ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - EC234801

PENGEMBANGAN SISTEM KENDALI KURSI RODA BERBASIS SIBI DAN *BRAKING SYSTEM* MENGGUNAKAN LSTM dan YOLOv11

I Putu Deva Febriana
NRP 5024 21 1016

Dosen Pembimbing

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.

NIP 19700313199512 1 001

Dr. Eko Mulyanto Yuniarno,S.T.,M.T.
NIP 19680601199512 1 009

Program Studi Strata 1 (S1) Teknik Komputer
Departemen Teknik Komputer
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya
2025

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR - EC234801

PENGEMBANGAN SISTEM KENDALI KURSI RODA BERBASIS SIBI DAN *BRAKING SYSTEM* MENGGUNAKAN LSTM dan YOLOv11

I Putu Deva Febriana
NRP 5024 21 1016

Dosen Pembimbing

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.

NIP 19700313199512 1 001

Dr. Eko Mulyanto Yuniarno,S.T.,M.T.
NIP 19680601199512 1 009

Program Studi Strata 1 (S1) Teknik Komputer

Departemen Teknik Komputer

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2025

[Halaman ini sengaja dikosongkan]

FINAL PROJECT - EC234801

DEVELOPMENT OF WHEELCHAIR CONTROL SYSTEM BASED ON SIBI AND BRAKING SYSTEM USING LSTM AND YOLOv11

I Putu Deva Febriana
NRP 5024 21 1016

Advisor

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.

NIP 19700313199512 1 001

Dr. Eko Mulyanto Yuniarno,S.T.,M.T.
NIP 19680601199512 1 009

Undergraduate Study Program of Computer Engineering

Department of Computer Engineering

Faculty of Intelligent Electrical And Informatics Technology

Sepuluh Nopember Institute of Technology

Surabaya

2025

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PENGEMBANGAN SISTEM KENDALI KURSI RODA BERBASIS SIBI DAN *BRAKING SYSTEM* MENGGUNAKAN LSTM dan YOLOv11

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat memperoleh gelar Sarjana Teknik pada Program Studi S-1
Teknik Komputer
Departemen Teknik Komputer
Fakultas Teknik Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh: **I Putu Deva Febriana**
NRP. 5024211016

Disetujui Oleh:

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T. (Pembimbing 1)
NIP: 19700313199512 1 001
.....

Dr. Eko Mulyanto Yuniarno,S.T.,M.T. (Pembimbing 2)
NIP: 19680601199512 1 009
.....

Arta Kusuma Hernanda, S.T., M.T. (Penguji I)
NPP : 1996202311024
.....

Dr. Arief Kurniawan, S.T., M.T. (Penguji II)
NIP: 19740907200212 1 001
.....

Yusril Izza, S.T., M.Sc. (Penguji III)
NPP: 1992202511059
.....

Mengetahui,
Kepala Departemen Teknik Komputer, FTEIC

Dr. Arief Kurniawan, S.T., M.T.
NIP: 19740907200212 1 001

SURABAYA
Juni, 2025

[Halaman ini sengaja dikosongkan]

APPROVAL SHEET

DEVELOPMENT OF WHEELCHAIR CONTROL SYSTEM BASED ON SIBI AND BRAKING SYSTEM USING LSTM AND YOLOv11

FINAL PROJECT

Submitted to fulfill one of the requirements for obtaining a degree Bachelor of Engineering at
Undergraduate Study Program of Computer Engineering
Department of Computer Engineering
Faculty of Intelligent Electrical and Informatics Technology
Sepuluh Nopember Institute of Technology

By: **I Putu Deva Febriana**
NRP. 5024211016

Approved by Final Project Proposal Examiner Team:

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.
NIP: 19700313199512 1 001 (Advisor 1)

.....

Dr. Eko Mulyanto Yuniarno,S.T.,M.T.
NIP: 19680601199512 1 009 (Advisor 2)

.....

Arta Kusuma Hernanda, S.T., M.T.
NPP : 1996202311024 (Examiner I)

.....

Dr. Arief Kurniawan, S.T., M.T.
NIP: 19740907200212 1 001 (Examiner II)

.....

Yusril Izza, S.T., M.Sc.
NPP: 1992202511059 (Examiner III)

.....

Verified by,
Head of the Computer Engineering Department, FTEIC

Dr. Arief Kurniawan, S.T., M.T.
NIP: 19740907200212 1 001

SURABAYA
June, 2025

[Halaman ini sengaja dikosongkan]

PERNYATAAN ORISINALITAS

Yang bertanda tangan dibawah ini:

Nama Mahasiswa / NRP	:	I Putu Deva Febriana / 5024 21 1016
Departemen	:	Teknik Komputer
Dosen Pembimbing / NIP	:	Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T. / 19700313199512 1 001

Dengan ini menyatakan bahwa Tugas Akhir dengan judul "PENGEMBANGAN SISTEM KENDALI KURSI RODA BERBASIS SIBI DAN *BRAKING SYSTEM* MENGGUNAKAN LSTM dan YOLOv11" adalah hasil karya sendiri, berfsifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, Juni 2025

Mengetahui
Dosen Pembimbing

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.
NIP. 19700313199512 1 001

Mahasiswa

I Putu Deva Febriana
NRP. 5024 21 1016

[Halaman ini sengaja dikosongkan]

STATEMENT OF ORIGINALITY

The undersigned below:

Name of student / NRP	:	I Putu Deva Febriana / 5024 21 1016
Department	:	Computer Engineering
Advisor / NIP	:	Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T. / 19700313199512 1 001

Hereby declared that the Final Project with the title of "*DEVELOPMENT OF WHEELCHAIR CONTROL SYSTEM BASED ON SIBI AND BRAKING SYSTEM USING LSTM AND YOLOv11*" is the result of my own work, is original, and is written by following the rules of scientific writing.

If in future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with provisions that apply at Sepuluh Nopember Institute of Technology.

Surabaya, June 2025

Acknowledged
Advisor

Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.
NIP. 19700313199512 1 001

Student

I Putu Deva Febriana
NRP. 5024 21 1016

[Halaman ini sengaja dikosongkan]

ABSTRAK

PENGEMBANGAN SISTEM KENDALI KURSI RODA BERBASIS *SIBI* DAN *BRAKING SYSTEM* MENGGUNAKAN *LSTM* dan *YOLOv11*

Nama Mahasiswa / NRP: **I Putu Deva Febriana / 5024211016**
Departemen **: Teknik Komputer FTEIC - ITS**
Dosen Pembimbing **: 1. Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.**
2. Dr. Eko Mulyanto Yuniarno,S.T.,M.T.

Abstrak

Penelitian ini bertujuan untuk mengembangkan sistem kendali kursi roda berbasis gestur Sistem Bahasa Isyarat Indonesia (SIBI) dan sistem pengereman otomatis (automatic braking system) dengan menggunakan *Long Short-Term Memory* (LSTM) dan YOLOv11. Sistem ini dirancang untuk memberikan solusi inovatif bagi pengguna kursi roda, khususnya bagi mereka yang mengalami keterbatasan fisik dan komunikasi, dengan memanfaatkan isyarat tangan SIBI sebagai perintah untuk menggerakkan dan mengendalikan kursi roda. Teknologi YOLOv11 digunakan untuk mendeteksi objek yang berada di depan pengguna kursi roda, sedangkan LSTM diterapkan untuk mengolah urutan gestur dan memprediksi perintah yang diinginkan pengguna menggunakan gestur SIBI. Selain itu, sistem pengereman otomatis dikembangkan untuk meningkatkan keamanan pengguna, dengan memanfaatkan kamera yang dapat mendeteksi rintangan atau situasi darurat. Pengujian menunjukkan bahwa sistem yang diusulkan memiliki tingkat akurasi dan responsivitas yang tinggi, sehingga dapat meningkatkan kemandirian dan mobilitas pengguna kursi roda.

Kata Kunci: ***Kursi roda cerdas, SIBI gesture, YOLOv11, LSTM, Braking System.***

[Halaman ini sengaja dikosongkan]

ABSTRACT

DEVELOPMENT OF WHEELCHAIR CONTROL SYSTEM BASED ON SIBI AND BRAKING SYSTEM USING LSTM AND YOLOv11

Student Name / NRP: **I Putu Deva Febriana / 5024211016**
Department **: Computer Engineering FTEIC - ITS**
Advisor **: 1. Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.**
2. Dr. Eko Mulyanto Yuniarno,S.T.,M.T.

Abstract

This research aims to develop a wheelchair control system based on Indonesian Sign Language (SIBI) gestures and a auto braking system using YOLOv11 and Long Short-Term Memory (LSTM). The system is designed to provide an innovative solution for wheelchair users, particularly those with physical and communication limitations, by utilizing SIBI hand gestures as commands to move and control the wheelchair. YOLOv11 technology is employed to detect object around the user of wheelchair, while LSTM is applied to process gesture sequences and predict the user's intended commands using SIBI gestures. Additionally, the automatic braking system is developed to enhance user safety by utilizing a camera to detect obstacles or emergency situations. the testing shows that the proposed system has high accuracy and responsiveness, which can improve the independence and mobility of wheelchair users.

Keywords: ***Intelligent wheelchair, SIBI gesture, YOLOv11, LSTM, Braking System.***

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji dan syukur kehadirat Tuhan Yang Maha Esa, atas segala rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan penelitian ini yang berjudul PENGEMBANGAN SISTEM KENDALI KURSI RODA BERBASIS SIBI DAN *BRAKING SYSTEM* MENGGUNAKAN LSTM dan YOLOv11

Penelitian ini disusun dalam rangka pemenuhan Tugas Akhir sebagai syarat kelulusan Mahasiswa ITS. Oleh karena itu, penulis mengucapkan banyak terima kasih kepada Oleh karena itu, penulis mengucapkan terima kasih kepada;

1. Bapak Dr. Arief Kurniawan, S.T., M.T. selaku Kepala Departemen Teknik Komputer, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember.
2. Bapak Dr. Supeno Mardi Susiki Nugroho,S.T.,M.T.selaku Dosen Pembimbing I dan Bapak Dr. Eko Mulyanto Yuniarno,S.T.,M.T.selaku Dosen Pembimbing II yang telah memberikan arahan dan membantu penulis selama pengerjaan tugas akhir ini.
3. Bapak-Ibu dosen pengajar Departemen Teknik Komputer, atas ilmu yang telah diberikan kepada penulis selama menjalani masa perkuliahan.
4. Ibu, Adik, Kakek, Nenek, dan keluarga yang telah memberikan doa serta dukungan selama penulis mengenyam pendidikan.
5. I Gusti Ngurah Agung Hari Vijaya Kusuma selaku mentor dan inspirasi saya dalam menulis topik tugas akhir ini.
6. Teman - teman Laboratorium Telematika dan Multimedia Cerdas (B201) serta Laboratorium Multimedia dan Internet Of Things (MIOT) yang membantu penulis dalam pengerjaan tugas akhir ini
7. Tidak lupa kepada teman - teman Teknik Komputer yang memberikan motivasi dan semangat selama masa perkuliahan.

Akhir kata, semoga penelitian ini dapat memberikan manfaat kepada banyak pihak. Penulis menyadari jika tugas akhir ini masih jauh dari kata sempurna. Untuk itu penulis mengharapkan saran dan kritik yang bersifat membangun untuk dapat menuai hasi yang lebih baik lagi.

Surabaya, Mei 2025

I Putu Deva Febriana

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB I

PENDAHULUAN

1.1 Latar Belakang

Mobilitas adalah aspek penting dalam menunjang kehidupan sehari-hari yang memungkinkan seseorang untuk menjalani berbagai aktivitas secara mandiri. Kemampuan untuk bergerak secara mandiri memungkinkan individu berpartisipasi dalam aktivitas sosial, ekonomi, dan budaya. Namun, bagi individu yang mengalami kelumpuhan atau gangguan mobilitas, kebutuhan untuk alat bantu gerak, seperti kursi roda, menjadi sangat penting. Data dari Organisasi Kesehatan Dunia (WHO) menunjukkan bahwa lebih dari 75 juta orang di seluruh dunia membutuhkan kursi roda, namun hanya sebagian kecil yang memiliki akses ke kursi roda yang memadai [2].

Kelumpuhan dapat disebabkan oleh berbagai kondisi, termasuk cedera tulang belakang, stroke, atau penyakit neurologis [3]. Dalam kondisi ini, individu kehilangan kemampuan untuk mengontrol gerakan tubuh mereka, yang secara signifikan mempengaruhi kemandirian dan aktivitas sehari-hari. Oleh karena itu, teknologi asistif seperti kursi roda telah berkembang dari waktu ke waktu untuk memenuhi kebutuhan mobilitas pengguna dengan berbagai tingkat kelumpuhan.

Perkembangan kursi roda saat ini telah beralih dari desain manual menuju kursi roda yang lebih cerdas dan otonom, yang dilengkapi dengan teknologi terbaru untuk memberikan kebebasan bergerak yang lebih baik [4]. Dalam beberapa dekade terakhir, konsep kursi roda cerdas telah mengalami kemajuan signifikan, terutama dengan integrasi teknologi seperti kontrol berbasis otak, sensorik, dan algoritma kecerdasan buatan.Penelitian oleh Choi,Chung, dan Oh menunjukkan bahwa kontrol gerak kursi roda elektrik yang diintegrasikandengan joystick dapat meningkatkan keselamatan dan kenyamanan pengguna [5]. Salah satu pendekatan inovatif adalah menggunakan deep learning dalam pengembangan sistem kendali kursi roda. YOLOv11 (You Only Look Once) adalah salah satu algoritma deteksi objek terbaru yang dapat diadaptasi untuk mengenali objek manusia sebagai penerapan sistem pengereman otomatis. LSTM (Long Short-Term Memory) digunakan sebagai algoritma pendekatan untuk mengenali pola gerakan sehingga pengguna dengan keterbatasan fisik dapat mengendalikan kursi roda mereka menggunakan gerakan tangan atau isyarat bahasa tubuh [6].

SIBI (Sistem Isyarat Bahasa Indonesia) adalah sistem komunikasi visual yang digunakan oleh penyandang disabilitas pendengaran di Indonesia. Bahasa ini memanfaatkan berbagai gerakan tangan untuk mewakili kata atau konsep tertentu, yang memudahkan komunikasi antar pengguna [7]. Dalam konteks pengembangan teknologi kursi roda cerdas, gesture SIBI menjadi salah satu metode potensial untuk meningkatkan kontrol dan aksesibilitas bagi penyandang disabilitas fisik yang menggunakan bahasa isyarat sebagai alat komunikasi sehari-hari . Implementasi gesture ini tidak hanya meningkatkan inklusivitas, tetapi juga memberikan alternatif kendali yang lebih intuitif bagi pengguna yang tidak mampu menggunakan metode konven-

sional seperti joystick atau tombol.

Selain itu, implementasi LSTM (Long Short-Term Memory), yang merupakan jenis Recurrent Neural Network (RNN), memungkinkan sistem untuk mengenali pola gerakan secara lebih akurat dan responsif dalam pengendalian kursi roda [8]. Kombinasi dari algoritma deep learning ini dapat memberikan solusi optimal bagi pengguna kursi roda yang tidak hanya membutuhkan mobilitas fisik tetapi juga sistem kendali yang adaptif dan responsif.

Lebih jauh, pengembangan kursi roda cerdas yang dilengkapi dengan smart braking system bertujuan untuk meningkatkan keselamatan pengguna. Sistem pengereman otomatis ini dapat mendeteksi objek atau hambatan di sekitar pengguna dengan cepat dan melakukan pengereman secara otomatis, mengurangi risiko kecelakaan.

Dengan demikian, penelitian ini bertujuan untuk mengembangkan sistem kendali kursi roda cerdas yang berbasis gesture SIBI dan *braking system* menggunakan LSTM dan YOLOv11. Penggunaan teknologi ini diharapkan dapat memberikan solusi mobilitas yang lebih aman, efisien, dan inklusif bagi individu dengan kelumpuhan.

1.2 Rumusan Masalah

Berdasarkan hal yang telah dipaparkan di latar belakang, meskipun bahasa isyarat seperti SIBI telah diakui sebagai bentuk komunikasi bagi penyandang disabilitas pendengaran dan fisik, implementasinya dalam pengendalian kursi roda masih minim. Tidak banyak sistem kursi roda yang mendukung pengendalian berbasis gestur, terutama gestur bahasa isyarat yang spesifik. Maka dari itu diperlukannya sebuah sistem yang dapat berjalan secara baik dalam komputer lokal atau *Next Unit Computing* (NUC) untuk mendeteksi gestur SIBI dan sistem pengereman otomatis.

1.3 Batasan Masalah

Terdapat beberapa Batasan masalah untuk memperjelas penelitian yang dilakukan. Batasan-batasannya adalah sebagai berikut:

1. Batasan pada penggunaan gesture SIBI sebagai metode utama kendali kursi roda cerdas.
2. Penelitian dibatasi pada penggunaan LSTM untuk pengenalan dan deteksi gesture tangan SIBI yang dikombinasikan dengan perangkat keras seperti kamera.
3. Penggunaan LSTM dibatasi pada pemrosesan urutan data gesture yang diambil dari kamera.
4. Fokus pada pengembangan *braking system* untuk meningkatkan keselamatan pengguna kursi roda dalam mendeteksi objek di sekitar kursi roda dan melakukan pengereman otomatis.
5. Sistem braking ini dibatasi pada deteksi manusia di lingkungan sekitarnya tanpa memperhitungkan aspek lain seperti objek-objek yang ada di lingkungan pengujian.
6. Penelitian difokuskan pada pengembangan dan integrasi antara perangkat keras (kamera, motor kursi roda) dan perangkat lunak kontrol kursi roda dan *braking system*.
7. Pengontrol kursi roda akan diimplementasikan untuk memberikan respon sesuai dengan gestur yang dipanggil dan berhenti otomatis ketika mendeteksi manusia di depannya.

1.4 Tujuan

Tujuan dari penelitian ini adalah membuat sebuah sistem yang dapat berjalan baik dalam komputer lokal maupun *Next Unit Computing* untuk mendeteksi gestur SIBI dan sistem pengereman otomatis.

1.5 Manfaat

Terdapat dua poin manfaat dari penelitian ini. Manfaat dari penelitian ini bagi masyarakat adalah memperluas opsi dari sistem kendali kursi roda yakni membuat sebuah sistem untuk mendeteksi gestur SIBI dan sistem pengereman otomatis. Manfaat dari penelitian ini bagi penulis yakni dapat mengasah kemampuan penulis dalam berinovasi dan memecahkan masalah yang dihadapi penulis penyelesaian penelitian ini.

[Halaman ini sengaja dikosongkan]

BAB II
TINJAUAN PUSTAKA

2.1 Hasil penelitian terdahulu

Pada subbab berikut akan dijabarkan mengenai penelitian terdahulu yang mengambil topik yang sama ataupun beririsan sebagai referensi, serta dasar pengembangan dan inovasi pada tugas akhir ini.

2.1.1 *Deep Learning-Based Object Detection, Localisation and Tracking for Smart Wheelchair Healthcare Mobility*

Penelitian dengan judul ”Deep Learning-Based Object Detection, Localisation and Tracking for Smart Wheelchair Healthcare Mobility” yang diteliti oleh Lecrosnier, L., Khemmar, R., Ragot, N., et al. Penelitian ini bertujuan untuk mengembangkan sistem bantuan pengemudi tingkat lanjut untuk mobil. Kursi roda elektrik pintar untuk meningkatkan kemandirian penyandang disabilitas. Penelitian ini berfokus pada deteksi objek dalam ruangan, lokalisasi, dan pelacakan. Kursi roda, terutama pintu dan gagang pintu. Tujuan utama dari penelitian ini adalah untuk meningkatkan fungsionalitas kursi roda otonom. Ini memungkinkan untuk mendeteksi objek-objek ini di sekitar secara akurat. langkah pertama penelitian ini bertujuan untuk mengadaptasi algoritma deteksi objek YOLOv3 dengan kasus kami. Penggunaannya. Selain itu, penelitian ini menggunakan kamera Intel RealSense untuk Metode estimasi kedalaman. Terakhir, sebagai langkah ketiga dan terakhir, penelitian ini Kami menggunakan pendekatan pelacakan objek 3D berdasarkan algoritma SORT. [9]

2.1.2 Deteksi Fitur Huruf Sistem Isyarat Bahasa Indonesia menggunakan Metode *Chain Code*

Berdasarkan penelitian yang dilakukan oleh Afifah Nur dan Hendro Nugroho berhasil dalam melakukan deteksi SIBI dengan menggunakan 119 data training dan 52 data testing dengan akurasi terbesar yakni 96.961% dan akurasi terendah yaitu 84.762%. Penggunaan metode *Chain Code*, *Chain Code* adalah teknik representasi kontur atau bentuk dalam citra digital menggunakan urutan arah tertentu. Teknik ini digunakan dalam pemrosesan citra untuk menggambarkan tepi objek secara efisien dan kompak. Chain code merepresentasikan kontur dengan merekam langkah-langkah yang mengikuti batas objek berdasarkan arah relatif antara titik-titik tetangga. [10]

2.1.3 Pendeteksian Bahasa Isyarat Indonesia Secara *Real-Time* menggunakan *Long Short-Term Memory* (LSTM)

Penelitian yang dilakukan oleh Husna Moetia Putri, Fadlisyah, dan Wahyu Fuadi berhasil melakukan pendeteksian bahasa isyarat Indonesia secara *real-time* menggunakan arsitektur Long Short-Term Memory (LSTM) dan *MediaPipe Holistic*. Pada pengujian menggunakan 10 kosakata isyarat BISINDO didapat evaluasi akurasi sebesar 92% dengan menggunakan *bidirectional* layer LSTM, epoch 1000, hidden layer 64, batch size 32. Sedangkan, pada pengujian meng-

gunakan 30 kosakata isyarat BISINDO didapat evaluasi akurasi sebesar 65% dengan menggunakan 2 buah layer LSTM epoch 500, hidden layer 64, batch size 64.

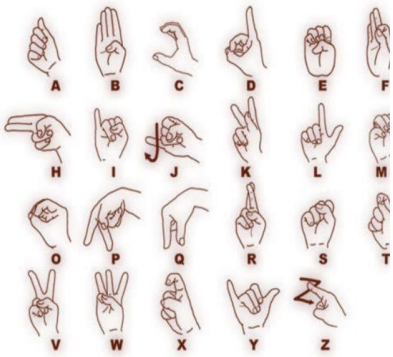
2.2 Teori Dasar

2.2.1 Gestur

Gestur adalah gerakan tubuh, terutama tangan, lengan, kepala, atau bagian tubuh lainnya, yang digunakan untuk menyampaikan informasi, ekspresi, atau niat tertentu, baik secara sengaja maupun tidak sengaja. Dalam konteks komunikasi non-verbal, gestur berfungsi untuk melengkapi atau menggantikan kata-kata, seperti melambaikan tangan untuk menyapa atau menganggukkan kepala sebagai tanda persetujuan. Dalam bidang teknologi, gesture sering digunakan sebagai metode interaksi manusia dengan komputer (HCI), di mana gerakan tubuh menjadi input untuk mengontrol sistem, misalnya melalui pengenalan gerakan tangan menggunakan sensor atau kamera. Gesture juga memiliki peran penting dalam psikologi dan studi kognitif, karena dapat mencerminkan proses berpikir dan emosi seseorang. Dalam penelitian berbasis analisis visual, gesture sering diidentifikasi untuk memahami pola gerakan atau mengembangkan teknologi interaktif. Dengan perannya yang luas, gesture menjadi elemen penting dalam berbagai bidang penelitian, terutama yang berfokus pada komunikasi, perilaku manusia, atau interaksi teknologi.

2.2.2 Sistem Isyarat Bahasa Indonesia (SIBI)

Sistem SIBI (Sistem Isyarat Bahasa Indonesia) adalah suatu sistem komunikasi yang menggunakan gerakan tangan, ekspresi wajah, dan posisi tubuh untuk menyampaikan pesan dalam bentuk bahasa [11]. SIBI dirancang sebagai representasi visual dari Bahasa Indonesia yang ditujukan untuk membantu komunikasi dengan atau di antara penyandang tunarungu. Sistem ini memadukan simbol-simbol isyarat yang sesuai dengan tata bahasa Bahasa Indonesia, sehingga setiap kata dan kalimat diisyaratkan berdasarkan urutan gramatikal Bahasa Indonesia yang baku.



Gambar 2.1: Gesture tangan untuk SIBI

2.2.3 Object Detection

Object Detection adalah teknik visi komputer yang membantu mengidentifikasi dan menemukan objek dalam gambar dan video. Dengan bentuk identifikasi dan pelokalan ini, deteksi objek dapat digunakan untuk menghitung jumlah item dalam sebuah skenario, menemukan dan

mengidentifikasi mereka dengan tepat, serta memberikan nama. Pendeteksian objek tetap menjadi salah satu aspek paling mendasar dan menantang dalam aplikasi visi komputer dan pemahaman citra. Kemajuan signifikan dalam deteksi objek telah dicapai melalui representasi objek yang lebih baik dan penggunaan model jaringan saraf dalam (*deep neural networks*). Metode deteksi objek telah berkembang pesat dengan penggunaan jaringan saraf konvolusional (CNN) dan algoritme khusus seperti YOLO (*You Only Look Once*) dan SSD (*Single Shot Multibox Detector*).

Selain kerangka YOLO, kemajuan juga telah dicapai di bidang deteksi objek dan pemrosesan gambar.Memperluas beberapa metode terkenal lainnya. Teknologi seperti R-CNN (Konvolusi Berbasis Wilayah), (Neural Networks nasional) dan penerusnya Fast R-CNN dan Faster R-CNN. Ini memainkan peran penting dalam meningkatkan akurasi deteksi objek. Metode ini bergantung pada dua proses tahap tersebut melakukan pencarian selektif untuk menghasilkan proposal wilayah dan menghasilkan jaringan saraf konvolusional. Klasifikasi dan penyempurnaan area ini. Pendekatan penting lainnya adalah *Single-Shot MultiBox Detector* (SSD), yang mirip dengan YOLO dalam mengutamakan kecepatan dan efisiensi dengan menghilangkan kebutuhan akan langkah proposal wilayah secara terpisah. Di sisi lain, metode seperti Mask R-CNN telah meningkatkan kemampuan deteksi objek hingga mencakup segmentasi instans, memungkinkan pelokalan objek yang lebih presisi hingga tingkat piksel. Perkembangan ini, bersama metode lainnya seperti RetinaNet dan EfficientDet, secara kolektif telah memperkaya keragaman algoritma deteksi objek. Setiap algoritma ini menawarkan keseimbangan yang berbeda antara kecepatan, akurasi, dan tingkat kompleksitas, yang dirancang untuk memenuhi berbagai kebutuhan aplikasi dan batasan komputasi [12].

2.2.4 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah jenis lapisan *deep learning* yang dirancang khusus untuk memproses data dalam bentuk grid seperti gambar. CNN merupakan lapisan dasar yang digunakan dalam membangun arsitektur YOLO. Dimana dalam penggunaannya CNN memiliki beberapa lapisan (*layers*) yang secara hierarkis mengekstrak fitur dari data masukan. Berikut adalah penjelasan mengenai komponen utama dari CNN:

2.2.4.1 Lapisan Konvolusi (*Convolutional Layer*)

Lapisan konvolusi adalah inti dari CNN. Pada lapisan ini, filter (*kernels*) diterapkan pada input untuk menghasilkan peta fitur (*feature map*). Filter ini bergerak melintasi input dengan operasi konvolusi, menangkap pola lokal seperti tepi, tekstur, dan bentuk. Setiap filter mempelajari fitur yang berbeda dari data. berikut merupakan rumus untuk operasi konvolusi dalam konteks matematika dan pemrosesan sinyal.

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau \tag{2.1}$$

2.2.4.2 Lapisan Aktivasi (*Activation Layer*)

Lapisan aktivasi memperkenalkan non-linearitas ke dalam model, memungkinkan jaringan untuk mempelajari hubungan yang kompleks. Fungsi aktivasi yang umum digunakan adalah ReLU (*Rectified Linear Unit*), yang mendefinisikan $f(x) = \max(0, x)$. Fungsi ini menggantikan nilai negatif dalam peta fitur dengan nol.

2.2.4.3 Lapisan Pooling (*Pooling Layer*)

Lapisan pooling mengurangi dimensi spasial dari peta fitur, mengurangi jumlah parameter dan komputasi dalam jaringan. Tipe pooling yang sering digunakan adalah *max pooling*, yang memilih nilai maksimum dalam setiap jendela pooling. Pooling membantu dalam membuat deteksi fitur lebih robust terhadap perubahan posisi dan skala.

2.2.4.4 Lapisan Fully Connected (*Fully Connected Layer*)

Pada lapisan *fully connected*, setiap *neuron* terhubung dengan semua *neuron* di lapisan sebelumnya. Lapisan ini biasanya terletak di akhir jaringan dan digunakan untuk menggabungkan fitur-fitur yang telah diekstraksi menjadi prediksi akhir.

2.2.4.5 Lapisan Output (*Output Layer*)

Lapisan output menghasilkan prediksi akhir dari model. Dalam konteks deteksi objek, lapisan ini dapat menghasilkan hasil kelas objek yang terdeteksi. Fungsi aktivasi softmax sering digunakan untuk klasifikasi multi-kelas, sedangkan fungsi sigmoid digunakan untuk deteksi objek dengan beberapa kelas.

2.2.5 YOLO(*You Look Only Once*)

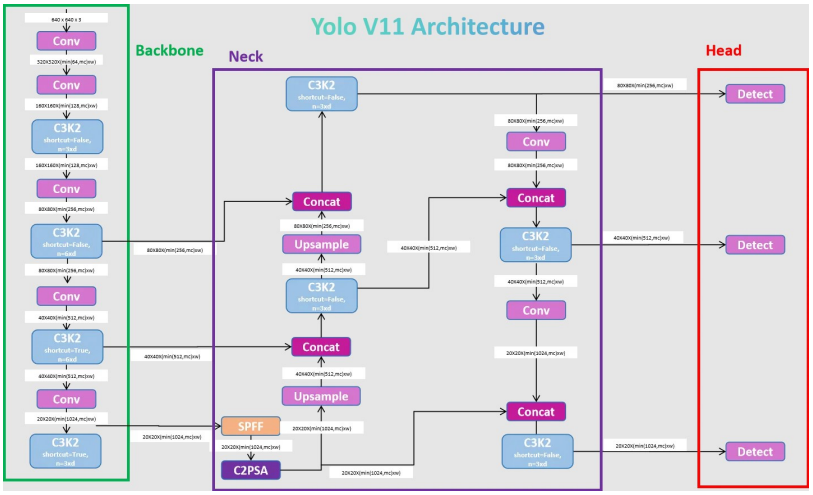
YOLO (*You Only Look Once*) adalah salah satu algoritma deteksi objek dalam visi komputer yang dirancang untuk mendeteksi dan mengidentifikasi objek dalam gambar atau video secara cepat dan akurat. YOLO bekerja dengan cara menganalisis gambar secara keseluruhan dalam satu kali pemrosesan (hence "*You Only Look Once*"), berbeda dengan metode lainnya yang sering memproses gambar secara bertahap atau melalui beberapa langkah, seperti mengidentifikasi wilayah objek terlebih dahulu baru kemudian mengklasifikasikan objek tersebut.

Pada dasarnya, YOLO membagi gambar menjadi grid dan, untuk setiap grid, memprediksi berbagai kotak pembatas (*bounding boxes*) dan kemungkinan kelas objek yang ada di dalamnya. Hal ini memungkinkan YOLO untuk mendeteksi beberapa objek dalam satu gambar secara bersamaan dalam satu waktu pemrosesan, sehingga sangat efisien dalam hal kecepatan.

2.2.6 YOLOv11

YOLOv11 merupakan iterasi terbaru dalam seri pengembangan YOLO yang memperkenalkan peningkatan arsitektur baru, termasuk mekanisme perhatian yang lebih baik, lapisan ekstraksi fitur yang lebih dalam, dan paradigma deteksi tanpa jangkar (*anchor-free*). Inovasi-inovasi ini dirancang untuk mengatasi tantangan dalam mendeteksi kendaraan yang lebih kecil, tersembunyi, atau bergerak cepat, sambil mempertahankan kemampuan inferensi waktu nyata. YOLOv11 juga dioptimalkan untuk percepatan perangkat keras, membuatnya lebih kompatibel dengan perangkat tepi yang digunakan dalam aplikasi kritis seperti deteksi emosi dan sistem transportasi cerdas [13].

Peningkatan dalam YOLOv11, khususnya dengan pengenalan deteksi tanpa jangkar dan mekanisme perhatian yang lebih baik, menandakan langkah maju dalam pengembangan sistem deteksi kendaraan yang kuat dan dapat diskalakan. Penggunaan YOLOv11 ini bertujuan untuk mengevaluasi kinerja YOLOv11 dalam konteks deteksi objek pada kursi roda, dengan fokus pada kemampuannya untuk menangani skenario deteksi yang kompleks dan waktu nyata. Hasil evaluasi YOLOv11 dibandingkan dengan pendahulunya, YOLOv8, menggunakan metrik utama seperti *presisi, recall, dan mean average precision* (mAP).

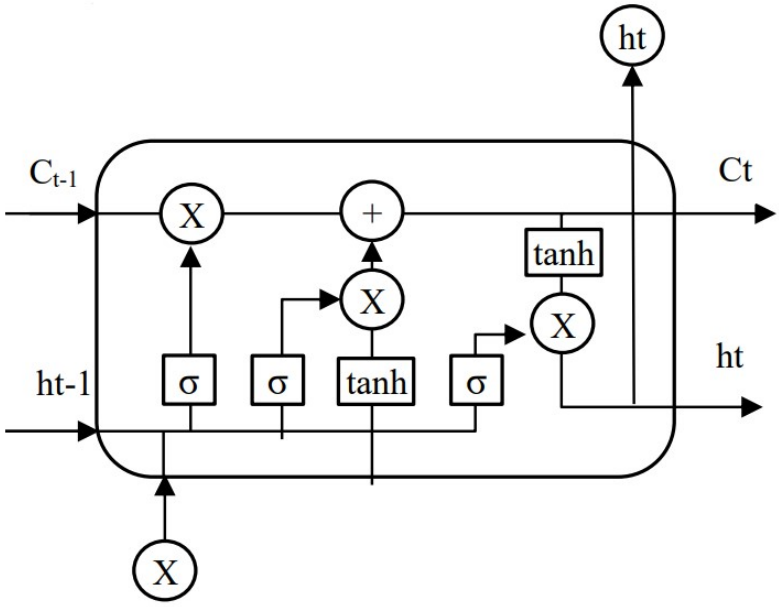


Gambar 2.2: Arsitektur dari YOLOv11

2.2.7 LSTM(*Long Short-Term Memory*)

Long Short-Term Memory (LSTM) (LSTM) adalah bentuk RNN yang sering digunakan untuk menghindari masalah penumpukan dengan gradasi, ketergantungan jangka panjang pada memproses atau membuat prediksi ke data deret waktu. RNN cenderung memiliki penumpukan pada gradien yang menyebabkan nilai gradien saling bertabrakan sehingga terdapat nilai gradien yang tidak jelas dan dapat menghilangkan nilai akumulasinya sehingga diperlukan LSTM untuk menghindari hal tersebut terjadi [14].

Menurut Sepp Hochreiter dan Jurgen Schmidhuber, model LSTM terbentuk dari berbagai rangkaian sel memori yang dapat menggantikan sel neuron pada hidden layer dari RNN. Model LSTM dapat menyaring data atau informasi memlalu struktur gates untuk dapat memperta-hankan informasi yang berhubungan dan mengubah keadaan dari sel memori. Struktur gerbang tersebut mencakup input gate, forgate gate, dan output gate [15].



Gambar 2.3: Arsitektur dari LSTM

LSTM terdiri dari rangkaian sel memori yang unik dan model LSTM menyaring informasi melalui struktur gerbang. Pada LSTM terdapat 3 gerbang atau Gates Unit dan 1 Cell states, yaitu input gates, forget gate, output gates, dan cell states. Persamaan dari setiap gate sebagai berikut:

2.2.7.1 Input gates

*Input gate*menerima informasi berupa *hidden state* yang berasal dari sel sebelumnya dan informasi baru yang berasal dari masukan saat ini, dan informasi tersebut digabungkan dan diproses menggunakan fungsi *sigmoid* dan *tanh*. Hasil dari fungsi sigmoid mengubah nilai dari 0 menjadi 1 dan menentukan informasi mana yang diperbarui. Nilai yang mendekati 0 berarti informasi tersebut tidak penting, dan nilai yang mendekati 1 berarti informasi tersebut penting. Hasil fungsi Tanh berupa nilai antara -1 dan 1 digunakan untuk membantu sel mempelajari informasi dengan lebih baik. Persamaan dari *input gate* adalah sebagai berikut:

$$i_t = \sigma(W_i \times X_i + U_i \times h_{t-1} + b_i) \tag{2.2}$$

Dalam persamaan ini, i_t mewakili nilai dari input gate, yang dihitung dengan menggunakan fungsi aktivasi sigmoid σ, yang menghasilkan output antara 0 dan 1. Ini memungkinkan model untuk mengontrol secara dinamis seberapa banyak informasi dari input saat ini (X_i) dan keadaan tersembunyi sebelumnya (h_{t-1}) yang akan diterima dan disimpan dalam memori. Faktor W_i dan U_i adalah bobot yang masing-masing terkait dengan input X_i dan keadaan tersembunyi sebelumnya h_{t-1}, sementara b_i adalah bias yang membantu menyesuaikan perhitungan tersebut.

2.2.7.2 Forget gates

Forget gates memutuskan informasi apa yang akan disimpan atau dibuang. Forget gate menerima informasi berupa *hidden state* yang berasal dari sel sebelumnya dan informasi baru yang berasal dari *input* saat ini. Informasi ini kemudian digabungkan dan diproses menggunakan fungsi sigmoid sehingga menghasilkan hasil dalam bentuk 0 hingga 1. Semakin dekat hasil yang diperoleh ke 0, semakin banyak informasi yang dibuang, dan sebaliknya, semakin dekat hasil yang diperoleh ke 1, semakin banyak informasi yang disimpan. Persamaan dari *forget gate* adalah sebagai berikut:

$$f_t = \sigma(W_f \times X_t + U_f \times h_{t-1} + b_f) \tag{2.3}$$

Pada dasarnya, forget gate memeriksa data baru (X_t) dan informasi historis dari hidden state sebelumnya (h_{t-1}) untuk memutuskan seberapa besar kontribusi c_{t-1} yang akan diteruskan ke sel memori pada waktu t. Ini adalah langkah penting dalam mengontrol "memori jangka panjang" pada LSTM, yang memungkinkan jaringan untuk menangani dependensi jangka panjang dalam data.

Jika nilai f_t mendekati 0, maka sebagian besar informasi dari memori sebelumnya akan dilupakan. Sebaliknya, jika f_t mendekati 1, informasi tersebut akan dipertahankan dalam memori selanjutnya. Ini membantu LSTM mengelola informasi yang relevan dan membuang informasi yang tidak berguna.

2.2.7.3 Output gates

Output gates menentukan *hidden state* yang dikirim ke sel berikutnya.*Output gate* menerima informasi berupa *hidden state* yang berasal dari sel sebelumnya, informasi baru yang be-

rasal dari masukan saat ini, dan informasi tersebut digabungkan dan diproses dengan fungsi sigmoid. Status sel baru ditangani melalui fungsi tanh. Hasil fungsi Tanh dikalikan dengan hasil fungsi sigmoid untuk memperoleh informasi yang disimpan dalam keadaan tersembunyi baru. Status tersembunyi dan status sel baru ditransfer ke sel berikutnya. Berikut adalah persamaan dari *output gate*:

$$o_t = \sigma(W_o \times X_i + U_o \times h_{t-1} + b_o) \tag{2.4}$$

Output gate pada LSTM mengontrol informasi yang akan diteruskan dari cell state c_t ke hidden state h_t dan akhirnya menjadi output model. Ini penting karena hidden state (h_t) membawa informasi yang diperlukan untuk prediksi pada langkah berikutnya, dan juga dapat digunakan sebagai output model dalam tugas-tugas seperti klasifikasi atau prediksi deret waktu.

Setelah output gate dihitung, informasi dari cell state c_t diubah dengan fungsi tanh (fungsi aktivasi hiperbolik tangen) untuk memastikan nilai berada dalam rentang yang sesuai sebelum diteruskan melalui output gate. Prosesnya adalah sebagai berikut:

$$h_t = o_t \times \tanh(c_t) \tag{2.5}$$

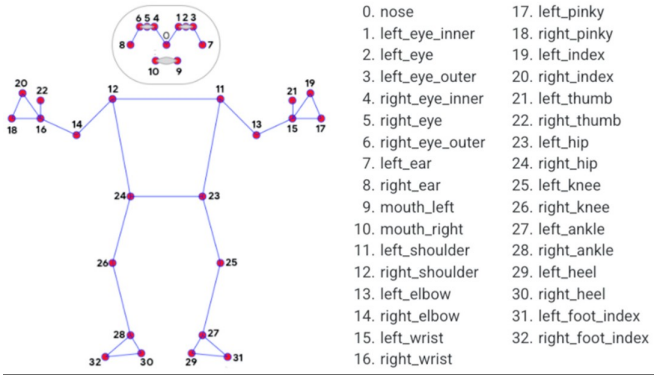
Di sini, $\tanh(c_t)$ mengubah nilai cell state menjadi nilai yang lebih terkontrol sebelum dihasilkan sebagai output.

2.2.8 Mediapipe

MediaPipe adalah kerangka kerja yang dirancang oleh Google untuk dikembangkan. Saluran pengenalan waktu nyata. MediaPipe memungkinkan pengembang untuk berintegrasi Integrasikan berbagai jenis data sensor seperti video, audio, dan data lainnya ke dalam satu platform ini efisien dan dapat berjalan di berbagai perangkat mulai dari seluler hingga desktop. Framework ini menggunakan konsep “grafik”, jadi setiap node dalam grafik Ini adalah ”kalkulator” yang melakukan tugas-tugas tertentu seperti deteksi objek, pelacakan, dll. pose, atau segmentasi gambar. Masing-masing node ini dapat dikonfigurasi melalui GraphConfig.Salah satu penerapan MediaPipe yang paling dikenal adalah pada estimasi pose manusia menggunakan MediaPipe Pose. Metode ini menggabungkan estimasi pose 2D dengan model humanoid yang lebih kompleks serta metode optimasi untuk mengestimasi sudut sendi pada pose 3D. Metode ini terbukti efektif dalam mengatasi masalah ambiguitas kedalaman pada es- timasi pose 3D dan dapat berjalan secara real-time bahkan pada perangkat tanpa GPU [16].

2.2.8.1 MediaPipe Pose

MediaPipe Pose (MPP) adalah kerangka kerja sumber terbuka yang disediakan oleh Google. Digunakan untuk mendapatkan perkiraan koordinat sendi manusia 2D dalam setiap bingkai gambar batang. MediaPipe Pose membangun saluran untuk memproses data kognitif dalam beberapa bagian untuk video menggunakan *Machine Learning* (ML). Penerapan MPP BlazePose mengekstrak 33 landmark 2D tubuh manusia seperti yang ditunjukkan. Dapat dilihat pada Gambar 2.4. BlazePose adalah arsitektur pembelajaran mesin ringan yang memungkinkan kesamaan pekerjaan real-time di ponsel dan PC dengan inferensi CPU.Ketika menggunakan koordinat yang dinormalisasi untuk estimasi pose, rasio invers harus dikalikan dengan nilai piksel sumbu-y [17].



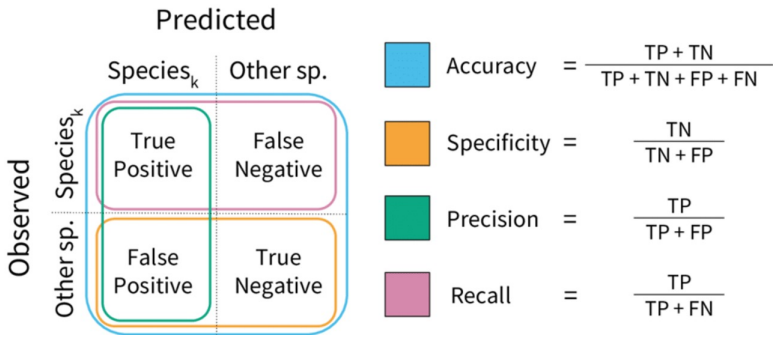
Gambar 2.4: Pose MediaPipe

2.2.9 Classification Performance

Classification Performance adalah sekumpulan metrik yang digunakan untuk mengevaluasi kinerja model klasifikasi. Metrik ini digunakan untuk menilai akurasi model, presisi, perolehan kembali, dan aspek lainnya. Ini sering digunakan untuk membandingkan model yang berbeda atau menyetel satu model untuk kinerja optimal. Metrik klasifikasi dapat dikelompokkan menjadi tiga kategori utama: Akurasi, sensitivitas, spesifisitas. Akurasi mengukur performa model secara keseluruhan dan biasanya merupakan metrik yang paling penting. Sensitivitas dan spesifisitas mengukur seberapa baik suatu model dapat membedakan kelas yang berbeda. Terakhir, metrik lain seperti skor AUC, skor F1, dan skor Kappa mengukur akurasi dan pengenalan model. Dalam hal ini, confusion matrix merupakan salah satu perhitungan yang sering digunakan dalam kasus pengklasifikasian.

2.2.9.1 Confusion Matrix

Confusion matrix adalah salah satu pengukuran paling sederhana Mencari tingkat nilai kebenaran dan keakuratan model. *Confusion matrix* merupakan tabel dua dimensi berisi data aktual dan prediksi, masing-masing dengan kelasnya. Data aktual ada di bagian kolom tabel dan data prediksi ada di bagian baris tabel. Gambar 2.5 merupakan representasi visual dari perhitungan *confusion matrix*.



Gambar 2.5: Confusion Matrix

2.2.10 Evaluation Matrix

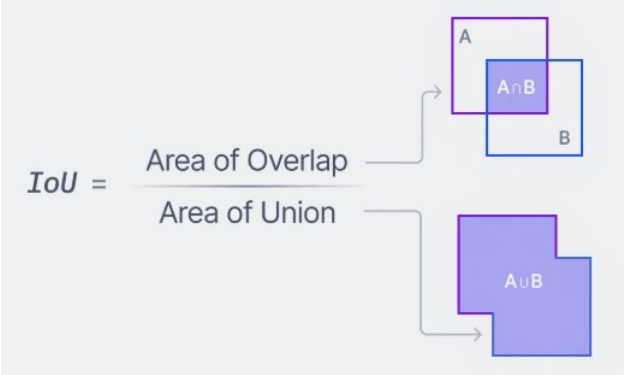
Evaluation Matrix adalah cara untuk mengevaluasi sejumlah besar peringkat secara objektif. Pilihan untuk beberapa kriteria. Kriteria ini akan diprioritaskan sebelum evaluasi. Ini telah

dibuat dengan lebih menekankan pada item yang paling penting.Terdapat dua tingkatan dalam *evaluation matrix*. tingkat pertama ini bertindak sebagai filter di mana setiap opsi dievaluasi berdasarkan kriteria yang diperlukan pilihan-pilihan itu yang memenuhi semua kriteria yang dipersyaratkan akan maju ke tingkat kedua dan dievaluasi kriteria prioritas.

Metrik seperti akurasi, presisi, *recall* adalah kunci saat mengevaluasi seberapa efektif suatu model dalam mendeteksi dan mengidentifikasi objek. Lebih lanjut, analisis akurasi memberikan wawasan kuantitatif yang signifikan mengenai kinerja algoritma deteksi objek, memberikan detail lebih lanjut mengenai kemampuan algoritma dalam menghasilkan deteksi yang akurat. Mengenali kesalahan melalui metrik evaluasi adalah langkah kritis dalam penelitian deteksi, seperti dalam kasus deteksi asap yang kami teliti. Langkah ini memfasilitasi identifikasi dan pemahaman tentang potensi kesalahan dalamalgoritma, yang dapat membuka jalan untuk perbaikan dan peningkatan metode deteksi. Selanjutnya, metrik evaluasi juga mendukung pengoptimalan hiperparameter algoritma. Dalam konteks penelitian ini, berbagai indikator evaluasi telah diterapkan, termasuk presisi, *recall*, dan *Mean Average Precision* (mAP). Dengan menggabungkan metode evaluasi ini,Penelitian ini bertujuan untuk menyajikan analisis komprehensif terhadap kinerja algoritma deteksi objek yang sedang ditinjau.

2.2.11 Intersection over Union(IoU)

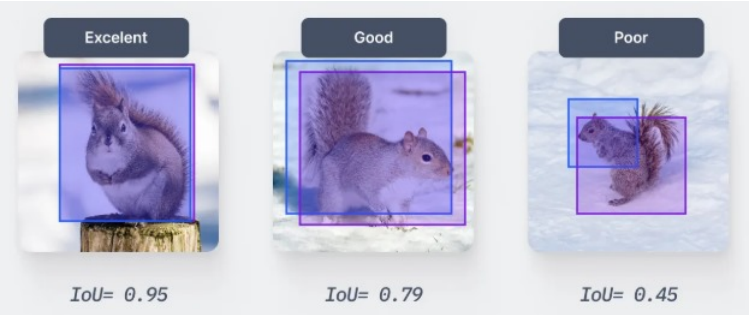
Intersection over Union adalah metrik populer untuk mengukur akurasi pelokalan dan menghitung kesalahan pelokalan dalam model deteksi objek. Ini menghitung jumlah tumpang tindih antara dua kotak pembataskotak pembatas yang diprediksi dan kotak pembatas kebenaran dasar. IoU adalah rasio perpotongan luas dua kotak dengan luas gabungannya. Kotak pembatas kebenaran dasar dan kotak pembatas yang diantisipasi keduanya mencakup area gabungan, yang merupakan penyebutnya.



Gambar 2.6: Penggambaran dari IoU

IoU atau *Intersection over Union* merupakan metode penilaian yang mengukur efektivitas model deteksi objek dengan membandingkan area *overlap* antara prediksi model dan posisi objek aktual. Skala nilai IoU berada antara 0 hingga 1, nilai yang lebih dekat ke 1 menandakan prediksi yang sangat akurat terhadap objek sebenarnya. Nilai IoU yang lebih tinggi menunjukkan bahwa model tersebut lebih tepat dalam mengidentifikasi dan menentukan lokasi objek. Dalam skenario penelitian, misalnya pengenalan otomatis seekor hewan dalam sebuah gambar, model pembelajaran mendalam akan menciptakan sebuah kotak pembatas sebagai prediksi lokasi hewan tersebut. Kotak pembatas ini lalu dibandingkan dengan kotak kebenaran dasar—area yang telah ditentukan secara manual sebagai lokasi sebenarnya dari hewan dalam

gambar. IoU kemudian dihitung untuk menilai seberapa baik kotak prediksi menutupi kotak kebenaran dasar, dengan mempertimbangkan area bersama dan area gabungan dari kedua kotak tersebut.



Gambar 2.7: Contoh perbandingan performa IoU

IoU menawarkan metode kuantitatif untuk mengevaluasi seberapa akurat model dalam mengenali dan menandai objek dalam gambar. Dalam proses pelatihan, nilai IoU minimal yang ditetapkan memungkinkan penetapan batas bagi kotak prediksi untuk dianggap cocok dengan deteksi yang benar, memfasilitasi penyesuaian antara tingkat akurasi deteksi dan insiden false positif. Penentuan batas IoU tidak bersifat tetap dan dapat disesuaikan, dengan 0,5 yang menjadi standar patokan awal. Kotak prediksi dengan IoU minimal 0,5 terhadap deteksi positif dianggap valid. Penyesuaian ambang nilai ini mempengaruhi balance antara presisi dan daya tangkap, dengan peningkatan nilai ambang cenderung mengurangi kesalahan positif namun berpotensi mengabaikan beberapa deteksi valid.Penggunaan nilai kebenaran dasar dalam IoU merupakan perbandingan standar antara prediksi dan kondisi aktual objek yang diidentifikasi. Penandaan kotak kebenaran dasar, dilakukan secara manual oleh pakar, menentukan batas pasti objek dalam gambar. Skor IoU yang dihasilkan dari perbandingan antara prediksi model dengan batas ini memberikan wawasan terhadap efektivitas model dalam deteksi objek. Dataset kebenaran dasar, yang meliputi kotak pembatas yang ditandai secara manual, menjadi kunci dalam proses evaluasi ini, memberikan dasar objektif untuk mengukur kinerja algoritme deteksi objek [18].

2.2.12 Tensorflow

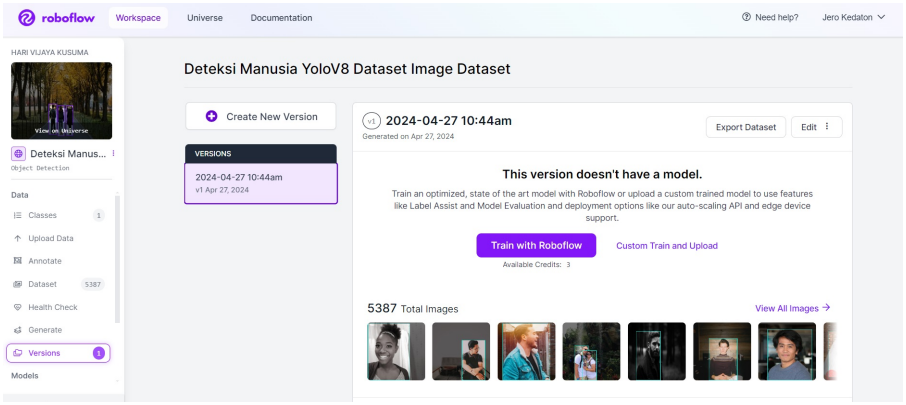
TensorFlow adalah sistem pembelajaran mesin yang beroperasi dalam skala besar dan di lingkungan yang heterogen [19]. TensorFlow menggunakan grafik aliran data untuk merepresentasikan komputasi, status bersama, dan operasi yang mengubah status tersebut. Ini memetakan node grafik aliran data di banyak mesin dalam sebuah cluster, dan di dalam mesin di beberapa perangkat komputasi, termasuk CPU *multicore*, general-purpose GPU, dan ASIC yang dirancang khusus yang dikenal sebagai Tensor Processing Units (TPU). Arsitektur ini memberikan fleksibilitas kepada pengembang aplikasi. TensorFlow memungkinkan pengembang bereksperimen dengan pengoptimalan baru dan algoritma pelatihan. TensorFlow mendukung berbagai aplikasi, dengan dukungan yang sangat kuat untuk pelatihan dan inferensi pada jaringan neural dalam. Beberapa layanan Google menggunakan TensorFlow dalam produksinya, dan telah banyak digunakan untuk penelitian pembelajaran mesin.

2.2.12.1 Tensorflow-Keras

Keras merupakan library yang dikembangkan oleh Francois Chollet. Keras ini dirancang oleh pengembang agar cepat, mudah diimplementasikan, dan modular secara alami. Keras diadopsi sebagai API tingkat tinggi untuk mengembangkan algoritma pembelajaran mendalam alias *deep learning* [19].

2.2.13 Roboflow

RoboFlow adalah platform yang mendukung pengembangan dan penyebaran aplikasi visi komputer dengan menyediakan fitur - fitur untuk manajemen dan peningkatan dataset. Platform ini dirancang untuk memudahkan pengolahan, analisis, dan augmentasi data visual, sehingga mempercepat siklus pengembangan dan peningkatan model pembelajaran mesin.



Gambar 2.8: Interface Roboflow

RoboFlow menyediakan serangkaian fungsi yang membantu dalam proses pengembangan model visi komputer, termasuk:

- Annotasi Data : RoboFlow memungkinkan pengguna untuk menandai gambar dengan alat bantu yang intuitif, mempercepat proses pembuatan label untuk dataset.
- Augmentasi Data: Melalui augmentasi data, RoboFlow dapat secara otomatis memodifikasi gambar dalam dataset untuk menciptakan variasi, yang membantu dalam meningkatkan robustness model yang dilatih.
- Konversi Format Data: Platform ini mendukung konversi antara berbagai format dataset yang populer, memudahkan pengguna dalam mempersiapkan data untuk berbagai jenis algoritma pembelajaran mesin.
- Pemisahan Dataset: RoboFlow menyediakan fungsi untuk membagi dataset menjadi set pelatihan, validasi, dan pengujian, yang merupakan langkah penting dalam validasi model.

RoboFlow menawarkan integrasi yang mulus dengan banyak kerangka kerja pembelajaran mesin populer seperti TensorFlow, PyTorch, dan YOLO. Integrasi ini memungkinkan pengembang:

- Ekspor Data: Pengguna dapat dengan mudah mengekspor dataset mereka dalam format yang siap digunakan oleh kerangka kerja pembelajaran mesin pilihan mereka.

- Pelatihan Model: Platform ini menyediakan alat yang memungkinkan pengguna untuk langsung melatih model mereka menggunakan dataset yang telah disiapkan dan dioptimalkan.
- Evaluasi Model: RoboFlow menyediakan metrik untuk mengukur kinerja model, membantu pengguna memahami efektivitas model mereka dan membuat penyesuaian yang diperlukan.

2.2.14 ESP-32

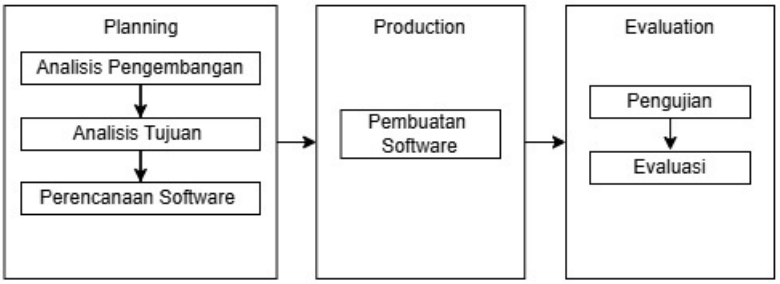
ESP32 adalah modul mikrokontroler terintegrasi yang memiliki fitur lengkap dan kinerja tinggi. Modul ini merupakan pengembangan dari ESP8266, yang merupakan modul WiFi populer. ESP32 memiliki dua prosesor komputasi, satu prosesor untuk mengelola jaringan WiFi dan Bluetooth, serta satu prosesor lainnya untuk menjalankan aplikasi. Dilengkapi dengan memori RAM yang cukup besar untuk menyimpan data. Fitur yang berguna seperti TCP/IP, HTTP, dan FTP. Modul ini juga dilengkapi fitur pemrosesan sinyal analog, dukungan untuk sensor, dan dukungan untuk perangkat masukan/keluaran (I/O) digital. ESP32 juga memiliki dukungan untuk konektivitas Bluetooth. Dapat digunakan untuk mengendalikan perangkat yang terhubung dengan Bluetooth.

BAB III
METODOLOGI

Penelitian ini dilakukan sesuai dengan desain sistem berikut beserta implementasinya. Desain sistem adalah konsep dari pembuatan dan perancangan infrastruktur dan kemudian diwujudkan dalam bentuk alur yang harus dikerjakan.

3.1 Metode Penelitian

Metode penelitian yang akan digunakan pada penelitian ini adalah penelitian pengembangan dari Richey dan Klein [20]. Metode ini memiliki tiga tahapan yaitu *planning*, *production*, dan *evaluation*. Berikut adalah diagram dari metode yang akan digunakan dalam penelitian ini.



Gambar 3.1: Diagram alur penelitian

3.1.1 Perencanaan(Planning)

Pada bagian perencanaan, peneliti melakukan analisis pengembangan terhadpat kursi roda yang sudah ada sebelumnya. Analisis pengembangan ini diperlukan agar penilit dapat mengetahui hal apa yang bisa dikembang dari kursi roda. Berdasarkan hasil dari analisis pengembangan, hal yang dapat dikembangkan dari kursi roda adalah bagian *software*.

3.1.2 Produksi(Production)

Tahan produksi meliputi pembuatan dari *software* dari kursi roda berdasarkan dari perencanaan *software* dan analisis tujuan yang telah ditentukan di tahan *planning*.

3.1.3 Evaluasi(Evaluation)

Pada tahap ini, dilakukan pengujian secara bertahap untuk memastikan kursi roda dapat beroperasi sesuai dengan tujuan yang telah ditetapkan.

3.2 Analisis sistem

Sebelum memulai pengembangan sistem, penting untuk melakukan analisis terhadap sistem. Analisis ini diperlukan untuk mengetahui kebutuhan dan tujuan yang ingin dicapai. Tujuan dari analisis kebutuhan adalah identifikasi berbagai elemen yang diperlukan untuk sistem, baik dari sudut pandang fungsional maupun non-fungsional. Untuk memastikan bahwa sistem yang

dikembangkan memenuhi persyaratan perlu ditetapkan spesifikasi dan persyaratan.

3.2.1 Analisis Kebutuhan

Analisis kebutuhan dilakukan dengan mengumpulkan data dari studi literatur terkait. Analisis ini bertujuan untuk mengidentifikasi fitur-fitur dan spesifikasi teknis yang diperlukan dalam pengembangan sistem. Berdasarkan hasil analisis, beberapa kebutuhan sistem yang harus dipenuhi adalah sebagai berikut:

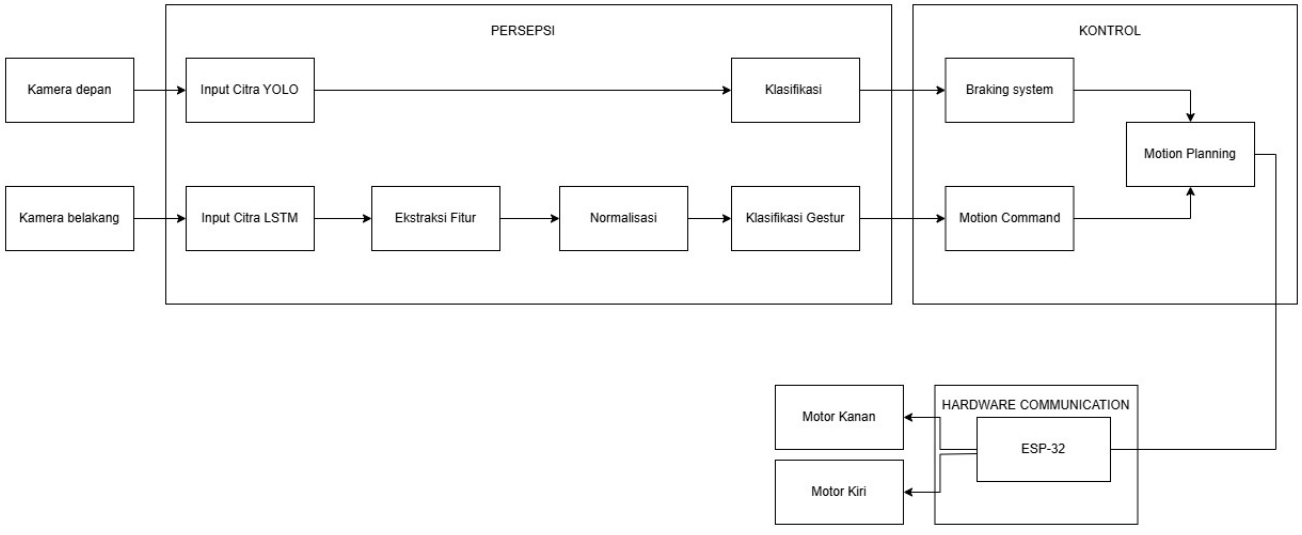
1. Penggunaan gesture SIBI sebagai metode utama kendali kursi roda.
2. Penelitian pada penggunaan LSTM untuk pengenalan dan deteksi gestur tangan SIBI yang dikombinasikan dengan perangkat keras seperti kamera.
3. Penggunaan LSTM pada pemrosesan urutan data gesture yang diambil dari kamera.
4. Pengembangan *braking system* untuk meningkatkan keselamatan pengguna kursi roda dalam mendeteksi objek di sekitar kursi roda dan melakukan pengereman otomatis.
5. Sistem braking ini akan berfungsi untuk mendeteksi manusia tanpa memperhitungkan aspek lain seperti objek-objek yang ada di lingkungan pengujian
6. Penelitian difokuskan pada pengembangan dan integrasi antara perangkat keras (kamera, motor kursi roda) dan perangkat lunak kontrol kursi roda dan braking system.
7. Pengontrol kursi roda akan diimplementasikan untuk memberikan respon sesuai dengan gestur yang dipanggil dan berhenti otomatis ketika mendeteksi manusia di depannya.

3.2.2 Analisis Tujuan

Berdasarkan hasil analisis kebutuhan, tujuan pengembangan dari sistem ini ditetapkan untuk memastikan bahwa sistem yang dikembangkan dapat memenuhi spesifikasi dan operasional yang diperlukan. Adapun tujuan dari ingin dicapai dalam pengembangan sistem ini adalah:

1. Membuat sebuah sistem untuk mendeteksi gestur SIBI menggunakan LSTM.
2. Membuat sebuah sistem untuk mendeteksi objek menggunakan YOLOv11 untuk sistem pengereman otomatis.

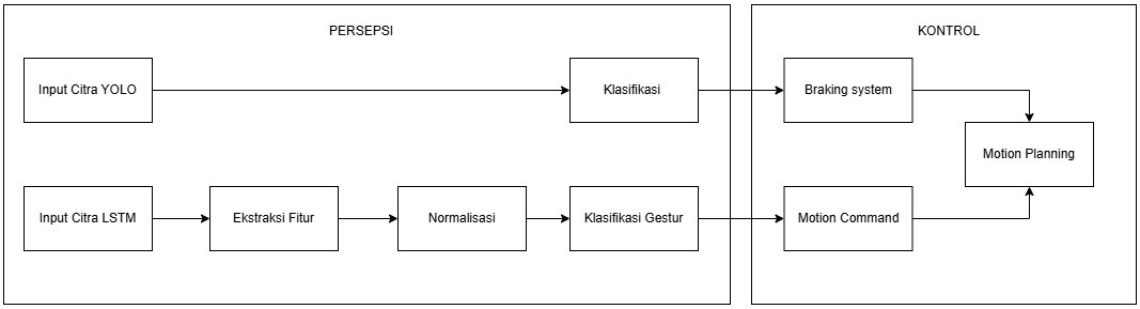
3.3 Rancangan Penelitian



Gambar 3.2: Blok diagram dari sistem

Pada tahap ini, sistem akan diterapkan sesuai dengan desain dan implementasi yang telah direncanakan sebelumnya. Diagram ini mencakup dari pengembangan *software*, alur sistem, dan implementasi infrastruktur.

3.4 Software



Gambar 3.3: Diagram dari *software*

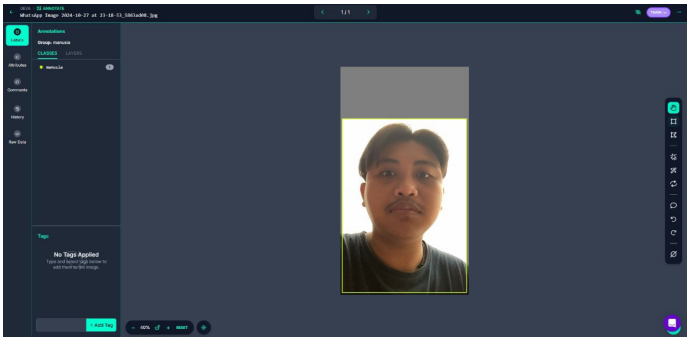
Pada tahap ini, merupakan pembuatan dari *software* yang telah ditentukan pada perencanaan *software*. Gambar 3.3 merupakan blok diagram dari alur yang akan dijelaskan sebagai berikut:

3.4.1 Input Citra YOLO

Input untuk YOLO biasanya berupa citra dalam format digital (misalnya JPG atau PNG) yang telah diubah ukurannya sesuai dengan kebutuhan model. Sebelum dimasukkan ke model, citra tersebut dinormalisasi, yaitu nilai pikselnya diubah menjadi rentang 0 hingga 1 untuk mempercepat proses komputasi. YOLO membagi citra menjadi grid, di mana setiap grid bertanggung jawab untuk memprediksi *bounding box, confidence score*, dan kelas objek di dalam grid tersebut. Proses ini menghasilkan output berupa koordinat *bounding box*, skor probabilitas, dan label kelas untuk setiap objek yang terdeteksi.

3.4.2 Labelling menggunakan Roboflow

Input citra yang didaptkan tadi lalu akan dilabeli dan diaugmentasi. Dimana Roboflow digunakan untuk melakukan proses labeli dan augmentasi ini. Roboflow memiliki tools yang mumpuni dalam melakukan labeling. Dimana terdapat beberapa proses yang akan dilakukan yaitu, import dataset, pelabelan dataset dan augmentasi dataset.



Gambar 3.4: Labeling pada Roboflow

Dalam Proses anotasi penamaan class yang digunakan haruslah sesuai dengan objek yang akan dideteksi. Dalam Konteksi tugas akhir kali ini objek yang akan dideteksi adalah manusia. Maka digunakanlah nama manusia sebagai penamaan dari kelas yang akan digunakan. Adapun dalam proses anotasi harus diperhatikan objek yang akan dianotasikan harus sesuai agar tidak ada kesalahan model dalam mendeteksi.

setelah seluruh gambar yang dipilih telah dianotasi. Maka disimpan dalam bentuk versi yang mudah diingat, apabila terjadi perubahan pada dataset maka akan disimpan dalam versi lain.

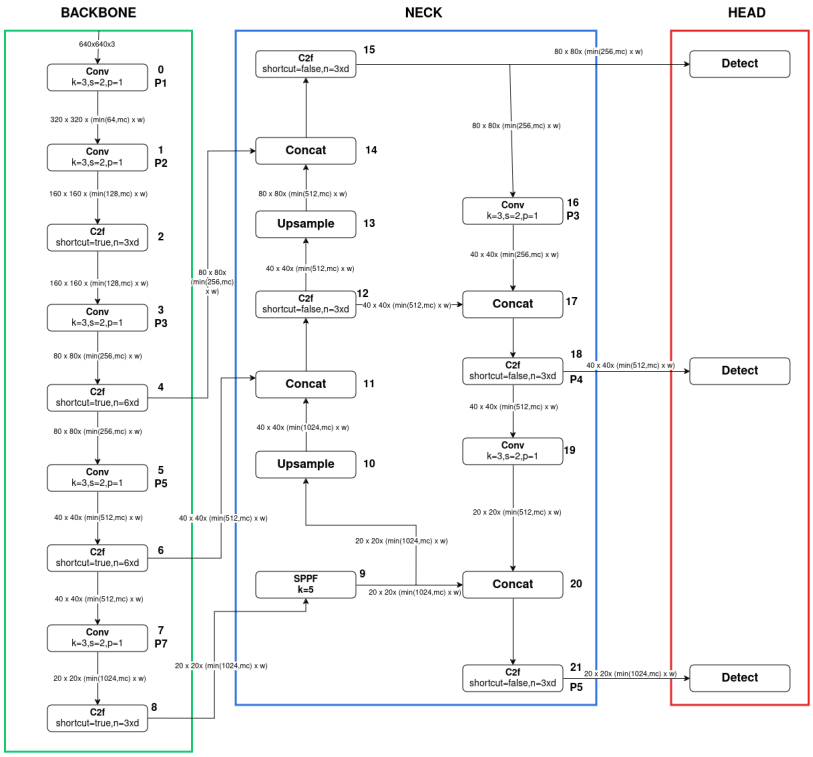
3.4.3 Klasifikasi

Klasifikasi dalam YOLO (*You Only Look Once*) adalah proses di mana model menentukan kategori atau kelas dari objek yang terdeteksi dalam sebuah citra. Setelah model YOLO mendeteksi keberadaan objek melalui *bounding box*, langkah berikutnya adalah mengklasifikasikan objek tersebut ke dalam salah satu kelas yang telah dilatih, seperti mobil, orang, hewan, atau benda lainnya, dalam penelitian kelas yang digunakan adalah manusia.

3.4.4 Klasifikasi YOLOV8

Klasifikasi dilakukan setelah melakukan proses labelling yang akan dikenali menggunakan YOLOV8 yang telah ditraining untuk mengenali manusia yang terdapat pada citra. Model akan memberikan output berupa kelas yaitu manusia, nantinya hasil dari klasifikasi akan dijadikan acuan untuk melakukan *braking*.

Model yang digunakan memiliki output kelas berupa manusia, adapun hasilnya berupa *bounding box* pada manusia yang terdeteksi. Adapun arsitektur yang digunakan pada YOLOv8.



Gambar 3.5: Blok diagram dari sistem

Arsitektur YOLOv8 menggunakan *convolutional neural network (CNN)* sebagai dasar dalam YOLO, jaringan CNN digunakan untuk ekstraksi fitur dari gambar yang telah diinputkan dan kemudian menerapkan jaringan lain untuk memprediksi bounding box dan kelas objek langsung dari gambar tersebut. Adapun selain lapisan CNN terdapat juga lapisan Concat, lapisan Upsampling, Blok SPPF, Blok C2f.

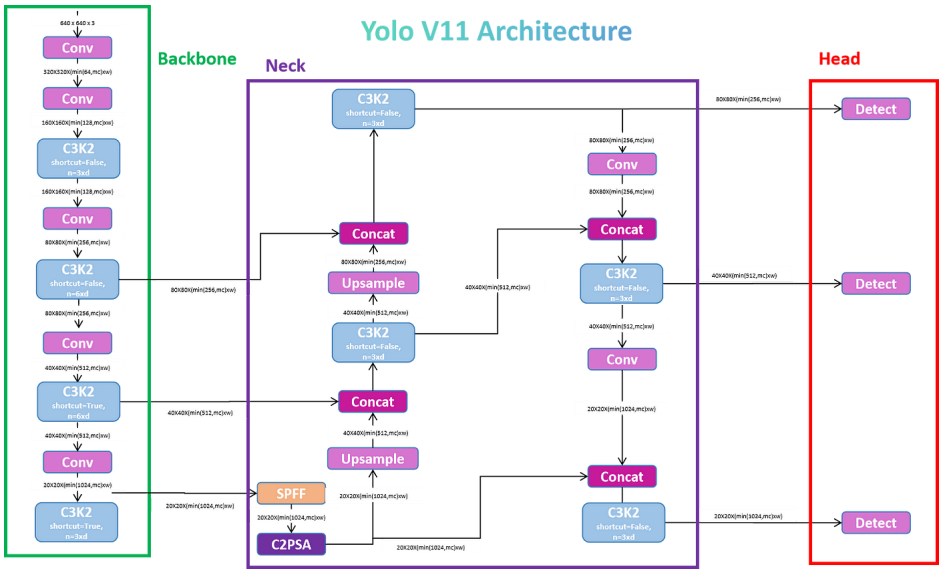
Adapun dalam arsitektur YOLOv8 dibagi menjadi bagian backbone, neck dan head. agar dapat memahami arsitektur YOLOV8 akan dibahas satu per satu fungsi dari lapisan dan blok yang digunakan pada YOLOV8. Lapisan Convolution (Conv) fungsinya yaitu melakukan operasi convolusi pada gambar input untuk mengekstraksi fitur pada gambar. operasi ini melibatkan filter pada gambar, mengkalikan dan menjumlahkan nilai nilai piksel untuk menghasilkan fitur map baru.

Blok C2f atau yang sering disebut dengan *Cross Stage Spatial Network*, dimana pada blok ini digabungka beberapa lapisan convolusi dengan skip connection, yang meningkatkan pemahaman konteks spasial dan fitur kompleks. Blok SPPF (*spatial pyramid pooling fast*) bertujuan untuk menangkap dari bebagai skala fitur map dengan menerapkan pooling pada beberapa tingkat. Sebagai contoh 1x1, 3x3 dan 5x5. Lapisan Upsampling meningkatkan resolusi fitur map dengan memperbesar dimensinya menggunakan metode seperti *nearest neighbor* atau *bilinear interpolation*, memungkinkan deteksi objek dengan resolusi yang lebih tinggi. Lapisan Concat (*concatenate*) menggabungkan dua atau lebih fitur map dari jalur berbeda dalam model. ini dilakukan dengan menggabungkan fitur map di sepanjang dimensi *channel*, sehingga mempertahankan informasi dari berbagai tahap pemrosesan sebelumnya sehingga menghasilkan konteks yang lebih kaya untuk deteksi objek.

3.4.5 Klasifikasi YOLOV11

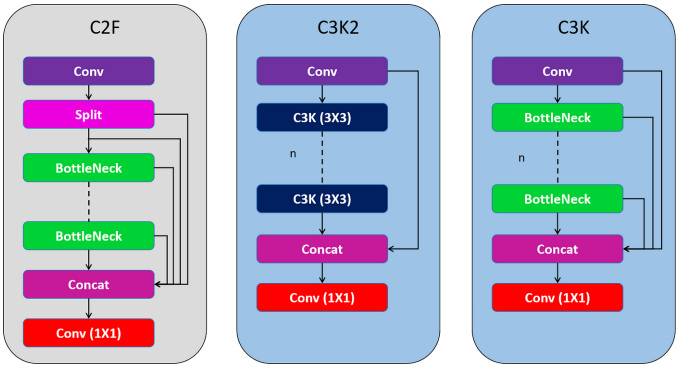
Klasifikasi pada YOLOV11 juga menggunakan data yang sudah dilabeling pada proses sebelumnya. Dimana output yang diharapkan dari training yaitu berupa kelas yaitu manusia.

YOLOV11 menggunakan arsitektur yang kurang lebih sama dengan YOLOV8 dengan perubahan pada Blok C2f yang diganti dengan Blok C3K2 yang merupakan upgrade untuk menghasilkan deteksi yang lebih baik dan lebih effisien. adapun arsitektur YOLOV11 sebagai



Gambar 3.6: Blok diagram dari sistem

Blok C3k2 berisi lapisan convolusi yang digabungkan dengan blok C3K dan lapisan concat. adapun lapisan ini digunakan untuk menghasilkan ekstraksi fitur yang lebih baik, terutama untuk objek yang berukuran kecil. Berikut merupakan isi dari Blok C2f pada YOLOV8 dan C3K2 pada YOLOV11.



Gambar 3.7: Blok diagram dari sistem

3.4.6 Preprocessing

Setelah citra diklasifikasi maka akan dilakukan Preprocessing yang menghasilkan jarak dari citra yang didapatkan. Adapun proses ini melibatkan perhitungan dari nilai nilai yang didapatkan dari bounding box dalam layar, Seperti tinggi bounding box dalam pixel maupun lebar bounding box dalam pixel.

Untuk mendapatkan jarak menggunakan bounding box dengan menerapkan konsep Fokus Panjang dalam Pixel *Focal Length Pixel*. Adapun fokus panjang dalam pixel ini adalah konversi dari fokus panjang lensa kamera yang biasanya diukur dalam meter ke dalam satuan pixel. Konsep ini biasa digunakan dalam visi komputer untuk menghubungkan informasi visual dari kamera ke ukuran fisik di dunia nyata. Adapun rumus Fokus panjang dalam pixel dapat dilihat pada rumusan 3.1.

$$f_p = \frac{D \times h_b}{H - o} \tag{3.1}$$

Dimana :

- f_p adalah *focal length pixel*.
- D adalah jarak sebenarnya dari kamera ke objek.
- h_b adalah tinggi bounding box dalam piksel.
- H_o adalah tinggi sebenarnya dalam objek.

Dengan mendapatkan nilai f_p kita dapat mengkonversi pengukuran dari ruang gambar (piksel) ke dimensi dunia nyata (meter atau centimeter). Nilai tinggi rata rata manusia juga digunakan untuk melakukan konversi. Adapun rumus konversinya seperti rumus 3.2

$$D = \frac{f_p \times H_o}{h_b} \tag{3.2}$$

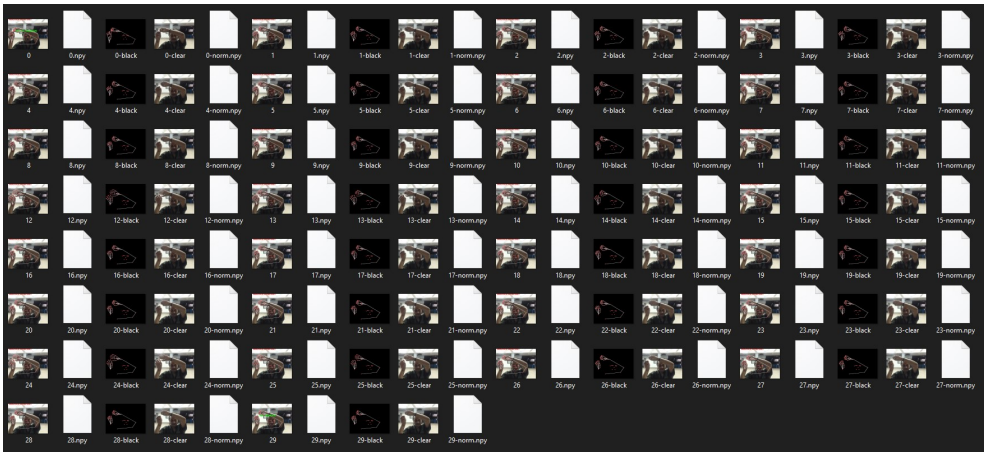
Dimana :

- H_o merupakan tinggi rata - rata manusia

Jarak yang didapatkan sekarang dapat digunakan dalam sistem *braking* yang akan diimplementasikan di kursi roda. Dimana pada penelitian ini digunakan Jarak 1.3 meter sebagai syarat kursi roda untuk berhenti. Yang mana apabila sudah dijarak 1.3 meter maka kursi roda akan berhenti.

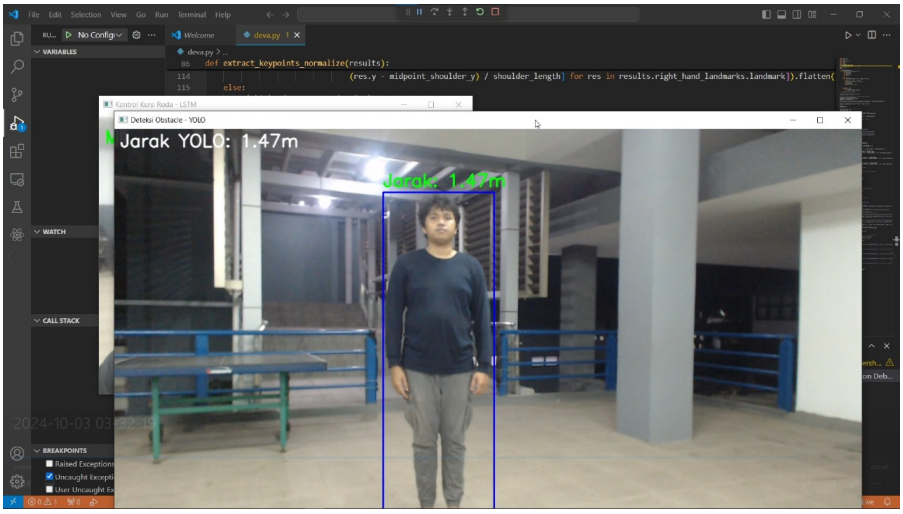
3.4.7 Input Citra LSTM

Pada tahap ini, diawali dengan pengambilan citra untuk dataset. Pengambilan dataset dilakukan langsung melalui kursi roda dengan menggunakan kamera yang menghadap ke pengguna kursi roda.



Gambar 3.8: Dataset untuk gestur kanan

Data citra kedua merupakan input YOLO untuk mendeteksi objek yang ada di depan kursi roda. Data citra ini akan digunakan untuk pendeteksian objek penghalang di depan kursi roda yang berguna untuk *braking system* dari kursi roda



Gambar 3.9: Hasil deteksi YOLO

3.4.8 Ekstraksi Fitur

Ekstraksi fitur dalam program LSTM (Long Short-Term Memory) adalah proses pengambilan informasi penting dari data mentah untuk digunakan sebagai input bagi model. Proses ini bertujuan untuk menyederhanakan data dan menyoroti pola atau karakteristik relevan yang dapat membantu model memahami hubungan temporal dalam data berurutan seperti teks, sinyal, atau deret waktu. Metode ekstraksi fitur bisa dilakukan secara manual, misalnya dengan menghitung statistik dasar seperti rata-rata atau transformasi sinyal seperti FFT, atau secara otomatis menggunakan model pembelajaran mendalam untuk menghasilkan representasi numerik, seperti word embeddings untuk teks atau representasi vektor dari gambar. Dalam konteks LSTM, fitur-fitur yang telah diekstraksi ini diorganisasi dalam urutan yang sesuai dengan waktu atau konteks data, sehingga LSTM dapat mempelajari hubungan temporal antar fitur secara efektif. Dengan ekstraksi fitur yang tepat, model LSTM dapat bekerja lebih efisien, menghasilkan kinerja yang lebih baik dengan memanfaatkan informasi relevan dan mengurangi gangguan dari data mentah.

3.4.9 Normalisasi

Normalisasi adalah proses transformasi data untuk menyelaraskan nilai-nilainya dalam rentang tertentu atau membuat distribusinya konsisten, sehingga mempermudah analisis dan meningkatkan kinerja model pembelajaran mesin. Proses ini bertujuan untuk mengurangi pengaruh skala fitur yang besar terhadap model, mempercepat konvergensi selama pelatihan, dan mencegah dominasi fitur tertentu. Metode normalisasi yang umum meliputi *Min-Max Scaling*, yang mengubah nilai ke rentang tertentu seperti 0 hingga 1, dan *Z-Score Normalization*, yang menyelaraskan data sehingga memiliki rata-rata 0 dan standar deviasi 1. Misalnya, dalam citra digital, normalisasi dilakukan dengan mengubah nilai piksel dari skala 0-255 menjadi 0 hingga 1 agar lebih mudah diproses oleh model. Dengan normalisasi, semua fitur dalam data berada pada skala yang seimbang, memungkinkan algoritma pembelajaran mesin untuk lebih fokus pada pola dan hubungan dalam data tanpa terganggu oleh perbedaan skala antar fitur.

3.4.10 Klasifikasi Gestur

Pada tahap ini, gestur yang sudah diambil selanjutnya diklasifikasikan sesuai dengan perintah yang ada pada kursi roda. Terdapat 5 kelas yang digunakan yang kelas maju, kelas mundur, kelas kanan, kelas kiri, dan kelas stop.

3.4.11 Braking System

Klasifikasi dari YOLO nantinya akan digunakan untuk melakukan pengereman otomatis jika mendeteksi kelas yang ditentukan, dalam penelitian ini kelas yang digunakan adalah manusia. Jarak untuk menentukan *braking* adalah 1.3 meter. Jika dalam jarak 1.3 meter terdeteksi manusia maka kursi roda akan berhenti secara otomatis, ini berguna untuk mengurangi resiko kecelakaan pengguna kursi roda.

3.4.12 Motion Command

Motion command adalah perintah yang digunakan untuk mengendalikan gerakan suatu objek atau perangkat, seperti kursi roda, dengan memberikan instruksi terkait arah, kecepatan, atau posisi yang harus dicapai. Dalam konteks penelitian ini, motion command berfungsi untuk mengubah input gestur menjadi perintah gerakan yang spesifik untuk kursi roda. Setiap kelas yang Anda gunakan (A, B, C, D, dan E) mewakili jenis perintah gerakan yang berbeda: Kelas

A untuk bergerak ke kiri, kelas B untuk bergerak maju, kelas C untuk berhenti, kelas D untuk bergerak mundur, dan kelas E untuk bergerak ke kanan. Setelah sistem mendeteksi gestur yang sesuai, seperti gerakan tangan atau tubuh, hasil klasifikasi tersebut diterjemahkan menjadi motion command yang dikirim ke sistem kendali kursi roda. Perintah ini kemudian dieksekusi oleh motor atau aktuator kursi roda untuk melaksanakan gerakan yang diinginkan. Dengan menggunakan motion command yang terhubung langsung ke kelas gestur, sistem dapat mengendalikan kursi roda secara responsif dan akurat berdasarkan input dari pengguna.

3.4.13 Motion Planning

Motion planning adalah proses perencanaan jalur atau gerakan bagi suatu objek atau perangkat untuk mencapai tujuan tertentu dengan mempertimbangkan berbagai faktor, seperti hambatan, batasan kecepatan, dan kondisi lingkungan. Dalam konteks robotika atau kendaraan otonom, motion planning bertujuan untuk merencanakan serangkaian langkah atau aksi yang harus diambil agar objek dapat bergerak dari titik awal menuju titik tujuan secara aman, efisien, dan sesuai dengan batasan yang ada. Proses ini melibatkan pembuatan peta lingkungan yang menggambarkan hambatan yang harus dihindari, serta perencanaan jalur yang menghubungkan posisi awal dan tujuan. Selain itu, motion planning juga mencakup optimasi jalur untuk meningkatkan efisiensi, seperti mengurangi waktu atau energi yang dibutuhkan, dan eksekusi jalur tersebut dengan mengontrol perangkat untuk mengikuti rute yang telah direncanakan. Dalam kendaraan otonom, misalnya, motion planning digunakan untuk merencanakan jalur yang menghindari kendaraan lain dan rintangan di jalan, sementara dalam robotika, motion planning memastikan robot dapat bergerak dari satu lokasi ke lokasi lain tanpa menabrak objek sekitarnya. Dengan demikian, motion planning memungkinkan perangkat bergerak secara cerdas, aman, dan responsif terhadap lingkungan sekitarnya.

3.4.14 Training Model

Pada tahap ini, dataset yang sudah diklasifikasi selanjutnya akan masuk ke tahap training ke sel LSTM.

Layer (type)	Output Shape	Param #
time_distributed (TimeDistributed)	(None, 30, 128)	13,952
lstm (LSTM)	(None, 30, 128)	131,584
dropout (Dropout)	(None, 30, 128)	0
lstm_1 (LSTM)	(None, 30, 64)	49,408
dropout_1 (Dropout)	(None, 30, 64)	0
lstm_2 (LSTM)	(None, 32)	12,416
dropout_2 (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 16)	528
dropout_3 (Dropout)	(None, 16)	0
dense_2 (Dense)	(None, 16)	272
dropout_4 (Dropout)	(None, 16)	0
dense_3 (Dense)	(None, 5)	85

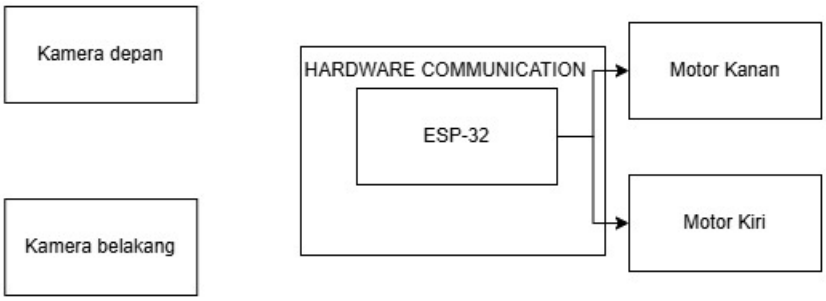
Gambar 3.10: Struktur LSTM untuk Training Gestur Tangan

Proses dimulai dengan lapisan TimeDistributed, yang menerapkan transformasi pada se-

tiap langkah waktu dari data sekuensial dengan panjang 30, menghasilkan keluaran dengan dimensi 128. Lapisan ini digunakan untuk mengolah data dengan fitur per frame, seperti hasil dari pose estimation. Selanjutnya, lapisan LSTM pertama memproses data sekuensial untuk menangkap pola temporal dengan keluaran 128 per langkah waktu, diikuti dengan lapisan Dropout untuk mencegah overfitting. Lapisan LSTM kedua mengurangi dimensi keluaran menjadi 64 unit untuk fokus pada pola yang lebih signifikan, diikuti oleh Dropout kedua. LSTM ketiga mengurangi dimensi menjadi 32 unit untuk mempertahankan informasi penting, dengan Dropout ketiga untuk menjaga generalisasi. Setelah itu, lapisan Dense pertama mengintegrasikan informasi dari lapisan sebelumnya dengan 16 unit, diikuti dengan Dropout untuk regularisasi. Lapisan Dense kedua juga memiliki 16 unit untuk memperdalam pengolahan informasi, disertai dengan Dropout keempat. Lapisan Dense ketiga, yang merupakan lapisan output, menghasilkan 5 neuron yang kemungkinan mewakili 5 kelas gestur atau tindakan berbeda. Dengan penggunaan Dropout di setiap tahap, model ini dirancang untuk mencegah overfitting dan meningkatkan generalisasi agar dapat bekerja dengan baik pada data baru.

3.5 Hardware

pada tahap ini, perancangan hardware dilakukan sesuai dengan alur yang telah direncanakan sebelumnya. Perancangan ini akan dipresentasikan dengan blok diagram alur yang telah mereprentasikan alur dari bagian hardware.



Gambar 3.11: Diagram Hardware

3.5.1 Kamera Depan

Kamera depan berfungsi sebagai hardware untuk mengambil citra untuk YOLO yang akan mendeteksi objek manusia. Setelah citra diproses, YOLO akan membagi gambar menjadi grid dan menghasilkan prediksi berupa bounding box, label kelas, dan skor probabilitas untuk setiap objek yang terdeteksi. Dengan cara ini, kursi roda atau sistem dapat mengidentifikasi objek di jalur depan dan berhenti ketika mendeteksi manusia. Kamera depan berfungsi sebagai ”mata” sistem untuk memberikan informasi visual yang diperlukan bagi YOLO untuk melakukan deteksi objek secara real-time.

3.5.2 Kamera Belakang

Kamera belakang berfungsi sebagai input citra untuk LSTM. Prosesnya dimulai dengan kamera belakang yang menangkap citra atau video dari pengguna kursi roda. MediaPipe akan menganalisis citra tersebut dan mendeteksi landmark tubuh, misalnya posisi tangan yang kemudian diterjemahkan menjadi input untuk model LSTM. LSTM akan memproses urutan data ini, memahami pola gerakan atau posisi tubuh pengguna, dan menghasilkan output yang sesuai,

seperti perintah untuk menggerakkan kursi roda maju, mundur, atau berbelok. Dengan menggunakan LSTM, model dapat mengingat dan merespons gerakan tubuh sebelumnya, memberikan kontrol yang lebih lancar dan alami atas kursi roda, berdasarkan input gestur atau posisi tubuh yang terdeteksi oleh kamera belakang.

3.5.3 Laptop

Pada tahap awal, data di peroleh dengan menggunakan kamera yang dipasang pada bracket khusus. Desain bracket ini memungkinkan kamera untuk menangkap citra tangan dengan sudut pandang yang optimal, sehingga memastikan pengambilan gambar yang akurat. Proses ini melibatkan pengambilan gambar dan video secara real-time, yang sangat penting untuk memastikan data yang dihasilkan relevan dan dapat digunakan untuk analisis lebih lanjut.

Setelah citra berhasil dikumpulkan, langkah berikutnya adalah pengolahan data tersebut dengan melakukan klasifikasi menggunakan Model LSTM yang telah dilatih sebelumnya, yang disimpan dalam format .h5. Model ini digunakan untuk mendeteksi gestur yang terdeteksi dalam citra. Untuk mendukung proses ini, beberapa library perlu diinstal terlebih dahulu, dan instalasinya dilakukan melalui serangkaian perintah yang telah ditentukan. Adapun penggunaan library yang akan digunakan ialah, Mediapipe, Scikit - learn, tensor- flow, keras, seaborn, numpy dan opencv-python. Library - library ini memastikan semua alat dan dependensi yang diperlukan untuk menjalankan model tersebut pada perangkat laptop.

Spesifikasi laptop yang digunakan dalam penelitian ini adalah laptop ROG Strix G512LU dengan spesifikasi sebagai Tabel berikut: Pada tugas akhir ini, data citra tangan yang telah

Spesifikasi	Detail
CPU	Intel Core I7-10750H
RAM	16 GB DDR4
GPU	GTX 1660 Ti

Tabel 3.1: Spesifikasi Laptop

diproses dan diklasifikasikan akan menghasilkan perintah yang dikirimkan dari laptop ke ESP32 melalui koneksi WiFi. Pengiriman data ini sangat penting karena hasil klasifikasi akan digunakan untuk mengendalikan motor pada kursi roda, memungkinkan kursi roda dikendalikan melalui gestur tangan.

Untuk memastikan pengiriman data berjalan lancar, laptop harus terlebih dahulu terhubung ke access point yang disediakan oleh ESP32. Proses ini melibatkan konfigurasi koneksi WiFi pada laptop agar dapat berkomunikasi dengan ESP32. Setelah koneksi berhasil, data klasifikasi mulai dikirim. Pengiriman data dilakukan menggunakan bahasa pemrograman Python, yang dipilih karena fleksibilitasnya dalam menangani operasi jaringan dan pengolahan data. Data dikirim dalam format string, yang sederhana namun efektif untuk transmisi data melalui jaringan.

Untuk membangun koneksi jaringan yang memungkinkan pengiriman dan penerimaan data, digunakan beberapa library Python seperti socket, time, dan datetime. Library socket menangani komunikasi jaringan, memungkinkan program untuk mengirim dan menerima data melalui koneksi TCP/IP. Library time digunakan untuk mengatur interval pengiriman data, memastikan data dikirim pada waktu yang tepat. Sementara itu, library datetime mencatat waktu pengiriman data untuk keperluan logging dan debugging.

IP Address dari access point ESP32 digunakan sebagai variabel host dalam program, sementara port 80 digunakan sebagai variabel port. Port 80 adalah port standar untuk komunikasi HTTP, yang memungkinkan penggunaan protokol HTTP untuk pengiriman data. Setelah semua konfigurasi selesai, program akan menghubungkan ke server sesuai dengan IP Address dan port yang telah ditentukan, memastikan pengiriman data dari laptop ke ESP32 berjalan dengan lancar.Program ini dirancang untuk berjalan secara terus-menerus, menerima input dari pengguna dan mengirimkan data ke ESP32 tanpa batas waktu.

3.5.4 ESP-32

Pada ESP-32 akan menerima data dari laptop untuk memberikan perintah pada kursi roda sesuai dengan kelas yang dipanggil secara *real-time*. Program dimulai dengan mengimpor beberapa library yang diperlukan untuk menjalankan berbagai fungsi, seperti WiFi, Arduino, dan JSON. Library WiFi digunakan untuk mengelola koneksi jaringan, library Arduino untuk mengontrol perangkat keras, dan library JSON untuk memproses data dalam format JSON. Setelah itu, dilakukan inisialisasi pada komunikasi serial, WiFi, dan server untuk memastikan bahwa ESP32 siap menerima dan mengirimkan data melalui jaringan.

Selanjutnya, program memeriksa apakah ada perangkat yang terhubung ke server ESP32 dengan menggunakan fungsi pemeriksa koneksi jaringan. Jika tidak ada perangkat yang terhubung, program akan terus memeriksa hingga perangkat berhasil terhubung. Setelah perangkat terhubung, program akan memeriksa apakah ada data yang dikirimkan. Jika ada data yang diterima, program akan masuk ke dalam loop untuk membaca data secara terus-menerus. Data yang diterima kemudian akan disimpan dalam variabel `receivedData`, dan string data akan dibaca hingga ditemukan karakter newline (/n), yang menandakan akhir dari data tersebut. Setelah data diterima dan disimpan, string data tersebut akan diekstraksi untuk mendapatkan informasi mengenai arah dan kecepatan. Proses ekstraksi ini sangat penting untuk mengonversi string data menjadi format yang dapat digunakan oleh program untuk mengendalikan motor pada kursi roda. Setelah ekstraksi selesai, data arah dan kecepatan yang telah diproses akan ditampilkan guna memastikan bahwa informasi yang diterima sesuai dengan yang diharapkan. Dengan data arah dan kecepatan yang sudah tersedia, ESP32 dapat mengirimkan instruksi ke motor kursi roda agar bergerak sesuai dengan perintah yang diterima. Proses ini akan terus berulang selama perangkat tetap terhubung dan data terus diterima. Berikut merupakan perintah sesuai dengan kelas yang dipanggil:

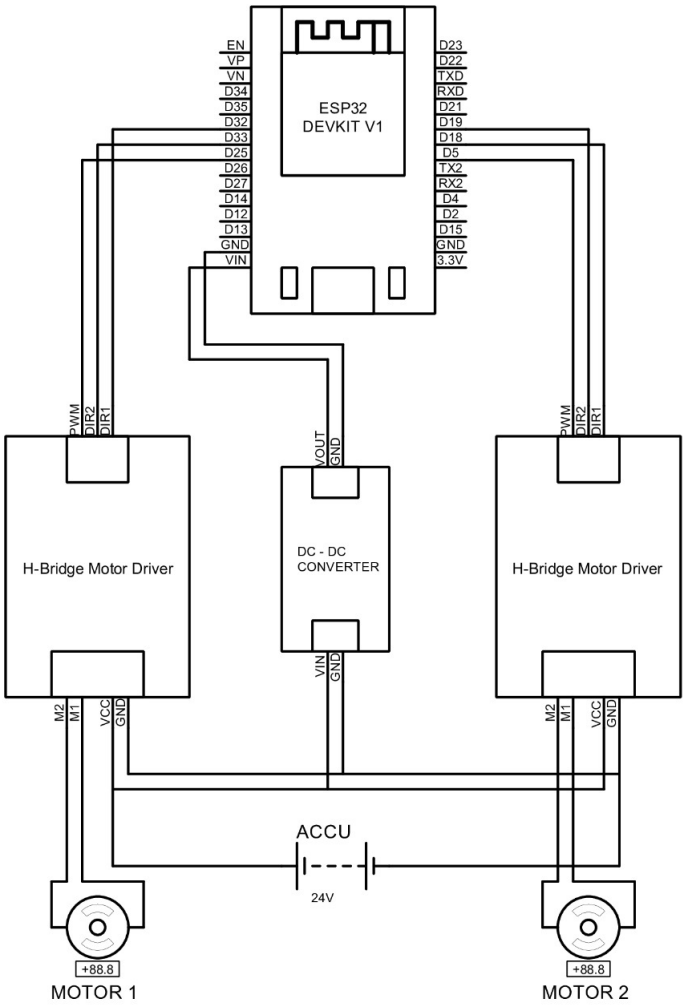
Klasifikasi gestur	**Kode Intruksi**
Kiri	A
Maju	B
Stop	C
Mundur	D
Kanan	E

Tabel 3.2: Kode Instruksi dari hasil klasifikasi

Sesuai dengan Tabel 3.2, kode intruksi tersebut diimplementasikan dengan hasil gestur yang didapatkan dari hasil training LSTM. Pengoperasian dari kursi roda diawali dengan intruksi kode B yakni maju, jika pengguna mengintruksikan kode A atau E kursi roda akan berbelok ke kiri atau kanan. Namun perlu diperhatikan, apabila ingin berganti arah agar motor tidak

mengalami kesalahan dalam menerima input yang dihasilkan model, maka perlu dilakukan pemanggilan kelas 'Stop' atau mengirim instruksi 'C'.

3.5.5 Skematik Alat



Gambar 3.12: Skematik kontrol motor kursi roda [1]

Dari penelitian yang telah dilakukan, telah dirancang sebuah sistem kontrol untuk motor kursi roda dengan skematik alat yang ditampilkan pada Gambar 3.8. ESP32 akan terhubung dengan dua buah H-Bridge Motor Driver dan sebuah DC-DC Converter. Setiap H-Bridge Motor Driver terhubung langsung ke motor roda kiri dan motor roda kanan untuk menggerakkan roda kursi roda secara efektif. Dalam skematik ini, ESP32 berfungsi sebagai otak dari sistem, mengirimkan sinyal kontrol ke motor driver berdasarkan data yang diterima melalui koneksi WiFi [1].

Program dirancang untuk mengendalikan motor DC dengan memproses perintah yang dikirim melalui jaringan WiFi. Tahapan pertama dalam program adalah mengimpor library yang dibutuhkan, seperti Arduino dan WiFi, untuk menyediakan fungsi-fungsi penting. Setelah itu, dilakukan inisialisasi PWM dan konfigurasi pin sebagai saluran untuk mengirim sinyal kontrol ke motor driver.

Setelah inisialisasi selesai, program mulai memeriksa koneksi dengan perangkat yang terhubung ke server ESP32. Jika belum ada perangkat yang terhubung, program akan terus memantau hingga koneksi berhasil. Ketika koneksi berhasil, program melanjutkan dengan memeriksa apakah ada data yang diterima dari perangkat tersebut. Jika data ditemukan, program membacanya hingga mendeteksi karakter newline (/n), yang menunjukkan akhir dari string data.

Data yang diterima selanjutnya diproses untuk mengekstrak informasi mengenai arah dan kecepatan. Tahap ini mengubah string data menjadi format yang dapat digunakan oleh program untuk mengontrol motor kursi roda. Berdasarkan informasi ini, ESP32 mengirimkan instruksi ke motor driver untuk menggerakkan roda sesuai arah dan kecepatan yang diinginkan.

Sistem kontrol kursi roda ini menggunakan dua metode pengendalian. Metode pertama adalah differential drive, di mana roda kanan bergerak mundur dan roda kiri maju saat berbelok ke kanan, serta sebaliknya untuk berbelok ke kiri. Metode kedua adalah gerakan normal, di mana roda kanan diam dan roda kiri bergerak maju saat berbelok ke kanan, atau sebaliknya saat berbelok ke kiri. Saat bergerak maju atau mundur, kedua roda berputar serentak ke arah yang sama. Dalam penelitian ini, metode kedua dipilih untuk mengendalikan pergerakan kursi roda.

Program ini memastikan ESP32 dapat berfungsi secara optimal sebagai pengontrol motor. Setiap langkah dalam flowchart dirancang untuk memastikan sistem bekerja dengan efisien dan responsif terhadap perintah yang diterima melalui jaringan WiFi. Dengan pengoperasian secara real-time, sistem mampu merespons perintah dengan cepat dan memberikan kontrol yang presisi untuk pergerakan kursi roda [1].

BAB IV
PENGUJIAN DAN ANALISIS

Pada bab ini, akan dijelaskan mengenai hasil dari pengujian dan analisis dari penelitian yang telah diuraikan pada metodologi. Selain itu, akan dipaparkan mengenai skenario dari pengujian yang dilakukan untuk evaluasi performa dari sistem. Pengujian ini dilakukan untuk memastikan bahwa sistem yang telah dirancang mampu berfungsi dengan baik dan situasi yang mungkin dihadapi oleh pengguna.

4.1 Skenario Pengujian

Pengujian dilakukan untuk menguji performa dari model dalam melakukan deteksi terhadap gestur untuk sistem kendali dan sistem pengereman otomatis. Skenario pengujian dirancang untuk mengukur berbagai aspek dari sistem seperti akurasi deteksi terhadap kelas yang dipanggil, *respond time*, kecepatan pemrosesan, dan respon sistem terhadap objek di depannya yang berpengaruh kepada pengereman otomatis. Skenario pengujian yang dilakukan adalah sebagai berikut;

1. Hasil Pengujian model
2. Pengujian berdasarkan FPS
3. Pengujian berdasarkan hasil *respond time*
4. Pengujian terhadap akurasi dari kelas yang dipanggil
5. Pengujian sistem pengereman otomatis pada jarak 150 cm
6. Pengujian sistem pengereman otomatis pada jarak 130 cm

4.2 Pengujian Performa Model YOLOv11

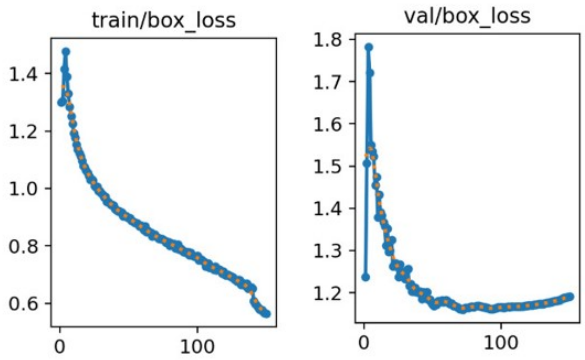
Pengujian model akan dilakukan untuk menguji model dengan menggunakan *epoch*150 dan *epoch* 100 untuk melihat performa yang lebih baik. Data yang digunakan sebagai set data berjumlah 5372 citra manusia dengan perbandingan 4359 set data train dan 1023 set data validasi. Setelah proses pemuatan set dilakukan di lanjutkan dengan proses pelatihan model. Proses pelatihan dilakukan dengan beberapa parameter yang nantinya akan diuji, yang dimana parameter ini akan dibandingkan performanya melalui nilai confusion matrix maupun nilai akurasi deteksi seperti skor mAP, *precision*, *box loss* dan F1-*confidence*. Gambar 4.1 merupakan *input layer* yang digunakan.

Input Layer : Citra Manusia	Input	[(16, 800, 800, 3)]
	Output	[(16, 800, 800, 3)]

Gambar 4.1: Input Layer Pelatihan

Pelatihan pertama menggunakan 150 epoch. *Input layer* memiliki *batch size* sebesar 16 dengan tindakan praproses merubah ukuran gambar pada dataset menjadi 800x800 piksel. Adapun tujuan dari pelatihan ini untuk mengetahui seberapa baik peningkatan model pre- trained yang digunakan dalam deteksi manusia berdasarkan jumlah *Epoch* yang ditentukan. Berikut

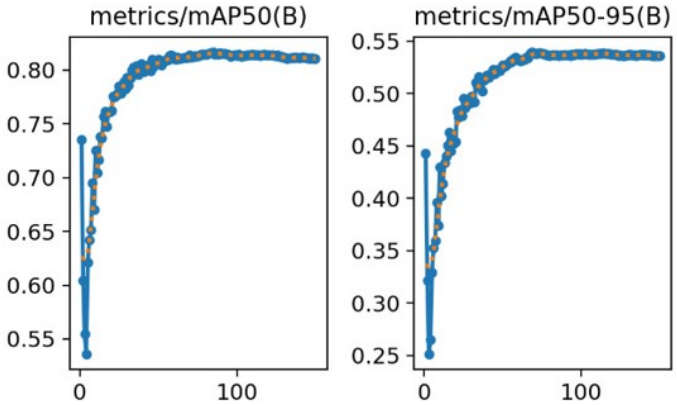
nilai box loss yang didapatkan dari hasil training pada Gambar 4.2



Gambar 4.2: Grafik *box loss*

Didapatkan nilai dari *box loss* pada hasil pelatihan YOLOv11 adalah sebesar 0.562 setelah 150 *epoch*. *Box loss* mengukur seberapa baik model memprediksi bounding box di sekitar objek. Tujuan dari loss ini adalah untuk mengoptimalkan model agar bounding box yang diprediksi sesuai dengan posisi dan ukuran objek sebenarnya dalam gambar. Adapun penurunan yang terlihat pada gambar pelatihan box loss menandakan bahwa hasil pelatihan yang dilakukan telah berhasil membuat model menentukan koordinat *bounding box* pada objek manusia secara akurat.

Nilai dari *box loss* pada proses validasi menggunakan YOLOv11 ini adalah 1.198, *box loss* validasi ini mengindikasi kemampuan mengenali objek pada data uji. Secara teori , penurunan objek loss pada tahap validasi menandakan bahwa objek mampu melakukan deteksi secara umum. Namun diakhir terlihat grafik sedikit naik. Hal ini menandakan ketika model mulai menghafal pola data training dengan baik, tapi gagal menggeneralisasi pada data validasi. Akibatnya, *loss training* tetap atau terus menurun, loss validasi justru naik sedikit karena model tidak mampu memprediksi data validasi dengan baik. Selanjutnya grafik dari mAP dapat dilihat pada Gambar 4.3



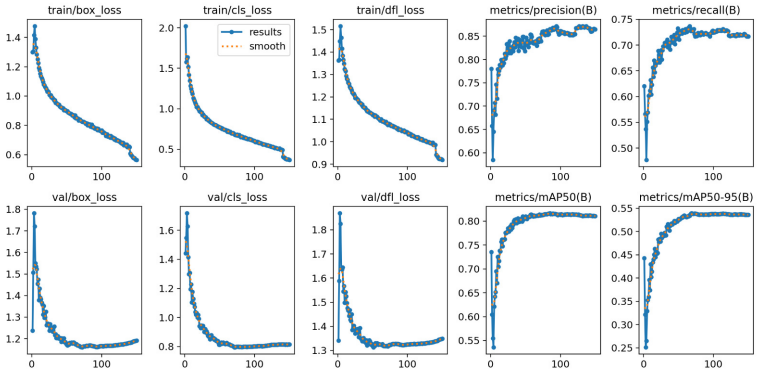
Gambar 4.3: Grafik mAP epoch 150

Skor mAP50 pada *threshold* 0.5 yang diperoleh dari proses ini adalah sebesar 0.806, ini mengindikasikan bahwa model memiliki kemampuan yang sangat baik dalam mendeteksi objek dengan kecepatan tinggi, syarat minimal dari kriteria IoU adalah 0.5 terpenuhi. Hal ini

menunjukan bahwa dalam percobaan kasus, *bounding box* yang diprediksi oleh model memiliki kesesuaian yang signifikan dengan *bounding box ground truth*.

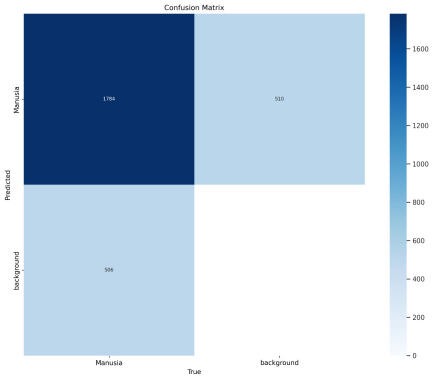
Pada grafik mAP50-95 dengan prediksi yang lebih presisi. Nilai mAP50-95 adalah 0.27 diawal dan mulai terlihat kenaikan yang signifikan mencapai 0.511 di epoch ke 92. Hal ini menunjukan bahwa walaupun model sudah mampu mendeteksi objek dengan cukup baik pada *threshold* rendah, namun masih terdapat tantangan dalam mempertahankan akurasi deteksi saat persyaratan presisi yang tumpang tindih semakin ketat.

Berikut merupakan grafik dari hasil yang didapatkan secara keseluruhan dari pelatihan model YOLOv11. Pada Gambar 4.4 merupakan grafik hasil yang didapatkan pada model secara keseluruhan. Didapatkan nilai- nilai pada masing masing grafik, pada *train box loss* didapatkan nilai 0.562, pada *train cls loss* didapatkan nilai 0.365, pada *train dfl loss* didapatkan nilai 0.916 , pada grafik *precision* didapatkan nilai 0.862, pada grafik *recall* 0.730, pada *val box loss* didapatkan nilai 1.182, pada *val cls loss* didapatkan nilai 0.804, dan pada *val dfl loss* didapatkan nilai 1.322, pada grafik mAP50 0.813, pada grafik mAP50-95 0.528



Gambar 4.4: Grafik keseluruhan pelatihan epoch 150

Berikut adalah visualisasi melalui *confusion matrix* terkait dengan pelatihan model YOLOv11 dengan 150 epoch. Pada Gambar 4.5 indikator yang digunakan pada *confusion matrix* pada setiap kotaknya merepresentasikan nilai *true positive, false positive, false negative, true negative*.

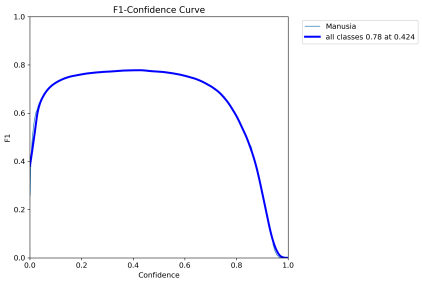


Gambar 4.5: *Confusion matrix* pelatihan epoch 150

Dari gambar 4.5 hasil pelatihan menunjukan dua kelas yakni manusia dan *background*. Dari hasil tersebut, model berhasil memprediksi hasil benar sebanyak 1784 sampel manu-

sia. Namun, terdapat 510 sampel *background* yang salah diklasifikasikan oleh model sebagai manusia. Disisi lain, model melewatkan 506 sampel manusia yang diklasifikasikan sebagai *background*. Selain itu, tidak ada prediksi yang benar-benar untuk kelas background, yang mengindikasikan bahwa model cenderung memprioritaskan prediksi kelas manusia dan kurang mampu membedakan objek background dengan baik. Hal ini menunjukkan adanya kecenderungan false positive dan false negative yang cukup signifikan

Selanjutnya adalah kurva F1-*Confidence* dari hasil pelatihan model YOLOv11 dengan *epoch* 150.

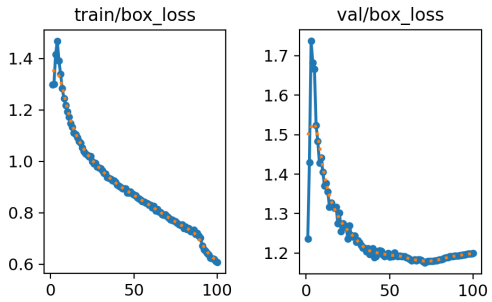


Gambar 4.6: Kurva F1-*Confidence* pelatihan epoch 150

Kurva F1-*Confidence* menunjukan hubungan antara nilai *confidence* dengan nilai F1-score. Pada kurva ini, terlihat nilai dari kurva tertinggi adalah 0.78 pada nilai *confidence* 0.424. Nilai F1-*score* yang tinggi ini menandakan model memiliki keseimbangan yang baik antara *precision* dan *recal* pada model.

Dengan kurva F1-*Confidence* pada Gambar 4.6 dapat disimpulkan bahwa model mampu mendeteksi manusia dengan baik pada tingkat kepercayaan (*confidence*) tertentu. Kurva yang menurun tajam setelah nilai 0.78 menunjukan bahwa nilai *confidence* yang sangat tinggi, model cenderung mengabaikan banyak deteksi positif yang sebenarnya (*true positive*). sehingga menurunkan nilai F1-score secara keseluruhan.

Selanjutnya pengujian model YOLOv11 dengan*epoch* 100 untuk melihat performanya. Dataset yang digunakan masih sama seperti yang digunakan pada *epoch* 150 tanpa ada pengubahan fitur. Berikut adalah nilai *box loss* dari pelatihan model menggunakan *epoch* 100.

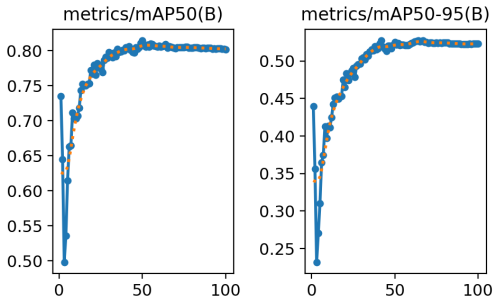


Gambar 4.7: *Train* dan *box loss* pelatihan epoch 100

Didapatkan nilai box loss pada proses training menggunakan *epoch* 100 ini adalah sebesar 0.607. Box loss mengukur seberapa baik model memprediksi *bounding boxes* seputar objek.

Tujuan dari *loss* ini adalah untuk mengoptimalkan model agar *bounding box* yang diprediksi sesuai dengan posisi dan ukuran objek sebenarnya dalam gambar. Adapun penurunan yang nilai *box loss* yang didapatkan selama *training* menandakan bahwa hasil *training* yang dilakukan telah berhasil membuat model menentukan koordinat bounding box pada object manusia dengan akurat.

Untuk nilai dari validasi adalah 1.175, nilai *box loss* pada validasi ini mengindikasikan kemampuan pengenalan objek dari model pada data uji. Selanjutnya Gambar 4.8 merupakan mAP50 dan mAP50-95 dari pelatihan menggunakan *epoch* 100.

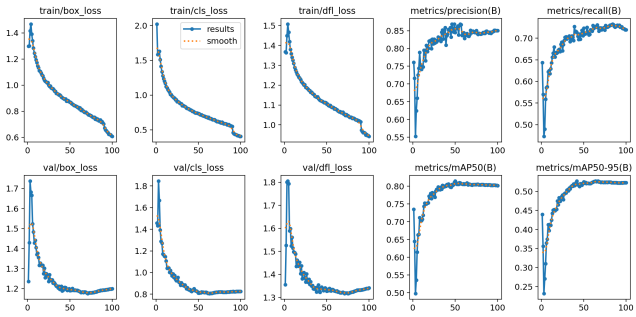


Gambar 4.8: Grafik mAP *epoch* 100

Skor mAP50 pada *threshold* 0.5 adalah 0.814, menunjukkan model mampu mendeteksi objek dengan kecepatan tinggi dan *bounding box* prediksi sangat sesuai dengan *ground truth*.

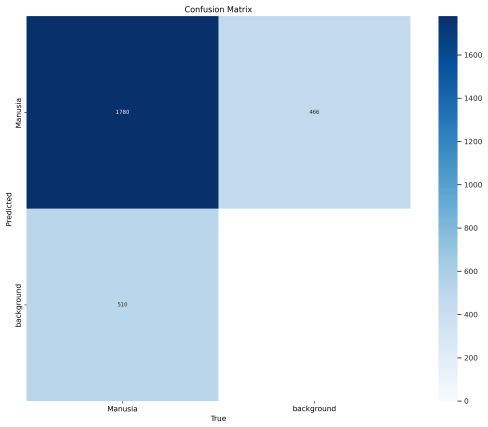
Nilai mAP50-95 adalah 0.527, menandakan tantangan menjaga akurasi deteksi saat presisi tumpang tindih semakin ketat.

Berikut adalah keseluruhan hasil *training* dari YOLOv11 dengan *epoch* 100. Pada Gambar 4.9 nilai dari *train box loss* didapatkan nilai 0.607, pada *train cls loss* didapatkan nilai 0.405, pada *train dfl loss* didapatkan nilai 0.942 , pada grafik *precision* didapatkan nilai 0.868, pada grafik *recall* 0.732, pada *val box loss* didapatkan nilai 1.175, pada *val cls loss* didapatkan nilai 0.806, dan pada *val dfl loss* didapatkan nilai 1.317, pada grafik mAP50 0.814, pada grafik mAP50-95 0.527.



Gambar 4.9: Hasil pelatihan *epoch* 100

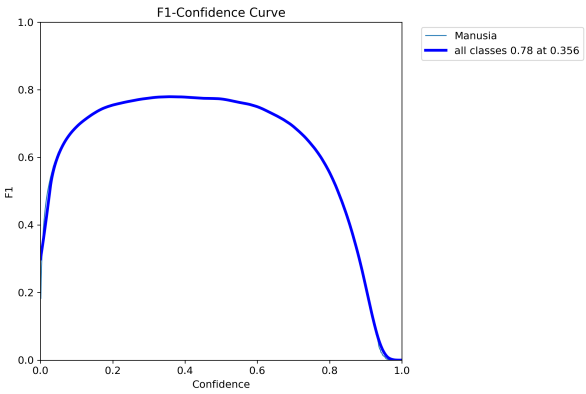
Berikut adalah visualisasi melalui *confusion matrix* terkait dengan pelatihan model YOLOv11 dengan 100 epoch.



Gambar 4.10: *Confusion matrix epoch* 100

Dari gambar 4.10 hasil pelatihan menunjukan dua kelas yakni manusia dan *background*. Dari hasil tersebut, model berhasil memprediksi hasil benar sebanyak 1780 sampel manusia. Namun, terdapat 510 sampel *background* yang salah diklasifikasikan oleh model sebagai manusia. Disisi lain, model melewatkan 466 sampel manusia yang diklasifikasikan sebagai *background*. Tidak ada prediksi yang benar-benar untuk kelas *background*, yang mengindikasikan bahwa model cenderung memprioritaskan prediksi kelas manusia dan kurang mampu membedakan objek *background* dengan baik. Hal ini menunjukkan adanya kecenderungan *false positive* dan *false negative* yang cukup signifikan.

Sebaliknya adalah kurva F1-*Confidence* dari hasil pelatihan model YOLOv11 dengan *epoch* 100.



Gambar 4.11: Kurva F1-*Confidence* pelatihan epoch 100

Kurva F1-*Confidence* menunjukan hubungan antara nilai *confidence* dengan nilai F1-score. Pada Gambar 4.11, terlihat nilai dari kurva tertinggi adalah 0.78 pada nilai *confidence* 0.365. Nilai F1-*score* yang tinggi ini menandakan model memiliki keseimbangan yang baik antara *precision* dan *recal* pada model.

Berdasarkan Gambar 4.4 dan Gambar 4.9 didapatkan tabel perbandingan sebagai berikut.

Tabel 4.1: Perbandingan Performa Model YOLO pada Epoch ke-150 dan ke-100

Metrik	**Epoch 150**	**Epoch 100**
Train Box Loss	0.562	0.607
Train Cls Loss	0.365	0.405
Train DFL Loss	0.916	0.942
Precision	0.862	0.868
Recall	0.730	0.732
Val Box Loss	1.182	1.175
Val Cls Loss	0.804	0.806
Val DFL Loss	1.322	1.317
mAP50	0.813	0.814
mAP50-95	0.528	0.527

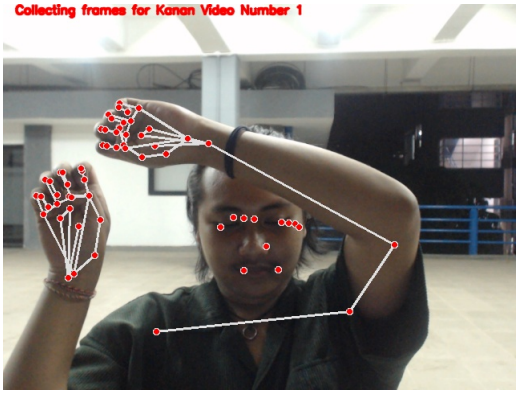
Berdasarkan perbandingan performa kedua model, model pada epoch ke-100 menunjukkan hasil yang sedikit lebih unggul dalam metrik evaluasi utama seperti (0.868), *recall* (0.732), dan mAP50 (0.814), meskipun selisihnya tipis. Sementara itu, model pada epoch ke-150 memiliki keunggulan pada nilai train loss dan mAP50-95 (0.528) yang mencerminkan ketelitian deteksi pada *threshold* yang lebih tinggi. Namun, karena perbedaan antara keduanya sangat kecil dan model pada epoch ke-100 menunjukkan performa validasi yang lebih konsisten, maka model tersebut dapat dianggap lebih optimal dan efisien untuk digunakan.

4.3 Pengujian Performa Model LSTM Menggunakan Confusion Matrix

Dataset untuk LSTM diambil dalam lima kelas dengan kelas yakni kelas kanan, stop, maju, mundur, dan kiri. Masing-masing kelas berjumlah 30 citra yang telah di normalisasi tiap citranya, sebagai contoh pada gambar 4.12 merupakan gambar asli atau data mentah.

Gambar 4.12: Dataset kelas kanan *clear*

Kemudian pada gambar 4.13 menunjukan citra wajah dan tangan dengan titik-titik *landmark* yang telah di deteksi menggunakan media pipe.



Gambar 4.13: Dataset kelas kanan *landmark*

Dari gambar 4.13 diubah kedalam format .npy yang berisi data mentah dari titik-titik *landmark* yang terdeteksi pada bagian wajah dan dan tangan pada *frame* video. File .npy ini berisi koordinat 2D dari landmark yang citra yang nantinya akan di normalisasi agar koordinat *landmark* menjadi relatif terhadap gestur tangan sehingga data menjadi skala yang kosisten dan tidak bergantung pada ukuran asli dari citra. Normalisasi ini penting untuk memastikan model LSTM dapat belajar pola gestur tanpa terpengaruh perubahan ukuran citra atau posisi objek.

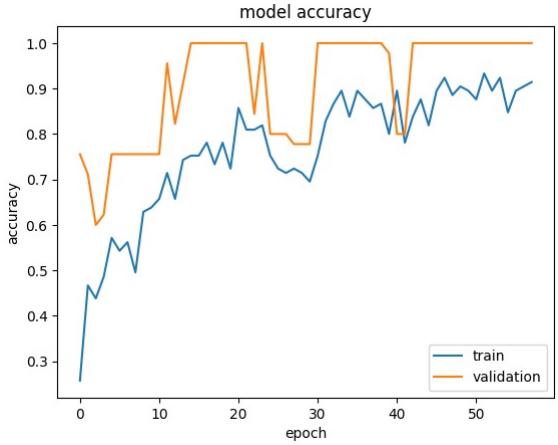
Tabel 4.2: Tabel Sequencial

Layer (type)	**Output Shape**	**Param #**
time_distributed (TimeDistributed)	(None, 30, 128)	13,952
lstm (LSTM)	(None, 30, 128)	131,584
dropout (Dropout)	(None, 30, 128)	0
lstm_1 (LSTM)	(None, 30, 64)	49,408
dropout_1 (Dropout)	(None, 30, 64)	0
lstm_2 (LSTM)	(None, 32)	12,416
dropout_2 (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 16)	528
dropout_3 (Dropout)	(None, 16)	0
dense_2 (Dense)	(None, 16)	272
dropout_4 (Dropout)	(None, 16)	0
dense_3 (Dense)	(None, 5)	85

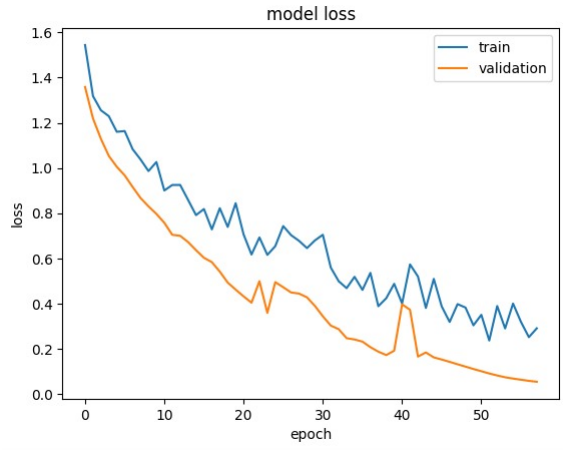
Tabel 4.2 menunjukkan ringkasan arsitektur model jaringan saraf yang terdiri dari beberapa layer berturut-turut. Dimulai dengan layer TimeDistributed yang menghasilkan output berukuran (None, 30, 128) dengan 13.952 parameter, yang berfungsi untuk menerapkan layer Dense pada setiap langkah waktu secara independen. Berikutnya terdapat tiga layer LSTM bertingkat, masing-masing dengan ukuran output (None, 30, 128), (None, 30, 64), dan (None, 32), serta jumlah parameter yang berkurang secara bertahap yaitu 131.584, 49.408, dan 12.416. Layer-layer ini menangani pemrosesan sekuensial data dengan mempertahankan urutan waktu dan mengekstrak fitur temporal yang kompleks. Setiap layer LSTM diikuti oleh layer Dropout yang tidak memiliki parameter, berfungsi sebagai regularisasi untuk mengurangi overfitting dengan mematikan sebagian neuron secara acak selama pelatihan. Setelah LSTM, model memiliki tiga layer Dense berturut-turut dengan ukuran output (None, 16), (None, 16), dan (None, 5)

dengan jumlah parameter masing-masing 528, 272, dan 85. Layer Dense ini berfungsi untuk memetakan fitur yang telah diekstrak menjadi output klasifikasi akhir dengan 5 kelas, yang ditentukan oleh neuron pada layer terakhir.

Berdasarkan hasil akurasi *training* yang telah dilakukan didapatkan bahwa model menghasilkan akurasi validasi bernilai 1 dan akurasi *traininf* bernilai 0.91. Data ini menunjukan peningkatan akurasi yang cukup baik pada data *training* namun pada akurasi validasi mencapai 1.0 beberapa kali yang biasa jarang terjadi pada data validasi yang benar-benar baru dan representatif. Hal ini menandakan data validasi terlalu kecil atau kurang sehingga model mudah menghafal. Untuk nilai *loss training* bernilai cukup tinggi di 2.56 dan *loss* validasi di di nilai 0.05. Data ini menunjukan nilai *loss training* secara umum menurun dari awal hingga akhir pelatihan, yang berarti model semakin baik dalam meminimalkan kesalahan terhadap data *training* seiring dengan berjalannya *epoch*. Untuk *loss* validasi juga menunjukan tren penurunan yang konsisten. Hal ini mengindikasikan bahwa model dapat belajar dari data *training* dengan cukup baik. Berikut merupakan grafik dari model akusari dan model *loss* dari pelatihan model LSTM.



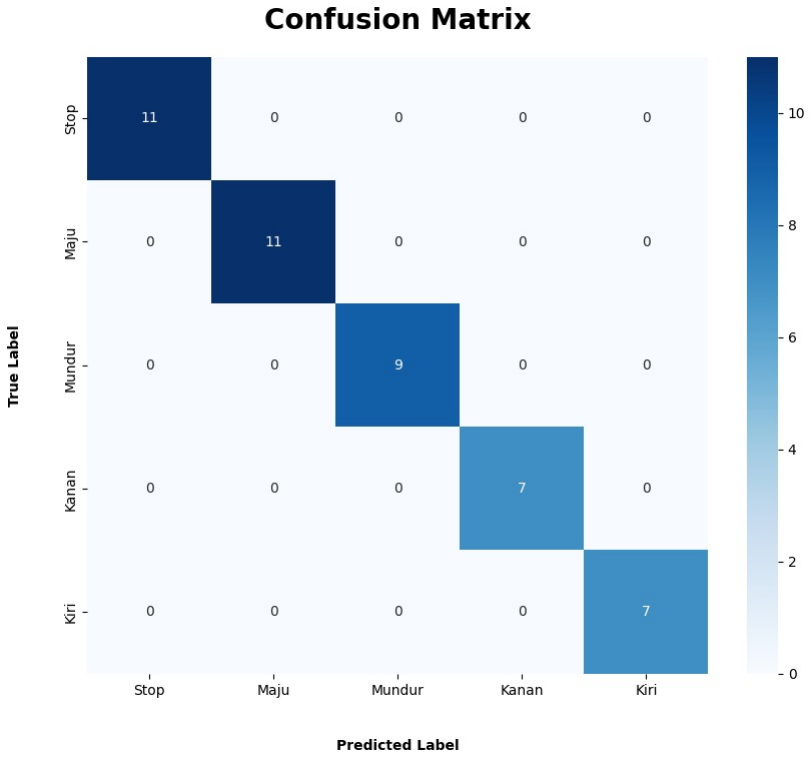
Gambar 4.14: Hasil *accuracy* model LSTM



Gambar 4.15: Hasil model *loss* LSTM

Kemudian berdasarkan model yang telah dilatih, dilakukan pengujian dengan dataset *testing* yang menghasilkan *confusion matrix*. Gambar 4.16 menunjukkan confusion matrix dari

sebuah model klasifikasi dengan lima kelas yaitu Stop, Maju, Mundur, Kanan, dan Kiri. Dari matriks ini dapat dilihat bahwa model mampu mengklasifikasikan setiap kelas dengan sangat baik tanpa terjadi kesalahan prediksi antar kelas. Misalnya, untuk kelas Stop, seluruh 11 sampel diklasifikasikan dengan benar sebagai Stop, dan hal yang sama berlaku untuk kelas Maju dengan 11 sampel yang benar, kelas Mundur dengan 9 sampel, serta kelas Kanan dan Kiri yang masing-masing diklasifikasikan benar sebanyak 7 sampel. Tidak terdapat nilai selain diagonal utama yang berarti tidak ada sampel yang salah diklasifikasikan ke kelas lain. Hal ini menunjukkan performa model yang sangat akurat dalam mengenali kelima kelas tersebut.



Gambar 4.16: Hasil *confusion matrix* LSTM

Tabel 4.3: Classification Report

Class	**Precision**	**Recall**	**F1-score**	**Support**
Stop	1.00	1.00	1.00	11
Maju	1.00	1.00	1.00	11
Mundur	1.00	1.00	1.00	9
Kanan	1.00	1.00	1.00	7
Kiri	1.00	1.00	1.00	7
Accuracy		1.00		
Macro avg	1.00	1.00	1.00	45
Weighted avg	1.00	1.00	1.00	45

Tabel 4.2 menunjukkan hasil evaluasi model klasifikasi untuk lima kelas, yaitu Stop, Maju, Mundur, Kanan, dan Kiri. Dari tabel tersebut terlihat bahwa model mencapai nilai precision, recall, dan f1-score sebesar 1.00 pada semua kelas, yang berarti model mampu mengklasifikasikan setiap kelas dengan sempurna tanpa kesalahan. Support menunjukkan jumlah data

uji pada masing-masing kelas, dengan total 45 sampel yang tersebar di kelima kelas tersebut. Nilai accuracy sebesar 1.00 mengindikasikan bahwa model memiliki akurasi 100% secara keseluruhan pada data uji ini. Selain itu, nilai macro average dan weighted average yang juga 1.00 memperkuat bahwa performa model sangat baik dan konsisten di seluruh kelas.

4.4 Pengujian Berdasarkan FPS

Pengujian FPS akan dilakukan pada dua device yang pertama adalah pengujian pada device Laptop, dan pengujian pada device NUC. Dalam pengujian ini akan diambil nilai FPS sebanyak 30 kali yang sesuai dengan output sistem dan sudah diolah pada bagian sebelumnya.

Pada pengujian ini akan diambil nilai FPS pada laptop dan FPS pada NUC. Spesifikasi laptop yang digunakan untuk pengujian ini pada tabel :

Tabel 4.4: Spesifikasi Laptop

Komponen	Spesifikasi
CPU	Intel(R) Core(TM) i7-10750H
RAM	16 GB DDR4
GPU	GTX 1660 Ti

Spesifikasi NUC yang digunakan untuk pengujian ini pada tabel :

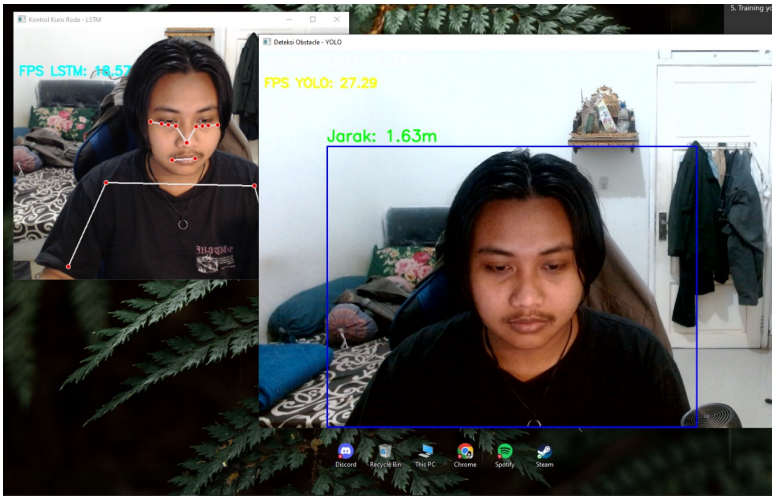
Tabel 4.5: Spesifikasi NUC

Komponen	Spesifikasi
CPU	Intel® Core™ i5-1240P
RAM	8GB
OS	Ubuntu

Baik system kontrol maupun system pengereman otomatis akan diambil nilai fpsnya, selain itu setiap jenis arsitektur YOLO juga akan dibandingkan untuk menilai efisiensi sistem berdasarkan jenis arsitektur YOLO yang digunakan. Pengujian ini dilakukan dengan cara menjalankan sistem pada laptop dan NUC, kemudian merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel.

4.4.1 Pengujian Model YOLOv11 Berdasarkan FPS pada Laptop

Pengujian dilakukan pada laptop dengan spesifikasi yang telah disebutkan sebelumnya. Pengujian ini dilakukan dengan cara menjalankan sistem pada laptop dan merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel. Gambar 4.17 merupakan gambar pengambilan data fps pada laptop.



Gambar 4.17: Pengujian FPS pada Laptop

Pada tabel 4.6 merupakan hasil dari pengujian model YOLOv11 berdasarkan FPS pada Laptop. Dari hasil tersebut didapatkan rata rata FPS sebesar 24,969 , dengan nilai tertinggi FPS sebesar 35,81 dan nilai terendah sebesar 12,8.

Tabel 4.6: Nilai Pengujian Model YOLOv11 Berdasarkan FPS pada Laptop

No	FPS	No	FPS	No	FPS
1	21,8	11	21,82	21	23,32
2	20,89	12	20,89	22	23,89
3	25,7	13	17,9	23	21,18
4	12,8	14	23,57	24	31,33
5	23,32	15	21,77	25	30,38
6	34,57	16	17,34	26	29,49
7	21,33	17	26,39	27	33,42
8	35,81	18	19,28	28	32,34
9	21,8	19	32,34	29	28,31
10	24,45	20	29,54	30	22,12

Pada pengujian menggunakan model YOLOv11 pada laptop ini, hasil yang didapatkan sangat baik dimana FPS menyentuh angka yang cukup tinggi. Hal ini menunjukan bahwa sistem dapat berjalan dengan baik dan efisien. Adapun ketidakstabilan pada FPS disebabkan oleh beberapa faktor seperti banyaknya objek yang terdeteksi, dan juga pengaruh dari suhu laptop selama pengujian.

4.4.2 Pengujian Model YOLOv8 Berdasarkan FPS pada Laptop

Pengujian dilakukan pada laptop dengan spesifikasi yang telah disebutkan sebelumnya. Pengujian ini dilakukan dengan cara menjalankan sistem pada laptop dan merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel.

Pada tabel 4.7 merupakan hasil dari pengujian model YOLOv8 berdasarkan FPS pada Laptop. Dari hasil tersebut didapatkan rata rata FPS sebesar 6,782, dengan nilai tertinggi FPS sebesar 8,89 dan nilai terendah sebesar 4,12.

Tabel 4.7: Nilai Pengujian Model YOLOv8 Berdasarkan FPS pada Laptop

No	FPS	No	FPS	No	FPS
1	6,68	11	6,24	21	5,12
2	7,43	12	4,77	22	8,43
3	6,68	13	7,43	23	6,73
4	8,89	14	8,36	24	8,36
5	6,08	15	6,68	25	6,43
6	7,43	16	7,23	26	5,33
7	8,36	17	4,12	27	6,59
8	5,12	18	6,88	28	7,1
9	6,14	19	7,48	29	5,43
10	7,87	20	7,42	30	6,65

Pada pengujian menggunakan model YOLOv8 pada laptop ini, hasil yang didapatkan kurang baik dimana FPS menyentuh angka yang cukup rendah. Hal ini menunjukan bahwa sistem dapat berjalan dengan baik namun tidak efisien.

4.4.3 Pengujian Model LSTM Berdasarkan FPS pada Laptop

Pengujian dilakukan pada laptop dengan spesifikasi yang telah disebutkan sebelumnya. Pengujian ini dilakukan dengan cara menjalankan sistem pada laptop dan merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel.

Pada tabel 4.8 merupakan hasil dari pengujian model LSTM berdasarkan FPS pada Laptop. Dari hasil tersebut didapatkan rata rata FPS sebesar 23,813 , dengan nilai tertinggi FPS sebesar 26,32 dan nilai terendah sebesar 22,19.

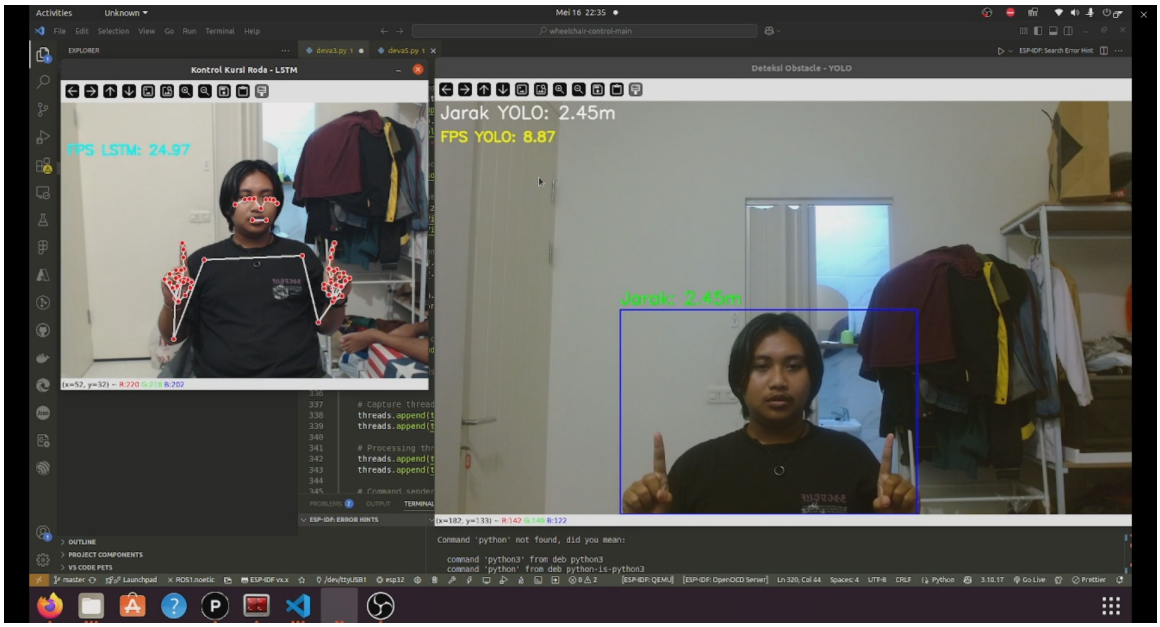
Tabel 4.8: Nilai Pengujian Model LSTM Berdasarkan FPS pada Laptop

No	FPS	No	FPS	No	FPS
1	25,91	11	23,09	21	22,38
2	22,43	12	24,67	22	22,79
3	23,25	13	22,76	23	24,57
4	23,98	14	25,74	24	22,73
5	26,32	15	23,46	25	22,62
6	22,19	16	25,3	26	25,03
7	23,32	17	25,96	27	23,17
8	22,57	18	22,51	28	25,16
9	22,76	19	24,97	29	25,06
10	24,78	20	22,43	30	22,48

Pada pengujian menggunakan model LSTM pada laptop ini, hasil yang didapatkan sangat baik dimana FPS menyentuh angka yang cukup tinggi. Hal ini menunjukan bahwa sistem dapat berjalan dengan baik dan efisien, hal ini akan menjadi sangat penting bahwa output klasifikasi gestur dari model LSTM akan digunakan dalam sistem kendali kursi roda. FPS yang didapatkan cendrung stabil tanpa drop yang signifikan, hal ini menunjukan bahwa sistem dapat berjalan dengan baik dan efisien.

4.4.4 Pengujian Model YOLOv11 Berdasarkan FPS pada NUC

Pengujian dilakukan pada NUC dengan spesifikasi yang telah disebutkan sebelumnya. Pengujian ini dilakukan dengan cara menjalankan sistem pada NUC dan merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel. Gambar 4.18 merupakan gambar pengambilan data fps pada NUC.



Gambar 4.18: Pengujian FPS pada NUC

Pada tabel 4.9 merupakan hasil dari pengujian model YOLOv11 berdasarkan FPS pada NUC. Dari hasil tersebut didapatkan rata rata FPS sebesar 10,042, dengan nilai tertinggi FPS sebesar 11,21 dan nilai terendah sebesar 9,12.

Tabel 4.9: Nilai Pengujian Model YOLOv11 Berdasarkan FPS pada NUC

No	FPS	No	FPS	No	FPS
1	9,12	11	10,4	21	9,59
2	10,02	12	11,21	22	9,97
3	10	13	10,87	23	10,09
4	9,94	14	9,87	24	9,97
5	10,28	15	9,23	25	10,21
6	9,54	16	10,65	26	9,89
7	9,95	17	10,02	27	10
8	9,86	18	9,92	28	10,19
9	10,32	19	10,03	29	10,22
10	9,94	20	10,09	30	9,87

Pada pengujian menggunakan model YOLOv11 pada NUC ini, hasil yang didapatkan sangat baik dimana FPS menyentuh angka yang cukup tinggi. Hal ini menunjukan bahwa sistem dapat berjalan dengan baik dan efisien.

4.4.5 Pengujian Model YOLOv8 Berdasarkan FPS pada NUC

Pengujian dilakukan pada NUC dengan spesifikasi yang telah disebutkan sebelumnya. Pengujian ini dilakukan dengan cara menjalankan sistem pada NUC dan merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel.

Pada tabel 4.10 merupakan hasil dari pengujian model YOLOv8 berdasarkan FPS pada NUC. Dari hasil tersebut didapatkan rata rata FPS sebesar 5,616 , dengan nilai tertinggi FPS sebesar 7,33 dan nilai terendah sebesar 3,65.

Tabel 4.10: Nilai Pengujian Model YOLOv8 Berdasarkan FPS pada NUC

No	FPS	No	FPS	No	FPS
1	5,43	11	6,34	21	5,32
2	3,65	12	5,76	22	6,43
3	5,67	13	5,98	23	5,65
4	6,54	14	5,34	24	6,19
5	5,34	15	6,32	25	5,71
6	4,98	16	4,94	26	4,31
7	5,74	17	5,53	27	4,67
8	5,01	18	5,67	28	5,43
9	6,43	19	6,32	29	4,81
10	7,33	20	5,12	30	6,53

Pada pengujian menggunakan model YOLOv8 pada NUC ini, hasil yang didapatkan kurang baik dimana FPS menyentuh angka yang cukup rendah. Hal ini menunjukan bahwa sistem dapat berjalan dengan baik namun tidak efisien.

4.4.6 Pengujian Model LSTM Berdasarkan FPS pada NUC

Pengujian dilakukan pada NUC dengan spesifikasi yang telah disebutkan sebelumnya. Pengujian ini dilakukan dengan cara menjalankan sistem pada laptop dan merekam hasil dari pengujian tersebut. Hasil dari pengujian ini akan ditampilkan dalam bentuk tabel.

Pada tabel 4.11 merupakan hasil dari pengujian model LSTM berdasarkan FPS pada NUC. Dari hasil tersebut didapatkan rata rata FPS sebesar 13,704 , dengan nilai tertinggi FPS sebesar 16,71 dan nilai terendah sebesar 11,43.

Tabel 4.11: Nilai Pengujian Model LSTM Berdasarkan FPS pada NUC

No	FPS	No	FPS	No	FPS
1	14,32	11	12,32	21	16,44
2	12,34	12	11,43	22	16,71
3	14,03	13	12,54	23	14,86
4	12,39	14	12,87	24	13,32
5	14,71	15	13,74	25	12,86
6	13,22	16	13,93	26	15,92
7	12,19	17	15,92	27	16,71
8	12,89	18	12,43	28	13,43
9	14,42	19	15,92	29	12,43
10	12,42	20	12,86	30	11,56

Pada pengujian menggunakan model LSTM pada NUC ini, hasil yang didapatkan sangat baik dimana FPS menyentuh angka yang cukup tinggi. Hal ini menunjukan bahwa sistem dapat berjalan dengan baik dan efisien, hal ini akan menjadi sangat penting bahwa output klasifikasi gestur dari model LSTM akan digunakan dalam sistem kendali kursi roda.

4.5 Pengujian berdasarkan hasil respond time

Pengujian respond time dilakukan dengan cara mencatat hasil klasifikasi dari model LSTM dan hasil deteksi objek dari Model. Output dari model tersebut sudah dilengkapi dengan Arah dan timestamp yang dihasilkan untuk model klasifikasi gestur, dan timestamp pengereman yang dihasilkan untuk model deteksi objek. Dimana timestamp tersebut dibandingkan hasil pengiriman dari sistem dan hasil penerimaan dari hardware, dimana kedua data tersebut diambil melalui Terminal Visual Studio dan Arduino IDE. Berikut merupakan contoh data mentah untuk kedua hasil output:

```
1
2  0: 480x800 1 Manusia, 101.8ms
3  Speed: 12.5ms preprocess, 101.8ms inference, 1.2ms postprocess per ↩
       image at shape (1, 3, 480, 800)
4  [LSTM]  Mundur (D) == 04:06:10.098
5  Aksi: Mundur
6
7  0: 480x800 2 Manusias, 95.5ms
8  Speed: 16.2ms preprocess, 95.5ms inference, 1.3ms postprocess per image↩
       at shape (1, 3, 480, 800)
9  [LSTM]  Stop (C) == 04:06:10.426
10 Aksi: Stop
11
12 0: 480x800 2 Manusias, 95.5ms
13 Speed: 3.9ms preprocess, 74.0ms inference, 1.3ms postprocess per image ↩
       at shape (1, 3, 480, 800)
14 [LSTM]  Maju (B) == 04:06:10.554
15 Aksi: Maju
```

Data diatas merupakan sampel data mentah yang didapatkan dari hasil klasifikasi gesture menggunakan model LSTM. Adapun output untuk model YOLO dipisahkan penulisannya dari hasil klasifikasi gestur, hal ini bertujuan untuk mempermudah dalam pengolahan data. Berikut merupakan contoh data mentah untuk hasil deteksi objek menggunakan model YOLOv11 dan YOLOv8

```
1
2  0: 480x800 1 Manusia, 81.6ms
3  Speed: 10.0ms preprocess, 81.6ms inference, 1.3ms postprocess per image↩
       at shape (1, 3, 480, 800)
4  STOP
5  [YOLO] BREAK Response ======== 04:06:10.553
6
7  0: 480x800 1 Manusia, 73.0ms
8  Speed: 5.5ms preprocess, 73.0ms inference, 1.4ms postprocess per image ↩
       at shape (1, 3, 480, 800)
9  STOP
10 [YOLO] BREAK Response ======== 04:06:10.737
```

Output yang didapatkan pada ardruino berisikan Timestamp. Timestamp yang dihasilkan

yaitu *Timestamp Receive ESP* dan *Timestamp Receive Motor*. contoh outputnya dapat dilihat sebagai berikut.

```
BELOK KANAN,0.8,2024-05-08 22:18:14.263
MAJU,0.4,2024-05-08 22:18:18.762
BELOK KANAN,0.0,2024-05-08 22:18:28.614
MAJU,0.0,2024-05-08 22:18:30.940
BELOK KANAN,1.2,2024-05-08 22:18:32.899
MAJU,0.8,2024-05-08 22:18:34.728
BELOK KANAN,0.0,2024-05-08 22:18:36.392
```

Pengujian ini akan dihitung *respond time* sejumlah 5 kali yang akan dijabarkan pada tabel 4.12 Hasil *respond time* akan diuji untuk mendapatkan waktu yang dibutuhkan untuk melakukan pendeteksian maupun klasifikasi gesture pada model yang kemudian diteruskan ke ESP32 hingga ke motor kursi roda mulai bergerak. Pengujian ini dilakukan secara real time pada perangkat Laptop, perhitungan delay didapatkan dari mulai dikirim hingga berhentinya motor pada kursi roda. Mengingat bahwa baik inference maupun response kirim dari kedua model berbeda maka tabel akan dibedakan menjadi dua tabel, yaitu tabel untuk model YOLO dan tabel untuk model LSTM.

Tabel 4.12: Hasil Pengujian Response Time Model LSTM

Inference	Sent	Receive	Motor	Delay	Arah
81.6 ms	04.06.10.553	04.06.10.580	04.06.10.600	47 ms	KANAN
73.0 ms	04.06.10.737	04.06.10.760	04.06.10.780	43 ms	KANAN
76.8 ms	04.06.10.098	04.06.10.130	04.06.10.160	62 ms	KANAN
84.5 ms	04.06.10.426	04.06.10.455	04.06.10.485	59 ms	KANAN
74.0 ms	04.06.10.554	04.06.10.580	04.06.10.610	56 ms	KANAN

Pada Tabel 4.12 Rata - rata waktu delay yang didapatkan adalah 53.4 ms dari data yang sudah didapatkan dari pengujian Laptop. Adapun nilai inference rata-rata yang didapatkan adalah 77.98 ms.

Tabel 4.13: Hasil Pengujian Response Time Model YOLO

Inference	Sent	Receive	Motor	Delay	Arah
62.4 ms	05.12.08.120	05.12.08.140	05.12.08.165	45 ms	STOP
59.8 ms	05.12.08.200	05.12.08.220	05.12.08.240	40 ms	STOP
70.1 ms	05.12.08.280	05.12.08.305	05.12.08.335	55 ms	STOP
66.7 ms	05.12.08.360	05.12.08.385	05.12.08.415	55 ms	STOP
58.9 ms	05.12.08.440	05.12.08.465	05.12.08.490	50 ms	STOP

Pada Tabel 4.13 Rata - rata waktu delay yang didapatkan adalah 49 ms dari data yang sudah didapatkan dari pengujian NUC, adapun hasil nya dapat dilihat pada tabel dibawah. Adapun nilai inference rata-rata yang didapatkan adalah 63.58 ms.

4.5.1 Pengujian Kesesuaian Jarak Pengereman Otomatis

Pengujian ini dilakukan pengetasan terhadap model dalam menghasilkan jarak berdasarkan hasil perhitungan pada *Bounding Box*. Pengujian ini dilakukan dengan membandingkan jarak

objek asli dengan jarak yang dihasilkan sistem pada Laptop dan Intel NUC terhadap manusia yang berdiri tegak. Kalibrasi telah dilakukan pada jarak 150 cm jarak ini diambil atas dasar visibilitas bounding box. Sehingga didapatkan nilai Focal Length sebesar 480. Nilai-nilai tersebut akan digunakan dalam pengujian kesesuaian jarak pada 150 cm dan 130 cm. Adapun tujuan dilakukan pengujian ini untuk menguji kemampuan model dalam mengukur jarak.

Pengujian ini dilakukan menggunakan alat ukur meteran yang ditancapkan pada kamera dan ditarik menuju peneliti untuk mendapatkan jarak. Nilai tersebut akan dihitung untuk diambil nilai rata rata *difference* atau perbedaan yang dihasilkan dari sistem terhadap pengukuran nyata.

4.5.2 Pengujian Kesesuaian Jarak Model YOLOv11 pada 150cm

Pada pengujian pertama jarak yang digunakan adalah 150cm atau 1.5m pada objek nyata yang jaraknya telah diukur sehingga bisa dilakukan perbandingan dan didapatkan tabel 4.14 untuk jarak menggunakan YOLOv11 *Bounding Box*.

Tabel 4.14: Hasil Pengujian kesesuaian Jarak YOLOv11 150 cm

Distance	*Yolo Bbox Distance*	*Difference*
150cm	151cm	1cm
150cm	151cm	1cm
150cm	150cm	0cm
150cm	151cm	1cm
150cm	152cm	2cm
150cm	153cm	3cm
150cm	151cm	1cm
150cm	150cm	0
150cm	150cm	0
150cm	152cm	2cm

4.5.3 Pengujian Kesesuaian Jarak Model YOLOv11 pada 130 cm

Pada pengujian pertama jarak yang digunakan adalah 130cm atau 1.3m pada objek nyata yang jaraknya telah diukur sehingga bisa dilakukan perbandingan dan didapatkan tabel 4.15 untuk jarak menggunakan YOLOv11 *Bounding Box*.

Tabel 4.15: Hasil Pengujian kesesuaian Jarak YOLOv11 130cm

Distance	*Yolo Bbox Distance*	*Difference*
130cm	128cm	2cm
130cm	128cm	2cm
130cm	129cm	1cm
130cm	127cm	3cm
130cm	130cm	0
130cm	130cm	0
130cm	131cm	1cm
130cm	132cm	2cm
130cm	131cm	1cm
130cm	130cm	0

4.5.4 Pengujian Kesesuaian Jarak Model YOLOv8 pada 150cm

Pada pengujian pertama jarak yang digunakan adalah 150cm atau 1.5m pada objek nyata yang jaraknya telah diukur sehingga bisa dilakukan perbandingan dan didapatkan tabel 4.16 untuk jarak menggunakan YOLOv8 *Bounding Box*.

Tabel 4.16: Hasil Pengujian kesesuaian Jarak YOLOv8 150 cm

Distance	*Yolo Bbox Distance*	*Difference*
150cm	154cm	4cm
150cm	142cm	8cm
150cm	144cm	6cm
150cm	144cm	6cm
150cm	151cm	1cm
150cm	150cm	0
150cm	149cm	1cm
150cm	141cm	9cm
150cm	151cm	1cm
150cm	147cm	3cm

4.5.5 Pengujian Kesesuaian Jarak Model YOLOv8 pada 130 cm

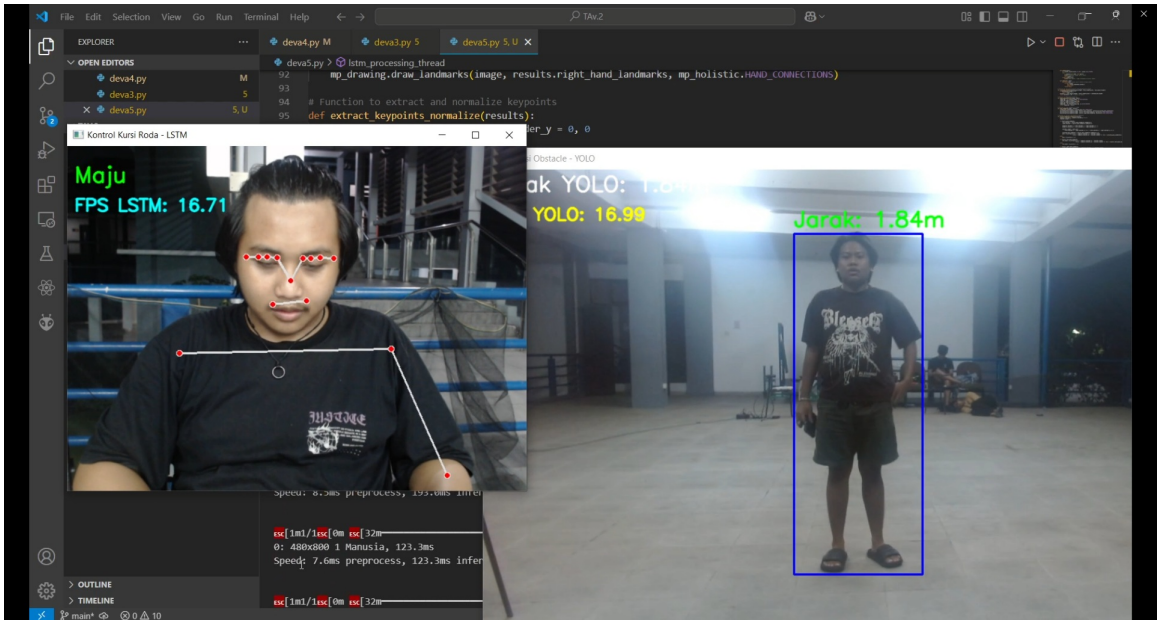
Pada pengujian pertama jarak yang digunakan adalah 130cm atau 1.3m pada objek nyata yang jaraknya telah diukur sehingga bisa dilakukan perbandingan dan didapatkan tabel 4.17 untuk jarak menggunakan YOLOv8 *Bounding Box*.

Tabel 4.17: Hasil Pengujian kesesuaian Jarak YOLOv8 130cm

Distance	*Yolo Bbox Distance*	*Difference*
130cm	129cm	1cm
130cm	127cm	3cm
130cm	128cm	2cm
130cm	122cm	8cm
130cm	129cm	1cm
130cm	130cm	0
130cm	130cm	0
130cm	134cm	4cm
130cm	132cm	4cm
130cm	132cm	2cm

4.5.6 Performa Sistem Pengereman Otomatis

Pada pengujian ini dilakukan pengetesan terhadap kursi roda dalam melakukan pengereman otomatis. Pengetesan ini akan dilakukan Pada Tower 2 ITS. Manusia yang dideteksi berdiri tegak. Adapun pengujian ini bertujuan untuk mengetahui keberhasilan pengereman dalam 10 sampel pengujian. Kursi roda sudah diatur agar tidak membahayakan subjek penelitian. Setiap model akan diuji untuk mengetahui performa sistem pengereman, serta mengetahui model manakah yang terbaik. Gambar 4.19 merupakan sampel pengujian yang akan dijadikan subjek penelitian.



Gambar 4.19: Sampel Performa Pengereman Otomatis

Selanjutnya hasil dari performa pengereman otomatis ini dicatat dalam tabel. Agar memudahkan pembaca dalam memahami isi tabel diberikan warna sesuai dengan hasil yang didapatkan. Berikut merupakan hasil pengujian tersebut.

Tabel 4.18: Tabel Performa Keberhasilan Pengereman Otomatis menggunakan YOLOv11

Percobaan	Hasil
1	Kursi Roda Berhasil Berhenti
2	Kursi Roda Berhasil Berhenti
3	Kursi Roda Berhasil Berhenti
4	Kursi Roda Berhasil Berhenti
5	Kursi Roda Berhasil Berhenti
6	Kursi Roda Berhasil Berhenti
7	Kursi Roda Berhasil Berhenti
8	Kursi Roda Berhasil Berhenti
9	Kursi Roda Berhasil Berhenti
10	Kursi Roda Berhasil Berhenti

Dari hasil yang didapatkan pada tabel 4.18 pada 10 kali sampel percobaan didapatkan rasio keberhasilan sebesar 100% yang menunjukan bahwa sistem pengereman otomatis menggunakan model YOLOv11 dapat berjalan dengan sempurna.

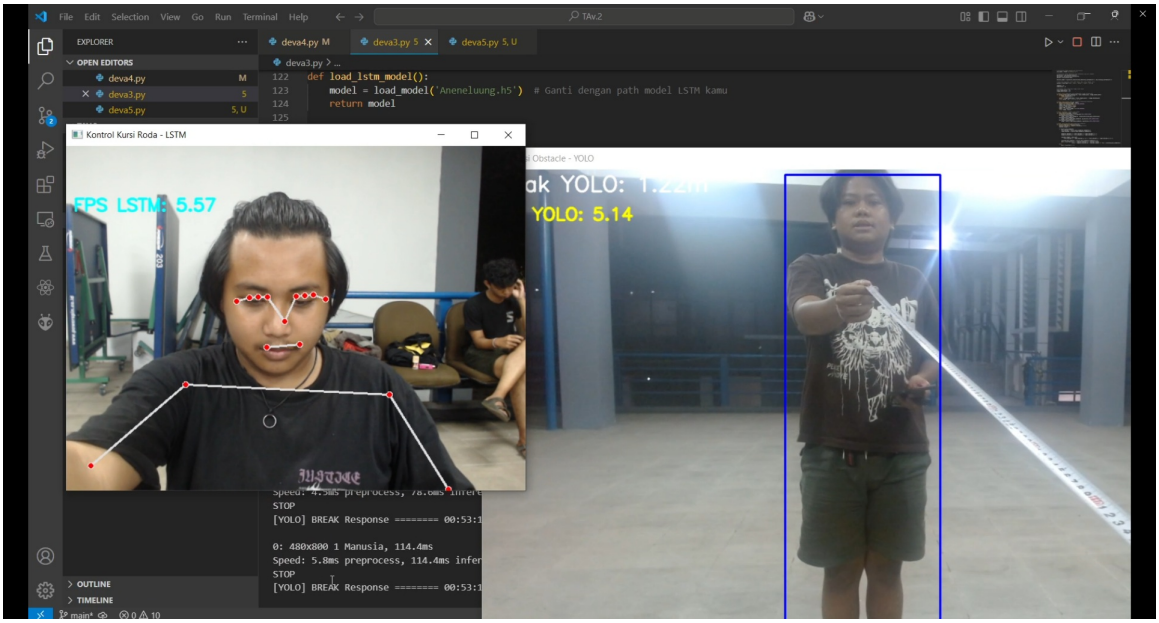
Tabel 4.19: Tabel Performa Keberhasilan Pengereman Otomatis menggunakan YOLOv8

Percobaan	Hasil
1	Kursi Roda Berhasil Berhenti
2	Kursi Roda Berhasil Berhenti
3	Kursi Roda Berhasil Berhenti
4	Kursi Roda Berhasil Berhenti
5	Kursi Roda Berhasil Berhenti
6	Kursi Roda Berhasil Berhenti
7	Kursi Roda Berhasil Berhenti
8	Kursi Roda Berhasil Berhenti
9	Kursi Roda Berhasil Berhenti
10	Kursi Roda Gagal Berhenti

Dari hasil yang didapatkan pada tabel 4.19 pada 10 kali sampel percobaan didapatkan rasio keberhasilan sebesar 90%. Adapun pada percobaan ke 10 sistem gagal dalam melakukan pengereman otomatis. Hal ini disebabkan oleh 2 faktor yang pertama adalah bounding box yang dihasilkan oleh YOLOv8 tidak sekonsisten YOLOv11, dan yang kedua adalah suhu laptop yang meningkat yang mengurangi performa sistem.

4.5.7 Pengujian Akurasi Sistem Pengereman Otomatis

Untuk mengetahui lebih dalam performa dari model dalam melakukan pengereman maka diperlukannya Uji akurasi dari sistem pengereman otomatis ini. Uji akurasi ini dilakukan dengan cara menghitung jarak yang didapatkan pada sistem dan jarak asli menggunakan meteran. Gambar 4.20 telah dilakukan pengukuran menggunakan meteran untuk mendapatkan jarak saat terjadi pengereman dan juga mengambil jarak pada sistem dengan mendokumentasikan hasil dari sistem dengan menggunakan OBS.



Gambar 4.20: Hasil Pengukuran Jarak Pengereman Otomatis

Jarak yang telah diukur lalu dibandingkan sehingga menghasilkan Tabel 4.20 yang berisi nilai jarak pengukuran nyata saat pengereman (*Real Distance*), Jarak pada sistem yang telah dicatat (*System Distance*), Error nyata, dan error sistem. Adapun tujuan dilakukannya pengujian ini yaitu untuk mengetahui kesalahan jarak yang dihasilkan dari sistem dan juga mengetahui penyebab kesalahan pengukuran jarak tersebut.

Tabel 4.20: Tabel hasil pengukuran jarak pengereman otomatis

Braking Distance	Real Distance	System Distance	Real Error	System Error
130 cm	127 cm	125 cm	3 cm	5 cm
130 cm	124 cm	123 cm	6 cm	7 cm
130 cm	115 cm	115 cm	15 cm	15 cm
130 cm	110 cm	110 cm	20 cm	20 cm
130 cm	108 cm	108 cm	22 cm	22 cm
130 cm	112 cm	112 cm	18 cm	18 cm
130 cm	111 cm	111 cm	19 cm	19 cm
130 cm	113 cm	113 cm	17 cm	17 cm
130 cm	104 cm	104 cm	26 cm	26 cm
130 cm	98 cm	99 cm	32 cm	31 cm

Dalam uji akurasi pengereman didapatkan hasil yaitu error pengukuran nyata sebesar 17.8 cm dan error sistem sebesar 18.0 cm. Dapat dilihat terdapat peningkatan nilai error yang terjadi pada sistem pengereman otomatis ini, hal tersebut disebabkan oleh beberapa faktor yaitu posisi kamera yang terguncang saat pengujian, delay pada sistem, laptop yang tidak di charge saat pengujian, dan suhu ruangan yang cukup panas menyebabkan laptop menjadi lambat.

BAB V
PENUTUP

Pada bab ini akan dipaparkan kesimpulan dari hasil pengujian yang telah dilakukan. Kesimpulan ini akan menjawab permasalahan yang diangkat dengan merujuk pada tujuan dari pelaksanaan tugas akhir ini. Pada bab ini juga dipaparkan saran mengenai hal yang dapat dilakukan untuk mengembangkan penelitian dengan topik yang sama.

5.1 Kesimpulan

Berdasarkan hasil pengujian yang telah dilakukan dalam tugas akhir ini, dapat diambil kesimpulan bahwa sistem kendali kursi roda berbasis SIBI dan *braking system* menggunakan LSTM dan YOLOv11 dapat berjalan dengan baik. Sistem dapat diimplementasikan pada laptop dan NUC dengan performa yang baik. Pengujian model YOLOv11 dengan epoch 100 lebih bagus sekitar 0.5%–1%. Model LSTM dapat memiliki akurasi 100% dalam pemanggilan kelas. Pengujian FPS dengan hasil terbaik adalah pengujian dengan YOLOv11 pada laptop dengan rata-rata FPS yang didapat sebesar 24,969. Pengujian *respond time* LSTM rata-rata berada di 53.4 ms. Pengujian kesesuaian jarak dengan eror paling kecil adalah pengujian dengan model YOLOv11 dengan jarak 150 cm dengan eror 0.73%. Performa pengereman otomatis terbaik adalah model YOLOv11 dengan akurasi 100%.

5.2 Saran

Berdasarkan hasil yang diperoleh dari tugas akhir ini, saran yang dapat dipertimbangkan untuk pengembang lebih lanjut adalah sebagai berikut:

1. Menambahkan kelas yang lebih beragam pada YOLO agar sistem pengereman lebih *safety*.
2. Menambahkan gestur yang lebih bervariasi selain SIBI untuk memperluas opsi sistem kendali kursi roda.
3. Mempertimbangkan penggunaan metode lain CNN-LSTM untuk ekstraksi fitur dalam bentuk citra sehingga dapat melakukan klasifikasi gerakan bahasa isyarat dnegan lebih baik dan akurat lagi.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] I. P. H. S. Ekatama. Perancangan sistem kontrol motor kursi roda secara nirkabel berbasis esp32. Other thesis, Institut Teknologi Sepuluh Nopember, 2024. Online.

[2] WHO. Geneva: World health organization. *World Report on Disability*, 2018.

[3] Henry Nweke, Teh Wah, Mohammed Al-Garadi, and Uzoma Alo. Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications*, 105, 04 2018.

[4] C. E. Lang. Annual review of neuroscience,. *Stroke: Mechanisms and Recovery of Movement.*, 2019.

[5] Jung Hyun Choi, Younghun Chung, and Sehoon Oh. Motion control of joystick interfaced electric wheelchair for improvement of safety and riding comfort. *Mechatronics*, 59:104–114, May 2019. Publisher Copyright: © 2019 Elsevier Ltd.

[6] M. Kim. Gesture recognition based on deep learning algorithms for control of smart devices. *IEEE Access*, 2021.

[7] S. P. Ratri. Penerapan bahasa isyarat indonesia untuk pengembangan aplikasi berbasis gesture recognition. *Jurnal Teknologi Informasi dan Pendidikan*, 2019.

[8] A. Graves. Hybrid computing using a neural network with dynamic external memory. *Nature, 538(7626)*, 2019.

[9] L. Lecrosnier. International journal of environmental research and public health. *Deep learning-based object detection, localisation and tracking for smart wheelchair healthcare mobility*, vol. 18:no. 1, p. 91, 2021.

[10] Afifah Nur and Hendro Nugroho. Deteksi fitur huruf sistem isyarat bahasa indonesia menggunakan metode chain code. *POSITIF : Jurnal Sistem dan Teknologi Informasi*, 8:36–40, 11 2022.

[11] A. S. Nugraheni, A. P. Husain, and H. Unayah. Optimalisasi penggunaan bahasa isyarat dengan sibi dan bisindo pada mahasiswa difabel tunarungu di prodi pgmi uin sunan kalijaga. *Jurnal Holistika*, 5(1):28–33, 2023.

[12] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Machine Learning and Knowledge Extraction*, 5(4):1680–1716, November 2023.

[13] Luhao He, Yongzhang Zhou, Lei Liu, and Jianhua Ma. Research and application of yolov11-based object segmentation in intelligent recognition at construction sites. *Buildings*, 14:3777, 11 2024.

[14] Imam Cholissodin and Arief Soebroto. Machine learning and deep learning. *Buku Ajar AI, Machine Learning and Deep Learning*, 2019.

[15] K. D. Larasati and A. H. Primandari. Forecasting bitcoin price based on blockchain information using long-short term method. *Parameter: Journal of Statistics*, 1(1):1–6, 2021.

[16] C. Lugaresi, J. Tang, H. Nash, and et al. Mediapipe: A framework for building perception pipelines. *CoRR*, abs/1906.08172, 2019.

[17] J.-W. Kim, J.-Y. Choi, E.-J. Ha, and J.-H. Choi. Human pose estimation using mediapipe pose and optimization method based on a humanoid model. *Applied Sciences*, 13(4):2700, 2023.

[18] D. Shah. Intersection over union (iou): Definition, calculation, code, 2023.

[19] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 3rd edition, 2022.

[20] Sugiyono. *Metode Penelitian dan Pengembangan: Research and Development (R&D)*. Alfabeta, Bandung, 2019.

BIOGRAFI PENULIS

I Putu Deva Febriana, atau yang biasa dikenal sebagai Depa, lahir di Tabanan, Bali pada 8 Februari 2002. Penulis merupakan anak pertama dari dua bersaudara yang tinggal dan tumbuh besar di pedesaan kecil di perbatasan antara Tabanan dan Negara, Bali. Penulis merupakan lulusan SMA Negeri 1 Tabanan dan melanjutkan pendidikan ke jenjang strata satu di Departemen Teknik Komputer Fakultas Teknologi Elektro dan Informatika Cerdas Institut Teknologi Sepuluh Nopember (ITS) pada tahun 2021.

Selama masa studinya, penulis menunjukkan minat yang kuat dalam bidang teknologi, terutama pada pengembangan kecerdasan buatan, dan *machine learning*. Ketertarikan ini mendorongnya untuk mendalami topik-topik seperti *machine learning* dan *Internet of Things* (IoT).

Pada penelitian tugas akhir ini, penulis memilih mengembangkan sistem kendali kursi roda berbasis SIBI dan memiliki sistem pengereman otomatis agak meminimalisir kecelakaan.

[Halaman ini sengaja dikosongkan]