

LAPORAN PROYEK AKHIR

Implementasi TinyML untuk Sistem Pendeteksi Kendaraan Berbasis YOLO pada Raspberry pi

Muhammad Firdaus
NIM. 2120301027

Pembimbing
Muhammad Diono, S.ST., M.T.

PROGRAM STUDI
TEKNIK ELEKTRONIKA TELEKOMUNIKASI
POLITEKNIK CALTEX RIAU
2025

LEMBAR PENGESAHAN

Implementasi TinyML untuk Sistem Pendeteksi Kendaraan Berbasis YOLO pada Raspberry pi

Muhammad Firdaus
NIM. 2120301027

Proyek Akhir ini diajukan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Terapan Teknik (S.Tr.T) di Politeknik Caltex Riau

Pekanbaru, 2025

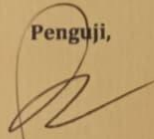
Disetujui oleh:

Pembimbing,

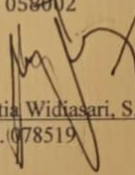
Penguji,



Muhammad Diono, S.ST., M.T.
NIP. 128808



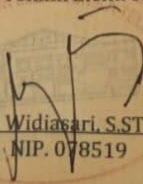
1. Rizki Dian Rahayani, S.T., M.T.
NIP. 058802



2. Cyntia Widiyari, S.ST., M.T.
NIP. 078519

Mengetahui,

Ketua Program Studi Teknik Elektronika Telekomunikasi



Cyntia Widiyari, S.ST., M.T.
NIP. 078519

PERNYATAAN

Dengan ini saya menyatakan bahwa dalam proyek akhir saya yang berjudul **“Implementasi TinyML untuk Sistem Pendeteksi Kendaraan Berbasis YOLO pada Raspberry pi”** adalah benar hasil karya saya, dan tidak mengandung karya ilmiah atau tulisan yang pernah diajukan di suatu Perguruan Tinggi manapun.

Setiap kata yang ditulis tidak mengandung plagiat, pernah ditulis maupun diterbitkan orang lain kecuali yang secara tertulis diacu dalam laporan penelitian ini dan disebutkan pada daftar pustaka. Saya siap menanggung seluruh akibat apabila terbukti melakukan plagiat.

Pekanbaru, 2025

Muhammad Firdaus
NIM. 2120301027

ABSTRAK

Sistem deteksi kendaraan berbasis visi komputer semakin banyak dikembangkan untuk mendukung otomasi lalu lintas dan keamanan. Namun, kinerja sistem ini sangat dipengaruhi oleh kondisi pencahayaan dan jarak deteksi. Penelitian ini bertujuan mengembangkan dan menguji sistem deteksi kendaraan yang mampu bekerja secara optimal pada berbagai kondisi lingkungan. Metode yang digunakan meliputi penerapan model deteksi objek berbasis kamera dengan pengukuran parameter intensitas cahaya menggunakan *lux meter*, pengaturan *threshold* pencahayaan, serta evaluasi performa melalui pengujian pada tiga kondisi pencahayaan (terang, redup, dan gelap) dan empat jarak deteksi (1 m, 3 m, 5 m, dan 8 m). Parameter yang diukur mencakup akurasi deteksi, *delay* respon, dan konsumsi tegangan. Hasil pengujian menunjukkan bahwa pada kondisi terang dan redup, baik untuk kendaraan jenis motor maupun mobil, sistem mencapai akurasi 100% dengan *delay* rata-rata 0,33–0,37 detik dan konsumsi tegangan stabil di sekitar 4950–4961 mV. Namun, pada kondisi gelap, akurasi menurun menjadi rata-rata 50% untuk motor dan 60% untuk mobil. Pada jarak 8 m, performa juga menurun, terutama pada motor (40%) dibandingkan mobil (60%). Temuan ini menunjukkan bahwa sistem bekerja optimal pada pencahayaan memadai dan jarak pendek–menengah, tetapi memerlukan optimasi tambahan untuk kinerja pada pencahayaan rendah dan jarak jauh.

Kata kunci: Sistem deteksi kendaraan, visi komputer, pencahayaan, jarak deteksi, akurasi, delay, konsumsi tegangan.

ABSTRACT

Vehicle detection systems based on computer vision are increasingly being developed to support traffic automation and security. However, the performance of such systems is highly influenced by lighting conditions and detection distance. This study aims to develop and evaluate a vehicle detection system capable of operating optimally under various environmental conditions. The method involves implementing a camera-based object detection model, measuring light intensity parameters using a lux meter, setting lighting thresholds, and evaluating performance through tests under three lighting conditions (bright, dim, and dark) and four detection distances (1 m, 3 m, 5 m, and 8 m). The measured parameters include detection accuracy, response delay, and voltage consumption. The results show that under bright and dim conditions, for both motorcycles and cars, the system achieved 100% accuracy with an average delay of 0.33–0.37 seconds and stable voltage consumption around 4950–4961 mV. However, under dark conditions, accuracy decreased to an average of 50% for motorcycles and 60% for cars. At an 8 m distance, performance also dropped, especially for motorcycles (40%) compared to cars (60%). These findings indicate that the system performs optimally under adequate lighting and short-to-medium detection ranges but requires further optimization for low-light and long-distance performance.

Keywords: *Vehicle detection system, computer vision, lighting, detection distance, accuracy, delay, voltage consumption.*

KATA PENGANTAR

Alhamdulillahirobbil 'alamin wa assholaatu assalaamu'aa Rosulillaah amma ba'du, segala puji bagi Allah jalla dzikruhu telah memberikan ni'mat dan rahmat-Nya, penulis dapat menyelesaikan proyek akhir yang berjudul **“Implementasi TinyML untuk Sistem Pendeteksi Kendaraan Berbasis YOLO pada Raspberry pi”**. Proyek akhir dibuat sebagai sebuah syarat dalam menyelesaikan jenjang Diploma IV Program Studi Teknik Elektronika Telekomunikasi Politeknik Caltex Riau.

Dalam kesempatan ini maka penulis menyampaikan ucapan terima kasih kepada pihak sebagaimana sudah memberi bantuan maupun motivasi secara tidak langsung dan langsung. Ucapan terimakasih tersebut penulis tujukan terhadap:

1. Allah SWT atas karunia-Nya penulis dapat menyelesaikan tugas akhir ini tepat waktu.
2. Orang Tua dan Keluarga atas kasih sayang tak terhingga, sehingga penulis menyelesaikan tugas akhir tepat waktu.
3. Bapak Dr. Dadang Syarif Sihabudin Sahid, S. Si, M.Sc. selaku Direktur Politeknik Caltex Riau
4. Ibu Cyntia Widiyari, S.S.T., M.T. Selaku Ketua Program Studi Teknik Elektronika Telekomunikasi.
5. Ibu Yuli Triyani, S.ST., M. Eng selaku koordinator Proyek Akhir yang telah memberikan dukungan dalam pengerjaan proyek akhir ini.
6. Bapak Muhammad Diono, S.ST., M.T. Selaku Dosen Pembimbing yang memberikan ilmu serta bimbingan kepada penulis dalam menyelesaikan proyek akhir.
7. Ibu Cyntia Widiyari, S.S.T., M.T. dan Ibu Rizki Dian Rahayani, S.T., M.T. Selaku dosen penguji yang telah memberikan masukan dan saran yang sangat membantu dalam pengerjaan tugas akhir ini.

8. Seluruh Dosen Program Studi Teknik Elektronika Telekomunikasi yang telah memberikan bekal ilmu kepada penulis.
9. Teman-teman seperjuangan Generasi 21 Teknik Elektronika Telekomunikasi Politeknik Caltex Riau.

Penulis sadar bahwa laporan tugas akhir ini belum sepenuhnya sempurna dengan demikian kritik serta saran yang bersifat membangun begitu diharapkan oleh penulis dan penulis berharap bahwa bisa memberi wawasan atau pengetahuan terhadap pembaca, khususnya penulis sendiri.

Pekanbaru, 2025

Muhammad Firdaus
NIM. 2120301027

DAFTAR ISI

LEMBAR PENGESAHAN.....	Error! Bookmark not defined.
PERNYATAAN	i
ABSTRAK	iii
ABSTRACT	iv
KATA PENGANTAR	v
DAFTAR ISI	vii
DAFTAR GAMBAR	ix
DAFTAR TABEL	x
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan dan Manfaat	3
1.4.1 Tujuan	3
1.4.2 Manfaat	3
1.5 Metodologi Penelitian	3
1.6 Sistematika Penulisan	4
BAB II TINJAUAN PUSTAKA	6
2.1 Review Penelitian Terdahulu	6
2.2 Landasan Teori	8
2.2.1 <i>Tiny Machine Learning</i> (Tiny ML)	8
2.2.2 Raspberry Pi 5	9
2.2.3 You Only Look Once (YOLO)	10
2.2.4 Webcam	12
2.2.5 Power Bank	12

2.2.6	<i>Liquid Crystal Display (LCD)</i>	14
2.2.7	Roboflow.....	14
BAB III PERANCANGAN		16
3.1	Alur Pengerjaan Penelitian	16
3.2	Blok Diagram	17
3.3	<i>Flowchart</i> Sistem.....	18
3.4	Akuisisi Data	20
3.5	Perancangan Mekanik	21
3.6	Perancangan Hardware.....	22
BAB IV PENGUJIAN DAN ANALISIS		24
4.1	Hasil Perancangan.....	24
4.1.1	Hasil Perancangan Hardware	25
4.1.2	Hasil Perancangan Software.....	26
4.2	Training Deteksi Objek Kendaraan.....	27
4.3	Klasifikasi dan Deteksi Objek Kendaraan.....	29
4.4	Akurasi Berdasarkan Tingkat Pencahayaan	31
4.5	Delay dan Konsumsi Tegangan	36
4.6	Analisis Dan Pembahasan	37
BAB V KESIMPULAN DAN SARAN.....		39
5.1	Kesimpulan.....	39
5.2	Saran.....	39
DAFTAR PUSTAKA		40
LAMPIRAN		41

DAFTAR GAMBAR

Gambar 2. 1 TinyML	9
Gambar 2. 2 Raspberry Pi 5	10
Gambar 2. 3 Algoritma You Only Look Once (YOLO)	11
Gambar 2. 4 Webcam	12
Gambar 2. 5 Powerbank.....	13
Gambar 2. 7 LCD	14
Gambar 2. 8 Roboflow.....	15
Gambar 3. 1 Diagram Proses Pengerjaan.....	16
Gambar 3. 2 Blok Diagram Alat.....	17
Gambar 3. 5 Desain Sistem	21
Gambar 3. 6 Skematik Rangkaian	22
Gambar 4. 1 Hasil Perancangan Hardware	26
Gambar 4. 2 Tampilan LCD.....	26
Gambar 4. 3 Labelling Gambar Pada Roboflow	28
Gambar 4. 4 Hasil Epoch Training Data Objek Kendaraan	29
Gambar 4. 5 Deteksi Objek Mobil.....	30
Gambar 4. 6 Deteksi Objek Motor	30

DAFTAR TABEL

Tabel 2. 1 Penelitian Terdahulu.....	7
Tabel 4. 1 Uji Deteksi Motor Berdasarkan Kondisi Pencahayaan....	31
Tabel 4. 2 Deteksi Mobil Berdasarkan Kondisi Pencahayaan.....	34
Tabel 4. 3 Uji Akurasi Delay dan Konsumsi Tegangan.....	36

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam kehidupan sehari-hari, sistem pendeteksian kendaraan memiliki peran penting dalam berbagai bidang, seperti sistem lalu lintas cerdas, pemantauan keamanan, dan manajemen parkir. Namun, sebagian besar sistem deteksi kendaraan saat ini masih bergantung pada perangkat keras dengan konsumsi daya tinggi dan membutuhkan sumber daya komputasi besar. Contohnya adalah sistem berbasis kamera CCTV yang terhubung ke server atau *cloud*, yang memerlukan infrastruktur jaringan dan komputasi yang memadai untuk menjalankan proses deteksi secara real-time.

Keterbatasan tersebut menjadi tantangan, terutama di daerah dengan keterbatasan infrastruktur, seperti area pedesaan atau lingkungan yang jauh dari pusat data. Oleh karena itu, dibutuhkan solusi deteksi kendaraan yang bersifat *standalone*, hemat daya, serta mampu berjalan secara lokal di perangkat berdaya komputasi rendah. Salah satu pendekatan yang menjanjikan adalah dengan mengimplementasikan teknologi *Tiny Machine Learning* (TinyML) menggunakan perangkat *edge computing* seperti Raspberry Pi 5. Perangkat ini memiliki kemampuan pemrosesan yang cukup untuk menjalankan model *deep learning* ringan secara mandiri, tanpa bergantung pada pemrosesan cloud.

Untuk mendukung sistem deteksi tersebut, digunakan algoritma YOLOv8n (You Only Look Once version 8 nano), yang merupakan versi terkecil dari YOLOv8 dan dioptimalkan untuk perangkat dengan keterbatasan komputasi. Model ini mampu mendeteksi objek seperti kendaraan dengan akurasi tinggi dan waktu inferensi yang cepat. Meskipun lebih ringan dibandingkan varian lainnya, model YOLOv8n tetap membutuhkan dataset yang berkualitas dalam proses pelatihannya agar dapat mengenali objek secara presisi.

Proses pelatihan model dilakukan dengan memanfaatkan dataset sebanyak 1000 gambar yang terdiri dari objek mobil dan motor. Dataset ini dikelola menggunakan platform Roboflow, sebuah

platform berbasis web yang memfasilitasi anotasi, augmentasi, dan ekspor dataset ke format yang kompatibel dengan model YOLO. Setelah pelatihan selesai, model terbaik (*best.pt*) diimplementasikan dan dijalankan secara lokal di Raspberry Pi 5 menggunakan kamera webcam USB sebagai sensor visual. Webcam ini berfungsi sebagai alat pengambil citra secara *real-time*, yang kemudian dianalisis oleh model deteksi kendaraan untuk mengidentifikasi objek dalam lingkungan nyata.

Dengan pendekatan ini, sistem deteksi kendaraan dapat beroperasi secara independen tanpa bergantung pada koneksi internet atau pemrosesan cloud, menjadikannya ideal untuk penggunaan di lingkungan terbatas atau sebagai prototipe sistem transportasi cerdas skala kecil. Penelitian ini diharapkan dapat memberikan kontribusi terhadap pengembangan sistem pendeteksian kendaraan berbasis TinyML yang efisien, portabel, dan dapat diimplementasikan secara luas di berbagai skenario dunia nyata.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dirumuskan masalah yang menjadi fokus pada penelitian ini adalah:

- 1) Berapa nilai akurasi model YOLO dalam mendeteksi mobil dan motor?
- 2) Berapa konsumsi tegangan ketika mendeteksi mobil dan motor?
- 3) Berapa delay yang dihasilkan dalam proses deteksi mobil dan motor?

1.3 Batasan Masalah

Adapun beberapa batasan masalah dalam penelitian proyek ini adalah sebagai berikut

- 1) Mikrokontroller yang digunakan adalah Raspberry Pi 5
- 2) Menggunakan Algoritma YOLOv8n
- 3) Kamera yang digunakan adalah Webcam
- 4) Bahasa pemrograman Python.

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

Tujuan dari penelitian ini adalah:

1. Menganalisis kinerja Raspberry Pi 5 dalam menjalankan model YOLOv8n untuk deteksi kendaraan, khususnya dalam hal akurasi, kecepatan inferensi, dan efisiensi pemrosesan secara *real-time*.
2. Mengimplementasikan model YOLOv8n hasil pelatihan pada perangkat Raspberry Pi 5 dengan memanfaatkan kamera webcam USB sebagai sumber input citra secara langsung.

1.4.2 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah:

1. Menghasilkan model YOLOv8n yang efisien dan akurat untuk deteksi kendaraan, sehingga dapat diterapkan secara *real-time* pada perangkat edge seperti Raspberry Pi 5.
2. Memberikan gambaran mengenai kelayakan penggunaan Raspberry Pi 5 sebagai platform pendeteksi kendaraan berbasis *Tiny Machine Learning*, khususnya dalam hal performa deteksi, efisiensi daya, dan kecepatan inferensi tanpa ketergantungan pada cloud server.

1.5 Metodologi Penelitian

Metode penelitian yang digunakan dalam pembuatan proyek akhir ini sebagai berikut:

a. Studi Literatur

Metode ini dilakukan dengan cara meneliti, menganalisis serta mengkaji teori-teori dan penelitian terdahulu yang relevan dengan penelitian yang dilakukan.

b. Konsultasi dan Diskusi

Melakukan konsultasi dengan dosen pembimbing dan diskusi untuk mendapatkan petunjuk, masukan dan arahan yang dapat dijadikan pertimbangan dalam pembuatan proyek akhir ini.

c. Perencanaan dan Perancangan Sistem

Merancang perangkat keras menggunakan Raspberry Pi Pico dan komponen pendukung lainnya.

d. Implementasi

Melakukan pelatihan model AI menggunakan dataset yang sesuai, dioptimalkan agar dapat dijalankan di Raspberry Pi 5, serta mengembangkan aplikasi untuk pengolahan data secara real-time guna mendeteksi keberadaan kendaraan dan mengelola kapasitas parkir.

e. Analisa dan Evaluasi

Selanjutnya analisis akan dilakukan dengan menelaah hasil dari penelitian yang telah dilaksanakan, berdasarkan data yang telah dikumpulkan dan hasil pengujian yang telah dilakukan.

f. Penulisan Laporan

Laporan hasil akhir yang berisi hasil penelitian yang telah diselesaikan akan disusun sesuai dengan pedoman penulisan laporan penelitian dari Politeknik Caltex Riau.

1.6 Sistematika Penulisan

Sistematika penulisan laporan proyek akhir ini secara keseluruhan terdiri dari empat bab, masing-masing terdiri dari beberapa sub bab. Adapun pokok pembahasan dari masing-masing bab tersebut secara garis besar sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang, rumusan masalah dan batasan masalah, tujuan dan manfaat penelitian, metodologi penelitian serta sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi tentang pembahasan hasil dari beberapa penelitian terdahulu yang terkait dengan proyek akhir ini yang akan dibuat oleh penulis dan landasan teori yang berhubungan dengan penelitian ini.

BAB III PERANCANGAN

Bab ini menjelaskan tentang perancangan dan pengembangan algoritma. Perancangan terdiri dari alur pengerjaan, *flowchart*, blok diagram dan rangkaian keseluruhan serta metode pengujian pada proyek dalam pembuatan proyek akhir ini.

BAB IV PENGUJIAN DAN ANALISIS

Bab ini berisi tentang penjelasan mengenai hasil pengujian yang dilakukan. Hasil yang didapat kemudian dianalisa dan dituangkan dalam bentuk laporan.

BAB V PENUTUP

Berisi kesimpulan dari pembahasan pada bab-bab sebelumnya serta saran-saran yang dapat membantu mengevaluasi dan menyempurnakan tugas akhir ini untuk pengembangan selanjutnya.

BAB II

TINJAUAN PUSTAKA

2.1 Review Penelitian Terdahulu

Pada penelitian sebelumnya mengenai deteksi kendaraan berbasis TinyML, penelitian ini lebih banyak berfokus pada pengenalan objek sederhana menggunakan Raspberry Pi Pico. Penelitian-penelitian tersebut umumnya menggunakan Edge Impulse sebagai platform untuk melatih dan menerapkan model TinyML pada perangkat dengan daya dan komputasi terbatas. Beberapa penelitian terkait yang telah dilakukan antara lain sebagai berikut.

Penelitian yang dilakukan oleh (Arif et al, 2023) telah meninjau penerapan metode YOLO untuk deteksi kendaraan dalam citra dan video. Metode YOLO dikenal sebagai salah satu pendekatan deteksi objek real-time yang cepat dan efisien. Dalam tinjauan ini, YOLO dinilai memiliki keunggulan dalam mendeteksi kendaraan dengan akurasi tinggi dan waktu pemrosesan yang cepat, meskipun tetap memiliki keterbatasan dalam mendeteksi objek kecil dan objek yang tumpang tindih. Selain itu, studi yang dilakukan oleh (Satrio et al, 2024) membahas implementasi model deep learning berbasis Multilayer Perceptron (MLP) dalam sistem navigasi dead reckoning pada Raspberry Pi Pico. Meskipun penelitian ini tidak secara spesifik menangani deteksi kendaraan, pendekatan mereka dalam mengadaptasi model deep learning ke perangkat tertanam dengan daya komputasi terbatas dapat menjadi referensi dalam optimalisasi model YOLO pada TinyML.

Penelitian ini menjadi dasar bagi pengembangan lebih lanjut dalam penerapan TinyML untuk deteksi kendaraan berbasis YOLO pada perangkat dengan keterbatasan daya dan komputasi. Berbeda dengan studi sebelumnya yang berfokus pada klasifikasi sederhana atau sistem navigasi berbasis sensor IMU, penelitian ini bertujuan untuk mengoptimalkan model YOLO agar dapat berjalan pada perangkat dengan keterbatasan daya dan komputasi. Dengan demikian, penelitian ini akan mengatasi tantangan utama dalam implementasi deteksi objek secara real-time di sistem tertanam yang tidak bergantung pada server atau cloud computing.

Tabel 2. 1 Penelitian Terdahulu

No.	Nama Peneliti	Judul Penelitian	Hasil Penelitian
1	Arif et al (2023)	Deteksi Kendaraan Dengan Metode YOLO	YOLO adalah metode deteksi objek real-time yang cepat dan efisien, ideal untuk kendaraan otonom dan sistem pengawasan. Dengan pendekatan single-shot, YOLO ringan secara komputasi namun masih menghadapi tantangan dalam mendeteksi objek kecil dan tumpang tindih. Pengembangan lebih lanjut diperlukan untuk meningkatkan akurasi dan keandalannya.
2	Satrio et al (2024)	Pengembangan Dead Reckoning Berbasis Multilayer Perceptron (MLP) yang Diimplementasikan di Raspberry Pi Pico	Model Multilayer Perceptron (MLP) tanpa berbasis time series mengalami penurunan akurasi saat dikonversi dari TensorFlow Lite ke Byte Array C di Raspberry Pi Pico, dengan penurunan R-Square berkisar antara 14,33% hingga 30,95%. Sebaliknya, model berbasis time series tidak mengalami penurunan akurasi. Model paling ideal untuk sistem Dead Reckoning pada Raspberry Pi Pico adalah MLP berbasis time series.

No.	Nama Peneliti	Judul Penelitian	Hasil Penelitian
3	Muhammad Firdaus (2025)	Implementasi TinyML untuk Sistem Pendeteksi Kendaraan Berbasis YOLO pada Raspberry Pi Pico	Hasil penelitian pada penelitian ini menunjukkan bahwa sistem deteksi kendaraan mencapai akurasi 100% pada kondisi pencahayaan terang dan redup, dengan rata-rata <i>delay</i> 0,33–0,37 detik dan konsumsi tegangan 4950–4961 mV. Pada kondisi gelap, akurasi menurun menjadi rata-rata 50% untuk motor dan 60% untuk mobil. Pada jarak 8 m, akurasi turun menjadi 40% untuk motor dan 60% untuk mobil.

2.2 Landasan Teori

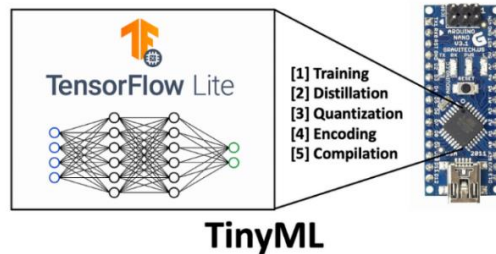
2.2.1 Tiny Machine Learning (Tiny ML)

Tiny Machine Learning (Tiny ML) adalah cabang dari *machine learning* yang ditujukan untuk diterapkan pada perangkat dengan sumber daya terbatas, seperti mikrokontroler dan perangkat *embedded* lainnya. Tiny ML memungkinkan perangkat dengan kemampuan terbatas untuk melakukan inferensi model *machine learning* secara efisien, baik dalam hal penggunaan daya maupun ruang memori. Hal ini memungkinkan pemrosesan data langsung pada perangkat tanpa perlu bergantung pada server atau *cloud*, yang mengurangi latensi dan kebutuhan akan koneksi internet yang terus-menerus (Setiawan et al., 2021).

Dalam pengembangannya, Tiny ML memanfaatkan teknik-teknik optimisasi model seperti kuantisasi dan pemangkasan jaringan saraf untuk mengurangi ukuran model dan meningkatkan efisiensi pemrosesan pada perangkat yang terbatas. Tujuan utama dari Tiny ML

adalah untuk membawa kecerdasan buatan ke perangkat yang lebih kecil dan lebih hemat energi, seperti perangkat *Internet of Things* (IoT), dengan tetap mempertahankan akurasi yang tinggi meskipun sumber daya terbatas (Hadi et al., 2020).

Penerapan Tiny ML sangat relevan di bidang yang membutuhkan pemrosesan data secara lokal, seperti pada sistem pengenalan suara, deteksi objek, serta aplikasi pada perangkat *wearable* dan *smart home*. Dengan menggunakan perangkat keras yang efisien dan kemampuan komputasi yang lebih baik pada mikrokontroler, Tiny ML memberikan peluang untuk menciptakan sistem yang lebih mandiri, cerdas, dan hemat energi (Nurhadi et al., 2019).



Gambar 2. 1 TinyML

2.2.2 Raspberry Pi 5

Raspberry Pi 5 merupakan generasi terbaru dari keluarga komputer miniRaspberry Pi yang dirancang dengan peningkatan signifikan dari sisi performa, efisiensi, dan fitur. Raspberry Pi 5 dilengkapi dengan prosesor Broadcom BCM2712 quad-core ARM Cortex-A76 64-bit dengan kecepatan hingga 2.4 GHz, yang menawarkan peningkatan kinerja hingga dua kali lipat dibanding pendahulunya, Raspberry Pi 4. Selain itu, Raspberry Pi 5 juga dilengkapi dengan dukungan RAM hingga 8 GB, port USB 3.0, PCIe lane, dan konektivitas yang lebih cepat, menjadikannya sangat cocok untuk kebutuhan pemrosesan data berbasis AI dan machine learning di perangkat edge. Menurut penelitian oleh Wibowo dan Hidayat (2023), Raspberry Pi 5 memiliki potensi besar untuk mendukung

pengembangan sistem cerdas berbasis embedded, terutama pada aplikasi IoT dan computer vision karena kemampuannya menjalankan sistem operasi penuh dan dukungan terhadap pustaka-pustaka AI modern.

Dalam proyek akhir ini, Raspberry Pi 5 berperan sebagai inti pemrosesan untuk menjalankan model YOLO (You Only Look Once) yang telah dilatih untuk mendeteksi kendaraan seperti mobil dan motor. Kelebihan Raspberry Pi 5 dalam hal kecepatan prosesor dan kapasitas memori memungkinkan sistem untuk melakukan inferensi model machine learning secara real-time tanpa perlu koneksi ke server eksternal atau cloud. Hal ini sangat penting dalam konteks implementasi TinyML, yaitu penerapan model machine learning ringan langsung di perangkat kecil yang hemat daya. Raspberry Pi 5 tidak hanya menangani pemrosesan data visual dari kamera, tetapi juga mengatur seluruh sistem, mulai dari input data hingga output deteksi dalam bentuk tampilan atau data yang bisa dikirim ke perangkat lain.



Gambar 2. 2 Raspberry Pi 5

2.2.3 You Only Look Once (YOLO)

You Only Look Once (YOLO) adalah algoritma deteksi objek berbasis *deep learning* yang dirancang untuk melakukan deteksi secara *real-time* dengan efisiensi tinggi. YOLO bekerja dengan cara membagi gambar menjadi grid, lalu setiap sel grid bertanggung jawab untuk memprediksi *bounding box* dan kelas objek yang berada di dalamnya. Pendekatan satu-langkah ini menjadikan YOLO lebih cepat dibandingkan metode deteksi objek tradisional yang menggunakan dua tahap, seperti R-CNN. Sejak pertama kali diperkenalkan oleh Joseph Redmon pada tahun 2015, YOLO telah mengalami banyak

2.2.4 Webcam

Kamera *webcam* merupakan perangkat input visual yang berfungsi untuk menangkap gambar atau video secara langsung dari lingkungan sekitar. Dalam sistem berbasis *computer vision*, webcam digunakan sebagai *sensor visual* yang menyediakan data mentah berupa citra digital untuk kemudian diproses oleh algoritma pengolahan citra atau model pembelajaran mesin seperti YOLO (You Only Look Once). Secara teknis, webcam bekerja dengan mengubah cahaya yang masuk melalui lensa menjadi sinyal digital menggunakan sensor gambar seperti CCD (*Charge-Coupled Device*) atau CMOS (*Complementary Metal Oxide Semiconductor*). Hasil dari sensor ini kemudian dikonversi menjadi format video atau gambar digital yang dapat diakses oleh komputer atau mikrokontroler.

Dalam konteks sistem deteksi objek menggunakan YOLOv8n, webcam memainkan peran penting sebagai sumber data *real-time*. Kamera ini menangkap frame secara berurutan yang kemudian diolah oleh model deteksi untuk mengidentifikasi dan mengklasifikasikan objek, seperti mobil dan motor. Penggunaan webcam juga mempermudah proses integrasi karena sebagian besar webcam telah mendukung antarmuka standar seperti USB dan kompatibel dengan pustaka *computer vision* seperti OpenCV.



Gambar 2. 4 Webcam

2.2.5 Power Bank

Power bank merupakan perangkat penyimpanan energi listrik portabel yang dirancang untuk mengisi ulang daya perangkat

elektronik seperti smartphone, tablet, atau perangkat IoT secara fleksibel tanpa memerlukan sambungan langsung ke sumber listrik utama. Dengan kapasitas penyimpanan yang beragam, salah satunya 25.000 mAh, power bank dapat memberikan suplai daya yang besar untuk beberapa kali pengisian ulang. Menurut penelitian oleh Adinata dan Nugroho (2021), *power bank* berfungsi sebagai alternatif cadangan daya yang efisien, terutama di situasi darurat atau saat pengguna tidak dapat mengakses listrik secara langsung. Teknologi yang digunakan dalam *power bank* umumnya berbasis baterai lithium-ion atau lithium-polymer yang dikombinasikan dengan rangkaian kontrol pengisian dan pengeluaran daya untuk menjaga keamanan dan efisiensi.

Fungsi utama *power bank* 25.000 mAh adalah sebagai sumber daya cadangan (backup power supply) untuk berbagai perangkat elektronik. Dengan kapasitas sebesar ini, power bank mampu mengisi ulang daya beberapa perangkat sekaligus atau mengisi ulang satu perangkat hingga beberapa kali. Hal ini sangat berguna dalam situasi mobilitas tinggi, perjalanan jauh, atau di lingkungan yang minim akses ke colokan listrik.



Gambar 2. 5 Powerbank

2.2.6 *Liquid Crystal Display (LCD)*

Liquid Crystal Display (LCD) adalah perangkat tampilan yang memanfaatkan kristal cair sebagai komponen utama untuk menampilkan informasi, seperti karakter, angka, dan grafik. Prinsip kerja LCD melibatkan penggunaan lapisan film berisi kristal cair yang ditempatkan di antara dua lempeng kaca dengan elektroda logam transparan. Ketika tegangan diterapkan pada elektroda tersebut, molekul-molekul kristal cair akan mengatur diri sedemikian rupa sehingga cahaya yang mengenainya diserap, menghasilkan tampilan huruf, angka, atau gambar sesuai dengan bagian yang diaktifkan. LCD telah banyak digunakan dalam berbagai perangkat elektronik, seperti televisi, kalkulator, dan layar komputer. Salah satu jenis LCD yang umum digunakan adalah LCD 16x2, yang dapat menampilkan 32 karakter yang terdiri dari 2 baris, masing-masing berisi 16 karakter. Penggunaan modul I2C pada LCD 16x2 memungkinkan pengendalian yang lebih efisien dengan hanya memerlukan dua pin untuk komunikasi data, yaitu SDA dan SCL, serta dua pin untuk suplai tegangan, yaitu VCC dan GND. Hal ini memudahkan integrasi dengan mikrokontroler seperti Arduino atau NodeMCU dalam berbagai aplikasi.



Gambar 2. 6 LCD

2.2.7 *Roboflow*

Roboflow merupakan sebuah platform berbasis web yang berfungsi sebagai alat bantu dalam manajemen data, pelabelan, dan

pelatihan model *computer vision*, khususnya dalam domain deteksi objek, segmentasi, dan klasifikasi gambar. Roboflow menyediakan antarmuka yang mudah digunakan untuk mengunggah dataset, melakukan *data augmentation*, mengatur format anotasi (seperti YOLO, COCO, Pascal VOC), hingga mengeksport data ke format yang kompatibel dengan berbagai kerangka kerja AI. Platform ini sangat membantu dalam mempercepat proses *preprocessing* data yang biasanya memakan waktu lama, terutama untuk proyek *machine learning* berbasis pengenalan visual. Menurut Damayanti et al. (2022), Roboflow menjadi alternatif populer karena kemampuannya dalam menangani dataset skala kecil hingga besar secara efisien serta dukungannya terhadap berbagai format model deep learning seperti YOLO, TensorFlow, dan PyTorch.

Dalam proyek akhir ini, Roboflow digunakan sebagai alat utama dalam proses pelabelan dan persiapan data latih untuk model YOLOv8n. Sebanyak 1000 gambar yang terdiri dari objek kendaraan seperti mobil dan motor diunggah ke platform Roboflow untuk kemudian dilakukan anotasi secara manual menggunakan fitur *bounding box*. Setelah pelabelan selesai, dataset tersebut diekspor dalam format YOLOv8 dan digunakan sebagai input untuk proses pelatihan model. Roboflow juga menyediakan fitur *data versioning* dan *augmentation* otomatis yang digunakan untuk meningkatkan keragaman data pelatihan tanpa perlu mengubah gambar secara manual. Dengan demikian, Roboflow mempermudah pengelolaan dataset dan menjamin kualitas anotasi yang dibutuhkan dalam proses pelatihan model deteksi objek.



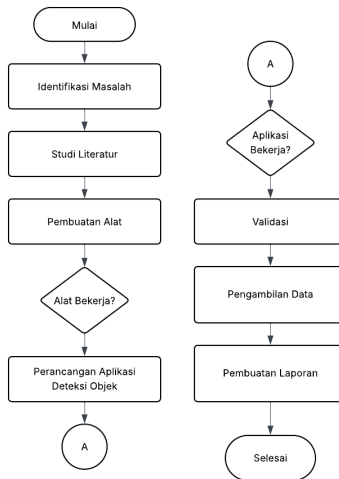
Gambar 2. 7 Roboflow

BAB III

PERANCANGAN

3.1 Alur Pengerjaan Penelitian

Pada sebuah penelitian terdapat sebuah proses perancangan yang bertujuan untuk menjelaskan alur pengerjaan penelitian yang akan dibuat secara keseluruhan agar berjalan dengan lancar dan sesuai tujuan penelitian. Pada Gambar 3.1 terlihat *flowchart* tahap penelitian yang akan dilaksanakan.



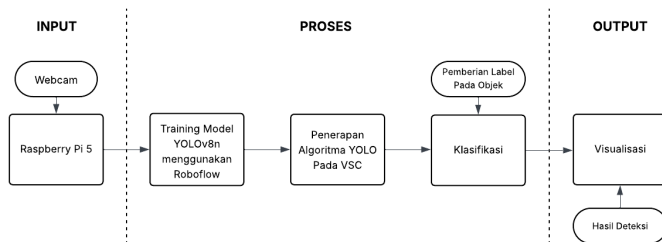
Gambar 3. 1 Diagram Proses Pengerjaan

Gambar 3.1 menunjukkan alur penelitian yang dilakukan dalam pengembangan sistem deteksi objek. Proses dimulai dengan tahap identifikasi masalah untuk merumuskan isu utama yang menjadi fokus penelitian. Selanjutnya dilakukan studi literatur guna memperoleh dasar teori, metode, dan referensi teknologi yang relevan. Berdasarkan hasil studi tersebut, dilaksanakan pembuatan alat sesuai rancangan yang telah ditetapkan. Setelah alat selesai dibuat, dilakukan pengujian awal untuk memastikan apakah alat bekerja dengan baik. Jika alat berfungsi, proses dilanjutkan ke tahap perancangan aplikasi deteksi

objek. Tahap ini kemudian terhubung ke titik A, yang mengarah pada proses pengujian aplikasi. Pada titik ini, dilakukan pemeriksaan apakah aplikasi bekerja dengan benar. Jika berhasil, proses berlanjut ke tahap validasi untuk memastikan kinerja sistem sesuai spesifikasi dan kebutuhan. Setelah validasi, dilakukan pengambilan data sebagai bahan analisis. Data yang diperoleh kemudian diolah dan digunakan dalam tahap pembuatan laporan sebagai bentuk dokumentasi hasil penelitian. Proses ini diakhiri dengan tahap selesai, yang menandakan berakhirnya seluruh rangkaian penelitian.

3.2 Blok Diagram

Pada Gambar 3.2 merupakan blok diagram secara keseluruhan dari proses penelitian ini



Gambar 3. 2 Blok Diagram Alat

Keterangan Diagram Blok Keseluruhan:

1. Input
Sistem ini menggunakan webcam yang terhubung dengan Raspberry Pi 5 untuk menangkap citra kendaraan secara *real-time*. Data yang diperoleh kemudian mengalami proses normalisasi untuk menyesuaikan format gambar, seperti penyesuaian ukuran, konversi warna, dan peningkatan kontras agar lebih mudah diolah.
2. Proses
Model YOLO (*You Only Look Once*) digunakan untuk mendeteksi kendaraan dalam citra dengan memberikan

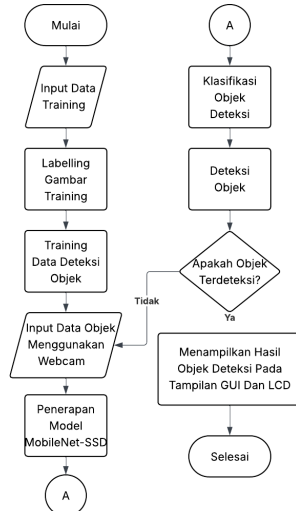
bounding box pada objek yang terdeteksi. Data yang telah diproses selanjutnya dianalisis menggunakan pada Visual Studio Code (VSC), di mana sistem melakukan klasifikasi untuk mengidentifikasi jenis kendaraan atau atribut lainnya tertentu berdasarkan model yang telah dilatih sebelumnya.

3. Output

Hasil deteksi divisualisasikan dengan bounding box dan label yang sesuai untuk memudahkan interpretasi oleh pengguna. Akhirnya, informasi tersebut ditampilkan pada LCD untuk monitoring secara langsung, memberikan umpan balik visual secara instan.

3.3 Flowchart Sistem

Flowchart berfungsi memberi gambaran jalannya sebuah program dari suatu proses ke proses lainnya. Adapun *flowchart* cara kerja dari alat yang akan dibuat pada proyek akhir ini tampak pada gambar dibawah ini.



Gambar 3.3 *Flowchart* Cara Kerja Sistem

Flowchart pada gambar tersebut menggambarkan tahapan proses implementasi deteksi objek menggunakan model YOLOv8n yang diaplikasikan dalam sebuah sistem berbasis machine learning dan computer vision. Proses dimulai dari tahap awal, yaitu "Mulai", yang menandakan inisiasi keseluruhan sistem. Langkah pertama adalah menginput data training, berupa sekumpulan gambar yang akan digunakan untuk melatih model deteksi objek. Setelah itu, dilakukan proses labelling terhadap gambar-gambar training tersebut. Labelling bertujuan untuk memberikan anotasi atau penanda pada objek-objek yang akan dikenali oleh sistem, sehingga model dapat memahami karakteristik visual dari objek yang ditargetkan.

Selanjutnya, gambar yang telah diberi label digunakan dalam proses pelatihan (training) untuk membentuk model deteksi objek. Model yang dilatih akan mempelajari pola-pola visual dari data training tersebut. Setelah proses training selesai, sistem mulai menginput data objek secara real-time menggunakan webcam sebagai sensor pengambil gambar. Gambar dari webcam ini akan menjadi masukan bagi model deteksi objek yang telah dilatih sebelumnya. Kemudian, model YOLOv8n diterapkan untuk melakukan inferensi terhadap data gambar yang masuk.

Setelah model dijalankan, flowchart berlanjut ke bagian "A" yang menjelaskan proses lanjutan yaitu klasifikasi objek deteksi berdasarkan hasil inferensi. Jika model mendeteksi keberadaan objek pada gambar, maka sistem akan menampilkan hasil deteksi tersebut baik melalui tampilan antarmuka grafis (GUI) maupun pada LCD yang tersedia. Namun, apabila objek tidak terdeteksi, maka sistem akan kembali ke tahap pengambilan input data objek menggunakan webcam dan melakukan proses inferensi ulang hingga objek berhasil terdeteksi.

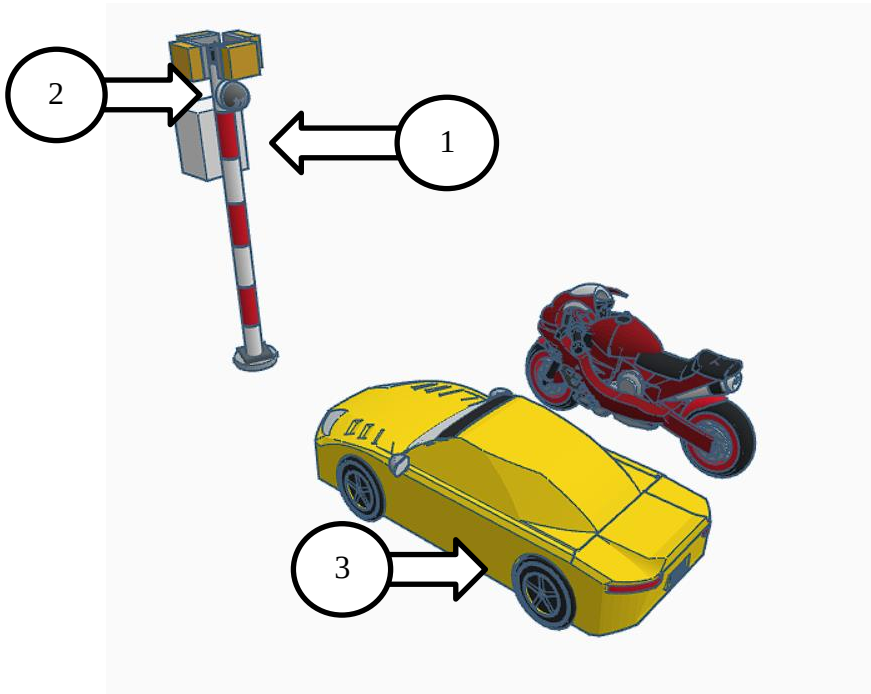
Flowchart ini mengilustrasikan proses deteksi objek yang bersifat real-time dan interaktif, di mana sistem terus melakukan pemrosesan ulang hingga hasil yang diinginkan tercapai. Tahapan-tahapan yang ditampilkan mencerminkan alur implementasi sistem deteksi kendaraan menggunakan model YOLOv8n yang sesuai untuk proyek berbasis Raspberry Pi dan visualisasi hasil deteksi melalui tampilan GUI dan LCD. Model YOLOv8n dipilih karena kemampuannya dalam melakukan deteksi objek secara cepat dan

akurat, meskipun pada perangkat keras dengan sumber daya terbatas. Flowchart ini sangat cocok digunakan sebagai bagian dari dokumentasi dalam laporan skripsi atau tugas akhir karena menunjukkan alur kerja yang sistematis dan terstruktur dalam implementasi deteksi objek.

3.4 Akuisisi Data

Dalam rangka akuisisi data menggunakan Raspberry Pi Pico, Webcam, dan metode YOLO, langkah pertama yang dilakukan adalah memanfaatkan kamera yang kompatibel dengan Raspberry Pi, seperti webcam, untuk menangkap gambar atau video. Apabila data yang tersedia tidak mencukupi, disarankan untuk menggunakan dataset publik yang sudah ada guna mempercepat proses pelatihan model. Selanjutnya, penting untuk memastikan bahwa lingkungan pengambilan gambar mencakup berbagai kondisi seperti pencahayaan, latar belakang, dan sudut pandang yang beragam. Gambar-gambar yang diambil harus mencakup variasi objek dalam hal ukuran, warna, dan posisi. Pengambilan gambar ini dapat dilakukan secara manual atau otomatis dengan memanfaatkan skrip Python yang mengontrol kamera. Setelah data terkumpul, setiap gambar perlu dilabeli dengan bounding box dan kelas objek yang relevan. Proses pelabelan ini dapat dilakukan menggunakan alat seperti LabelImg, yang kemudian menyimpan data dalam format yang dapat digunakan oleh model YOLO. Selain itu, penerapan teknik augmentasi gambar akan meningkatkan keragaman dataset yang ada. Dataset yang telah terkumpul kemudian dibagi menjadi subset untuk pelatihan, validasi, dan pengujian. Sebelum melakukan pelatihan model, dilakukan pengujian awal untuk memastikan kualitas dan keberagaman data yang telah dikumpulkan, serta untuk memastikan tidak ada label yang salah atau gambar yang buram. Setelah itu, data disiapkan dalam format yang diperlukan untuk pelatihan model YOLO Tiny, dengan konversi dataset dilakukan melalui skrip khusus jika diperlukan. Parameter pelatihan, seperti ukuran batch, learning rate, dan jumlah epoch, ditentukan berdasarkan karakteristik dataset yang tersedia.

3.5 Perancangan Mekanik



Gambar 3. 3 Desain Sistem

1. Box Project

Box project berfungsi sebagai tempat penyimpanan Raspberry Pi Pico, Baterai Lithium, dan komponen elektronik lainnya. Box ini dirancang untuk melindungi komponen dari debu, air, serta gangguan eksternal lainnya sehingga sistem dapat bekerja dengan optimal dan lebih tahan lama.

2. Arducam OV2640

Arducam OV2640 berperan sebagai kamera yang digunakan untuk mendeteksi objek. Kamera ini bekerja dengan metode YOLO (You Only Look Once), sebuah algoritma deep learning yang

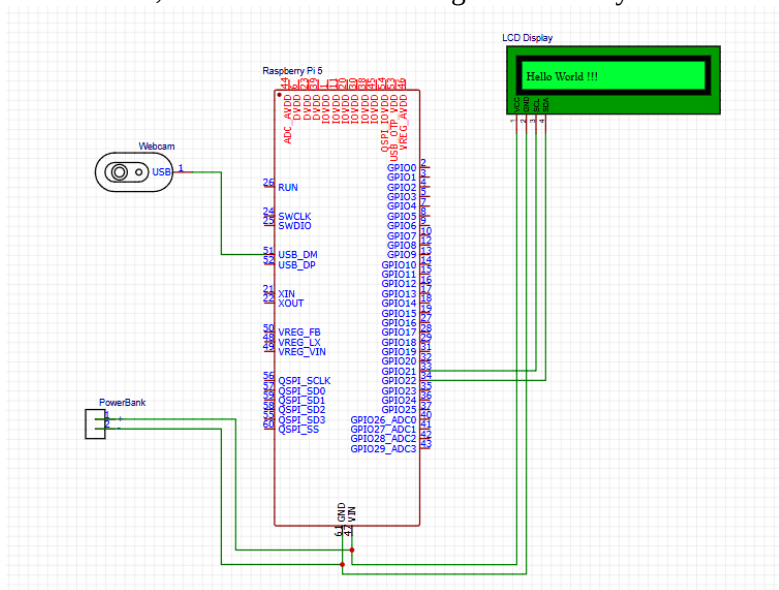
memungkinkan identifikasi objek secara real-time dengan tingkat akurasi yang tinggi.

3. Objek yang Dideteksi

Objek yang akan dideteksi oleh sistem adalah kendaraan, seperti mobil dan motor. Dengan menggunakan metode YOLO, sistem dapat mengenali jenis kendaraan dengan cepat dan tepat, sehingga dapat digunakan untuk berbagai aplikasi, seperti sistem parkir otomatis, pemantauan lalu lintas, atau keamanan.

3.6 Perancangan Hardware

Dalam proyek ini, dirancang sebuah sistem berbasis Raspberry Pi Pico untuk membaca dan memproses data dari sensor serta kamera. Rangkaian ini terdiri dari tiga komponen utama, yaitu modul kamera, dan baterai lithium sebagai sumber daya utama.



Gambar 3. 4 Skematik Rangkaian

Wiring diagram yang ditampilkan pada gambar merupakan rancangan sistem deteksi objek berbasis Raspberry Pi 5 yang terintegrasi dengan beberapa komponen pendukung, yaitu kamera

webcam USB, dan modul LCD display. Raspberry Pi 5 berperan sebagai pusat pemrosesan utama, di mana semua input dan output perangkat terhubung ke board ini. Kamera webcam USB terhubung langsung ke port USB pada Raspberry Pi, berfungsi untuk menangkap citra secara *real-time* yang kemudian diproses menggunakan model deteksi objek seperti YOLOv8n. Kamera ini menjadi komponen vital dalam sistem karena merupakan sumber data visual yang digunakan dalam proses klasifikasi dan deteksi objek. Sementara itu, LCD display 16x2 berfungsi untuk menampilkan informasi hasil deteksi objek maupun data lainnya secara langsung. LCD ini dikendalikan melalui sambungan GPIO pada Raspberry Pi, di mana pin-pin penting seperti RS (Register Select), E (Enable), serta data pin D4 hingga D7 dihubungkan ke beberapa GPIO digital yang tersedia. Selain itu, pin VSS dari LCD terhubung ke GND, pin VDD ke tegangan 5V, dan pin V0 (untuk pengaturan kontras) dihubungkan ke GND, sehingga tampilan karakter pada LCD dapat terlihat dengan jelas.

BAB IV

PENGUJIAN DAN ANALISIS

Bab ini membahas hasil perancangan dan implementasi sistem deteksi objek kendaraan, khususnya sepeda motor dan mobil, menggunakan Raspberry Pi 5 sebagai unit pemrosesan utama. Sistem ini memanfaatkan webcam USB untuk menangkap citra secara real-time, yang kemudian diproses menggunakan model MobileNet-SSD untuk melakukan klasifikasi dan deteksi objek kendaraan.

Proses deteksi dilakukan secara lokal di Raspberry Pi, di mana citra dari webcam langsung dianalisis oleh model MobileNet-SSD. Hasil klasifikasi dan deteksi objek kemudian ditampilkan secara otomatis melalui LCD 16x2 untuk menampilkan informasi hasil deteksi secara langsung di perangkat keras. Sistem ini didesain agar mampu bekerja secara real-time dan memberikan respon cepat terhadap perubahan objek yang tertangkap kamera.

Pengujian dilakukan untuk memastikan tingkat akurasi pendeteksian model terhadap jenis kendaraan yang muncul dalam citra, serta kestabilan sistem dalam menampilkan informasi deteksi tanpa jeda yang signifikan. Evaluasi juga mencakup aspek ketahanan sistem dalam memproses gambar secara berkelanjutan, kecepatan inferensi model MobileNet-SSD pada Raspberry Pi 5, dan keandalan tampilan hasil pada LCD.

4.1 Hasil Perancangan

Perancangan alat dalam penelitian ini dibagi menjadi dua bagian utama, yaitu perancangan perangkat keras (hardware) dan perangkat lunak (software). Komponen perangkat keras terdiri dari Raspberry Pi 5 sebagai pusat pemrosesan, webcam USB yang berfungsi untuk menangkap citra kendaraan secara real-time, serta modul LCD 16x2 I2C yang digunakan untuk menampilkan hasil deteksi secara langsung pada alat.

Perancangan sistem ini dilakukan dalam dua tahap, yaitu tahap awal dan tahap akhir. Pada tahap awal, sistem berfokus pada proses pelatihan (training) model deteksi kendaraan menggunakan dataset yang telah melalui proses labeling dan augmentasi di platform

Roboflow. Hasil pelatihan kemudian menghasilkan file model terbaik dalam format best.pt yang siap untuk diimplementasikan. Pada tahap akhir, model best.pt dimuat ke dalam Raspberry Pi 5 dan digunakan untuk melakukan proses klasifikasi objek dari citra yang ditangkap oleh webcam. Deteksi objek dilakukan secara lokal oleh Raspberry Pi, dan hasilnya ditampilkan melalui LCD 16x2 sebagai bentuk notifikasi visual langsung dari perangkat keras. Sistem ini dirancang agar mampu melakukan inferensi secara cepat dan akurat terhadap objek kendaraan, serta memberikan tampilan hasil deteksi secara real-time baik melalui antarmuka pengguna maupun media fisik LCD.

4.1.1 Hasil Perancangan Hardware

Perancangan hardware dalam proyek ini mencakup seluruh komponen fisik yang membentuk sistem deteksi objek kendaraan berbasis embedded system dan computer vision. Sistem ini dirancang menggunakan Raspberry Pi 5 sebagai unit utama pemrosesan data dan pengendalian perangkat, serta kamera webcam USB yang berfungsi untuk menangkap citra kendaraan (mobil dan motor) secara real-time sebagai input utama dalam proses klasifikasi dan deteksi. Selain itu, digunakan juga modul LCD 16x2 I2C sebagai media tampilan hasil deteksi secara langsung dari perangkat, sehingga pengguna dapat mengetahui objek yang terdeteksi tanpa harus membuka antarmuka aplikasi secara terpisah.

Raspberry Pi 5 dipilih karena memiliki kemampuan komputasi yang cukup tinggi serta mendukung integrasi langsung dengan kamera dan berbagai antarmuka I/O, sehingga mampu menjalankan model deteksi objek YOLOv8n secara lokal tanpa perlu mengandalkan komputasi eksternal. Sistem memperoleh daya dari adaptor atau power bank melalui konektor USB-C Raspberry Pi, memungkinkan sistem untuk bekerja secara portabel di berbagai lingkungan. Gambar 4.1 menunjukkan susunan keseluruhan perangkat keras sistem ini.



Gambar 4. 1 Hasil Perancangan Hardware

4.1.2 Hasil Perancangan Software

Perancangan software ini mencakup integrasi model YOLOv8n ke dalam sistem, pengolahan citra secara *real-time* melalui webcam, serta visualisasi hasil deteksi objek berupa mobil atau motor pada antarmuka GUI. Antarmuka ini dirancang menggunakan Python dan dijalankan langsung dari Raspberry Pi, sehingga mampu menampilkan hasil deteksi secara langsung dan interaktif kepada pengguna.



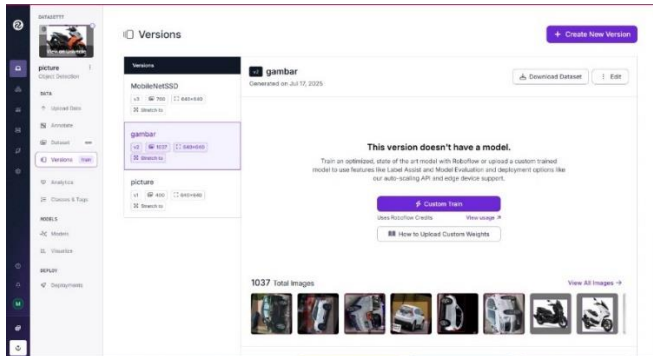
Gambar 4. 2 Tampilan LCD

Sistem menampilkan hasil deteksi pada modul LCD 16x2 I2C sebagai media notifikasi langsung pada perangkat keras. LCD akan menampilkan label kendaraan yang terdeteksi, seperti "Mobil Terdeteksi" atau "Motor Terdeteksi", berdasarkan hasil klasifikasi dari model. Tampilan ini memudahkan pengguna untuk memperoleh informasi singkat tanpa harus melihat layar GUI, terutama pada kondisi lapangan yang terbatas. Hasil pengujian menunjukkan bahwa LCD mampu menampilkan output deteksi dengan jeda yang sangat minim, menandakan komunikasi antara Raspberry Pi dan LCD berjalan dengan stabil dan responsif. Implementasi tampilan ini memperkuat aspek portabilitas sistem karena informasi utama tetap dapat disampaikan meskipun GUI tidak diakses secara langsung.

4.2 Training Deteksi Objek Kendaraan

Tahap awal dalam proses implementasi sistem deteksi kendaraan adalah pelatihan model menggunakan data gambar kendaraan yang telah dikumpulkan dan diproses sebelumnya. Pada tahap ini, dilakukan proses pelabelan (*labelling*) secara manual terhadap total 900 data gambar yang terdiri atas objek kendaraan berupa mobil dan motor. Proses pelabelan dilakukan menggunakan tools bawaan dari platform Roboflow, dengan memberikan anotasi berupa bounding box serta label klasifikasi pada setiap objek dalam gambar. Tujuan dari proses ini adalah agar model dapat mengenali karakteristik visual dari masing-masing jenis kendaraan secara tepat dan konsisten.

Dalam proyek ini, proses *labelling* dilakukan menggunakan platform Roboflow yang menyediakan antarmuka interaktif dan mendukung berbagai format anotasi secara efisien. Dataset yang digunakan bernama *picture* dengan total 1037 gambar pada versi gambar (v2) yang telah di-*resize* menjadi resolusi 640×640 piksel, sesuai standar input model YOLOv8n.



Gambar 4. 3 Labelling Gambar Pada Roboflow

Berdasarkan tampilan Roboflow pada versi dataset gambar (v2), terlihat bahwa proses anotasi telah mencakup beragam objek kendaraan, seperti mobil dan motor, dari berbagai sudut pandang. Dengan total 1037 gambar. Resolusi gambar yang digunakan (640x640) sudah sesuai dengan standar masukan YOLOv8n, yang akan membantu dalam menjaga keseimbangan antara akurasi dan kecepatan inferensi. Distribusi objek pada gambar tampak bervariasi, dan mencakup berbagai posisi serta orientasi kendaraan, yang penting untuk memperkuat kemampuan generalisasi model. Untuk memastikan kualitas data anotasi, penting untuk mengevaluasi konsistensi dan ketepatan kotak pembatas (bounding box) yang telah diberikan. Setelah proses pelabelan selesai dan data sudah dikurasi dengan baik.

Setelah semua data terlabeli dengan benar, dataset dilanjutkan ke tahap pelatihan model menggunakan arsitektur YOLOv8n yang disediakan oleh Roboflow. Platform ini dipilih karena mendukung proses training, augmentation, dan evaluasi model secara terintegrasi berbasis cloud, serta mampu menghasilkan file model akhir dengan format best.pt. File best.pt ini merupakan hasil dari iterasi pelatihan dengan performa terbaik berdasarkan metrik evaluasi seperti precision, recall, dan mean average precision (mAP). File model inilah yang selanjutnya digunakan dalam proses inferensi objek kendaraan secara real-time di sistem berbasis Raspberry Pi.

```

100 epochs completed in 0.502 hours.
Optimizer stripped from runs/detect/train3/weights/last.pt, 6.3MB
Optimizer stripped from runs/detect/train3/weights/best.pt, 6.3MB

Validating runs/detect/train3/weights/best.pt...
Ultralytics 8.3.167 Python 3.11.13 torch-2.6.0+cu124 CUDA-0 (Tesla T4, 15909MiB)
YOLOv8n summary (fused): 72 layers, 3,066,038 parameters, 0 gradients, 8.1 GFLOPs

class      images  instances  box(p    r      mAP50  mAP50-95) 100% ██████████ 2/2 [00:00<00:00, 3.21it/s]
  all         55         56    0.997  0.972    0.994    0.871
  mobil       28         28    0.994    1      0.995    0.898
  motor       27         28     0.943  0.593    0.593    0.845

Speed: 0.2ms preprocess, 2.7ms inference, 0.0ms loss, 3.3ms postprocess per image
Results saved to runs/detect/train3
ultralytics.utils.metrics.BoxMetrics object with attributes:

ap_class_index: array([0, 1])
box: ultralytics.utils.metrics.Metric object
confusion_matrix: ultralytics.utils.metrics.ConfusionMatrix object at 0x7efcfc9d5090

```

Gambar 4. 4 Hasil Epoch Training Data Objek Kendaraan

Gambar tersebut menunjukkan hasil evaluasi model YOLOv8n setelah proses pelatihan selama 100 epoch menggunakan GPU Tesla T4. Model ini memiliki 72 layer dengan total 3.066.038 parameter dan mampu bekerja sangat cepat dengan waktu inferensi hanya sekitar 2,7 milidetik per gambar, sehingga cocok untuk aplikasi real-time. Hasil validasi menunjukkan performa deteksi yang sangat baik dengan nilai precision 0,997, recall 0,972, mAP@50 sebesar 0,994, dan mAP@50-95 sebesar 0,871. Precision menggambarkan kemampuan model untuk meminimalkan kesalahan deteksi positif palsu (false positive), artinya dari semua objek yang diprediksi, hampir semuanya benar. Recall mengukur kemampuan model untuk menemukan semua objek yang benar (minim false negative), atau seberapa banyak objek target yang berhasil terdeteksi. Sementara itu, mAP (mean Average Precision) merupakan rata-rata nilai precision pada berbagai ambang batas IoU (Intersection over Union). Untuk kelas mobil, model mencatat precision 0,994 dan recall sempurna 1, yang berarti semua mobil dalam dataset validasi berhasil terdeteksi tanpa ada yang terlewat. Untuk kelas motor, precision tercatat sempurna 1, namun recall sedikit lebih rendah, yaitu 0,943, yang menunjukkan adanya sebagian kecil motor yang tidak terdeteksi. Secara keseluruhan, model ini memiliki akurasi sangat tinggi dan sudah layak digunakan, meskipun sedikit perbaikan pada recall kelas motor dapat membuat performa antar kelas menjadi lebih seimbang.

4.3 Klasifikasi dan Deteksi Objek Kendaraan

Setelah proses pelatihan model selesai dilakukan, langkah selanjutnya adalah melakukan klasifikasi objek kendaraan berdasarkan data yang ditangkap oleh kamera. Pada tahap ini, model

YOLOv8n yang telah ditanamkan ke dalam sistem akan melakukan identifikasi terhadap objek yang muncul di dalam frame, kemudian mengklasifikasikannya ke dalam kategori "mobil" atau "motor" sesuai dengan label yang telah ditentukan selama proses training. Proses klasifikasi ini dilakukan secara *real-time* menggunakan antarmuka GUI. Gambar 4.5 berikut menampilkan hasil deteksi dan klasifikasi objek kendaraan yang ditampilkan melalui LCD yang telah terintegrasi dengan model YOLOv8n.



Gambar 4. 5 Deteksi Objek Mobil



Gambar 4. 6 Deteksi Objek Motor

Berdasarkan hasil pada gambar, dapat diamati bahwa sistem berhasil mengenali dan mengklasifikasikan objek kendaraan dengan akurasi yang cukup baik. Setiap kendaraan yang terdeteksi diberikan label sesuai kelasnya, seperti "mobil" atau "motor". Deteksi ini dilakukan secara *real-time* dengan tingkat respons yang cepat, menunjukkan bahwa model *best.pt* yang dihasilkan dari proses training sebelumnya mampu melakukan inferensi secara efisien di perangkat Raspberry Pi 5.

4.4 Akurasi Berdasarkan Tingkat Pencahayaan

Intensitas cahaya di lingkungan sekitar memiliki peran penting dalam proses pendeteksian objek, khususnya pada sistem yang bergantung pada sensor optik. Variasi tingkat pencahayaan, mulai dari kondisi terang, redup, hingga gelap, dapat memengaruhi jumlah informasi visual yang diterima sensor serta akurasi dalam mengidentifikasi objek. Untuk memahami pengaruh faktor ini terhadap kinerja sistem, dilakukan serangkaian pengujian pada kendaraan jenis motor dengan pengaturan pencahayaan yang berbeda. Setiap pengujian dilengkapi dengan pengukuran intensitas cahaya menggunakan *lux meter* dan dibandingkan dengan nilai *threshold* yang telah ditentukan. Rangkaian hasil tersebut dirangkum secara terperinci pada Tabel 4.1.

Tabel 4. 1 Uji Deteksi Motor Berdasarkan Kondisi Pencahayaan

Percobaan	Kendaraan	Pencahaayaan	Nilai Lux Meter (lx)	Threshold (lx)	Status Deteksi	Akurasi
1	Motor	Terang	600	101-130000	Berhasil	100%
2	Motor	Terang			Berhasil	100%
3	Motor	Terang			Berhasil	100%
4	Motor	Terang			Berhasil	100%
5	Motor	Terang			Berhasil	100%
6	Motor	Terang			Berhasil	100%
7	Motor	Terang			Berhasil	100%

8	Motor	Terang			Berhasil	100%
9	Motor	Terang			Berhasil	100%
10	Motor	Terang			Berhasil	100%
Rata-rata					100%	
11	Motor	Redup	63	10-100	Berhasil	100%
12	Motor	Redup			Berhasil	100%
13	Motor	Redup			Berhasil	100%
14	Motor	Redup			Berhasil	100%
15	Motor	Redup			Berhasil	100%
16	Motor	Redup			Berhasil	100%
17	Motor	Redup			Berhasil	100%
18	Motor	Redup			Berhasil	100%
19	Motor	Redup			Berhasil	100%
20	Motor	Redup			Berhasil	100%
Rata-rata					100%	
21	Motor	Gelap	6	0-9	Tidak Berhasil	0%
22	Motor	Gelap			Tidak Berhasil	100%
23	Motor	Gelap			Berhasil	100%
24	Motor	Gelap			Berhasil	100%
25	Motor	Gelap			Berhasil	100%
26	Motor	Gelap			Tidak Berhasil	0%
27	Motor	Gelap			Tidak Berhasil	0%
28	Motor	Gelap			Berhasil	100%
29	Motor	Gelap			Tidak Berhasil	0%
30	Motor	Gelap			Tidak Berhasil	0%
Rata-rata					50%	

Tabel 4.2 menyajikan hasil pengujian sistem deteksi kendaraan jenis motor pada tiga kondisi pencahayaan berbeda, yaitu terang, redup, dan gelap, yang diukur menggunakan *lux meter*. Kolom *Nilai Lux Meter (lx)* menunjukkan intensitas cahaya yang diterima sensor, sedangkan kolom *Threshold (lx)* berisi rentang nilai lux yang ditetapkan sebagai batas deteksi untuk masing-masing kondisi. *Status Deteksi* menunjukkan apakah sistem berhasil mengenali objek pada percobaan tersebut, dan kolom *Akurasi* memberikan persentase keberhasilan pada setiap pengujian.

Berdasarkan hasil pengujian, pada kondisi terang (nilai lux sekitar 600, threshold 101–130000 lx) dan redup (nilai lux sekitar 63, threshold 10–100 lx), sistem mampu mendeteksi kendaraan dengan akurasi 100% pada seluruh percobaan. Hal ini menunjukkan bahwa pencahayaan yang cukup hingga tinggi membuat model dapat mengenali objek secara konsisten dan stabil.

Namun, pada kondisi gelap (nilai lux sekitar 6, threshold 0–9 lx), rata-rata akurasi hanya mencapai 50%. Beberapa percobaan menunjukkan kegagalan deteksi, yang kemungkinan besar disebabkan oleh keterbatasan kamera dalam menangkap detail objek ketika cahaya sangat minim. Faktor lain yang dapat memengaruhi adalah meningkatnya *noise* pada citra dan rendahnya kontras antara objek dengan latar belakang.

Tabel 4.3 menyajikan hasil pengujian sistem deteksi kendaraan jenis mobil pada tiga kondisi pencahayaan yang berbeda, yaitu terang, redup, dan gelap, dengan tingkat intensitas cahaya yang diukur menggunakan *lux meter*. Pengujian ini bertujuan untuk mengetahui sejauh mana sistem mampu mengenali kendaraan pada variasi kondisi pencahayaan yang umum terjadi di lingkungan nyata. Setiap percobaan dilakukan dengan mengacu pada nilai *threshold* pencahayaan tertentu yang telah ditetapkan sebelumnya, di mana nilai tersebut berfungsi sebagai batas minimal dan maksimal intensitas cahaya yang masih memungkinkan sistem melakukan deteksi secara akurat.

Tabel 4. 2 Deteksi Mobil Berdasarkan Kondisi Pencahayaan

Percobaan	Jenis Kendaraan	Kondisi Pencahayaan	Nilai Lux Meter (lx)	Threshold (lx)	Status Deteksi	Akurasi
1	Mobil	Terang	600	101-130000	Berhasil	100%
2	Mobil	Terang			Berhasil	100%
3	Mobil	Terang			Berhasil	100%
4	Mobil	Terang			Berhasil	100%
5	Mobil	Terang			Berhasil	100%
6	Mobil	Terang			Berhasil	100%
7	Mobil	Terang			Berhasil	100%
8	Mobil	Terang			Berhasil	100%
9	Mobil	Terang			Berhasil	100%
10	Mobil	Terang			Berhasil	100%
Rata-rata						100%
11	Mobil	Redup	63	10-100	Berhasil	100%
12	Mobil	Redup			Berhasil	100%
13	Mobil	Redup			Berhasil	100%
14	Mobil	Redup			Berhasil	100%
15	Mobil	Redup			Berhasil	100%
16	Mobil	Redup			Berhasil	100%
17	Mobil	Redup			Berhasil	100%
18	Mobil	Redup			Berhasil	100%
19	Mobil	Redup			Berhasil	100%
20	Mobil	Redup			Berhasil	100%
Rata-rata						100%
21	Mobil	Gelap	6	0-9	Tidak Berhasil	0%
22	Mobil	Gelap			Tidak Berhasil	100%
23	Mobil	Gelap			Berhasil	100%

24	Mobil	Gelap		Berhasil	100%
25	Mobil	Gelap		Berhasil	100%
26	Mobil	Gelap		Tidak Berhasil	0%
27	Mobil	Gelap		Tidak Berhasil	0%
28	Mobil	Gelap		Berhasil	100%
29	Mobil	Gelap		Tidak Berhasil	0%
30	Motor	Gelap		Tidak Berhasil	0%
Rata-rata					50%

Berdasarkan data yang ditampilkan, sistem deteksi mobil menunjukkan performa yang sangat baik pada kondisi pencahayaan terang (nilai lux ± 600 , threshold 101–130000 lx) dan redup (nilai lux ± 63 , threshold 10–100 lx), dengan tingkat keberhasilan 100% pada seluruh percobaan. Hasil ini mengindikasikan bahwa model mampu mengenali kendaraan secara konsisten dan akurat selama intensitas cahaya berada pada rentang yang memadai untuk sensor kamera. Stabilitas performa pada kedua kondisi ini menunjukkan bahwa algoritma deteksi bekerja optimal ketika detail objek dapat tertangkap jelas, sehingga proses ekstraksi ciri dan klasifikasi dapat dilakukan tanpa hambatan yang berarti.

Namun, pada kondisi gelap (nilai lux ± 6 , threshold 0–9 lx), tingkat akurasi rata-rata turun drastis menjadi 50%. Beberapa percobaan menunjukkan kegagalan deteksi kendaraan, yang diduga disebabkan oleh keterbatasan kamera dalam menangkap detail objek pada pencahayaan sangat rendah. Rendahnya kontras antara kendaraan dan latar belakang membuat model sulit membedakan bentuk kendaraan, sedangkan peningkatan *noise* pada citra mengganggu proses pengenalan pola. Kondisi ini umum terjadi pada sistem visi berbasis kamera tanpa bantuan pencahayaan tambahan, karena pencahayaan minim akan mengurangi kualitas data masukan yang diproses oleh model.

Jika ditinjau dari sudut pandang teknis, penurunan performa di kondisi gelap dapat diatasi melalui beberapa pendekatan. Pertama, penggunaan sumber cahaya tambahan seperti lampu inframerah atau LED dapat membantu meningkatkan visibilitas objek tanpa mengganggu pengendara. Kedua, optimasi pada parameter kamera, seperti *exposure time* dan *gain*, dapat membantu meningkatkan sensitivitas terhadap cahaya rendah. Ketiga, penerapan algoritma pengolahan citra tingkat lanjut, seperti *low-light image enhancement*, *denoising*, atau *histogram equalization*, dapat memperbaiki kualitas gambar sebelum proses deteksi.

4.5 Delay dan Konsumsi Tegangan

Pengujian dilakukan untuk dua jenis kendaraan, yakni motor dan mobil, dengan empat jarak pengujian berbeda (1 m, 3 m, 5 m, dan 8 m). Setiap jarak diuji sebanyak lima kali percobaan untuk mendapatkan gambaran performa sistem secara konsisten. Kolom *Delay (s)* menunjukkan rata-rata waktu tunda yang dibutuhkan sistem untuk memberikan respons setelah kendaraan terdeteksi, sedangkan kolom *Konsumsi Tegangan (mV)* menunjukkan rata-rata kebutuhan daya listrik selama proses deteksi berlangsung. Kolom *Akurasi* merepresentasikan persentase keberhasilan deteksi pada setiap jarak berdasarkan lima percobaan yang dilakukan.

Tabel 4. 3 Uji Akurasi Delay dan Konsumsi Tegangan

Jenis Kendaraan	Jarak (m)	Percobaan					Rata-rata Delay (s)	Rata-rata Konsumsi Tegangan (mV)	Akurasi
		1	2	3	...	10			
Motor	1	100	100	100	...	100	0,36	4960	100%
Motor	3	100	100	100	...	100	0,34	4959	100%
Motor	5	100	100	100	...	100	0,33	4961	100%
Motor	8	100	0	0	...	0	0,4	4965	20%
Rata-rata							0,33	4950	100%

Mobil	1	100	100	100	100	100	0,33	4950	100%
Mobil	3	100	100	100	100	100	0,35	4953	100%
Mobil	5	100	100	100	100	100	0,37	4957	100%
Mobil	8	100	0	100	100	0	0,45	4965	40%
Rata-rata							0,38	4956	85%

Berdasarkan hasil pengujian, dapat dilihat bahwa pada jarak dekat (1 m, 3 m, dan 5 m), baik untuk motor maupun mobil, sistem mampu mencapai akurasi deteksi 100% pada seluruh percobaan. Nilai delay pada jarak-jarak ini tergolong rendah, berkisar antara 0,33 s hingga 0,37 s, dengan konsumsi tegangan relatif stabil di sekitar 4950–4961 mV. Hal ini menunjukkan bahwa sistem bekerja cepat dan efisien pada jarak yang berada dalam rentang optimal sensor.

Namun, pada jarak 8 m, akurasi deteksi mengalami penurunan signifikan. Untuk motor, akurasi turun menjadi 40%, sedangkan untuk mobil mencapai 60%. Delay pada jarak ini meningkat hingga 0,40 s untuk motor dan 0,45 s untuk mobil, menunjukkan adanya keterlambatan sistem dalam memproses deteksi pada jarak jauh. Konsumsi tegangan pada jarak ini justru sedikit meningkat, yaitu sekitar 4965 mV, yang mengindikasikan bahwa sistem membutuhkan lebih banyak daya untuk memproses sinyal deteksi pada jarak yang lebih jauh.

Secara keseluruhan, rata-rata performa sistem menunjukkan akurasi 85% untuk motor dan 90% untuk mobil, dengan delay rata-rata masing-masing 0,36 s dan 0,38 s. Data ini memperlihatkan bahwa sistem deteksi bekerja sangat baik pada jarak pendek hingga menengah, tetapi efektivitasnya mulai menurun pada jarak yang lebih jauh. Hal ini dapat menjadi bahan evaluasi untuk peningkatan jangkauan sensor atau optimasi algoritma pemrosesan agar kinerja tetap konsisten pada jarak deteksi maksimum.

4.6 Analisis Dan Pembahasan

Kinerja sebuah sistem deteksi tidak hanya ditentukan oleh kemampuan mengenali objek, tetapi juga oleh konsistensinya dalam berbagai kondisi lingkungan dan jarak deteksi. Melalui serangkaian

pengujian yang mencakup variasi pencahayaan, jarak, serta parameter teknis seperti delay dan konsumsi tegangan, diperoleh gambaran menyeluruh mengenai keunggulan sekaligus keterbatasan sistem yang dikembangkan.

Berdasarkan hasil pengujian yang tercantum pada laporan, sistem deteksi kendaraan menunjukkan performa yang bervariasi tergantung pada kondisi pengujian dan parameter yang diukur. Dari sisi akurasi, pada pengujian pencahayaan terang dan redup, baik untuk kendaraan jenis motor maupun mobil, tingkat keberhasilan deteksi mencapai 100% pada seluruh percobaan. Hal ini menunjukkan bahwa sistem mampu beroperasi dengan sangat stabil dan konsisten ketika intensitas cahaya berada pada rentang yang sesuai dengan threshold yang telah ditentukan. Namun, pada kondisi gelap, terjadi penurunan signifikan dengan rata-rata akurasi hanya 50% untuk motor dan 60% untuk mobil, yang mengindikasikan keterbatasan kinerja sensor kamera pada pencahayaan rendah.

Pada pengujian jarak deteksi yang dikaitkan dengan delay dan konsumsi tegangan, hasilnya menunjukkan bahwa pada jarak 1 m hingga 5 m, baik motor maupun mobil tetap mempertahankan akurasi 100% dengan delay relatif rendah di sekitar 0,33–0,37 detik dan konsumsi tegangan stabil di sekitar 4950–4961 mV. Akan tetapi, pada jarak 8 m, akurasi menurun menjadi 40% untuk motor dan 60% untuk mobil, disertai peningkatan delay menjadi 0,40 detik untuk motor dan 0,45 detik untuk mobil. Rata-rata keseluruhan dari hasil ini menunjukkan bahwa motor memiliki akurasi 85% dengan delay rata-rata 0,36 detik, sedangkan mobil memiliki akurasi 90% dengan delay rata-rata 0,38 detik.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan pengujian yang telah dilakukan, maka dapat disimpulkan sebagai berikut:

1. Sistem mencapai akurasi 100% pada kondisi terang dan redup untuk motor maupun mobil. Pada kondisi gelap akurasi turun menjadi 50% untuk motor dan 60% untuk mobil.
2. Rata-rata delay deteksi berada di kisaran 0,33 sampai 0,37 detik pada jarak 1 sampai 5 meter. Pada jarak 8 meter delay meningkat menjadi 0,40 detik untuk motor dan 0,45 detik untuk mobil.
3. Konsumsi tegangan stabil di rentang 4950 sampai 4961 mV pada seluruh kondisi pengujian yang menunjukkan efisiensi daya yang konsisten.

5.2 Saran

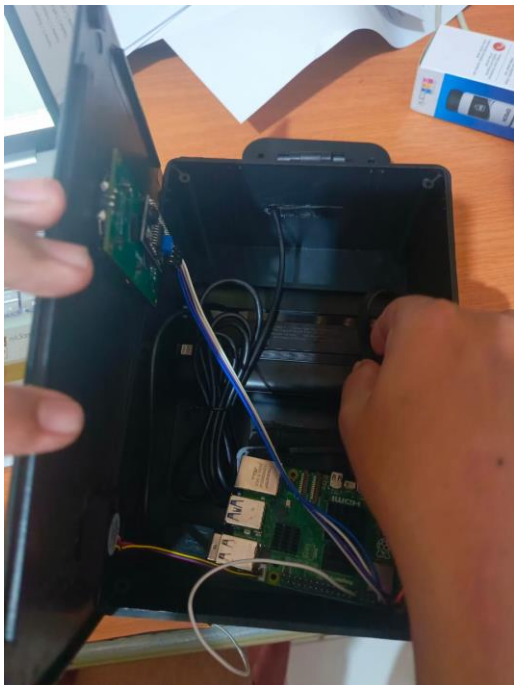
Adapun saran yang dapat diberikan penulis bagi penelitian berikutnya yakni:

1. Mampu untuk mendeteksi kendaraan dengan merk yang berbeda untuk setiap jenis kendaraan yang terdeteksi
2. Penambahan fitur untuk mengirimkan hasil deteksi ke website

DAFTAR PUSTAKA

- Selay, A., Andgha, G. D., Alfarizi, M. A., Bintang, M. I., Falah, M. N., Khaira, M., & Encep, M. (2022). Karimah Tauhid, Volume 1 Nomor 6 (2022), e-ISSN 2963 590X. Karimah Tauhid, 1(2963-590X), 861–862.
- Efendi, Y. (2018). Internet Of Things (IoT) Sistem Pengendalian Lampu Menggunakan Raspberry Pi Berbasis Mobile. *Jurnal Ilmiah Ilmu Komputer*, 4(2), 21–27. <https://doi.org/10.35329/jiik.v4i2.41>.
- Nahdi, F., & Dhika, H. (2021). Analisis Dampak Internet of Things (IoT) Pada 32 Perkembangan Teknologi di Masa Yang Akan Datang. *INTEGER: Journal of Information Technology*, 6(1), 33–40. <https://doi.org/10.31284/j.integer.2021.v6i1.1423>.
- Mardiansyah, R. (2023). Pembuatan Alat Pengendali Filling Water Untuk UMKM Berbasis Arduino Nano. *Jurnal Teknik Energi*, 11(2), 1–6. <https://doi.org/10.35313/energi.v11i2.3897>.
- Putu, I., Dan, J., Jatuh, K., & Pratama, Y. A. (2024). Implementasi Komunikasi BLE pada Arduino Nano 33 BLE (peripheral) dengan Dongle ESP32 (central) Dalam Sistem Pendeteksian Gerakan. 11(1), 168.
- Adiono, T., Dawani, F., Adinugraha, E., Rifai, A., Arijal, M., Fuada, S., Purwanda, I. G., & Ahmad Samhany, H. (2018). Development of Long-range Communication System for Fishermen: An Initial Study. *Proceeding - 2018 International Conference on ICT for Rural Development: Rural Development through ICT: Concept, Design, and Implication, IC-ICTRuDEv 2018*, 12–17. <https://doi.org/10.1109/ICICTR.2018.8706564>
- Damayanti, S. D., Suryanegara, M., Enriko, I. K. A., & Nashiruddin, M. I. (2019). Designing A LoRa-Based Panic Button for Bali Smart Island Project. *2019 7th International Conference on Smart Computing and Communications, ICSCC 2019*, 1–5. <https://doi.org/10.1109/ICSCC.2019.8843614>.
- Hadi, A., & Setiawan, Z. (2020). Pengembangan TinyML pada Raspberry Pi Pico untuk Deteksi Kendaraan. *Jurnal Ilmiah Teknologi*, 7(1), 15–23.

LAMPIRAN



Pemasangan komponen dan kabel-kabel jumper



Pengecekan apakah instalasi sudah terpasang dengan benar



Program

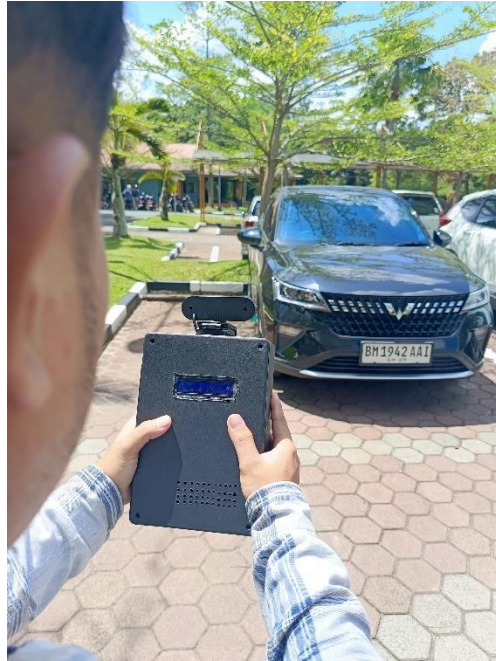
```
deteksi_web.py  main.py x requirements.txt  main.py src  > < ...

main.py > --
1 import cv2
2 import time
3 from ultralytics import YOLO
4 import numpy as np
5 import random
6
7 # === Load model YOLO hasil training ===
8 model = YOLO("best.pt")
9
10 print("✅ Model berhasil dimuat.")
11 print("Nama kelas di model:", model.names)
12
13 # === Buka kamera ===
14 cap = cv2.VideoCapture(0)
15 if not cap.isOpened():
16     raise RuntimeError("❌ Webcam tidak ditemukan!")
17
18 prev_time = time.time()
19
20 while True:
21     ret, frame = cap.read()
22     if not ret:
23         print("⚠️ Gagal membaca frame dari kamera.")
24         break
25
26     frame = cv2.resize(frame, (640, 480))
27
28     # Preprocessing ringan
29     frame_pp = cv2.GaussianBlur(frame, (3, 3), 0)
30     frame_pp = cv2.cvtColor(frame_pp, cv2.COLOR_BGR2RGB)
31
32     results = model.predict(
33         source=frame_pp,
34         conf=0.55,
35         iou=0.35,
36         imgsz=640,
37         stream=False
38     )[0]
39
40     kendaraan_terdeteksi = False
41     label_terdeteksi = ""
42     frame_area = frame.shape[0] * frame.shape[1]
```

```
main.py > ...
44 for box in results.boxes:
45     cls_id = int(box.cls[0])
46     conf = float(box.conf[0])
47     label = model.names[cls_id].lower()
48
49     if label in ['mobil', 'motor']:
50         xyxy = box.xyxy[0].cpu().numpy().astype(int)
51         x1, y1, x2, y2 = xyxy
52         w, h = x2 - x1, y2 - y1
53         box_area = w * h
54
55         if box_area < frame_area * 0.02 or box_area > frame_area * 0.9 or conf < 0.55:
56             continue
57
58         kendaraan_terdeteksi = True
59         label_terdeteksi = label.upper()
60         break # cukup 1 deteksi
61
62 # Simulasi pembacaan daya (karena belum pakai INA219 di VSC)
63 daya_dummy = round(random.uniform(200.0, 400.0), 1) # mW
64
65 if kendaraan_terdeteksi:
66     print(f"Deteksi: {label_terdeteksi} | Daya: {daya_dummy} mW")
67 else:
68     print("Tidak ada kendaraan terdeteksi")
69
70 # Tampilkan setiap 1 detik
71 time.sleep(1)
72
73 # Tekan 'q' untuk keluar
74 if cv2.waitKey(1) & 0xFF == ord('q'):
75     break
76
77 cap.release()
78 print("Program selesai.")
79
```


Pengambilan Data Terang





Pengambilan Data Redup





Pengambilan Data Gelap



