkan oleh beberapa faktor. Salah satu dari faktor kemacetan yaitu pewaktuan lampu lalu lintas yang tidak berdasarkan kepadatan arus lalu lintas secara waktu nyata. Penelitian ini bertujuan untuk memberikan solusi untuk kemacetan dengan memanfaatkan *Deep Learning* dalam proses penentuan waktu lampu hijau lalu lintas berdasarkan jumlah kendaraan yang terdeteksi dan klasifikasinya. Penelitian ini menggunakan metode deteksi objek dengan algoritma YOLOv8-large yang dilatih menggunakan Google Colaboratory sebanyak 50 epoch untuk menghitung waktu nyala lampu hijau secara langsung berdasarkan jumlah kendaraan yang terdeteksi. Sistem yang digunakan yaitu Laptop HP Pavilion dk-1041tx dengan GPU NVIDIA GTX 1650 yang dihubungkan dengan webcam Xiaovv untuk mendeteksi objek berupa diecast kendaraan. Terdapat 3 jenis pengujian yang dilakukan, yaitu pengujian gambar, video, dan alat peraga. Alat peraga dibuat menyerupai perempatan jalan dengan mobil diecast sebagai objek yang digunakan untuk menghitung waktu hijau lampu lalu lintas. Hasil training model YOLOv8-large menunjukkan nilai MAP50 sebesar 0,977 dan MAP50-95 sebesar 0,79. Pada pengujian gambar, performa deteksi berbeda pada gambar yang bervariasi. Hasil deteksi lebih baik berjalan pada suasana jalan raya pada siang hari, dan model mengalami kesulitan deteksi pada sampel gambar dengan latar parkiran mobil dengan warna mobil yang lebih gelap. Pengujian pada video menunjukkan bahwa objek yang terdeteksi dapat dihitung di dalam zona penghitungan, dan dapat diolah dalam wujud waktu nyala lampu lalu lintas. Pengujian menggunakan webcam dan alat peraga menunjukkan bahwa deteksi objek pada zona penghitungan juga dapat dilakukan secara langsung pada *runtime* lokal.
Kata kunci: *computer vision*; *deep learning*; kecerdasan buatan; lampu lalu lintas; yolov8

Abstract

Congestion is a problem that hinders people from carrying out fast and effective mobility, which is caused by several factors. One of the factors of congestion is the timing of traffic lights which is not based on real-time traffic density. This research aims to provide a solution to traffic jams by utilizing Deep Learning in the process of determining the timing of green traffic lights based on the number of vehicles detected and their classification. This research uses an object detection method with the YOLOv8-large algorithm which was trained using Google Colaboratory for 35 epochs to calculate the green light on time directly based on the number of vehicles detected. The system used is an HP Pavilion dk-1041tx laptop with an NVIDIA GTX 1650 GPU connected to a Xiaovv webcam to detect objects in the form of vehicle diecasts. There are 3 types of testing carried out, namely image, video and visual aid testing. The props are made to resemble a crossroads with diecast cars as objects used to calculate the green time for traffic lights. The training results of the YOLOv8-large model show a MAP50 value of 0.977 and MAP50-95 of 0.79. In image testing, detection performance is different on various images. Detection results are better when running on a highway during the day, and the model has difficulty detecting image samples with a car parking background with darker car colors. Tests on the video show that the detected objects can be counted within the calculation zone, and can be processed in the form of traffic light times. Tests using webcams and props show that detection of objects in the calculation zone can also be done directly at local runtime.
***Keywords**: artificial intelligence; computer vision; deep learning; traffic light; yolov8*

1. Pendahuluan

Kemacetan merupakan salah satu fenomena lalu lintas di mana arus lalu lintas terhambat dikarenakan beberapa faktor. Seperti banyaknya volume kendaraan, *timing* lampu hijau yang kurang efisien, dan bentuk persimpangan. Sejauh ini di Indonesia, lampu lalu lintas menggunakan *timer* untuk mengendalikan laju arus lalu lintas. Akan tetapi, waktu hijau yang diberikan tidak mampu menyesuaikan dengan volume kendaraan sehingga tidak menyelesaikan kemacetan.

Mayoritas lampu lalu lintas saat ini menggunakan sistem timer yang dikendalikan melalui pusat kontrol dengan alat PLC. PLC ini telah diprogram untuk memberikan waktu sekian detik pada fase lampu merah, kuning, dan hijau dengan mengamati kepadatan pada ruas jalan, kemudian waktu di-setting sesuai perkiraan kepadatan jalan. Akan tetapi, kerugian utama dari hal ini adalah pengatur waktu hanya dapat digunakan ketika lampu lalu lintas beroperasi dengan waktu yang sudah tetap (*fixed-time*), sehingga mengurangi fleksibilitas. Hal ini karena lama nyala lampu hijau atau merah tidak diketahui secara pasti.

Terdapat banyak parameter waktu sinyal yang mempengaruhi efisiensi persimpangan termasuk panjang siklus, waktu

hijau pergerakan, dan interval jarak bebas. Meningkatkan waktu hijau suatu jalur lalu lintas dapat mengurangi penundaan dan jumlah kendaraan yang berhenti. Namun, mengorbankan peningkatan penundaan dan pemberhentian pada jalur lainnya. Oleh karena itu, diperlukan pengaturan waktu sinyal yang baik berdasarkan permintaan di persimpangan dan menjaga panjang siklus seminimal mungkin [1].

Sebagai solusi dari permasalahan tersebut, PLC dapat dioptimasikan melalui integrasi dengan kamera untuk mengamati kepadatan lalu lintas dan artificial intelligence untuk mengolah data tersebut menjadi output berupa lama nyala lampu merah atau hijau. Kamera akan mengambil data berupa gambar kendaraan yang melintas pada ruas jalan dan data tersebut dikumpulkan untuk diolah dengan *Deep Learning*, salah satu metode AI yang bertujuan membuat prediksi berdasarkan data yang terkumpul. *Deep Learning* merupakan cakupan *Machine Learning* di mana sebuah arsitektur yang terdiri dari beberapa lapisan tersembunyi untuk mengenali beberapa pola dari fitur yang berbeda dengan abstraksi yang tinggi [2]. Algoritma *Deep Learning* mengenali pola yang terdapat pada kumpulan data dan membuat prediksi berdasarkan pola tersebut. Beberapa penelitian *Deep Learning* telah dibuat terkait dengan *timing* lampu lalu lintas, seperti algoritma EP-D3QN [3] dan optimasi siklus lalu lintas pada perempatan dengan simulasi Pygame [4].

Deep Learning, bersama dengan *Computer Vision*, dapat diaplikasikan dalam bentuk deteksi objek untuk mengoptimalkan pengaturan waktu hijau pada lampu lalu lintas. Sebuah algoritma/model *Deep Learning* yang sudah dilatih dengan dataset kendaraan akan mendeksi objek kendaraan secara *real time* dan memberikan output berupa lama waktu hijau berdasarkan jumlah kendaraan yang terdeteksi. Dengan memasukkan model tersebut ke dalam sistem kamera, lampu lalu lintas dapat diatur dengan waktu hijau tersebut.

Secara umum, jaringan saraf yang menyusun algoritma *Deep Learning* terbagi menjadi tiga lapisan, yaitu lapisan input, output, dan *hidden layer* [5]. Algoritma *Deep Learning* memiliki beberapa bobot yang ditentukan yang dapat memengaruhi hasilnya, diantaranya yaitu *learning rate*, yang merupakan perbaikan untuk mengurangi *error*. Propagasi balik memungkinkan informasi dari fungsi biaya kemudian mengalir mundur melalui jaringan, untuk menghitung gradien [6]. Berbeda dengan *learning rate*, propagasi balik mengatur bobot berdasarkan *error*. Jika model yang sudah dilatih memiliki akurasi lebih tinggi daripada validasinya, maka model tersebut mengalami *overfitting*, di mana model *overfitted* cenderung menghafal semua data, termasuk gangguan yang tidak dapat dihindari pada set training [7]. Performa *training* dari suatu model ditunjukkan dalam matriks konfusi, yaitu matriks yang membandingkan hasil prediksi model dengan label aktualnya. Beberapa ukuran performa, termasuk akurasi, presisi, perolehan, dan F-measure, dapat dihitung dengan memeriksa nilai-nilai dalam matriks konfusi untuk mengevaluasi efektivitas model [8]. Matriks ini juga membantu untuk mengidentifikasi apakah model mengalami kesalahan pelabelan atau kesulitan dalam mengenali pola yang terdapat pada *dataset* [9].

Model YOLOv8 merupakan versi terbaru dari Ultralytics YOLO [10]. Model ini memiliki beberapa fitur terbaru, seperti augmentasi data mosaik, deteksi *anchor-free*, modul C2f, *decoupled head*, dan fungsi kerugian DFL serta CIoU. Augmentasi data mosaik menggunakan empat foto untuk mosaik untuk membuat gambar komposit menggunakan empat gambar asli, sehingga meningkatkan efektivitas training model. Selain itu, teknik augmentasi data mosaik menawarkan opsi yang berubah bergantung pada berapa banyak objek dalam satu gambar sumber [11]. Deteksi *anchor-free* memungkinkan model meningkatkan generalisasi dengan mengurangi *hyperparameter* [12]. Modul C2f menggunakan basis modul C2 dengan fungsi seperti modul C3 [13], sehingga menurunkan biaya komputasi dan mempercepat proses *training*. *Decoupled Head* mencegah konflik antara tugas klasifikasi dan regresi dengan *head* yang terpisah [14].

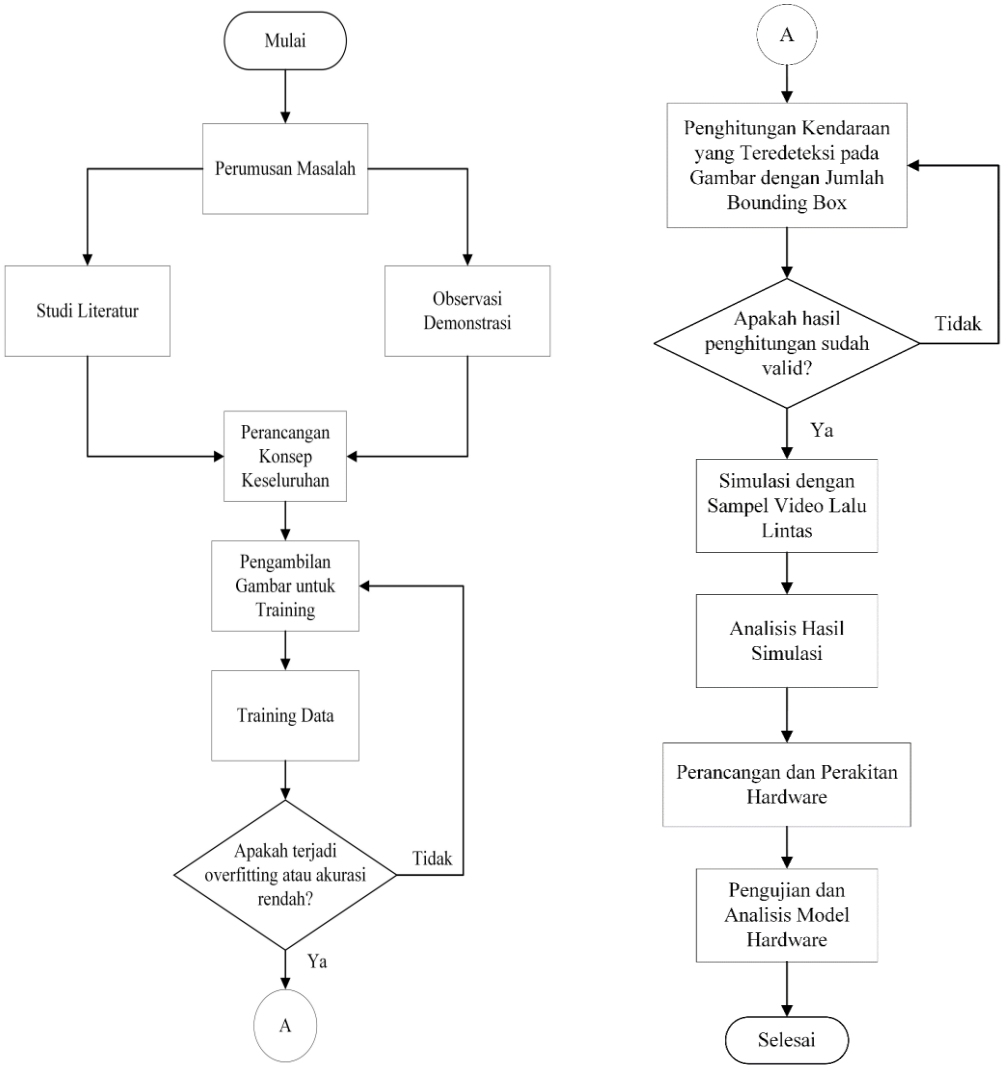
Model YOLOv8-large digunakan sebagai algoritma deteksi objek pada penelitian ini. Dataset yang digunakan berasal dari Kaggle [15] untuk melatih model hingga mendapatkan parameter yang baik. Model dijalankan pada *runtime* lokal pada Google Colaboratory melalui laptop untuk melakukan deteksi secara langsung dengan *webcam*.

Penulis berharap bahwa penelitian ini dapat dikembangkan lebih lanjut dan bermanfaat untuk mengurangi kepadatan lalu lintas di Indonesia yang salah satu faktornya yaitu sistem lampu lalu lintas berbasis *fixed time*. Selain itu, penulis juga berharap bahwa penelitian ini dapat menjadi *support* dalam proses otomatisasi sistem pengaturan lalu lintas dengan menggunakan AI.

2. Material dan Metode Penelitian

2.1 Diagram Alir

Pada penelitian ini terdapat diagram alir yang ditujukan untuk menggambarkan proses perancangan dan pemecahan masalah dari awal sampai akhir berdasarkan seluruh rangkaian proyek Tugas Akhir, diagram alir pemecahan masalah dapat dilihat pada Gambar 1.



Gambar 1. Diagram alir penelitian

Berdasarkan diagram alir, penelitian Deep Learning untuk Traffic Light control dimulai dengan mempelajari literatur, model yang sudah ada, software, dan bahasa pemrograman yang diperlukan. Setelah sistem dirancang, dilakukan pengujian dengan sampel video lalu lintas. Pembuatan model hardware dilakukan setelah hasil pengujian memenuhi rumusan masalah.

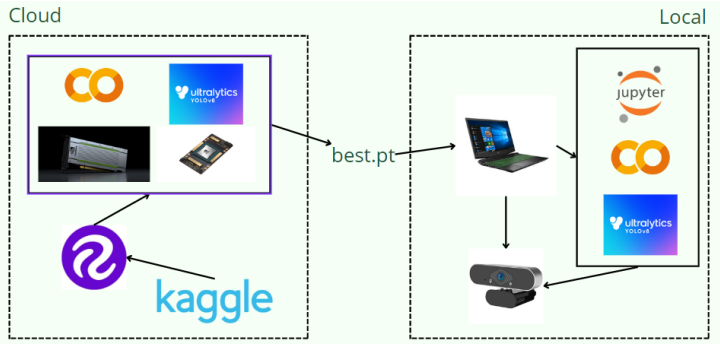
2.2 Perangkat Keras dan Lunak

Hardware yang dipakai dalam penelitian ini yaitu Laptop HP Pavilion dk-1041tx, yang menjadi kontroler utama di mana algoritma YOLOv8-large terinstal dan berjalannya *coding* deteksi objek. Kamera Xiaovv webcam dengan resolusi maksimum 1080p (FHD) berfungsi untuk mendeteksi diecast kendaraan secara langsung. Selain itu, hardware pada Google Colaboratory berupa Nvidia Tesla T4 dan Tesla L4 berfungsi sebagai pemroses tensor dalam proses training model YOLOv8-large.

Software yang dipakai pada penelitian ini merupakan software yang digunakan dalam aplikasi Deep Learning, computer vision, image labelling, dan proses menjalankan kode pemrograman. Software yang digunakan meliputi bahasa pemrograman Python 3, Miniconda3, Roboflow, dan Google Colaboratory. Software bawaan GPU berupa CUDA dan CuDNN juga berperan dalam memberikan tenaga dalam proses deteksi objek.

2.3 Setup Sistem

Sistem terdiri dari dua jenis *runtime*, yaitu cloud dan local. Sistem berjalan pada kondisi *cloud* ketika melakukan training model YOLOv8-large dan pengujian deteksi objek pada gambar dan video, sementara *rumtime* lokal berjalan saat pengujian secara langsung pada alat peraga. Setup system ditunjukkan pada gambar 2.



Gambar 2. Setup sistem

2.4 Training Data

Proses Training dimulai dengan melatih algoritma YOLOv8-large terlebih dahulu dengan penentuan parameter training kemudian pembagian dataset dalam tiga bagian, yaitu training, validasi, dan testing. Parameter training mencakup penentuan jumlah epos atau jumlah training yang dilakukan. Pembagian dataset mencakup data training yaitu jumlah gambar yang akan dijadikan bahan untuk melatih algoritma. Algoritma yang dilatih kemudian dilakukan validasi dengan beberapa gambar dataset untuk validasi. Kemudian, algoritma dilakukan testing untuk memastikan hasil akurasi testing mendekati akurasi pada saat training. Apabila akurasi testing lebih tinggi daripada saat training, maka terjadi overfitting. Dalam penelitian ini, proporsi pembagian dataset yaitu sebesar 76% untuk training, 13% untuk validasi, dan 11% untuk testing. Training model YOLOv8-large dilakukan pada Google Colaboratory menggunakan GPU NVIDIA A100 sebanyak 35 epos.

2.5 Metode Pengujian

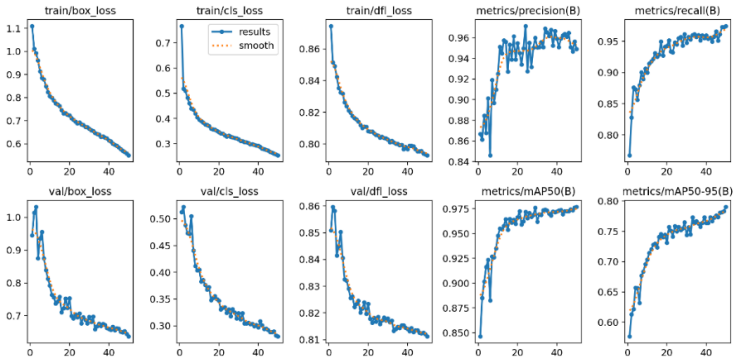
Pengujian sistem dilakukan dengan simulasi, di mana gambar dan rekaman video lalu lintas dari dijadikan bahan uji untuk deteksi kendaraan dan hasil berupa waktu hijau. Setelah simulasi, dilakukan pengujian real time menggunakan rekaman langsung dari kamera. Program Deep Learning dijalankan dengan input berupa rekaman, dan berdasarkan model yang sudah di training, divalidasi, dan testing, akan dilakukan deteksi objek dan menghasilkan waktu hijau.

Pengujian secara real time dengan kamera webcam dilakukan dengan melakukan deteksi objek kendaraan pada alat peraga berupa model perempatan sederhana. Desain model dibuat dengan ukuran 70x70 cm dengan lebar jalan 10 cm. Jalan dibagi menjadi 2 ruas dengan arah maju sebelah kiri. Alat peraga dibuat dengan triplex setebal 5 mm dengan susunan seperti puzzle untuk memudahkan transportasi alat.

3. Hasil dan Pembahasan

3.1 Hasil Training

Dengan GPU NVIDIA Tesla L4 yang dimiliki oleh server Google Colaboratory, training model YOLOv8-large memakan waktu hingga 0,506 jam dan penggunaan VRAM rata-rata sebesar 16,7 GB. Proses training menghasilkan dua model, yaitu “best.pt” dan “last.pt” yang merupakan model yang dihasilkan dengan weight terbaik dan model yang dihasilkan dengan weight terakhir secara berturut-turut. Model terbaik “best.pt” menghasilkan nilai mAP50 (Mean Average Precision dengan IoU sebesar 0,5) dari seluruh kelas sebesar 0,977 atau 97,7% dan mAP50-95 (MAP dengan IoU dari 0,5 hingga 0,95) sebesar 0,721 yang berarti memiliki nilai akurasi yang baik. Sebagai patokan, nilai akurasi mAP sebesar 0,5 merupakan kriteria minimum untuk sebuah model untuk dikatakan baik, karena model dengan akurasi 0,5 sama saja dengan menebak secara acak.



Gambar 3. Grafik parameter hasil training

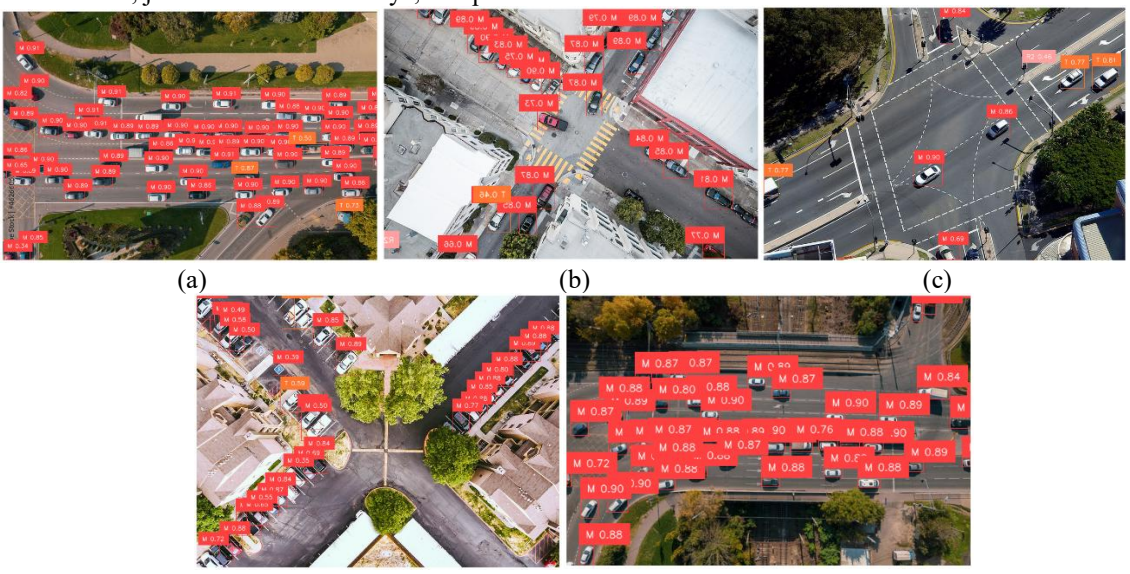
3.2 Hasil Validasi

Validasi model dilakukan pada model dengan weight terbaik, untuk memastikan akurasinya apabila diberikan gambar dari dataset validasi. Validasi juga digunakan untuk memeriksa apakah model mengalami overfitting, underfitting, ketidakpresisian atau tidak. Hasil dari validasi model “best.pt” menunjukkan bahwa untuk semua kelas, nilai mAP50 adalah 0,965 dan mAP50-95 sebesar 0,784 yang berarti sedikit lebih kecil daripada hasil training, demikian pula untuk masing-

masing kelas. Hal tersebut menandakan bahwa model "best.pt" sedikit mengalami overfitting, dan dapat dipakai dengan kondisi yang baik.

3.3 Hasil Pengujian Gambar

Hasil pengujian gambar diperoleh ketika lima gambar uji sudah dilakukan anotasi. Jumlah klasifikasi kendaraan yang terdeteksi oleh model YOLOv8 yang sudah dilatih dibandingkan dengan jumlah sebenarnya. Dari pengujian gambar, diperoleh persentase deteksi secara manual dari deteksi objek. Gambar (a) menunjukkan salah satu sampel hasil pengujian gambar. Jumlah deteksi, jumlah mobil sebenarnya, dan persentase deteksi ditabulasikan dalam tabel 1.



(a) (b) (c)

(d) (e)

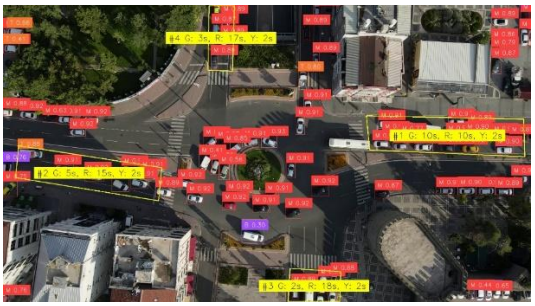
Gambar 4. Hasil pengujian gambar

Tabel 1. Persentase deteksi objek pengujian gambar

Gambar	Terdeteksi	Sebenarnya	Deteksi (%)
a	62	64	96,88
b	24	29	82,76
c	7	8	87,5
d	30	34	88,24
e	35	41	92,68

3.4 Hasil Pengujian Video

Hasil pengujian video diperoleh ketika video output dari proses anotasi tersimpan dalam runtime. Video hasil menunjukkan setiap kendaraan yang terdeteksi saat memasuki zona penghitungan akan langsung terhitung sebagai waktu hijau, sehingga berdasarkan jumlah kendaraan yang berhenti pada lampu lalu lintas, dapat dihasilkan waktu hijau yang akan dijadikan lama nyala lampu hijau pada ruas tersebut setelah ruas lainnya beralih menjadi lampu merah.

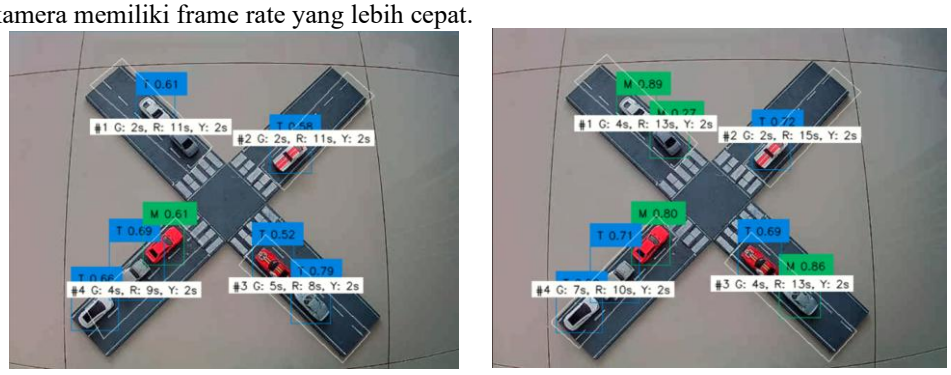


Gambar 5. Hasil pengujian video

3.5 Hasil Pengujian Alat Peraga

Gambar 6 menunjukkan proses deteksi secara langsung dengan webcam, yang berjalan dengan resolusi 640 x 480, dengan pencahayaan alami dalam ruangan berintensitas 82 lux, dan ketinggian kamera 85 cm. Pada hasil deteksi, dapat terlihat bahwa tidak semua kendaraan mendapatkan label klasifikasi yang benar, hal ini disebabkan oleh banyak faktor, seperti kekuatan prosesor, intensitas cahaya, ketinggian kamera, dan warna objek. Di sini, kekuatan prosesor menjadi

penentu kuat atau tidaknya deteksi. Pada deteksi dengan CPU, objek dapat terdeteksi, namun dengan klasifikasi yang tidak tepat dan rekaman berjalan dengan frame rate yang lambat. Hal ini disebabkan karena informasi berupa frame diproses pada inti CPU. Sementara dengan GPU, frame diproses pada inti tensor sehingga objek terdeteksi dengan klasifikasi yang lebih baik dan rekaman kamera memiliki frame rate yang lebih cepat.



(a) (b)

Gambar 6. Hasil *screenshot* rekaman langsung (a) tanpa GPU dan (b) dengan GPU

4. Kesimpulan

Penelitian ini memberikan kesimpulan atas masalah yang menjadi dasar penelitian ini, yaitu Deep Learning melakukan proses deteksi dan penghitungan objek pada rekaman kamera, yang mengubahnya menjadi waktu hijau proporsional dengan jumlah objek atau kendaraan yang terhitung. Hal ini dapat mengatasi keterbatasan dari sistem PLC di mana waktu hijau diatur terlebih dahulu tanpa memerhatikan kondisi real time. Untuk kedepannya, keluaran berupa waktu nyala lampu dapat diaplikasikan ke dalam sistem kontrol yang terintegrasi dengan lampu.

5. Daftar Pustaka

[1] Koonce, P. et al . (2008) Signal Timing Manual. Portland: Kittelson & Associates, Inc.
[2] Wani, M. A. et al . (2019) Advances in Deep Learning. doi: 10.1007/978-981-13-6794-6.
[3] Wang, B. et al. (2022) 'Deep Reinforcement Learning for Traffic Light Timing Optimization', Processes, 10(11). doi: 10.3390/pr10112458.
[4] Gandhi, M. M., Solanki, D. S., Daptardar, R. S., & Baloorkar, N. S. (2020, December 1). Smart Control of Traffic Light Using Artificial Intelligence. 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering, ICRAIE 2020 - Proceeding. https://doi.org/10.1109/ICRAIE51050.2020.9358334
[5] Verdhan, V. (2021) Computer Vision Using Deep Learning. Apress Media LLC.
[6] Goodfellow, I., Bengio, Y. and Courville, A. (2016) Deep Learning. Cambridge, Massachusetts: The MIT Press.
[7] Ying, X. (2019) 'An Overview of Overfitting and its Solutions An Overview of Overfitting and its Solutions'. doi: 10.1088/1742-6596/1168/2/022022.
[8] Tan, M. and Shao, P. (2015). Prediction of student dropout in e-learning program through the use of machine learning method. International Journal of Emerging Technologies in Learning (Ijet), 10(1), 11. https://doi.org/10.3991/ijet.v10i1.4189
[9] Gebrehiwot, A., Hashemi-Beni, L., Thompson, G., Kordjamshidi, P., & Langan, T. (2019). Deep convolutional neural network for flood extent mapping using unmanned aerial vehicles data. Sensors, 19(7), 1486. https://doi.org/10.3390/s19071486
[10] Ultralytics. (2023). YOLOv8 Docs., from https://docs.ultralytics.com/
[11] Hao, W., & Zhili, S. (2020). Improved mosaic: Algorithms for more complex images. Journal of Physics: Conference Series, 1684(1). https://doi.org/10.1088/1742-6596/1684/1/012094
[12] Zhang, J., Huang, B., Ye, Z., Kuang, L. D., & Ning, X. (2021). Siamese anchor-free object tracking with multiscale spatial attentions. Scientific Reports, 11(1). https://doi.org/10.1038/s41598-021-02095-4
[13] Zhu, J., Hu, T., Zheng, L., Zhou, N., Ge, H., & Hong, Z. (2024). YOLOv8-C2f-Faster-EMA: An Improved Underwater Trash Detection Model Based on YOLOv8. Sensors, 24(8), 2483. https://doi.org/10.3390/s24082483
[14] Huang, L., Li, W., Shen, L., Fu, H., Xiao, X., & Xiao, S. (2023). YOLOCS: Object Detection based on Dense Channel Compression for Feature Spatial Solidification. http://arxiv.org/abs/2305.04170
[15] Puertas, E., De-Las-heras, G., Fernández-Andrés, J., & Sánchez-Soriano, J. (2022). Dataset: Roundabout Aerial Images for Vehicle Detection. Data, 7(4). https://doi.org/10.3390/data7040047