(2026), 4 (3): 218-293 **Scientica** 3021-8209

Jurnal Ilmiah Sain dan Teknologi

KENDALI MOTOR DC PADA SEPEDA LISTRIK SECARA OTOMATIS BERBASIS YOLO V8 DAN ARDUINO UNO

Yessa Vika Franesya [1], Widi Aribowo [2]
Program Studi D4 Teknik Listrik, Fakultas Vokasi
Universitas Negeri Surabaya
yessavika2017@gmail.com

Abstract (English)

The increasing use of electric bicycles requires an intelligent safety system capable of automatically detecting obstacles and controlling vehicle speed to reduce the risk of accidents. This study aims to design and implement an automatic speed control system for an electric bicycle based on object detection using the YOLOv8 algorithm. The proposed system employs a Raspberry Pi 5 for image processing, a USB webcam as the vision sensor, an Arduino Uno R3 as the main controller, an MCP4725 Digital-to-Analog Converter (DAC) for throttle signal generation, an electronic brake relay, and a DFPlayer module for voice warnings. An experimental quantitative research method was applied, including hardware and software development, system implementation, and performance evaluation through object detection accuracy, system latency, and BLDC motor speed control tests based on predefined safety zones. The experimental results demonstrate that the developed system successfully performs real-time detection of pedestrians, motorcycles, and cars and classifies them into STOP, Z3, Z5, Z8, and NONE safety zones for automatic speed control. Object detection tests achieved 100% accuracy for adult male and female subjects up to a distance of 25 meters, while the detection accuracy for children and rear-view motorcycles decreased at longer distances due to reduced image detail and environmental conditions. The system successfully reduced the BLDC motor speed gradually until reaching 0 RPM in the STOP zone for all rider weight variations. The overall system response time ranged from 263.7 ms to 321.3 ms, enabling obstacle response within less than one second. Furthermore, the system was capable of detecting multiple objects simultaneously and consistently prioritizing the nearest object for decision-making. Therefore, the proposed system effectively improves electric bicycle safety through real-time object detection and automatic speed control.

Article History
Submitted: 1 Agustus 2026
Accepted: 3 Agustus 2026
Published: 4 Agustus 2026

Key Words
YOLOv8, Raspberry Pi 5, Arduino Uno R3, electric bicycle, object detection, automatic speed control.

Abstrak (Indonesia)

Peningkatan penggunaan sepeda listrik di Indonesia menuntut adanya sistem keselamatan yang mampu mendeteksi objek dan mengendalikan kecepatan kendaraan secara otomatis untuk mengurangi risiko kecelakaan. Penelitian ini bertujuan merancang dan mengimplementasikan sistem kendali kecepatan otomatis pada sepeda listrik berbasis deteksi objek menggunakan algoritma YOLOv8. Sistem dikembangkan menggunakan Raspberry Pi 5 sebagai pemroses citra, *webcam USB* sebagai sensor visual, Arduino Uno R3 sebagai pengendali utama, modul DAC MCP4725 sebagai pembangkit sinyal throttle analog, relay *e-brake* sebagai sistem pengereman, serta DFPlayer sebagai pemberi peringatan suara. Metode penelitian yang digunakan adalah penelitian kuantitatif eksperimental melalui tahapan perancangan perangkat keras dan perangkat lunak, implementasi sistem, serta pengujian akurasi deteksi objek, latensi sistem, dan pengendalian kecepatan motor BLDC berdasarkan zona keselamatan. Hasil penelitian menunjukkan bahwa sistem berhasil bekerja secara real-time dalam mendeteksi objek manusia, sepeda motor, dan mobil, kemudian mengklasifikasikan objek ke dalam zona keselamatan STOP, Z3, Z5, Z8, dan NONE sebagai dasar pengendalian kecepatan. Hasil pengujian menunjukkan akurasi deteksi objek pria dan wanita mencapai 100% hingga jarak 25 meter, sedangkan akurasi objek anak-anak dan sepeda motor tampak belakang menurun pada jarak yang lebih jauh akibat berkurangnya detail citra dan pengaruh kondisi lingkungan. Sistem juga mampu mengurangi putaran motor secara bertahap hingga mencapai 0 RPM pada zona STOP untuk seluruh variasi berat pengguna. Waktu

Sejarah Artikel
Submitted: 1 Agustus 2026
Accepted: 3 Agustus 2026
Published: 4 Agustus 2026

Kata Kunci
YOLOv8, Raspberry Pi 5, Arduino Uno R3, sepeda listrik, deteksi objek, kendali kecepatan otomatis.

respons sistem berada pada rentang 263,7–321,3 ms, sehingga mampu memberikan respons kurang dari satu detik. Selain itu, sistem mampu mendeteksi lebih dari satu objek secara bersamaan dan memprioritaskan objek dengan jarak terdekat sebagai dasar pengambilan keputusan. Dengan demikian, sistem yang dikembangkan mampu meningkatkan keselamatan berkendara melalui deteksi objek dan pengendalian kecepatan otomatis secara *real-time*.

Pendahuluan

Penggunaan sepeda listrik di Indonesia mengalami peningkatan yang signifikan dalam beberapa tahun terakhir sebagai salah satu alternatif transportasi yang ramah lingkungan dan relatif hemat biaya. Namun, peningkatan popularitas tersebut turut diikuti oleh tantangan dalam aspek keselamatan. Data mencatat bahwa sejak Januari hingga Agustus 2023 terdapat 107 kasus kecelakaan yang melibatkan sepeda listrik. Angka tersebut menunjukkan masih adanya kekurangan dalam aspek pengoperasian, perlengkapan keselamatan, serta pengawasan terhadap perilaku pengguna. Kondisi ini menimbulkan kekhawatiran terhadap keselamatan pengguna dan mendorong perlunya pengembangan solusi teknologi yang mampu merespons dinamika lalu lintas secara real-time (Hermawati et al., 2024).

Untuk meningkatkan efektivitas sistem deteksi objek dalam mendukung keselamatan berkendara, algoritma You Only Look Once versi 8 atau YOLOv8 menjadi salah satu pendekatan yang relevan. Algoritma ini merupakan pengembangan dari versi sebelumnya dengan peningkatan dalam aspek akurasi, efisiensi, dan fleksibilitas. Melalui arsitektur Convolutional Neural Network (CNN) yang telah dikembangkan, YOLOv8 mampu mendeteksi dan mengklasifikasikan objek secara real-time pada kondisi lingkungan yang dinamis. Berdasarkan penelitian Ikbal dan Saputra (2024), YOLOv8 mampu mencapai nilai precision sebesar 0,993 dan recall sebesar 0,999 dalam pengenalan rambu lalu lintas. Oleh karena itu, YOLOv8 dipilih dalam penelitian ini untuk membangun sistem keselamatan otomatis pada sepeda listrik yang mampu mendeteksi objek di sekitar kendaraan secara cepat dan efisien.

Agar proses deteksi menggunakan YOLOv8 dapat berjalan secara optimal, diperlukan perangkat komputasi yang mampu memproses data citra secara real-time. Raspberry Pi 5 dipilih karena memiliki peningkatan kinerja pemrosesan dan efisiensi dibandingkan generasi sebelumnya. Penelitian Prakoso dan Prasetio (2025) menunjukkan bahwa Raspberry Pi 5 mampu menjalankan model Gated Recurrent Unit (GRU) untuk pengenalan suara secara real-time dengan tingkat akurasi yang tinggi, termasuk pada kondisi lingkungan yang bising. Hal tersebut menunjukkan bahwa Raspberry Pi 5 memiliki kemampuan yang memadai untuk menjalankan model kecerdasan buatan pada sistem tertanam secara responsif dan efisien.

Dalam perancangan sistem deteksi objek, pemilihan sensor visual menjadi salah satu faktor yang menentukan kualitas citra dan akurasi deteksi. Webcam USB dipilih karena memiliki harga yang relatif terjangkau, mudah diperoleh, serta mendukung pemasangan secara plug-and-play. Perangkat ini juga memiliki kompatibilitas yang luas dengan berbagai sistem operasi dan pustaka pengolahan citra, sehingga memudahkan proses integrasi dengan Raspberry Pi 5. Penelitian Abdullah et al. (2021) menunjukkan bahwa penggunaan webcam dapat mendukung pelacakan objek secara real-time dengan hasil yang memadai, sedangkan Nikouei et al. (2018) menunjukkan bahwa sistem berbasis kamera dapat digunakan untuk mendukung proses deteksi manusia secara efisien. Berdasarkan pertimbangan tersebut, webcam USB dinilai sesuai untuk digunakan sebagai sensor visual dalam sistem deteksi objek berbasis YOLOv8.

Sistem pengendalian kecepatan memiliki peranan penting dalam memberikan respons terhadap objek yang terdeteksi di depan sepeda listrik. Dalam penelitian ini, Arduino Uno R3 digunakan sebagai unit kontrol yang menerima perintah zona jarak dari Raspberry Pi 5.

Arduino Uno R3 tidak mengendalikan proses komutasi motor BLDC secara langsung, melainkan mengatur nilai digital yang dikirimkan menuju modul Digital-to-Analog Converter (DAC) MCP4725 serta mengendalikan relay e-brake. Kecepatan motor tetap dikendalikan oleh controller bawaan sepeda listrik berdasarkan tegangan yang diterima pada jalur sinyal throttle. Dengan demikian, pengendalian kecepatan dilakukan dengan mengubah tegangan sinyal throttle melalui keluaran analog DAC sesuai dengan zona jarak objek yang terdeteksi.

Penggunaan DAC diperlukan karena Arduino Uno R3 tidak memiliki keluaran tegangan analog murni. Modul MCP4725 merupakan DAC 12-bit yang mampu mengubah data digital dari Arduino menjadi tegangan analog. Supriyo et al. (2024) menunjukkan bahwa Arduino Uno yang diintegrasikan dengan DAC MCP4725 dapat menghasilkan tegangan analog sebesar 0–5 V berdasarkan nilai digital yang diberikan, dengan rata-rata kesalahan sebesar 0,83%. Hasil tersebut menunjukkan bahwa MCP4725 dapat digunakan untuk menghasilkan keluaran tegangan analog yang terukur dan proporsional terhadap data yang diproses oleh Arduino.

Pengaturan tegangan pada sinyal throttle relevan untuk digunakan karena perubahan tegangan throttle berpengaruh terhadap kinerja motor BLDC. Penelitian Wardana dan Amalia (2024) menunjukkan bahwa variasi tegangan keluaran throttle memengaruhi kecepatan putar motor BLDC dan konsumsi arus pada kendaraan listrik. Penelitian tersebut tercatat pada halaman Google Scholar penulis yang ditampilkan melalui SINTA serta memiliki DOI dan publikasi jurnal yang dapat ditelusuri. Berdasarkan hasil tersebut, keluaran DAC dalam penelitian ini digunakan untuk menggantikan dan mengatur tegangan sinyal throttle menuju controller motor BLDC. Ketika jarak objek semakin dekat, Arduino Uno R3 menurunkan nilai keluaran DAC secara bertahap sehingga tegangan sinyal throttle dan kecepatan kendaraan turut berkurang. Ketika objek memasuki zona STOP, keluaran DAC ditetapkan pada nilai minimum dan relay e-brake diaktifkan untuk menghentikan penggerak sepeda listrik.

Integrasi algoritma YOLOv8 yang dijalankan pada Raspberry Pi 5, sistem akuisisi citra menggunakan webcam USB, serta sistem pengendalian kecepatan menggunakan Arduino Uno R3, DAC MCP4725, dan relay e-brake membentuk suatu sistem keselamatan yang terintegrasi. Sistem ini dirancang untuk mendeteksi objek, memperkirakan jarak, menentukan zona keselamatan, serta menyesuaikan tegangan sinyal throttle secara otomatis berdasarkan jarak objek di depan kendaraan. Dengan demikian, penerapan sistem tersebut diharapkan dapat membantu mengurangi risiko tabrakan dan meningkatkan keselamatan pengguna sepeda listrik. Penelitian ini juga dapat menjadi dasar bagi pengembangan sistem kendaraan cerdas yang lebih adaptif melalui penambahan sensor, peningkatan metode estimasi jarak, dan penerapan metode pengendalian yang lebih lanjut.

Metode Penelitian

Penelitian ini merupakan kuantitatif eksperimental yang mengembangkan dan menguji sistem kendali motor DC pada sepeda listrik berbasis YOLO v8 dan Arduino Uno. Pengujian dilakukan dengan memanipulasi jarak deteksi objek dan menganalisis respons kecepatan motor secara numerik.

Hasil dan Pembahasan

Hasil Penelitian

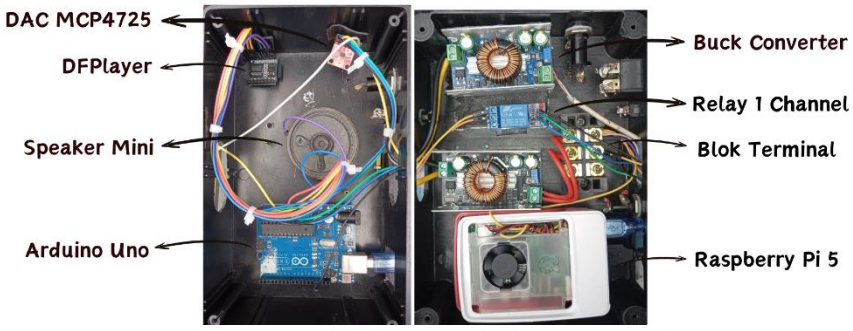
Pada penelitian ini telah berhasil dirancang dan direalisasikan sistem kendali kecepatan otomatis pada sepeda listrik berbasis deteksi objek menggunakan algoritma YOLOv8 dan Arduino Uno. Sistem yang dikembangkan mengintegrasikan perangkat keras dan perangkat lunak sehingga mampu bekerja secara real-time dalam mendeteksi objek serta mengendalikan kecepatan motor BLDC sesuai kondisi di depan kendaraan. Hasil perancangan hardware menunjukkan bahwa seluruh komponen utama, yaitu Raspberry Pi 5, Arduino Uno, webcam,

modul DAC, relay, serta DFPlayer dan speaker, dapat terhubung dan bekerja secara terintegrasi. Sistem catu daya menggunakan baterai lithium-ion dan buck converter juga mampu menyuplai tegangan secara stabil ke seluruh rangkaian. Dari sisi perangkat lunak, sistem berbasis YOLOv8 pada Raspberry Pi 5 berhasil mendeteksi objek seperti manusia, sepeda motor, dan mobil secara real-time. Hasil deteksi digunakan untuk menghitung estimasi jarak objek, kemudian diklasifikasikan ke dalam zona keselamatan yaitu STOP, Z3, Z5, Z8, dan NONE sebagai dasar pengambilan keputusan.

Komunikasi antara Raspberry Pi dan Arduino melalui serial berjalan dengan baik. Arduino mampu mengolah data zona untuk mengatur kecepatan motor BLDC secara bertahap melalui modul DAC serta mengaktifkan sistem pengereman (E-Brake) menggunakan relay ketika objek berada pada jarak berbahaya. Selain itu, sistem juga memberikan peringatan suara melalui speaker sebagai notifikasi kepada pengguna. Berdasarkan hasil pengujian, sistem dapat merespons kondisi lingkungan secara adaptif, yaitu kendaraan berjalan normal saat tidak ada objek, melambat ketika objek terdeteksi dalam jarak tertentu, dan berhenti otomatis saat jarak terlalu dekat. Secara keseluruhan, sistem yang dikembangkan telah berjalan sesuai dengan perancangan dan mampu meningkatkan aspek keselamatan pada sepeda listrik.

Hasil Perancangan Hardware

Rancangan alat yang dikembangkan terdiri dari sepeda listrik sebagai media implementasi serta dua kotak kontrol yang berisi rangkaian elektronik pendukung. Komponen-komponen tersebut berfungsi untuk pemrosesan citra, pengendalian motor DC, serta pemberian peringatan pada sistem. Gambar 3.1 merupakan hasil perancangan hardware pada alat :



Gambar 3. 1 Keseluruhan Sepeda dengan Kontrol
(***Sumber**: Dokumen Pribadi*)

Gambar 3. 2 Kotak Kontrol Pertama dan Kedua
(***Sumber :** Dokumen Pribadi)*

Pada kotak gambar kotak kontrol pertama terdapat komponen Arduino Uno, modul DAC MCP4725, DFPlayer Mini, dan speaker. Arduino Uno menerima data dari Raspberry Pi 5 melalui komunikasi serial (UART) untuk mengendalikan perangkat keluaran (output device) pada sistem. Arduino berkomunikasi dengan modul DAC menggunakan protokol I2C untuk menghasilkan tegangan analog 0–5 volt sebagai sinyal throttle menuju ESC (Electronic Speed Controller) guna mengatur kecepatan motor BLDC. Selain itu, Arduino juga terhubung dengan modul DFPlayer Mini melalui komunikasi serial untuk memutar audio yang dikeluarkan melalui speaker sebagai peringatan suara.

Sementara itu, pada kotak gambar kotak kontrol kedua, terdapat komponen yang berfungsi sebagai bagian pemrosesan utama dan pengaturan catu daya, yakni Raspberry Pi 5, dua buah modul buck converter, modul relay 1 channel, dan terminal distribusi tegangan. Raspberry Pi 5 terhubung dengan webcam melalui USB untuk melakukan proses deteksi objek. Raspberry Pi memperoleh suplai tegangan dari buck converter pertama, sedangkan buck converter kedua menyuplai tegangan ke modul DAC MCP4725. Relay 1 channel digunakan sebagai pengaktif sistem pengereman elektronik (E-Brake) ketika menerima sinyal dari Arduino, sementara terminal berfungsi sebagai titik distribusi tegangan ke setiap komponen dalam sistem.

Hasil Perancangan Software Pengaturan Kecepatan

Perancangan perangkat lunak pada penelitian ini dilakukan menggunakan bahasa pemrograman Python yang dijalankan pada Raspberry Pi. Perancangan ini berfungsi untuk melakukan deteksi objek secara real-time menggunakan algoritma YOLO, menghitung estimasi jarak objek dari kamera, menentukan zona keselamatan berdasarkan jarak objek, serta mengirimkan perintah kendali ke Arduino melalui komunikasi serial. Proses tersebut berlangsung secara berulang dimulai dari pengambilan citra oleh kamera, proses deteksi objek, perhitungan jarak objek, hingga penentuan tindakan sistem sesuai kondisi yang terdeteksi.

Penggunaan Library pada Sistem

Program yang dirancang menggunakan beberapa library Python untuk mendukung proses deteksi objek, pengolahan citra, komunikasi serial, serta pencatatan data sistem. Library yang digunakan dalam program ini dapat dilihat pada potongan kode berikut:

```
from ultralytics import YOLO
import cv2
import time
import serial
import glob
import statistics
import csv
import datetime
from collections import deque
```

Library YOLO digunakan untuk melakukan deteksi objek berbasis deep learning, sedangkan cv2 digunakan untuk mengambil dan menampilkan video dari webcam. Library lain seperti statistics digunakan untuk menghitung nilai median sebagai proses smoothing data jarak, dan deque digunakan untuk menyimpan histori jarak dalam jumlah terbatas agar perhitungan lebih stabil. Beberapa library seperti serial, csv, dan datetime disiapkan untuk pengembangan lanjutan seperti komunikasi perangkat atau pencatatan data.

Konfigurasi Parameter Sistem

Pada tahap awal program dilakukan pengaturan beberapa parameter yang digunakan dalam sistem deteksi objek dan komunikasi perangkat keras. Parameter ini meliputi indeks kamera, model YOLO yang digunakan, nilai ambang batas kepercayaan deteksi objek, serta

kecepatan komunikasi serial dengan Arduino. Potongan kode konfigurasi parameter tersebut adalah sebagai berikut:

```
CAMERA_INDEX = 0
MODEL_PATH = "yolov8n.pt"
CONF_THRESHOLD = 0.45

BAUDRATE = 115200
SAFETY_RELEASE_DELAY = 3.0
```

Selain itu, sistem juga menentukan batas jarak untuk setiap zona keselamatan sebagai dasar pengambilan keputusan sekaligus pengaturan awal agar sistem dapat berjalan sesuai dengan rancangan yang telah ditentukan.

```
ZONE_DISTANCE_LIMITS = {
   "STOP": 2.5,
   "Z3":   5.0,
   "Z5":   8.0,
   "Z8":   12.0
}
```

Implementasi Deteksi Objek YOLO

Sistem melakukan proses deteksi objek secara real-time menggunakan model YOLO. Kamera akan mengambil citra yang kemudian diproses oleh model YOLO untuk mendeteksi objek seperti manusia, sepeda motor, dan mobil. Hasil deteksi berupa bounding box yang menunjukkan posisi objek pada citra. Berikut merupakan potongan kode proses deteksi objek:

```
model = YOLO(MODEL_PATH)

results = model.predict(
   frame,
   imgsz=640,
   conf=CONF_THRESHOLD,
   classes=[0, 2, 3, 5, 7],
   verbose=False
)
```

Hasil deteksi objek tersebut kemudian digunakan sebagai dasar untuk melakukan perhitungan jarak objek terhadap kamera.

Perhitungan Estimasi Jarak

Setelah objek terdeteksi, sistem melakukan estimasi jarak objek berdasarkan tinggi bounding box yang diperoleh dari citra kamera. Perhitungan jarak dilakukan menggunakan persamaan kalibrasi yang telah ditentukan sebelumnya untuk setiap jenis objek. Potongan kode yang digunakan untuk menghitung estimasi jarak objek adalah sebagai berikut:

```
def estimate_distance(h_px, class_name):
   if h_px < 1:
      return 999.0
   coeffs = CALIBRATION_COEFFS.get(class_name,
DEFAULT_COEFF)
```

```
return coeffs["A"] * (h_px ** coeffs["B"])
```

Untuk menstabilkan nilai jarak, sistem menggunakan metode median dari beberapa data jarak terakhir guna mengurangi fluktuasi akibat noise pada proses deteksi objek.

```
dist_hist = deque(maxlen=5)
smooth_dist = statistics.median(dist_hist)
```

Penentuan Zona Keselamatan

Berdasarkan nilai jarak yang diperoleh, sistem kemudian menentukan zona keselamatan kendaraan. Zona ini digunakan sebagai dasar pengambilan keputusan untuk mengatur kecepatan motor maupun mengaktifkan sistem pengereman. Berikut merupakan potongan kode penentuan zona keselamatan:

```
def decide_zone(dist):
    if dist < ZONE_DISTANCE_LIMITS["STOP"]:
        return "STOP"
    elif dist < ZONE_DISTANCE_LIMITS["Z3"]:
        return "Z3"
    elif dist < ZONE_DISTANCE_LIMITS["Z5"]:
        return "Z5"
    elif dist < ZONE_DISTANCE_LIMITS["Z8"]:
        return "Z8"
    else:
        return "NONE"
```

Setiap zona keselamatan memiliki nilai DAC yang berbeda untuk mengatur kecepatan motor melalui Arduino.

```
ZONE_TO_DAC = {
    "NONE": 3000,
    "Z8":   2000,
    "Z5":   1500,
    "Z3":   950,
    "STOP": 0
}
```

Komunikasi Serial dengan Arduino

Setelah zona keselamatan ditentukan, sistem akan mengirimkan perintah kendali ke Arduino melalui komunikasi serial. Perintah ini digunakan untuk mengatur nilai DAC sebagai pengendali kecepatan motor serta mengaktifkan sistem pengereman elektronik (E-Brake) ketika objek berada pada jarak yang berbahaya. Berikut merupakan potongan kode komunikasi serial pada program:

```
def send_dac(ser, value):
    try:
        ser.write(f"{value}\n".encode())
    except:
        pass
```

Selain itu, sistem juga mengirimkan perintah untuk mengaktifkan atau menonaktifkan

pengereman melalui komunikasi serial sehingga sistem deteksi objek pada Raspberry Pi dapat mengendalikan perangkat keras yang terhubung dengan Arduino.

```
def send_brake(ser, on):
    try:
        ser.write(b"b\n" if on else b"r\n")
    except:
        pass
```

Hasil Perancangan Software Akurasi Deteksi

Penggunaan Library

Bagian ini bertugas memanggil semua library yang dibutuhkan oleh program agar dapat berjalan. Library seperti cv2 digunakan untuk mengakses webcam dan mengambil frame gambar secara real-time, sedangkan YOLO dari Ultralytics digunakan untuk melakukan proses deteksi objek pada gambar tersebut. Selain itu, Path disiapkan untuk pengelolaan direktori penyimpanan, dan csv disiapkan jika hasil pengujian ingin disimpan ke file. Berikut merupakan potongan kode penggunaan library :

```
import time
import csv
from pathlib import Path

import cv2
from ultralytics import YOLO
```

Konfigurasi Parameter

Bagian konfigurasi berfungsi sebagai pusat pengaturan parameter yang digunakan dalam program. Di sini ditentukan sumber kamera, resolusi frame, serta model YOLO yang akan digunakan. Selain itu, bagian ini juga mengatur kelas objek yang akan diuji, daftar jarak pengujian, jumlah sampel frame untuk setiap jarak, serta batas minimal confidence untuk menentukan apakah suatu deteksi dianggap valid. Dengan adanya konfigurasi ini, pengguna dapat mengubah perilaku program tanpa harus mengedit logika utama. Berikut merupakan potongan kode konfigurasi parameter :

```
CAMERA_INDEX = 0
FRAME_WIDTH = 1280
FRAME_HEIGHT = 720
MODEL_PATH = "yolov8n.pt"

TARGET_CLASSES = {
   "person": "Manusia",
    "car": "Mobil",
    "motorcycle": "Sepeda motor",
}
CLASS_WHITELIST=set(TARGET_CLASSES.keys())
DISTANCES_M = [10, 15, 20, 25, 30]
SAMPLES_PER_DISTANCE = 50
CONF_THRESHOLD = 0.3
SAVE_CSV = True
```

Fungsi Pilih Objek

Fungsi ini bertugas untuk menerima input dari pengguna terkait objek apa yang ingin diuji. Program akan menampilkan pilihan yang tersedia, kemudian menunggu pengguna memasukkan angka yang sesuai. Jika input valid, fungsi akan mengembalikan nama kelas objek sesuai format yang digunakan oleh model YOLO. Jika tidak valid, program akan terus meminta input hingga pengguna memberikan pilihan yang benar. Berikut merupakan potongan kode fungsi pilih objek :

```
def pilih_objek():
    print("=== PILIH OBJEK UJI ===")
    print("1. Manusia (person)")
    print("2. Mobil (car)")
    print("3. Sepeda motor (motorcycle)")

    while True:
        pilihan = input("Masukkan pilihan (1/2/3): ").strip()

        if pilihan == "1":
            return "person"
        elif pilihan == "2":
            return "car"
        elif pilihan == "3":
            return "motorcycle"
        else:
            print("Pilihan tidak valid")
```

Inisialisasi Kamera

Bagian ini bertanggung jawab untuk membuka dan menyiapkan webcam sebelum digunakan dalam proses pengujian. Program akan mencoba mengakses kamera berdasarkan indeks yang telah ditentukan, lalu mengatur resolusi frame sesuai konfigurasi. Jika kamera tidak berhasil dibuka, fungsi akan menghentikan program dengan memberikan error, sehingga mencegah proses berikutnya berjalan tanpa sumber gambar yang valid. Berikut merupakan potongan kode inisialisasi kamera :

```
def init_camera():
    cap = cv2.VideoCapture(CAMERA_INDEX)
    cap.set(cv2.CAP_PROP_FRAME_WIDTH, FRAME_WIDTH)
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, FRAME_HEIGHT)

    if not cap.isOpened():
        raise RuntimeError("Webcam gagal dibuka")

    return cap
```

Load Model YOLO

Fungsi ini digunakan untuk memuat model YOLO yang akan digunakan dalam proses deteksi objek. Model diambil dari file yang telah ditentukan pada konfigurasi, dan kemudian dikembalikan agar bisa digunakan di bagian lain program. Dengan memisahkan proses ini, model hanya perlu dimuat sekali sehingga lebih efisien. Berikut merupakan potongan kode load model yolo :

```
def load_model():
    return YOLO(MODEL_PATH)
```

Deteksi Objek Dalam Frame

Fungsi ini melakukan proses utama deteksi objek pada setiap frame yang diambil dari kamera. Frame akan diproses oleh model YOLO untuk menghasilkan daftar objek yang terdeteksi beserta nilai confidence-nya. Program kemudian memeriksa setiap objek untuk memastikan bahwa objek tersebut termasuk dalam daftar yang diizinkan, sesuai dengan target yang sedang diuji, dan memiliki confidence di atas ambang batas. Jika ditemukan objek yang memenuhi semua kriteria, fungsi akan mengembalikan nilai benar sebagai indikasi bahwa deteksi berhasil. Berikut merupakan potongan kode deteksi objek dalam frame :

```
def deteksi_target(model, frame, target_class_name):
    results = model(frame, imgsz=640, conf=CONF_THRESHOLD)[0]

    for box in results.boxes:
        cls_id = int(box.cls[0])
        conf = float(box.conf[0])
        cls_name=model.names.get(cls_id, str(cls_id))

        if cls_name not in CLASS_WHITELIST:
            continue

        if cls_name == target_class_name and conf >= CONF_THRESHOLD:
            return True

    return False
```

Loop Pengujian Tiap Jarak

Bagian ini bertugas melakukan pengujian akurasi pada satu jarak tertentu. Program akan meminta pengguna untuk memposisikan objek pada jarak yang ditentukan, kemudian mulai mengambil sejumlah frame sesuai jumlah sampel. Setiap frame dianalisis untuk melihat apakah objek target berhasil dideteksi. Hasil deteksi yang benar akan dihitung dan dibandingkan dengan total frame untuk menghasilkan nilai akurasi secara bertahap hingga semua sampel selesai diproses. Berikut merupakan potongan kode loop pengujian tiap jarak:

```
def uji_per_jarak(cap, model, target_class_name, distance):
    input(f"Posisikan objek di {distance}m lalu tekan ENTER...")

    total = 0
    correct = 0

    while total < SAMPLES_PER_DISTANCE:
        ret, frame = cap.read()
        if not ret:
            continue
        found = deteksi_target(model, frame, target_class_name)

        total += 1
```

```
        if found:
            correct += 1

        accuracy = (correct / total) * 100
        print(f"{distance}m | {total} | {accuracy:.2f}%")

    return accuracy
```

Fungsi Utama Pengujian

Fungsi ini mengatur keseluruhan alur pengujian dari awal hingga akhir. Program akan menampilkan informasi pengujian, memuat model, dan mengaktifkan kamera. Setelah itu, dilakukan proses pemanasan kamera agar hasil frame stabil. Selanjutnya program menjalankan pengujian untuk setiap jarak yang telah ditentukan, lalu menyimpan hasil akurasi ke dalam list. Di akhir proses, kamera akan ditutup untuk memastikan tidak ada resource yang tertinggal. Berikut merupakan potongan kode fungsi utama pengujian :

```
def uji_akurasi(target_class_name):
    target_label = TARGET_CLASSES[target_class_name]
print("\n====================================")
    print("Mulai uji akurasi")
    print("Objek :", target_label)
    print("Model :", MODEL_PATH)
    print("====================================")

    model = load_model()
    cap = init_camera()

    results = []

    try:
        for _ in range(30):
            cap.read()

        for distance in DISTANCES_M:
            acc = uji_per_jarak(cap, model, target_class_name, distance)
            results.append((distance, acc))

        return results

    finally:
        cap.release()
        cv2.destroyAllWindows()
```

Program Sistem Utama

Bagian ini berfungsi sebagai pengendali utama program. Program akan memulai dengan meminta pengguna memilih objek yang akan diuji, kemudian menjalankan proses pengujian akurasi berdasarkan pilihan tersebut. Setelah seluruh proses selesai, hasil akurasi untuk setiap jarak akan ditampilkan kepada pengguna dalam bentuk ringkasan. Berikut merupakan potongan kode program sistem utama :

```
def main():
    target = pilih_objek()
    hasil = uji_akurasi(target)

    print("\n=== HASIL AKHIR ===")
    for d, acc in hasil:
        print(f"{d}m : {acc:.2f}%")
```

Eksekusi Program

Bagian ini memastikan bahwa fungsi utama hanya dijalankan ketika file Python dieksekusi secara langsung, bukan ketika diimpor sebagai modul di file lain. Dengan demikian, struktur program menjadi lebih fleksibel dan dapat digunakan kembali jika diperlukan dalam proyek yang lebih besar. Berikut merupakan potongan kode eksekusi program :

```
if __name__ == "__main__":
    main()
```

Hasil Perancangan Software Modul DAC dan DFPlayer

Penggunaan Library

Bagian ini bertugas memanggil semua library yang diperlukan untuk mengendalikan perangkat keras. Library Wire digunakan untuk komunikasi I2C dengan modul DAC MCP4725, sedangkan Adafruit_MCP4725 digunakan untuk mengatur output analog ke motor controller. Library SoftwareSerial memungkinkan komunikasi serial tambahan selain port utama, yang digunakan untuk berkomunikasi dengan modul DFPlayer. Sementara itu, DFRobotDFPlayerMini digunakan untuk mengontrol pemutaran file audio sebagai sistem peringatan suara. Berikut merupakan potongan kode penggunaan library :

```
#include <Wire.h>
#include <Adafruit_MCP4725.h>
#include <SoftwareSerial.h>
#include <DFRobotDFPlayerMini.h>
```

Konfigurasi Pin

Bagian ini mendefinisikan koneksi antar komponen dengan mikrokontroler. Pin A1 dan A2 digunakan sebagai jalur komunikasi serial ke DFPlayer, sementara pin A0 digunakan untuk membaca input throttle (gas). Pin A3 digunakan untuk mengontrol relay yang berfungsi sebagai sistem pengereman elektronik, dan pin 13 digunakan sebagai indikator LED untuk debugging. Selain itu, objek untuk DFPlayer dan DAC juga dibuat agar dapat digunakan di seluruh program. Berikut merupakan potongan kode konfigurasi pin :

```
SoftwareSerial mySoftwareSerial(A1, A2);

DFRobotDFPlayerMini myDFPlayer;
Adafruit_MCP4725 dac;

const int throttlePin = A0;
const int relayPin = A3;
const int ledDebugPin = 13;
```

Variabel Sistem

Bagian ini berisi variabel yang digunakan untuk menyimpan kondisi sistem selama program berjalan. Nilai input throttle dari pengguna akan dibaca dan diolah menjadi nilai yang sesuai dengan DAC. Variabel raspiLimit berfungsi sebagai batas maksimum yang dapat dikirim ke motor, biasanya berasal dari sistem lain seperti Raspberry Pi. Variabel isBraking menentukan apakah sistem sedang dalam kondisi pengereman, sedangkan finalOutput adalah nilai akhir yang dikirim ke DAC. Berikut merupakan potongan kode variabel sistem :

```
int userThrottleInput = 0;
int userThrottleMapped = 0;
int raspiLimit = 3000;
int finalOutput = 0;
bool isBraking = false;
unsigned long lastDebugTime = 0;
```

Buffer Input Serial

Bagian ini digunakan untuk menampung data yang diterima melalui komunikasi serial dalam bentuk karakter. Data ini biasanya berupa angka yang dikirim sebagai limit kecepatan. Buffer ini akan mengumpulkan karakter satu per satu hingga membentuk angka utuh, kemudian dikonversi menjadi nilai integer untuk diproses lebih lanjut. Berikut merupakan potongan kode buffer input serial :

```
char inputBuffer[10];
byte indexBuf = 0;
```

Fungsi setup

Bagian awal fungsi setup berfungsi untuk menginisialisasi komunikasi serial utama dengan kecepatan tinggi agar responsif terhadap data dari komputer atau Raspberry Pi. Selain itu, komunikasi serial kedua untuk DFPlayer juga diaktifkan, serta komunikasi I2C untuk DAC dimulai. Berikut merupakan potongan kode fungsi setup :

```
void setup() {
  Serial.begin(115200);
  mySoftwareSerial.begin(9600);
  Wire.begin();
```

Inisialisasi DAC, DFPlayer, dan Output

Bagian ini mengecek apakah modul DAC berhasil terhubung melalui alamat I2C. Jika tidak terdeteksi, sistem akan memberikan pesan error melalui serial monitor, namun tetap melanjutkan program agar sistem tidak berhenti total. Kemudian memastikan modul DFPlayer siap digunakan untuk memutar suara. Jika modul berhasil terhubung, maka volume akan diatur ke tingkat tertentu. Jika gagal, sistem akan menampilkan pesan error agar pengguna dapat memeriksa koneksi atau SD card. Lalu bagian inisialisasi output berfungsi untuk mengatur mode pin sebagai output dan menetapkan kondisi awal sistem. Relay diset ke kondisi tidak aktif (mode jalan normal), LED dimatikan, dan output DAC diatur ke nol agar motor tidak bergerak saat awal sistem dinyalakan. Berikut merupakan potongan kode Inisialisasi DAC, DFPlayer dan Output :

```
  if (!dac.begin(0x60)) {
    Serial.println("!!! ERROR: DAC MCP4725 TIDAK TERDETEKSI !!!");
```

```
}
if (!myDFPlayer.begin(mySoftwareSerial)) {
  Serial.println(F("!!! ERROR: DFPlayer TIDAK KONEK (Cek Kabel/SD Card) !!!"));
} else {
  Serial.println(F("OK: DFPlayer Siap."));
  myDFPlayer.volume(25);
}
pinMode(relayPin, OUTPUT);
pinMode(ledDebugPin, OUTPUT);

digitalWrite(relayPin, HIGH);
digitalWrite(ledDebugPin, LOW);
dac.setVoltage(0, false);
```

Pembacaan Perintah Serial

Bagian ini terus memantau apakah ada data yang masuk melalui serial. Setiap karakter yang diterima akan dibaca satu per satu untuk kemudian diproses sesuai dengan jenis perintah yang diberikan. Berikut merupakan potongan kode pembacaan perintah serial :

```
while (Serial.available()) {
  char c = Serial.read();
```

Fitur Suara dan Rem

Bagian fitur suara berfungsi menangani perintah untuk memutar suara berdasarkan jarak tertentu. Ketika pengguna mengirim karakter tertentu, sistem akan memutar file audio yang sesuai melalui DFPlayer, sehingga dapat digunakan sebagai peringatan suara berbasis jarak. Sedangkan bagian fitur rem berfungsi mengatur kondisi pengereman sistem. Ketika perintah rem diterima, sistem akan mengaktifkan mode braking dengan cara mengaktifkan relay dan membatasi output motor menjadi nol. Sebaliknya, ketika perintah pelepasan rem diterima, sistem akan kembali ke kondisi normal. Berikut merupakan potongan kode fitur suara dan rem :

```
if (c == 'L') { ... }
else if (c == 'b') { ... }
```

Input Angka (Limit Speed)

Bagian ini mengumpulkan karakter angka yang dikirim melalui serial untuk membentuk nilai batas kecepatan. Setelah angka lengkap diterima, nilai tersebut akan dikonversi menjadi integer dan digunakan sebagai batas maksimum output motor. Berikut merupakan potongan kode input angka (limit speed) :

```
else if (isDigit(c)) { ... }
```

Logika Kontrol Hardware

Bagian ini merupakan inti kontrol sistem. Jika sistem dalam kondisi pengereman, maka relay akan diaktifkan dan output motor dihentikan. Jika tidak, sistem akan membaca input throttle dari pengguna, mengubahnya menjadi skala DAC, lalu membandingkannya dengan batas maksimum yang diberikan. Nilai yang lebih kecil akan dipilih sebagai output akhir untuk menjaga keamanan. Berikut merupakan potongan kode logika kontrol hardware:

```
if (isBraking) { ... }
else { ... }
```

Output ke DAC

Bagian ini mengirimkan nilai akhir yang telah dihitung ke DAC, yang kemudian diteruskan sebagai sinyal analog ke motor controller. Nilai ini menentukan seberapa besar kecepatan motor yang dihasilkan. Berikut merupakan potongan kode output ke DAC :

```
dac.setVoltage(finalOutput, false);
```

Monitoring Serial dan Delay Loop

Bagian ini menampilkan informasi kondisi sistem secara berkala setiap satu detik. Informasi yang ditampilkan meliputi nilai throttle, batas kecepatan, output DAC, dan status pengereman, sehingga memudahkan proses monitoring dan debugging. Delay loop untuk memberikan jeda singkat dalam loop utama untuk menjaga stabilitas sistem dan mencegah pembacaan data yang terlalu cepat, yang dapat menyebabkan beban berlebih pada mikrokontroler. Berikut merupakan potongan kode monitoring serial dan delay loop :

```
if (millis() - lastDebugTime > 1000) {
delay(10);
```

Pengujian Alat

Pengujian alat dilakukan untuk memastikan bahwa sistem kendali kecepatan sepeda listrik berbasis YOLOv8 dan Arduino Uno dapat bekerja sesuai dengan perancangan. Pengujian yang dilakukan meliputi pengujian akurasi deteksi objek dan pengujian respon sistem untuk kontrol kecepatan motor berdasarkan jarak.

Pengujian Akurasi Deteksi Objek

Pengujian akurasi dilakukan untuk mengetahui tingkat keberhasilan sistem dalam mendeteksi objek menggunakan algoritma YOLOv8. Pengujian ini dilakukan dengan menggunakan webcam sebagai input dan Raspberry Pi 5 sebagai pemroses data. Variabel yang digunakan dalam pengujian adalah jarak objek manusia, motor, dan mobil terhadap kamera, yaitu pada jarak 2 meter, 5 meter, 10 meter, 15 meter, 20 meter, 25 meter, dan 30 meter.

Pada setiap variasi jarak, dilakukan pengambilan sebanyak 50 sampel data berupa citra (frame) yang kemudian diproses oleh sistem untuk dideteksi objeknya. Hasil deteksi kemudian akan dihitung berapa kali deteksi benar dalam 50 sampel. Perhitungan nilai akurasi pada sistem ini dilakukan secara otomatis di dalam program menggunakan rumus :

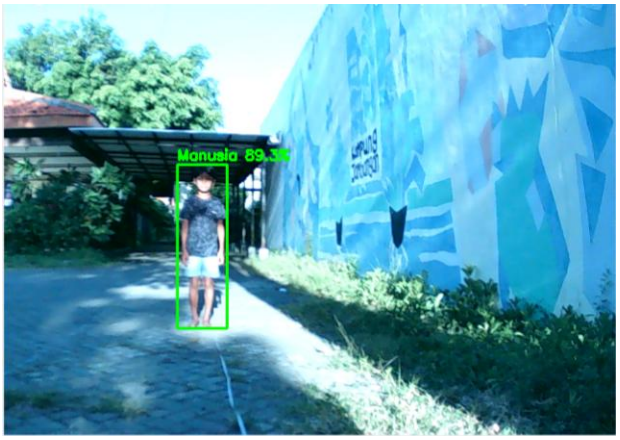
$$\text{Akurasi (\%)} = \frac{\text{Jumlah Deteksi Benar}}{\text{Total Sampel}} \times 100\%$$

Rumus tersebut diimplementasikan langsung pada kode program untuk menghitung persentase akurasi berdasarkan jumlah deteksi yang benar dari total sampel pada setiap pengujian jarak. Pengujian dilakukan pada tiga jenis objek yaitu manusia, mobil dan motor. Untuk akurasi manusia dilakukan pada anak – anak usia 11 tahun, pria usia 24 tahun, dan wanita usia 23 tahun. Hasil pengujian akurasi untuk objek anak – anak dapat dilihat pada Tabel 3.1 berikut :

Tabel 3. 1 Pengujian Akurasi Deteksi Anak – Anak

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%
15	50	50	100%
20	50	43	86%
25	50	13	26%
30	50	1	2%

Gambar 3. 3 Deteksi Anak - Anak Jarak 2 Meter
(***Sumber :*** *Dokumen Pribadi)*

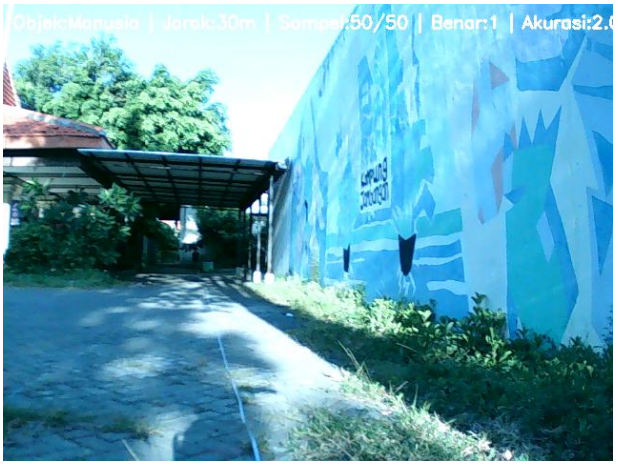


Gambar 3. 4 Deteksi Anak - Anak Jarak 5 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 5 Deteksi Anak - Anak Jarak 10 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 6 Deteksi Anak - Anak Jarak 15 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 7 Deteksi Anak - Anak Jarak 20 Meter
(***Sumber :*** *Dokumen Pribadi)*



Gambar 3. 8 Deteksi Anak - Anak Jarak 25 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 9 Deteksi Anak - Anak Jarak 30 Meter
(***Sumber :*** *Dokumen Pribadi)*

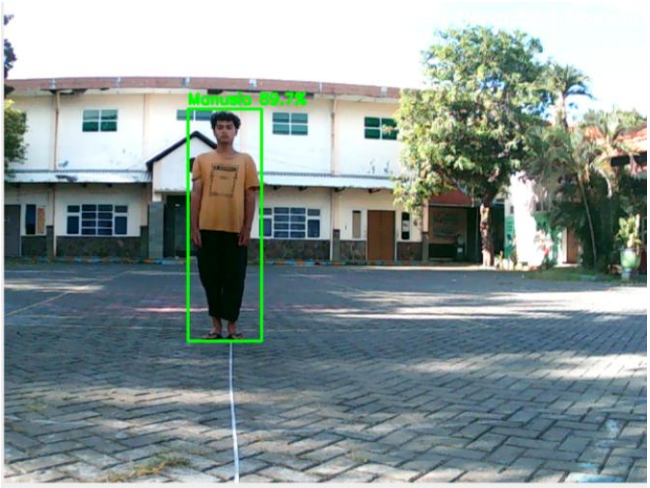
Berdasarkan Tabel 3.1, hasil pengujian akurasi deteksi objek anak-anak usia 11 tahun menunjukkan bahwa sistem memiliki performa yang sangat baik pada jarak 2 meter hingga 15 meter. Pada rentang jarak tersebut, seluruh sampel yang diuji sebanyak 50 objek berhasil terdeteksi dengan benar, sehingga menghasilkan tingkat akurasi sebesar 100%. Hal ini menunjukkan bahwa sistem mampu mengenali objek anak-anak secara konsisten dan stabil pada jarak pengujian tersebut tanpa mengalami kesalahan deteksi.

Namun, pada jarak 20 meter hingga 30 meter terjadi penurunan performa deteksi. Pada jarak 20 meter, sebanyak 43 dari 50 sampel berhasil terdeteksi dengan benar sehingga diperoleh akurasi sebesar 86%. Pada jarak 25 meter, hanya 13 objek yang berhasil terdeteksi dengan tingkat akurasi 26%, sedangkan pada jarak 30 meter hanya 1 objek yang terdeteksi dengan akurasi 2%. Penurunan tersebut menunjukkan bahwa semakin jauh jarak objek dari kamera, semakin sulit sistem dalam mengenali objek secara akurat. Hal ini dapat dipengaruhi oleh ukuran objek yang semakin kecil pada citra, berkurangnya detail visual, serta kondisi lingkungan saat pengujian. Kemudian hasil pengujian akurasi untuk objek pria dapat dilihat pada Tabel 3.2, sebagai berikut :

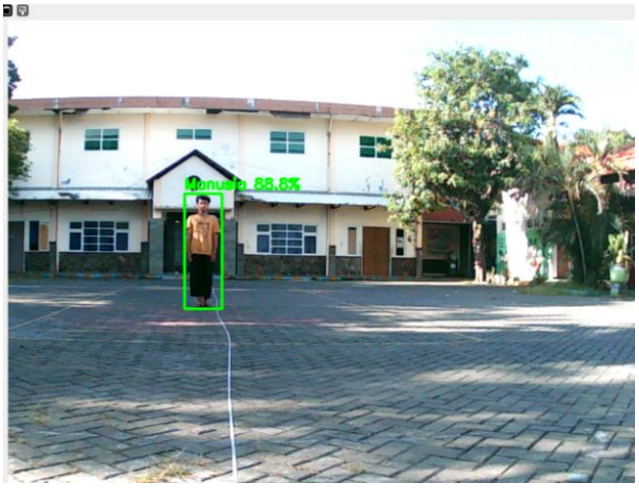
Tabel 3. 2 Pengujian Akurasi Objek Pria

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%
15	50	50	100%
20	50	50	100%
25	50	50	100%
30	50	47	94%

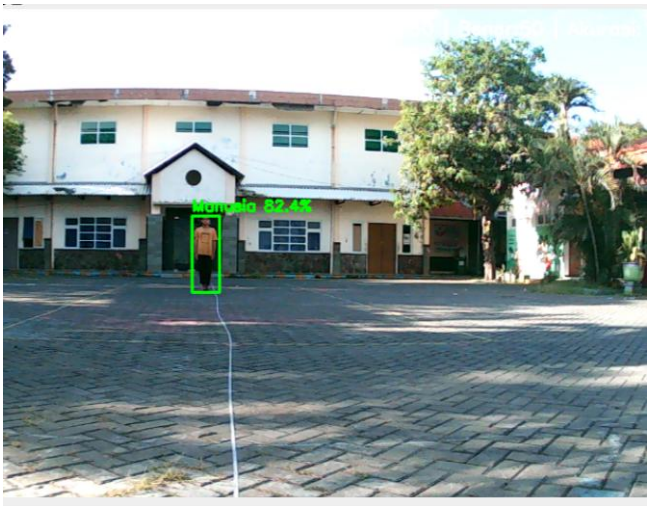
Gambar 3. 10 Deteksi Pria Jarak 2 Meter
(***Sumber :*** *Dokumen Pribadi)*



Gambar 3. 11 Deteksi Pria Jarak 5 Meter
(***Sumber :*** *Dokumen Pribadi)*



Gambar 3. 12 Deteksi Pria Jarak 10 Meter
(***Sumber :*** *Dokumen Pribadi)*



Gambar 3. 13 Deteksi Pria Jarak 15 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 14 Deteksi Pria Jarak 20 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 15 Deteksi Pria Jarak 25 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 16 Deteksi Pria Jarak 30 Meter
(***Sumber :*** *Dokumen Pribadi)*

Berdasarkan Tabel 3.2, hasil pengujian akurasi deteksi objek pria menunjukkan bahwa sistem memiliki performa yang sangat baik pada hampir seluruh jarak pengujian. Pada jarak 2 meter, 5 meter, 10 meter, 15 meter, 20 meter, dan 25 meter, seluruh sampel yang diuji sebanyak 50 objek berhasil terdeteksi dengan benar, sehingga diperoleh tingkat akurasi sebesar 100%.

Pada jarak 30 meter terjadi sedikit penurunan performa. Dari 50 sampel yang diuji, sebanyak 47 objek berhasil terdeteksi dengan benar sehingga menghasilkan tingkat akurasi sebesar 94%. Meskipun terjadi penurunan dibandingkan jarak sebelumnya, nilai akurasi tersebut masih tergolong sangat tinggi dan menunjukkan bahwa sistem tetap mampu melakukan deteksi objek pria dengan baik pada jarak yang relatif jauh. Kemudian hasil pengujian akurasi untuk objek wanita dapat dilihat pada Tabel 3.3 berikut :

Tabel 3. 3 Pengujian Akurasi Objek Wanita

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%

15	50	50	100%
20	50	50	100%
25	50	50	100%
30	50	44	88%

Gambar 3. 17 Deteksi Wanita Jarak 2 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 18 Deteksi Wanita Jarak 5 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 19 Deteksi Wanita Jarak 10 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 20 Deteksi Wanita Jarak 15 Meter
(***Sumber :*** *Dokumen Pribadi)*

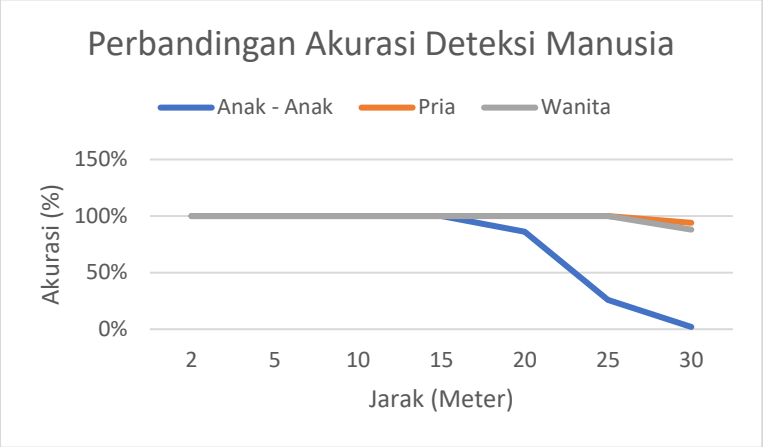
Gambar 3. 21 Deteksi Wanita Jarak 20 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 22 Deteksi Wanita Jarak 25 Meter
(***Sumber :*** *Dokumen Pribadi)*

Gambar 3. 23 Deteksi Wanita Jarak 30 Meter
(Sumber : Dokumen Pribadi)

Berdasarkan Tabel 3. 3, hasil pengujian akurasi deteksi objek wanita menunjukkan bahwa sistem memiliki tingkat performa yang sangat baik pada hampir seluruh jarak pengujian. Pada jarak 2 meter, 5 meter, 10 meter, 15 meter, 20 meter, dan 25 meter, seluruh sampel yang diuji sebanyak 50 objek berhasil terdeteksi dengan benar, sehingga diperoleh tingkat akurasi sebesar 100%. Hasil ini menunjukkan bahwa sistem mampu mengenali objek wanita secara konsisten dan stabil pada rentang jarak tersebut tanpa mengalami kesalahan deteksi.

Pada jarak 30 meter terjadi sedikit penurunan performa. Dari 50 sampel yang diuji, sebanyak 44 objek berhasil terdeteksi dengan benar, sehingga menghasilkan tingkat akurasi sebesar 88%. Meskipun mengalami penurunan dibandingkan jarak sebelumnya, nilai akurasi tersebut masih tergolong tinggi dan menunjukkan bahwa sistem tetap mampu melakukan deteksi dengan baik pada jarak yang relatif jauh. Gambar 3. 24 merupakan perbandingan akurasi deteksi manusia.



Gambar 3. 24 Grafik Perbandingan Akurasi Deteksi Manusia
(***Sumber :*** *Dokumen Pribadi*)

Untuk akurasi mobil dilakukan pada mobil Mitsubishi Expander, Suzuki Ertiga, dan Suzuki Karimun. Hasil pengujian akurasi untuk mobil Expander dapat dilihat pada tabel 3. 4 berikut :

Tabel 3. 4 Pengujian Akurasi Deteksi Expander

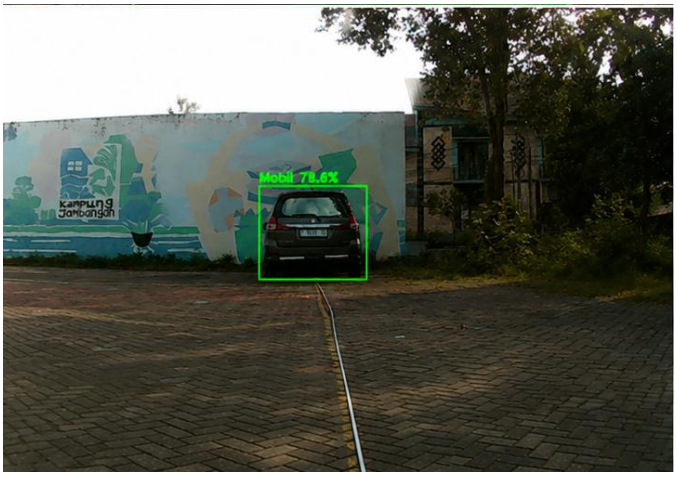
Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%
15	50	50	100%
20	50	49	98%
25	50	49	98%
30	50	47	94%

Gambar 3. 25 Deteksi Mobil Expander Jarak 2 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 26 Deteksi Mobil Expander Jarak 5 Meter
(Sumber : Dokumen Pribadi)

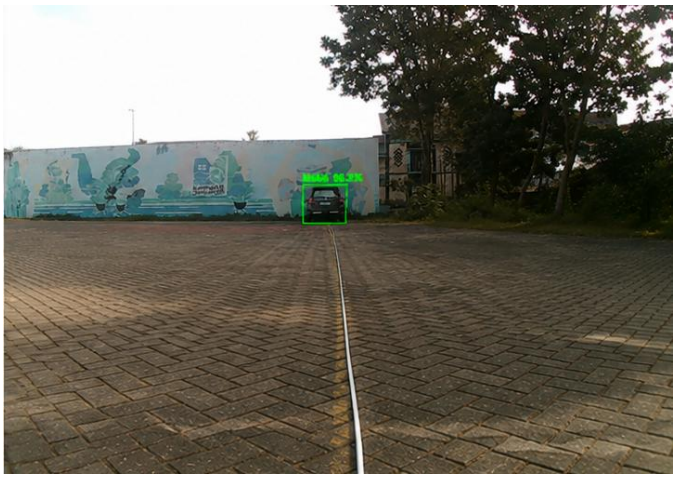
Gambar 3. 27 Deteksi Mobil Expander Jarak 10 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 28 Deteksi Mobil Expander Jarak 15 Meter
(***Sumber** : Dokumen Pribadi)*



Gambar 3. 29 Deteksi Mobil Expander Jarak 20 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 30 Deteksi Mobil Expander Jarak 25 Meter
(Sumber : Dokumen Pribadi)



Gambar 3. 31 Deteksi Mobil Expander Jarak 30 Meter
(***Sumber*** *: Dokumen Pribadi)*

Berdasarkan Tabel 3. 4, hasil pengujian akurasi deteksi Expander menunjukkan bahwa

sistem memiliki tingkat performa yang sangat baik pada sebagian besar jarak pengujian. Pada jarak 2 meter, 5 meter, 10 meter, dan 15 meter, seluruh sampel yang diuji sebanyak 50 kali berhasil terdeteksi dengan benar sehingga diperoleh tingkat akurasi sebesar 100%. Hasil ini menunjukkan bahwa sistem mampu melakukan deteksi Expander secara konsisten dan stabil pada rentang jarak tersebut tanpa mengalami kesalahan deteksi. Selanjutnya, pada jarak 20 meter dan 25 meter terjadi sedikit penurunan performa, di mana dari 50 sampel yang diuji terdapat 49 deteksi yang berhasil sehingga menghasilkan tingkat akurasi sebesar 98%.

Pada jarak 30 meter terjadi penurunan performa yang lebih terlihat dibandingkan jarak sebelumnya. Dari 50 sampel yang diuji, sebanyak 47 sampel berhasil terdeteksi dengan benar sehingga menghasilkan tingkat akurasi sebesar 94%. Meskipun demikian, tingkat akurasi yang masih berada di atas 90% menunjukkan bahwa sistem deteksi Expander yang dikembangkan memiliki tingkat keandalan yang tinggi dan tetap mampu beroperasi dengan baik meskipun digunakan pada jarak yang relatif jauh. Hasil pengujian akurasi untuk mobil Suzuki Ertiga dapat dilihat pada tabel 3. 5 berikut :

Tabel 3. 5 Pengujian Akurasi Deteksi Suzuki Ertiga

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	25	50%
5	50	50	100%
10	50	23	46%
15	50	45	90%
20	50	16	32%
25	50	47	94%
30	50	25	50%

Gambar 3. 32 Deteksi Mobil Ertiga Jarak 2 Meter
(*Sumber : Dokumen Pribadi*)

(2026), 4 (3): 218-293 **Scientica** 3021-8209
Jurnal Ilmiah Sain dan Teknologi

Gambar 3. 33 Deteksi Mobil Ertiga Jarak 5 Meter
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 34 Deteksi Mobil Ertiga Jarak 10 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 35 Deteksi Mobil Ertiga Jarak 15 Meter
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 36 Deteksi Mobil Ertiga Jarak 20 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 37 Deteksi Mobil Ertiga Jarak 25 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 38 Deteksi Mobil Ertiga Jarak 30 Meter
(Sumber : Dokumen Pribadi)

Berdasarkan Tabel 3. 5, hasil pengujian akurasi deteksi Suzuki Ertiga menunjukkan bahwa performa sistem mengalami variasi pada setiap jarak pengujian. Pada jarak 5 meter, sistem

mampu mendeteksi seluruh sampel dengan benar sebanyak 50 dari 50 sampel sehingga memperoleh tingkat akurasi tertinggi sebesar 100%. Selain itu, pada jarak 15 meter dan 25 meter sistem juga menunjukkan performa yang sangat baik dengan tingkat akurasi masing-masing sebesar 90% dan 94%, di mana jumlah deteksi benar mencapai 45 dan 47 sampel dari total 50 sampel yang diuji. Hasil ini menunjukkan bahwa model mampu mengenali objek Suzuki Ertiga dengan baik pada beberapa rentang jarak tertentu.

Namun, pada jarak 2 meter, 10 meter, 20 meter, dan 30 meter terjadi penurunan performa deteksi yang cukup signifikan. Pada jarak 2 meter dan 30 meter, sistem hanya berhasil mendeteksi 25 dari 50 sampel sehingga menghasilkan akurasi sebesar 50%. Sementara itu, pada jarak 10 meter akurasi turun menjadi 46% dengan 23 deteksi benar, dan pada jarak 20 meter diperoleh akurasi terendah sebesar 32% dengan 16 deteksi benar dari 50 sampel. Fluktuasi nilai akurasi ini menunjukkan bahwa kemampuan deteksi model terhadap objek Suzuki Ertiga belum sepenuhnya stabil pada seluruh jarak pengujian. Hasil pengujian akurasi untuk mobil Suzuki Karimun dapat dilihat pada tabel 3. 6 berikut :

Tabel 3. 6 Pengujian Akurasi Deteksi Suzuki Karimun

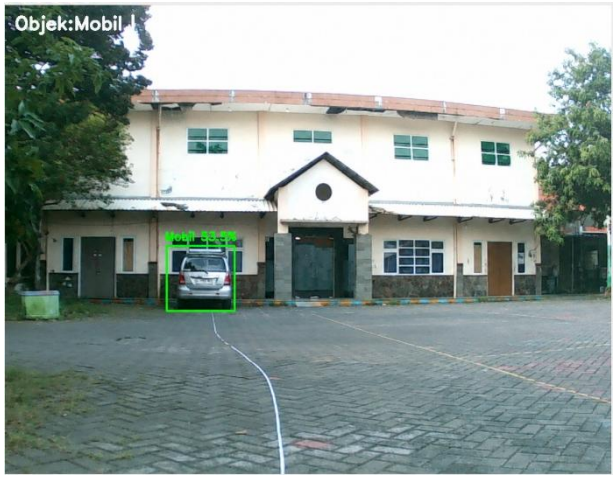
Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%
15	50	48	96%
20	50	46	92%
25	50	39	78%
30	50	33	66%

Gambar 3. 39 Deteksi Mobil Karimun Jarak 2 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 40 Deteksi Mobil Karimun Jarak 5 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 41 Deteksi Mobil Karimun Jarak 10 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 42 Deteksi Mobil Karimun Jarak 15 Meter
(Sumber : Dokumen Pribadi)



Gambar 3. 43 Deteksi Mobil Karimun Jarak 20 Meter
(Sumber : Dokumen Pribadi)

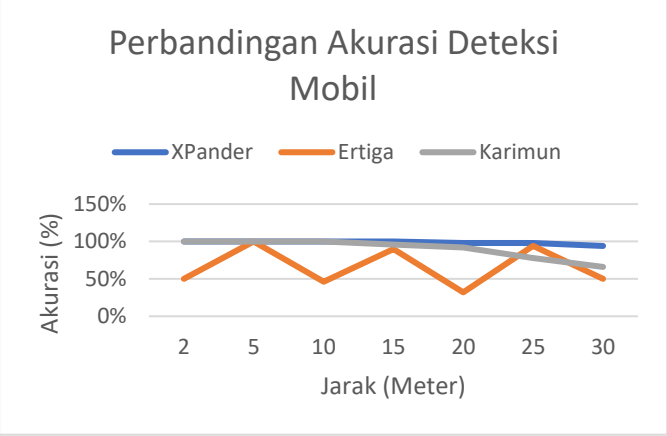
Gambar 3. 44 Deteksi Mobil Karimun Jarak 25 Meter
(Sumber : Dokumen Pribadi)

Gambar 3. 45 Deteksi Mobil Karimun Jarak 30 Meter
(Sumber : Dokumen Pribadi)

Berdasarkan hasil pengujian pada tabel, sistem deteksi menunjukkan performa yang sangat

baik pada jarak dekat hingga menengah. Pada jarak 2 meter, 5 meter, dan 10 meter, seluruh sampel yang diuji sebanyak 50 kali berhasil terdeteksi dengan benar sehingga diperoleh tingkat akurasi sebesar 100%. Hasil ini menunjukkan bahwa sistem mampu mengenali objek secara konsisten dan stabil tanpa mengalami kesalahan deteksi pada rentang jarak tersebut. Selanjutnya, pada jarak 15 meter dan 20 meter terjadi sedikit penurunan performa, di mana sistem berhasil mendeteksi masing-masing 48 dan 46 objek dari 50 sampel yang diuji sehingga menghasilkan tingkat akurasi sebesar 96% dan 92%. Meskipun mengalami penurunan, nilai akurasi tersebut masih tergolong sangat tinggi dan menunjukkan bahwa sistem tetap mampu melakukan deteksi dengan baik.

Pada jarak yang lebih jauh, yaitu 25 meter dan 30 meter, performa sistem mengalami penurunan yang lebih signifikan. Pada jarak 25 meter, sistem berhasil mendeteksi 39 dari 50 sampel sehingga menghasilkan akurasi sebesar 78%, sedangkan pada jarak 30 meter jumlah deteksi benar menurun menjadi 33 sampel dengan tingkat akurasi sebesar 66%. Penurunan akurasi ini menunjukkan bahwa bertambahnya jarak objek terhadap kamera memengaruhi kemampuan sistem dalam mengenali objek secara tepat. Faktor seperti ukuran objek yang semakin kecil pada citra dan kualitas pencahayaan, dapat menjadi penyebab menurunnya performa deteksi. Namun demikian, sistem masih mampu mempertahankan tingkat akurasi di atas 60% pada jarak 30 meter, sehingga dapat dikatakan bahwa sistem memiliki kemampuan deteksi yang cukup baik untuk digunakan pada berbagai rentang jarak pengujian. Grafik perbandingan akurasi deteksi mobil dapat dilihat pada Gambar 3. 46 berikut :



Gambar 3. 46 Perbandingan Akurasi Deteksi Objek Mobil
(***Sumber** : Dokumen Pribadi)*

Selanjutnya, pengujian akurasi objek motor menggunakan motor Honda Beat, Honda Supra X 125 dan Yamaha NMax. Untuk hasil pengujian akurasi motor Honda Beat dapat dilihat pada tabel 3. 7 berikut :

Tabel 3. 7 Pengujian Akurasi Deteksi Honda Beat

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%
15	50	50	100%
20	50	10	20%
25	50	2	4%
30	50	1	2%

Gambar 3. 47 Deteksi Motor Beat Jarak 2 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 48 Deteksi Motor Beat Jarak 5 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 49 Deteksi Motor Beat Jarak 10 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 50 Deteksi Motor Beat Jarak 15 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 51 Deteksi Motor Beat Jarak 20 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 52 Deteksi Motor Beat Jarak 25 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 53 Deteksi Motor Beat Jarak 30 Meter
(***Sumber*** *: Dokumen Pribadi)*

Berdasarkan tabel tersebut, terlihat bahwa sistem deteksi objek memiliki performa yang sangat baik pada jarak 2 meter, 5 meter, 10 meter, dan 15 meter, seluruh sampel yang diuji sebanyak 50 objek berhasil terdeteksi dengan benar sehingga diperoleh tingkat akurasi sebesar 100%. Hasil ini menunjukkan bahwa sistem mampu mengenali objek secara konsisten dan stabil pada rentang jarak tersebut tanpa mengalami kesalahan deteksi.

Namun, ketika jarak ditingkatkan menjadi 20 meter, performa sistem mengalami penurunan yang cukup signifikan. Dari 50 sampel yang diuji, hanya 10 objek yang berhasil terdeteksi dengan benar sehingga menghasilkan tingkat akurasi sebesar 20%. Penurunan akurasi semakin terlihat pada jarak 25 meter, di mana hanya 2 objek yang berhasil terdeteksi dengan benar dengan akurasi sebesar 4%. Sementara itu, pada jarak 30 meter sistem hanya mampu mendeteksi 1 objek dari 50 sampel yang diuji sehingga menghasilkan akurasi sebesar 2%. Untuk hasil pengujian akurasi motor NMax dapat dilihat pada tabel 3. 8 berikut :

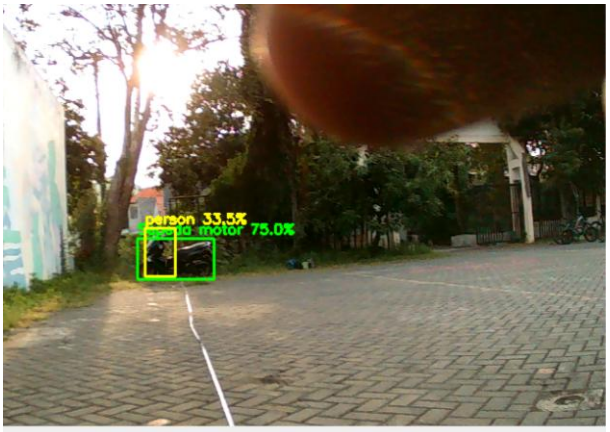
Tabel 3. 8 Hasil Pengujian Akurasi Motor Nmax

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	15	30%
10	50	22	44%
15	50	49	98%
20	50	6	12%
25	50	41	82%
30	50	34	68%

Gambar 3. 54 Deteksi Motor NMax Jarak 2 Meter
*(**Sumber** : Dokumen Pribadi)*

Gambar 3. 55 Deteksi Motor NMax Jarak 5 Meter
*(**Sumber** : Dokumen Pribadi)*

Gambar 3. 56 Deteksi Motor NMax Jarak 10 Meter
*(**Sumber** : Dokumen Pribadi)*



Gambar 3. 57 Deteksi Motor NMax Jarak 15 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 58 Deteksi Motor NMax Jarak 20 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 59 Deteksi Motor NMax Jarak 25 Meter
(***Sumber*** *: Dokumen Pribadi)*

Gambar 3. 60 Deteksi Motor NMax Jarak 30 Meter
(***Sumber*** *: Dokumen Pribadi)*

Berdasarkan tabel pengujian tersebut, terlihat bahwa sistem deteksi objek pada tabel 3. 8, hasil pengujian akurasi motor NMax menunjukkan bahwa performa sistem bervariasi pada setiap jarak pengujian. Pada jarak 2 meter, seluruh sampel yang diuji sebanyak 50 objek berhasil terdeteksi dengan benar sehingga diperoleh tingkat akurasi sebesar 100%. Hasil ini menunjukkan bahwa sistem mampu mendeteksi objek motor NMax dengan sangat baik pada jarak dekat. Pada jarak 5 meter terjadi penurunan performa yang cukup signifikan, di mana dari 50 sampel yang diuji hanya 15 objek yang berhasil terdeteksi dengan benar sehingga menghasilkan akurasi sebesar 30%. Selanjutnya, pada jarak 10 meter sistem berhasil mendeteksi 22 objek dengan tingkat akurasi sebesar 44%. Meskipun mengalami peningkatan dibandingkan jarak 5 meter, nilai tersebut masih menunjukkan bahwa kemampuan deteksi belum optimal. Performa sistem kembali meningkat pada jarak 15 meter, dengan 49 objek berhasil terdeteksi dari 50 sampel yang diuji sehingga menghasilkan akurasi sebesar 98%. Namun, pada jarak 20 meter akurasi kembali mengalami penurunan drastis menjadi 12%, karena hanya 6 objek yang berhasil terdeteksi dengan benar. Pada jarak 25 meter, sistem mendeteksi 41 objek dengan akurasi sebesar 82%, kemudian kembali menurun pada jarak 30 meter menjadi 68% dengan jumlah deteksi benar sebanyak 34 objek. Hasil pengujian menunjukkan bahwa akurasi deteksi objek motor NMax belum menunjukkan pola konsisten terhadap jarak. Kondisi ini mengindikasikan bahwa selain faktor jarak, terdapat faktor lain yang memengaruhi kinerja sistem deteksi. Untuk hasil pengujian akurasi motor Supra X 125 dapat dilihat pada tabel 3. 9 berikut :

Tabel 3. 9 hasil pengujian akurasi motor Supra

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	50	100%
10	50	50	100%
15	50	50	100%
20	50	50	100%
25	50	42	84%
30	50	8	4%

Gambar 3. 61 Deteksi Motor Supra Jarak 2 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 62 Deteksi Motor Supra Jarak 5 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 63 Deteksi Motor Supra Jarak 10 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 64 Deteksi Motor Supra Jarak 15 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 65 Deteksi Motor Supra Jarak 20 Meter
(***Sumber*** *: Dokumen Pribadi*)

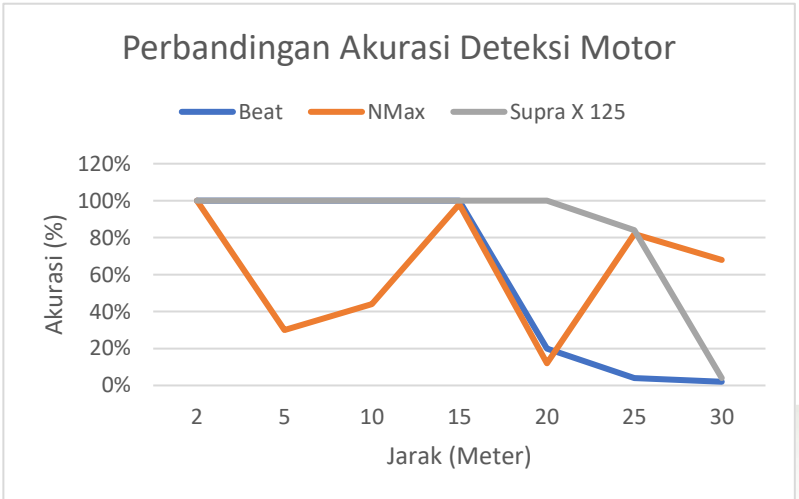
Gambar 3. 66 Deteksi Motor Supra Jarak 25 Meter
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 67 Deteksi Motor Supra Jarak 30 Meter
(***Sumber*** *: Dokumen Pribadi*)

Berdasarkan Tabel 3. 6, hasil pengujian akurasi deteksi objek motor Supra menunjukkan bahwa sistem memiliki performa yang sangat baik pada sebagian besar jarak pengujian. Pada jarak 2 meter, 5 meter, 10 meter, 15 meter, dan 20 meter, seluruh sampel yang diuji sebanyak 50 objek berhasil terdeteksi dengan benar, sehingga diperoleh tingkat akurasi sebesar 100%. Hasil ini menunjukkan bahwa sistem mampu mengenali objek motor Supra secara konsisten dan stabil pada rentang jarak tersebut tanpa mengalami kesalahan deteksi.

Pada jarak 25 meter terjadi penurunan performa. Dari 50 sampel yang diuji, sebanyak 42 objek berhasil terdeteksi dengan benar sehingga menghasilkan tingkat akurasi sebesar 84%. Meskipun mengalami penurunan dibandingkan dengan jarak sebelumnya, nilai akurasi tersebut masih tergolong baik dan menunjukkan bahwa sistem tetap mampu melakukan deteksi objek pada jarak yang relatif jauh.

Penurunan yang lebih signifikan terjadi pada jarak 30 meter. Dari 50 sampel yang diuji, hanya 8 objek yang berhasil terdeteksi dengan benar sehingga menghasilkan tingkat akurasi sebesar 4%. Hasil ini menunjukkan bahwa kemampuan sistem dalam mendeteksi objek motor Supra menurun secara drastis pada jarak tersebut. Penurunan ini kemungkinan disebabkan oleh ukuran objek yang semakin kecil pada citra, berkurangnya detail visual yang dapat dikenali oleh sistem. Untuk grafik perbandingan akurasi deteksi motor dapat dilihat pada Gambar 3. 68 :



Gambar 3. 68 Perbandingan Akurasi Deteksi Motor
(***Sumber*** *: Dokumen Pribadi)*

Selanjutnya, dilakukan pengujian tambahan dari sisi belakang untuk mengetahui kemampuan sistem YOLOv8 dalam mendeteksi objek pada sudut pandang yang berbeda. Hasil pengujian ini digunakan untuk membandingkan tingkat akurasi deteksi antara tampilan samping dan tampilan belakang pada setiap variasi jarak pengujian. Berikut pada tabel 3. 10 merupakan hasil pengujian akurasi sepeda motor Beat nampak belakang :

Tabel 3. 10 Hasil Pengujian Akurasi Motor Beat

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	50	100%
5	50	41	82%
10	50	48	96%
15	50	4	8%
20	50	0	0%
25	50	0	0%
30	50	0	0%

Gambar 3. 69 Deteksi Motor Beat Tampak Belakang Jarak 2 m
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 70 Deteksi Motor Beat Tampak Belakang Jarak 5 m
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 71 Deteksi Motor Beat Tampak Belakang Jarak 10 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 72 Deteksi Motor Beat Tampak Belakang Jarak 15 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 73 Deteksi Motor Beat Tampak Belakang Jarak 20 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 74 Deteksi Motor Beat Tampak Belakang Jarak 25 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 75 Deteksi Motor Beat Tampak Belakang Jarak 30 m
(***Sumber*** *: Dokumen Pribadi*)

Berdasarkan Tabel 3. 10, hasil pengujian akurasi deteksi sepeda motor Beat dari sisi belakang menunjukkan bahwa performa sistem tidak menurun secara linier seiring bertambahnya jarak. Pada jarak 2 meter, seluruh 50 sampel berhasil terdeteksi dengan benar sehingga diperoleh akurasi sebesar 100%. Akurasi kemudian menurun menjadi 82% pada jarak 5 meter, tetapi kembali meningkat menjadi 96% pada jarak 10 meter. Pola tersebut menunjukkan bahwa hasil deteksi tidak hanya dipengaruhi oleh jarak, tetapi juga dapat dipengaruhi oleh posisi objek dalam citra, sudut pengambilan gambar, pencahayaan, serta kejelasan karakteristik bagian belakang sepeda motor terhadap latar belakang.

Penurunan signifikan terjadi pada jarak 15 meter, ketika hanya 4 dari 50 sampel yang berhasil terdeteksi sehingga menghasilkan akurasi sebesar 8%. Pada jarak 20 meter, 25 meter, dan 30 meter, tidak terdapat sampel yang berhasil terdeteksi sehingga akurasi menjadi 0%. Kondisi tersebut menunjukkan kemampuan sistem mendeteksi sepeda motor Beat dari sisi belakang mulai menurun tajam pada jarak 15 meter. Hal ini diduga terjadi karena ukuran objek pada citra semakin kecil sehingga detail bagian belakang, seperti bentuk bodi, lampu, roda, dan spion, semakin sulit dikenali oleh sistem. Sebagai pengujian tambahan dari sudut pandang yang sama, Tabel 3. 11 menunjukkan hasil pengujian akurasi deteksi sepeda motor Supra dari sisi belakang :

Tabel 3. 11 Hasil Pengujian Akurasi Motor Supra

Jarak (Meter)	Sampel	Deteksi Benar	Akurasi
2	50	49	98%
5	50	50	100%
10	50	14	28%
15	50	0	0%
20	50	0	0%
25	50	0	0%
30	50	0	0%

Gambar 3. 76 Deteksi Motor Supra Tampak Belakang Jarak 2 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 77 Deteksi Motor Supra Tampak Belakang Jarak 5 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 78 Deteksi Motor Supra Tampak Belakang Jarak 10 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 79 Deteksi Motor Supra Tampak Belakang Jarak 15 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 80 Deteksi Motor Supra Tampak Belakang Jarak 20 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 81 Deteksi Motor Supra Tampak Belakang Jarak 25 m
(***Sumber*** *: Dokumen Pribadi*)

Gambar 3. 82 Deteksi Motor Supra Tampak Belakang Jarak 30 m
(***Sumber*** *: Dokumen Pribadi*)

Berdasarkan Tabel 3. 11, hasil pengujian akurasi deteksi sepeda motor Supra dari sisi belakang menunjukkan bahwa sistem bekerja sangat baik pada jarak dekat. Pada jarak 2 meter, sebanyak 49 dari 50 sampel berhasil terdeteksi sehingga diperoleh akurasi sebesar 98%, sedangkan pada jarak 5 meter seluruh sampel berhasil terdeteksi dengan akurasi 100%. Perbedaan kecil tersebut menunjukkan bahwa hasil deteksi tidak hanya dipengaruhi oleh jarak, tetapi juga oleh posisi objek dalam citra, kondisi pencahayaan, serta tingkat keyakinan model pada setiap frame.

Penurunan yang tajam terjadi pada jarak 10 meter, ketika hanya 14 dari 50 sampel yang berhasil terdeteksi sehingga akurasi turun menjadi 28%. Pada jarak 15 meter hingga 30 meter, tidak terdapat objek yang berhasil terdeteksi dan nilai akurasi menjadi 0%. Kondisi tersebut menunjukkan bahwa deteksi sepeda motor Supra dari sisi belakang mulai tidak stabil pada jarak 10 meter dan tidak lagi bekerja secara efektif pada jarak 15 meter atau lebih. Penurunan tersebut dapat disebabkan oleh ukuran objek yang semakin kecil pada citra sehingga detail bagian belakang semakin sulit dikenali. Selain itu, paparan panas matahari dalam waktu yang cukup lama selama pengujian dapat menurunkan kejernihan tampilan webcam sehingga citra menjadi sedikit kabur. Kondisi tersebut semakin mengurangi detail visual objek dan menurunkan kemampuan sistem dalam melakukan deteksi pada jarak yang lebih jauh.

Pengujian Latensi Deteksi Objek

Pengujian latensi dilakukan untuk mengetahui waktu respon sistem dalam memproses deteksi objek. Latensi diukur mulai dari proses pengambilan gambar oleh webcam, pemrosesan YOLOv8 pada Raspberry Pi, hingga hasil ditampilkan pada terminal.

Pengujian dilakukan pada beberapa zona sistem di siang hari dan malam hari, yaitu zona STOP (2,5 meter), zona Z3 (3 meter), zona Z5 (5 meter), dan zona Z8 (8 meter). Pada setiap kondisi, diambil 10 data nilai latensi yang ditampilkan pada terminal, kemudian dihitung nilai rata-ratanya untuk memperoleh representasi waktu respon sistem. Tabel 3. 6 – 4.8 merupakan rata-rata latensi deteksi objek manusia pada pengujian siang dan malam hari :

Tabel 3. 12 Rata Rata Latensi Deteksi Anak - Anak

Mode	Siang	Malam	Foto
Stop	263,7 ms	285,4 ms	 
Z3	288,6 ms	297,3 ms	 
Z5	289,9 ms	300,2 ms	

Mode	Siang	Malam	Foto
			
Z8	287,3 ms	298,3 ms	 

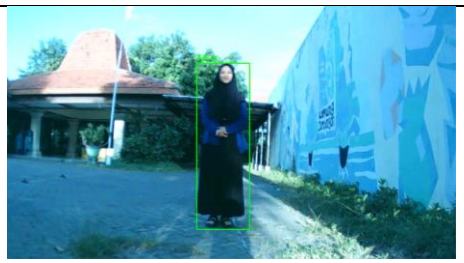
Tabel 3. 13 Rata Rata Latensi Deteksi Pria

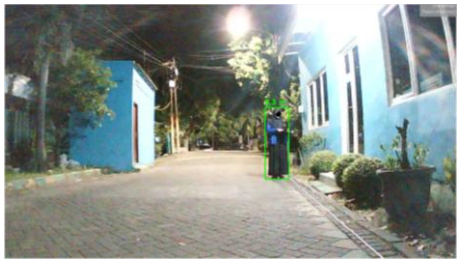
Mode	Siang	Malam	Foto
Stop	285,4 ms	305,2 ms	 

Mode	Siang	Malam	Foto
Z3	273,9 ms	296,6 ms	 
Z5	271,1 ms	299,8 ms	 
Z8	276,4 ms	289,2 ms	

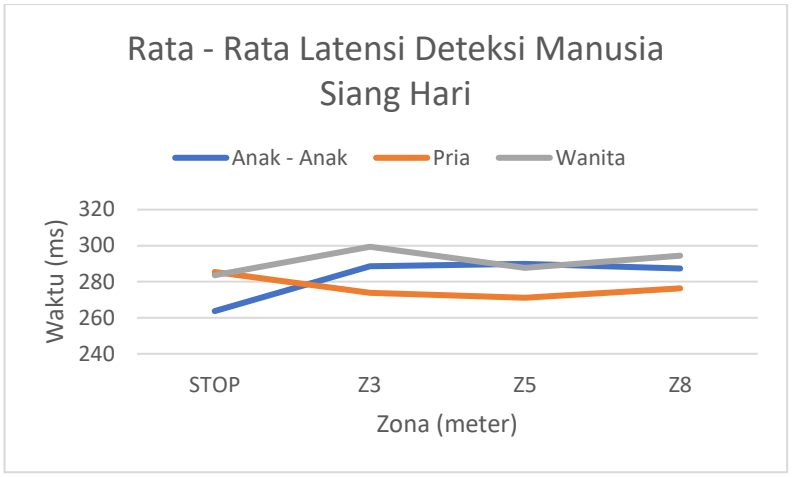
Mode	Siang	Malam	Foto
			

Tabel 3. 14 Rata Rata Latensi Deteksi Wanita

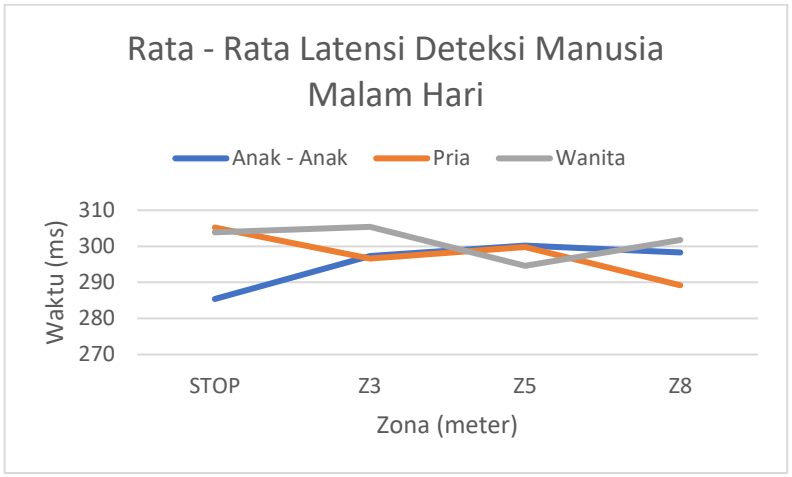
Mode	Siang	Malam	Foto
Stop	283,6 ms	303,9 ms	 
Z3	299,4 ms	305,4 ms	 
Z5	287,7 ms	294,6 ms	

Mode	Siang	Malam	Foto
			 
Z8	294,4 ms	301,8 ms	 

Berdasarkan Tabel 3. 12 - 4.14, hasil pengujian menunjukkan bahwa rata-rata latensi deteksi objek berada pada kisaran 263 ms hingga 305 ms untuk seluruh kondisi pengujian. Secara umum, sistem menunjukkan performa yang stabil baik pada siang maupun malam hari. Namun, terdapat kecenderungan bahwa latensi pada malam hari sedikit lebih tinggi dibandingkan siang hari, dengan selisih sekitar 6–20 ms. Hal ini kemungkinan disebabkan oleh kondisi pencahayaan yang lebih sederhana pada malam hari sehingga kompleksitas visual yang diproses sistem menjadi lebih rendah. Gambar 3. 83 dan 4.84 merupakan perbandingan rata – rata latensi pada siang hari dan malam hari :



Gambar 3. 83 Rata – Rata Latensi Deteksi Manusia Siang Hari
(***Sumber** : Dokumen Pribadi)*

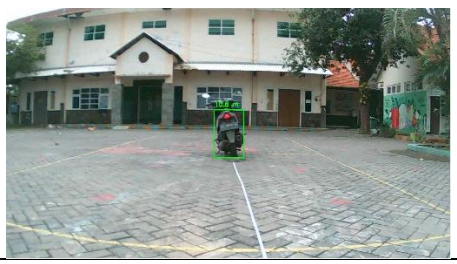


Gambar 3. 84 Rata – Rata Latensi Deteksi Manusia Malam Hari
(***Sumber** : Dokumen Pribadi*)

Tabel 3. 15 merupakan rata-rata latensi deteksi objek motor Beat pada pengujian siang dan malam hari :

Tabel 3. 15 Rata Rata Latensi Deteksi Objek Motor Beat

Mode	Siang	Malam	Foto
Stop	277,8 ms	287,6 ms	

Mode	Siang	Malam	Foto
			
Z3	271,4 ms	294,7 ms	 
Z5	279,5 ms	285,9 ms	 
Z8	283,6 ms	299,7 ms	

Mode	Siang	Malam	Foto
			

Tabel 3. 16 merupakan rata-rata latensi deteksi objek motor Nmax pada pengujian siang dan malam hari :

Tabel 3. 16 Rata Rata Latensi Deteksi Objek Motor Nmax

Mode	Siang	Malam	Foto
Stop	267,5 ms	287,4 ms	 
Z3	279,1 ms	294,2 ms	 

Mode	Siang	Malam	Foto
Z5	278,8 ms	286,7 ms	 
Z8	282,9 ms	306,3 ms	 

Tabel 3. 17 merupakan rata-rata latensi deteksi objek motor Supra pada pengujian siang dan malam hari :

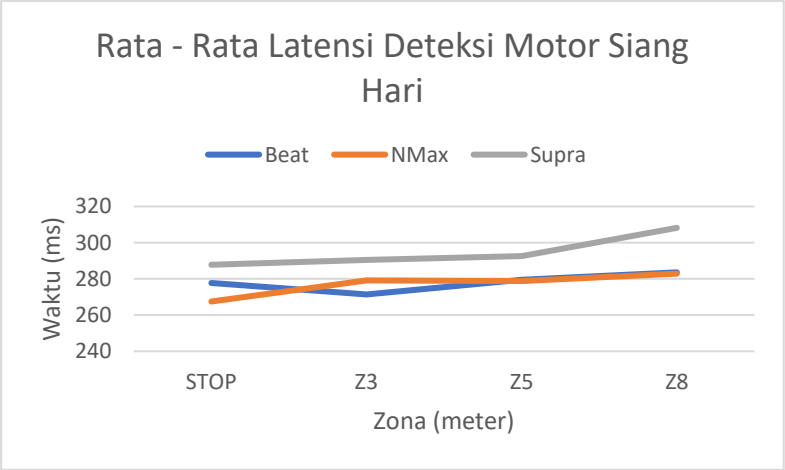
Tabel 3. 17 Rata Rata Latensi Deteksi Objek Motor Supra

Mode	Siang	Malam	Foto
Stop	287,8 ms	296,7 ms	

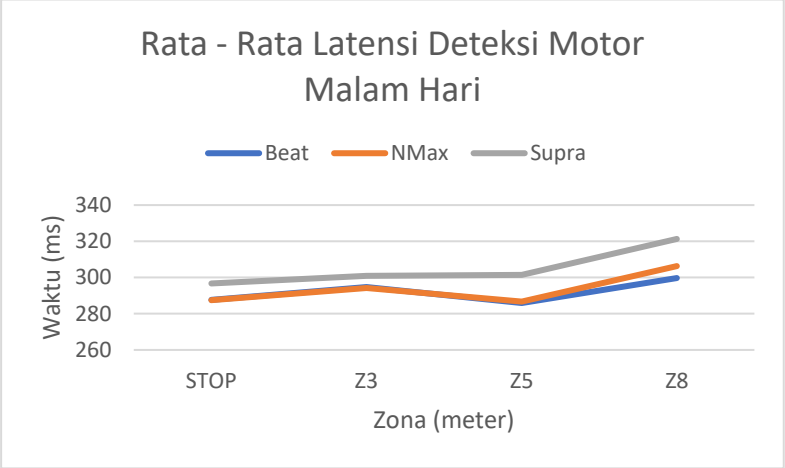
Mode	Siang	Malam	Foto
			
Z3	290,4 ms	300,9 ms	 
Z5	292,6 ms	301,5 ms	 
Z8	308,6 ms	321,3 ms	

Mode	Siang	Malam	Foto
			

Berdasarkan Tabel 3. 15 - 4.17, rata – rata latensi deteksi objek motor berada pada rentang 267,5 hingga 321,3 ms. Rata-rata keseluruhan pada siang hari sebesar 283,3 ms, sedangkan pada malam hari meningkat menjadi 296,9 ms. Terdapat perbedaan pola di mana pada deteksi manusia latensi malam hari cenderung lebih rendah, sedangkan pada deteksi motor pada kondisi tertentu justru lebih tinggi. Hal ini kemungkinan dipengaruhi oleh kompleksitas bentuk motor yang lebih detail sehingga lebih sensitif terhadap kondisi pencahayaan rendah. Gambar 3. 85 – 4.86 merupakan grafik rata – rata deteksi motor siang dan malam hari.



Gambar 3. 85 Rata – Rata Latensi Deteksi Motor Siang Hari
(***Sumber*** *: Dokumen Pribadi)*



Gambar 3. 86 Rata – Rata Latensi Deteksi Motor Malam Hari
(***Sumber*** *: Dokumen Pribadi)*

Tabel 3. 18 merupakan rata-rata latensi deteksi objek mobil Agya pada pengujian siang dan malam hari :

Tabel 3. 18 Rata Rata Latensi Deteksi Objek Mobil Agya

Mode	Siang	Malam	Foto
Stop	264,6 ms	296,5 ms	 
Z3	275,2 ms	291,7 ms	 
Z5	266,8 ms	280,9 ms	

Mode	Siang	Malam	Foto
			
Z8	276,4 ms	291,3 ms	 

Tabel 3. 19 merupakan rata-rata latensi deteksi objek mobil Avanza pada pengujian siang dan malam hari :

Tabel 3. 19 Rata Rata Latensi Deteksi Objek Mobil Avanza

Mode	Siang	Malam	Foto
Stop	276,8 ms	294,1 ms	 

Mode	Siang	Malam	Foto
Z3	287,4 ms	309,6 ms	 
Z5	297,5 ms	301,8 ms	 
Z8	288,9 ms	305,5 ms	 

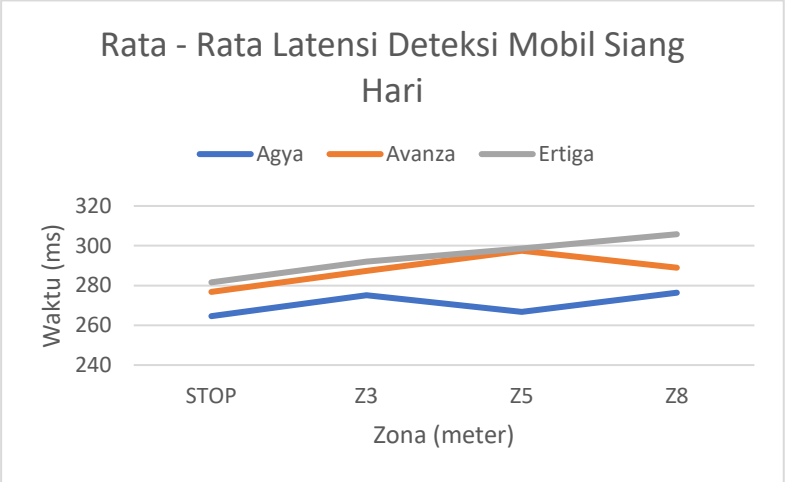
Tabel 3. 20 merupakan rata-rata latensi deteksi objek mobil Ertiga pada pengujian siang dan malam hari :

Tabel 3. 20 Rata Rata Latensi Deteksi Objek Mobil Ertiga

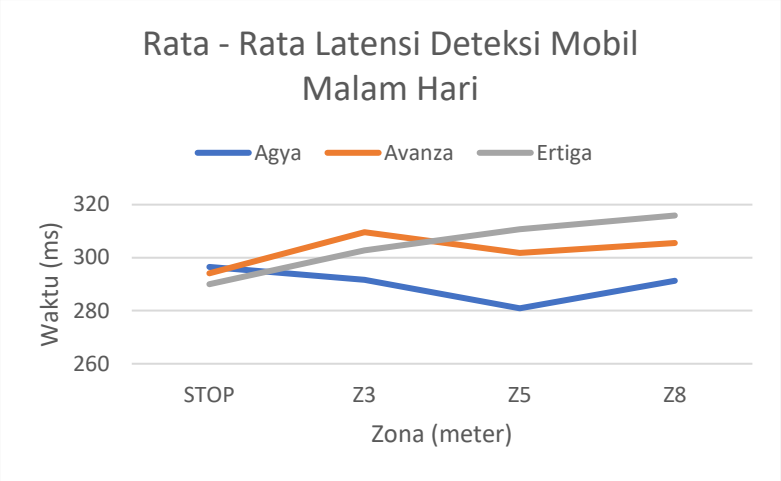
Mode	Siang	Malam	Foto
Stop	281,6 ms	290,0 ms	 
Z3	292,1 ms	302,8 ms	 
Z5	298,6 ms	310,7 ms	

Mode	Siang	Malam	Foto
			
Z8	305,8 ms	315,9 ms	 

Berdasarkan Tabel 3. 18 – 4.20, rata rata latensi deteksi objek mobil berada pada kisaran sekitar 264 ms hingga 315 ms, yang menunjukkan performa sistem sedikit lebih cepat dibandingkan deteksi sepeda motor. Hal ini mengindikasikan bahwa objek mobil cenderung lebih mudah dideteksi, kemungkinan karena ukuran yang lebih besar dan bentuk yang lebih konsisten sehingga mempermudah proses identifikasi oleh model. Gambar 3. 87 dan 4.88 merupakan perbandingan rata – rata latensi objek mobil pada siang dan malam hari :



Gambar 3. 87 Rata – Rata Latensi Deteksi Mobil Siang Hari
(***Sumber** : Dokumen Pribadi)*



Gambar 3. 88 Rata – Rata Latensi Deteksi Mobil Malam Hari
(***Sumber** : Dokumen Pribadi)*

Pengujian Kinerja Pengendalian Kecepatan Motor Berdasarkan Zona Jarak Objek

Pengujian tersebut dilakukan untuk mengetahui kemampuan sistem dalam mengatur kecepatan putaran motor berdasarkan zona jarak objek yang terdeteksi. Pengujian dilakukan menggunakan tiga jenis objek, yakni manusia, sepeda motor, dan mobil, dengan variasi berat pengendara sebesar 48 kg, 65 kg, dan 113 kg. Setiap objek diuji pada zona Z8 (8 meter), zona Z5 (5 meter), zona Z3 (3 meter), serta zona STOP (2,5 meter). Pada setiap variasi objek, berat pengendara, dan zona jarak, dilakukan sebanyak 4 kali pengujian. Nilai kecepatan putaran motor dalam satuan *revolution per minute* (RPM) diperoleh dari data yang ditampilkan pada terminal Raspberry Pi. Keempat nilai RPM pada setiap kondisi pengujian kemudian dihitung nilai rata-ratanya untuk merepresentasikan kinerja pengendalian kecepatan motor pada masing-masing zona.

Selain nilai RPM, terminal Raspberry Pi juga menampilkan nilai latensi pemrosesan sistem. Pada zona STOP, nilai RPM diamati selama proses penurunan kecepatan hingga motor benar-benar mencapai 0 RPM. Dalam setiap variasi berat pengguna dilakukan 4 kali pengujian, dengan mencatat 1 nilai latensi yang ditampilkan pada terminal ketika nilai RPM pertama kali mencapai 0 RPM. 4 sampel nilai latensi tersebut kemudian dihitung nilai rata-ratanya untuk merepresentasikan latensi sistem pada zona STOP pada masing-masing variasi berat pengguna. Hasil pengujian selanjutnya digunakan untuk membandingkan respons sistem pada kondisi beban pengguna yang berbeda. Tabel 3. 21 merupakan hasil pengujian pada jenis objek manusia berdasarkan variasi berat pengguna :

Tabel 3. 21 Pengujian RPM dan Latensi Zona STOP Objek Manusia

Berat Pengguna (Kg)	Rata-Rata RPM				Rata-Rata Latensi STOP (ms)
	Z8	Z5	Z3	STOP	
48	132,4	118,7	102,5	0	282,5
65	84,3	76,3	67,3	0	276,6
113	72,8	65,2	57,4	0	264,2

Gambar 3. 89 Pengujian RPM Objek Manusia dengan Berat Pengguna 48 kg
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 90 Pengujian RPM Objek Manusia dengan Berat Pengguna 65 kg
(***Sumber** : Dokumen Pribadi*)

Berdasarkan Tabel 3. 21, pengujian pada objek manusia menunjukkan bahwa sistem mampu menurunkan putaran motor secara bertahap pada seluruh variasi berat pengguna. Dari zona Z8 ke Z3, RPM berkurang sebesar 29,9 RPM pada berat 48 kg, 17,0 RPM pada berat 65 kg, dan 15,4 RPM pada berat 113 kg. Secara persentase, penurunannya relatif konsisten, yaitu sekitar 20–23%. Hal ini menunjukkan bahwa pembatasan keluaran DAC berdasarkan zona jarak tetap bekerja meskipun beban pengguna berbeda. Selain itu, pada zona yang sama, pengguna dengan berat lebih besar menghasilkan RPM yang lebih rendah karena motor menerima beban mekanis yang lebih tinggi. Pada zona STOP, seluruh kondisi mencapai 0 RPM, sehingga fungsi penghentian motor dapat bekerja pada ketiga variasi berat pengguna.

Rata-rata latensi yang dicatat pada pembacaan pertama saat RPM menunjukkan 0 adalah 282,5 ms, 276,6 ms, dan 264,2 ms. Selisih tertinggi dan terendah hanya sebesar 18,3 ms, sehingga perbedaannya masih tergolong sebagai variasi respons sistem pada setiap pengujian. Meskipun tabel menunjukkan latensi yang menurun seiring bertambahnya berat pengguna, perbedaan tersebut juga dapat dipengaruhi oleh waktu pemrosesan frame, beban kerja Raspberry Pi, dan komunikasi data antara Raspberry Pi dengan Arduino Uno. Selanjutnya, pada tabel 3. 22 merupakan hasil pengujian pada jenis objek sepeda motor berdasarkan variasi berat pengguna :

Tabel 3. 22 Pengujian RPM dan Latensi Zona STOP Objek Sepeda Motor

Berat Pengguna (Kg)	Rata-Rata RPM				Rata-Rata Latensi STOP (ms)
	Z8	Z5	Z3	STOP	
48	101,1	88,0	65,3	0	290,4
65	90,4	77,4	59,2	0	278,8
113	77,0	61,5	53,8	0	265,3

Gambar 3. 91 Pengujian RPM Objek Sepeda Motor dengan Berat Pengguna 48 kg
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 92 Pengujian RPM Objek Sepeda Motor dengan Berat Pengguna 65 kg
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 93 Pengujian RPM Objek Sepeda Motor dengan Berat Pengguna 113 kg
(***Sumber** : Dokumen Pribadi*)

Data pada Tabel 3. 22 memperlihatkan bahwa respons pengendalian terhadap objek sepeda motor membentuk pola penurunan RPM yang konsisten seiring perpindahan zona pada setiap variasi berat pengguna. Dari zona Z8 ke Z3, RPM berkurang sebesar 35,8 RPM pada berat 48

kg, 31,2 RPM pada berat 65 kg, dan 23,2 RPM pada berat 113 kg. Secara persentase, penurunannya berada pada kisaran 30–35%, sehingga pembatasan keluaran DAC berdasarkan zona tetap bekerja secara konsisten. Pada zona yang sama, peningkatan berat pengguna juga diikuti penurunan RPM karena motor menerima beban mekanis yang lebih besar. Seluruh kondisi mencapai 0 RPM pada zona STOP, yang menunjukkan bahwa fungsi penghentian dapat bekerja pada setiap variasi beban.

Rata-rata latensi saat RPM pertama kali terbaca 0 adalah 290,4 ms, 278,8 ms, dan 265,3 ms, dengan selisih pada nilai tertinggi dan terendah sebesar 25,1 ms. Selanjutnya, pada tabel 3. 23 merupakan hasil pengujian pada jenis objek mobil berdasarkan variasi berat pengguna :

Tabel 3. 23 Pengujian RPM dan Latensi Zona STOP Objek Mobil

Berat Pengguna (Kg)	Rata-Rata RPM				Rata-Rata Latensi STOP (ms)
	Z8	Z5	Z3	STOP	
48	78,5	74,8	72,6	0	301,3
65	76,8	69,6	54,6	0	282,4
113	59,5	57,1	44,5	0	267,7

Gambar 3. 94 Pengujian RPM Objek Mobil dengan Berat Pengguna 48 kg
(***Sumber** : Dokumen Pribadi*)

Gambar 3. 95 Pengujian RPM Objek Mobil dengan Berat Pengguna 65 kg
(***Sumber** : Dokumen Pribadi*)

(2026), 4 (3): 218-293 **Scientica** 3021-8209
Jurnal Ilmiah Sain dan Teknologi

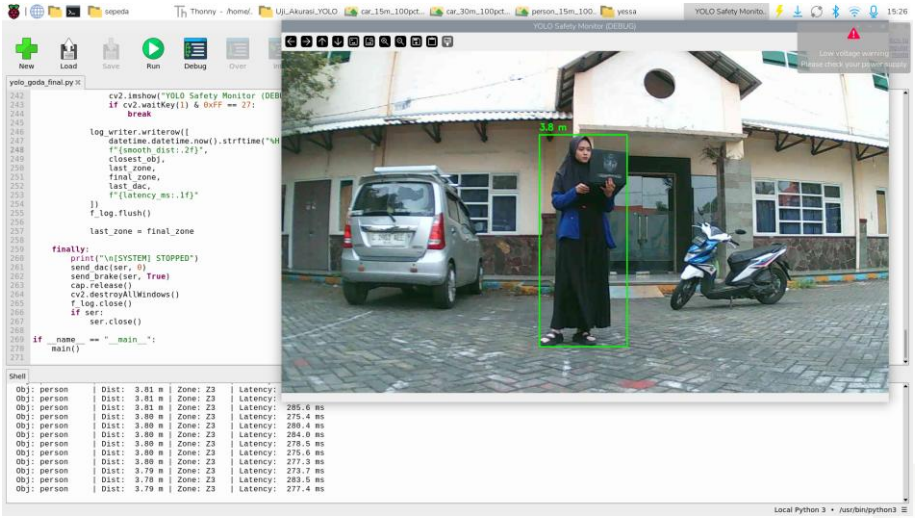
Gambar 3. 96 Pengujian RPM Objek Mobil dengan Berat Pengguna 113 kg
(***Sumber** : Dokumen Pribadi*)

Berdasarkan hasil pengujian pada Tabel 3. 23, penurunan paling kecil dari selisih Z8 dan Z3, terjadi pada berat 48 kg, yakni 5,9 RPM, sedangkan pada berat 65 kg dan 113 kg masing-masing sebesar 22,2 RPM dan 15,0 RPM. Perbedaan tersebut menunjukkan bahwa respons pembatasan kecepatan tidak selalu memiliki besar penurunan yang sama, tetapi tetap mengikuti pola sesuai zona. Nilai RPM juga cenderung lebih rendah pada beban yang lebih besar karena motor menerima beban mekanis yang meningkat. Pada zona STOP, seluruh variasi berat berhasil mencapai 0 RPM.

Pengujian Deteksi Multi Objek

Setelah dilakukan pengujian terhadap masing-masing objek secara terpisah, tahap selanjutnya adalah pengujian deteksi multi objek. Pengujian ini bertujuan untuk mengetahui kemampuan sistem dalam mendeteksi lebih dari satu jenis objek secara bersamaan dalam satu frame citra. Selain itu, pengujian ini juga dilakukan untuk memastikan bahwa keberadaan beberapa objek dalam satu bidang pandang tidak memengaruhi kemampuan sistem dalam mengenali objek yang berada pada jarak terdekat maupun objek lainnya.

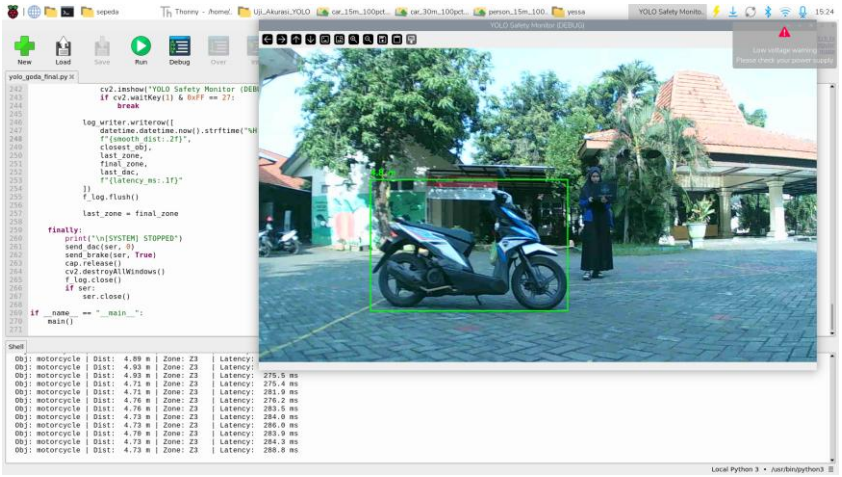
Pada pengujian ini digunakan tiga kategori objek, yaitu manusia, sepeda motor, dan mobil. Setiap objek ditempatkan pada posisi dan jarak yang berbeda sehingga sistem harus mampu mengidentifikasi seluruh objek yang muncul secara bersamaan. Gambar 3. 97 merupakan pengujian deteksi tiga objek dalam satu frame citra :



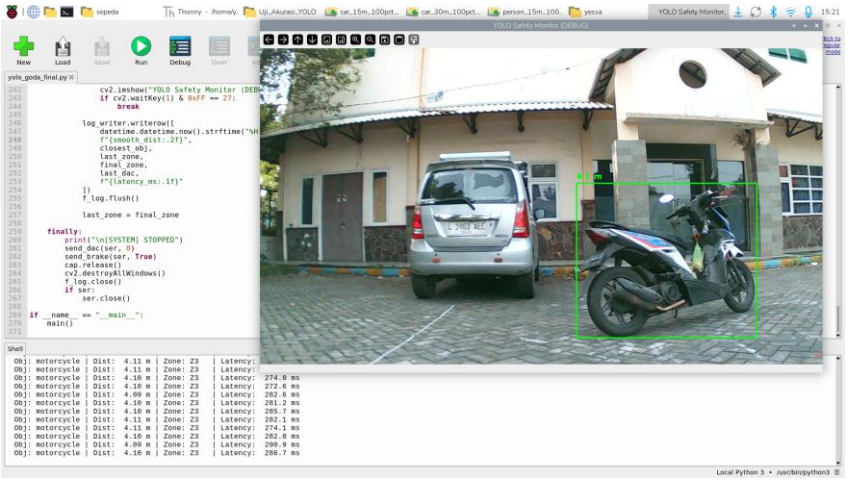
Gambar 3. 97 Pengujian Deteksi Tiga Objek
(***Sumber** : Dokumen Pribadi*)

Pada Gambar 3. 97, sistem diuji dengan tiga jenis objek yang berada dalam satu frame, yaitu manusia, mobil, dan sepeda motor. Pada skenario ini, objek manusia ditempatkan pada posisi yang paling dekat dengan kamera, sedangkan mobil dan sepeda motor berada pada jarak yang lebih jauh. Hasil pengujian menunjukkan bahwa sistem berhasil mendeteksi objek manusia dengan memberikan label **person** beserta *bounding box* yang sesuai dengan posisi objek. Selain itu, pada tampilan terminal juga terlihat bahwa sistem secara konsisten membaca objek sebagai **person**, lengkap dengan informasi jarak objek, zona deteksi dan *latency*. Selain pengujian tiga objek dalam satu frame, dilakukan pula pengujian terhadap kombinasi dua objek, yaitu manusia dengan mobil, manusia dengan sepeda motor, serta sepeda motor dengan mobil. Gambar 3. 98 – 4.99 merupakan pengujian deteksi dua objek dalam satu frame citra :

Gambar 3. 98 Pengujian Deteksi Manusia dengan Mobil
(***Sumber** : Dokumen Pribadi*)



Gambar 3. 99 Pengujian Deteksi Motor dengan Manusia
(***Sumber** : Dokumen Pribadi)*



Gambar 3. 100 Pengujian Deteksi Mobil dengan Motor
(***Sumber** : Dokumen Pribadi)*

Berdasarkan seluruh hasil pengujian deteksi dua objek, sistem menunjukkan kemampuan yang baik dalam mengenali objek yang berada pada jarak paling dekat dengan kamera. Pada pengujian manusia dan mobil, sistem berhasil mendeteksi objek **person** ketika manusia menjadi objek terdekat. Pada pengujian manusia dan sepeda motor serta mobil dan sepeda motor, sistem berhasil mendeteksi objek **motorcycle** ketika sepeda motor ditempatkan lebih dekat dibandingkan objek lainnya. Hasil pengujian ini menunjukkan bahwa sistem tidak hanya mampu membedakan kelas objek yang berbeda, tetapi juga secara konsisten memprioritaskan deteksi terhadap objek yang memiliki jarak paling dekat dengan kamera. Dengan demikian, sistem memiliki kemampuan yang baik dalam mendukung proses deteksi objek pada kondisi nyata yang melibatkan lebih dari satu objek dalam satu bidang pandang.

KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem kendali kecepatan sepeda listrik berbasis deteksi objek menggunakan algoritma YOLOv8 dan Arduino Uno, dapat diperoleh beberapa kesimpulan sebagai berikut :

a. Sistem berbasis YOLOv8 dan Raspberry Pi 5 mampu mendeteksi objek manusia, sepeda motor, dan mobil serta menentukan zona jarak sebagai dasar pengendalian kecepatan. Namun, akurasi belum merata pada seluruh kondisi. Deteksi pria dan wanita mencapai 100% hingga jarak 25 meter, sedangkan objek anak-anak menurun dari 86% pada jarak 20 meter menjadi 2% pada jarak 30 meter. Pada sepeda motor tampak belakang, penurunan terjadi lebih cepat, bahkan mencapai 0% mulai jarak 15–20 meter. Penurunan tersebut dipengaruhi oleh jarak, sudut objek, pencahayaan, serta citra webcam yang sedikit kabur akibat terlalu lama terpapar sinar matahari. Meskipun demikian, sistem tetap menunjukkan kinerja deteksi yang baik pada jarak dekat hingga menengah.
b. Sistem kendali menggunakan Arduino Uno R3 dan DAC MCP4725 mampu menurunkan putaran motor secara bertahap dari zona Z8, Z5, Z3, hingga STOP dengan berat pengguna yang bervariasi. Pada objek manusia dengan berat pengguna 48 kg, putaran motor menurun dari 132,4 RPM menjadi 118,7 RPM, 102,5 RPM, dan akhirnya 0 RPM. Seluruh kombinasi objek dan berat pengguna juga berhasil mencapai 0 RPM pada zona STOP.
c. Waktu respons deteksi sistem berada pada rentang 263,7–321,3 ms. Nilai terendah diperoleh pada deteksi anak-anak di zona STOP pada siang hari, sedangkan nilai tertinggi terjadi pada deteksi motor Supra di zona Z8 pada malam hari. Hasil tersebut menunjukkan bahwa sistem mampu merespons dalam waktu kurang dari satu detik, tetapi latensinya

belum sepenuhnya stabil karena dipengaruhi oleh jenis objek, pencahayaan, dan beban pemrosesan Raspberry Pi 5.

Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran yang dapat dipertimbangkan untuk pengembangan lebih lanjut, antara lain :

a. Untuk meningkatkan kestabilan sistem, terutama pada kondisi jalan yang tidak rata, disarankan penggunaan perangkat tambahan seperti stabilizer atau mounting peredam getaran pada kamera. Hal ini penting karena getaran dapat mempengaruhi kualitas citra yang ditangkap dan berdampak pada performa deteksi objek. Selain itu, pengujian lebih lanjut pada berbagai kondisi lingkungan dan permukaan jalan juga perlu dilakukan untuk memastikan keandalan sistem secara menyeluruh.
b. Dalam peningkatan kualitas akuisisi citra disarankan menggunakan kamera dengan resolusi dan kualitas sensor yang lebih baik serta memiliki ketahanan terhadap paparan panas. Karena peningkatan suhu pada webcam dapat menurunkan ketajaman citra menjadi sedikit kabur dan memengaruhi kemampuan sistem dalam mengenali objek.
c. Pengembangan sistem selanjutnya disarankan untuk menambahkan sensor pendukung seperti LiDAR, sehingga memungkinkan akurasi deteksi pada objek kecil dan jarak jauh dapat ditingkatkan.

DAFTAR PUSTAKA

Abdullah, Q., Shah, N. S. M., Mohamad, M., Ali, M. H. K., Farah, N., Salh, A., ... & Saif, A. (2021). Real-time autonomous robot for object tracking using vision system. *arXiv preprint arXiv:2105.00852*.

Afif, M. T., & Pratiwi, I. A. P. (2015). Analisis perbandingan baterai lithium-ion, lithium-polymer, lead acid dan nickel-metal hydride pada penggunaan mobil listrik-review. *Rekayasa Mesin*, *6*(2), 95-99.

Alam, H., Angga, M., & Widya, H. (2022). Penggunaan Arduino Uno Untuk Mendeteksi In dan Out Pengunjung Ruang Kantor. *JET (Journal of Electrical Technology)*, *7*(2), 96-99.

Ali Rohman, F., Ir Abraham Lomi, E., & Widodo Pudji Muljanto, I. (2020*) Rancang Bangun Prototipe Sistem Mnnitoring PJUTS Berbasis IoT Menggunkan Aplikasi Bot Telegram*. Jurnal JASTEN ITN Malang

Amir Hossein Ranjbar. Anahita Banei. Amir Khoobroo. Babak Fahimi. "Online Estimation of State of Charge Li-Ion Batteries Using Impuls Response Concept". IEEE Transactions on Smart Grid, Vol. 3, No. 1. March 2011.

Artanto, G. A., Hanafi, Z., Iman, R. M., & Stefanie, A. (2025). IMPLEMENTASI YOLOV8 PADA SISTEM DETEKSI PENYAKIT IKAN MAS KOKI MENGGUNAKAN RASPBERRY PI 5. *Jurnal Informatika dan Teknik Elektro Terapan*, *13*(3).

Beta, S., & Astuti, S. (2019). Modul timbangan benda digital dilengkapi led rgb dan dfplayer mini. *Orbith: Majalah Ilmiah Pengembangan Rekayasa dan Sosial*, *15*(1), 10-15.

Chen, C., Jin, J., & He, L. (2008, November). A new battery management system for li-ion battery packs. In *APCCAS 2008-2008 IEEE Asia Pacific Conference on Circuits and Systems* (pp. 1312-1315). IEEE.

Cristri, A. W., & Iskandar, R. F. (2017). Analysis and design of dynamic buck converter with change in value of load impedance. *Procedia engineering*, *170*, 398-403.

Diwan, T., Anirudh, G., & Tembhurne, J. V. (2023). Object detection using YOLO: challenges, architectural successors, datasets and applications. *multimedia Tools and Applications*, *82*(6), 9243-9275.

Djuandi, F. (2011). Pengenalan arduino. *E-book. www. tobuku*, *24*.

Firdausy, K., Riyadi, S., & Sutikno, T. (2008). Aplikasi Webcam Untuk Sistem Pemantauan Ruang Berbasis Web. *Telkomnika*, *6*(1), 39.

Hayati, N. J., Singasatia, D., & Muttaqin, M. R. (2023). Object tracking menggunakan algoritma you only look once (yolo) v8 untuk menghitung kendaraan. *Komputa: Jurnal Ilmiah Komputer dan Informatika*, *12*(2), 91-99.

Hermawati, M., Nuhi, M. H., Andari, A., Marito, E. E., Farros, N., & Josua, H. (2024). Penegakan hukum bagi pengguna sepeda listrik di jalan raya dalam perspektif hukum positif indonesia (Undang-Undang Lalu Lintas). *Media Hukum Indonesia (MHI)*, *2*(2), 66-73.

Hilal, Y. N., Muliandhi, P., & Ardina, E. N. (2023). Analisa Balancing Bms (Battery Management System) Pada Pengisian Baterai Lithium-Ion Tipe Inr 18650 Dengan Metode Cut Off. *Simetris: Jurnal Teknik Mesin, Elektro Dan Ilmu Komputer*, *14*(2), 367-374.

Hudson, T. (2022). The Difference Between CCM and DCM Explained. *Article# A-0022 Rev*, *1*.

Husin, S., Lutfi, I., & Anisah, M. (2025). PENGUJIAN PENGENALAN WAJAH REAL-TIME DENGAN DLIB PADA RASPBERRY PI 5. *Jurnal Riset Multidisiplin Edukasi*, *2*(6), 254-269.

Husnan, H., Fatichah, C., & Dikairono, R. (2023). Deteksi Objek Menggunakan Metode YOLO dan Implementasinya pada Robot Bawah Air. *Jurnal Teknik ITS*, *12*(3), A221-A226.

Ikbal, M., & Saputra, R. A. (2024). Pengenalan Rambu Lalu Lintas Menggunakan Metode Yolov8. *JIKA (Jurnal Informatika)*, *8*(2), 204-212.

Indroasyoko, N., Wulandari, I. Y., Rudiansyah, H., & Fauzi, M. (2026). Implementasi Sistem Kendali Kecepatan Motor Induksi Berbasis IoT (ESP32-Blynk) pada Indoor Wall Climbing. *REMIK: Riset dan E-Jurnal Manajemen Informatika Komputer*, *10*(1), 105-115.

Iqbal, M. (2012). *Pembuatan Sistem Pendeteksi Wajah Menggunakan Sensor Kamera Face Detector Berbasis Arduino Atmega328* (Doctoral dissertation, Universitas Pendidikan Indonesia).

Jaelani, I., Sompie, S. R., & Mamahit, D. J. (2016). Rancang bangun rumah pintar otomatis berbasis sensor suhu, sensor cahaya, dan sensor hujan. *Jurnal Teknik Elektro dan Komputer*, *5*(1), 1-10.

Julivano, H. Y., & Gunawan Ariyanto, S. T. (2024). *Evaluasi Performa Algoritma YOLOv8 dalam Deteksi dan Klasifikasi Posisi Tidur* (Doctoral dissertation, Universitas Muhammadiyah Surakarta).

Masudi, N. (2014). Desain Controller Motor Bldc Untuk Meningkatkan Performa (Daya Output) Sepeda Motor Listrik. *Institut Teknologi Sepuluh Nopember, Surabaya*.

Medina-Perez, A., Sánchez Rodríguez, D. C., & Alonso González, I. G. (2021). An internet of thing architecture based on message queuing telemetry transport protocol and node-red: A case study for monitoring radon gas. *Smart Cities*.

Medjaldi, A., Slimani, Y., & Karkar, N. (2025). Cost-Effective Real-Time Obstacle Detection and Avoidance for AGVs using YOLOv8 and RGB-D Sensors. *Engineering, Technology & Applied Science Research*, *15*(2), 21738-21745.

Mucthar, F., Wibowo, S. A., & Ariwibisono, A. (2021). Penerapan IoT (Internet of Thing) Terhadap Rancang Bangun Sangkar Burung Pintar Untuk Burung Teriep. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *5*(1), 162-170.

Mulyanto, T. A., Habiby, M., Kusnadi, K., & Adam, R. (2021). Home Automation System Dengan Menggunakan Raspberry Pi 4. *Jurnal Digit: Digital of Information Technology*, *11*(1), 60-73.

Muslihudin, M., Renvillia, W., Taufiq, T., Andoyo, A., & Susanto, F. (2018). Implementasi Aplikasi Rumah Pintar Berbasis Android Dengan Arduino Microcontroller. *Jurnal Keteknikan dan Sains (JUTEKS)*, *1*(1), 23-31.

Musthofa, A. M. R., & Aribowo, W. (2025). Rancang Bangun Sistem Kontrol dan Monitoring Lampu Penerangan Jalan Umum (PJU) Menggunkan Yolo dan Node-red. *JURNAL TEKNIK ELEKTRO*, *14*(1), 95-101.

Nainggolan, B., Inaswara, F., Pratiwi, G., & Ramadhan, H. (2016). Rancang bangun sepeda listrik Menggunakan panel surya sebagai pengisi baterai. *Jurnal Poli-Teknologi*, *15*(3).

Nikouei, S. Y., Chen, Y., Song, S., Xu, R., Choi, B. Y., & Faughnan, T. R. (2018, July). Real-time human detection as an edge service enabled by a lightweight cnn. In *2018 IEEE International Conference on Edge Computing (EDGE)* (pp. 125-129). IEEE.

Noviansyah, M., & Saiyar, H. (2019). Perancangan Alat Kontrol Relay Lampu Rumah Via Mobile (Vol. 4).

Nugroho, H., Kurniawan, M., & Saidatin, N. (2019, September). Deteksi Wajah dan Mata dengan Menggunakan Metode Fitur Haar-Like pada Kamera WebCam. In *Prosiding Seminar Nasional Sains dan Teknologi Terapan* (Vol. 1, No. 1, pp. 261-266).

Okano, M. T., Lopes, W. A. C., Ruggero, S. M., Vendrametto, O., & Fernandes, J. C. L. (2025). Edge AI for Industrial Visual Inspection: YOLOv8-Based Visual Conformity Detection Using Raspberry Pi. *Algorithms*, *18*(8), 510.

Otong, M. (2019). Perancangan modular baterai lithium ion (Li-ion) untuk beban lampu LED. *Setrum: Sistem Kendali-Tenaga-elektronika-telekomunikasi-komputer*, *8*(2), 260-273.

Pesik, W. H. S., Mamuaya, S., Putra, A. N., & Mulyaman, E. (2025). Implementasi YOLOv8 untuk Deteksi dan Hitung Objek Manusia dengan Convolutional Neural Network. *Buffer Informatika*, *11*(1), 16-26.

Prakoso, K. S. B., & Prasetio, B. H. (2025). Implementasi Speech Recognition berbasis Raspberry Pi 5 pada Ekosistem Smart-Home menggunakan Algoritma Gated Recurrent Unit (GRU). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *9*(3).

Reis, D., Kupec, J., Hong, J., & Daoudi, A. (2023). Real-time flying object detection with YOLOv8. *arXiv preprint arXiv:2305.09972*.

Sabdi, A. (2023). Rancang Bangun Kendali Fuzzy Logic untuk Tegangan Keluaran Buck Converter. *MSI Transaction on Education*, *4*(2), 95-106.

Saputra, JWW, Naufal, MA, & Putra, OA (2024). Implementasi Yolo V8 pada Prototipe Autonomous Underwater Robot Berbasis Raspberry Pi 5 Guna Menanggulangi Pencemaran Sampah Plastik di Daerah Perairan. *Jurnal Sintaks Kekaguman* , *5* (11), 4482-4493.

Shinde, N., Sankad, S., & Patil, S. L. (2018, May). Design and study voltage characteristics of buck converter by Matlab Simulink. In *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 680-683). IEEE.

Sihombing, R. S. I., Harahap, W. A., & Rahman, W. K. (2024). Implementasi Yolo V8 Untuk Mendeteksi Mata Uang Rupiah Emisi Tahun 2022 Ber-Output Audio. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(4), 5900-5905.

Silalahi, M. R., Virgono, A., & Saputra, R. E. (2023). Alat Pengolahan Informasi MP3 Sistem Pengumuman RT/RW Berbasis Arduino. *eProceedings of Engineering*, *10*(1).

Sodikin, N. H., Samosir, A. S., & Komalasari, E. (2015). Rancang Bangun Prototipe Emulator Sel Surya Menggunakan Buck Converter Berbasis Arduino. *Jurnal Rekayasa dan Teknik Elektro*, *9*(3), 172-180.

Srinu, V., Mounica, P., Varalakshmi, S. K. S., & Teja, K. (2017). A Novel Speed Control of Brushless DC Motor Using Arduino UNO R3 and BOT. *Asian Journal of Applied Science and Technology (AJAST)*, *1*(7), 10-14.

Supriyo, B., Sihono, S., Utomo, K., & Krishna, B. (2024). RANCANG BANGUN ALAT PENGUBAH RPM DARI PULSA ENCODER KE TEGANGAN. *Orbith: Majalah Ilmiah Pengembangan Rekayasa dan Sosial*, *20*(2), 127-133.

Susanto, H., Pramana, R., & Mujahidin, M. (2013). Perancangan Sistem Telemetri Wireless untuk Mengukur Suhu dan Kelembaban Berbasis Arduino Uno R3 ATmega328p dan XBee Pro. *Universitas Maritim Raja Ali Tanjung Pinang*, *4*(1).

SYAFRIADI, M. (2017). *Rancang Bangun Alat Pendeteksi Warna Menggunakan Kamera Dan Output Suara Berbasis Komputer* (Doctoral dissertation, POLITEKNIK NEGERI SRIWIJAYA).

Topan, P. A., Fardila, D., Rohman, S. A., Bahri, S., Jenal, J., & Febriansyah, Y. (2022). Pemanfaatan Teknologi Arduino Dan DFPlayer Mini Untuk Perangkat Pemutar Audio Di Masjid Raudhatul Jannah Desa Gontar, Kabupaten Sumbawa, Nusa Tenggara Barat. *Jurnal Abdi Insani*, *9*(4), 1797-1807.

Wahyudi, A. A., Khumaidi, A., Rahmat, M. B., Riananda, D. P., Syai'in, M., & Endrasmono, J. (2024). Implementasi Robot Operating System (ROS) Untuk Meningkatkan Akurasi Deteksi Bola Menggunakan YOLO V5 Pada KRSBI-Beroda. *Jurnal Elektronika Dan Otomasi Industri*, *11*(2), 590-603.

Wang, G., Jin, P., Qi, Z., & Li, X. (2025). Traffic sign detection method based on improved YOLOv8. *Scientific Reports*, *15*(1), 19385.

Wardana, S. D. R., & Amalia, Z. (2024). The Effect of Throttle Signal Output Voltage Variation and Load on Current Consumption. *International Journal of Frontier Technology and Engineering*, *3*(1), 69-76.

Widhi Winata Sakti, W. (2023). Pengembangan Sistem Deteksi Otomatis FOD dengan YOLOv5 di Lingkungan Landasan Bandara. *SKYHAWK: Jurnal Aviasi Indonesia*, *3*(2), 286-296.

Wu, S., Yang, J., Wang, X., & Li, X. (2022). Iou-balanced loss functions for single-stage object detection. *Pattern Recognition Letters*, *156*, 96-103.

Yana, D. R., & Salcedo, E. (2025). Low-Cost Machine Vision System for Sorting Green Lentils (Lens Culinaris) Based on Pneumatic Ejection and Deep Learning. *arXiv preprint arXiv:2507.20531*.

Yanto, Y., Aziz, F., & Irmawati, I. (2023). YOLO-V8 peningkatan algoritma untuk deteksi pemakaian masker wajah. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(3), 1437-1444.