



**SISTEM KONTROL ROBOT MOBIL MELALUI DETEKSI BENTUK
GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE**

LAPORAN TUGAS AKHIR

Program Studi

S1 Teknik Komputer

Oleh:

Aldi Ramadhani

20410200008

UNIVERSITAS
Dinamika

FAKULTAS TEKNOLOGI DAN INFORMATIKA

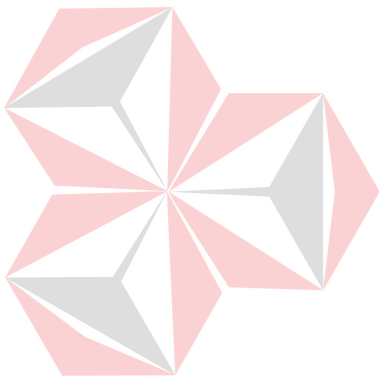
UNIVERSITAS DINAMIKA

2024

**SISTEM KONTROL ROBOT MOBIL MELALUI DETEKSI BENTUK
GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE**

TUGAS AKHIR

**Diajukan sebagai salah satu syarat untuk menyelesaikan
Program Sarjana Teknik**



UNIVERSITAS
Dinamika

Disusun Oleh:

Nama : Aldi Ramadhani
NIM : 20410200008
Program : S1 (Strata Satu)
Jurusan : Teknik Komputer

**FAKULTAS TEKNOLOGI DAN INFORMATIKA UNIVERSITAS
DINAMIKA
2024**

**SISTEM KONTROL ROBOT MOBIL MELALUI DETEKSI BENTUK
GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE**

Dipersiapkan dan disusun oleh:

Aldi Ramadhani

NIM : 20410200008

Telah diperiksa, dibahas, dan disetujui oleh Dewan Pembahas

Pada:

Susunan Dewan Pembahas

Pembimbing

I. Heri Pratikno, M.T., MTCNA., MTCRE.

NIDN. 0716117302



Digitally signed by Heri Pratikno, M.T.
DN: cn=Heri Pratikno, M.T.,
o=Universitas Dinamika, ou=S1
Teknik Komputer,
email=heri@dinamika.ac.id, c=ID
Date: 2024.08.06 11:34:23 +07'00'
Adobe Acrobat version: 11.0.23

II. Musayyanah, S.ST., M.T.

NIDN. 0730069102



Digitally signed by
Musayyanah
DN: cn=Musayyanah,
o=Universitas Dinamika,
ou=S1 Teknik Komputer,
email=musayyanah@din
amika.ac.id, c=ID
Date: 2024.08.06 11:36:48
+07'00'

Pembahas

I. Harianto, S.Kom., M.Eng.

NIDN. 0722087701



cn=Harianto Harianto,
o=Universitas Dinamika,
ou=Prodi S1 Teknik Komputer,
email=hari@dinamika.ac.id, c=ID
2024.08.06 11:42:49 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan

Untuk memperoleh gelar sarjana

Digitally signed by

Anjik Sukmaaji

Date: 2024.08.07

Dr. Anjik Sukmaaji, S.Kom., M.Eng.

NIDN. 0731057301

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA



PERNYATAAN
PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa **Universitas Dinamika**, Saya :

Nama : Aldi Ramadhani
NIM : 20410200008
Program Studi : S1 Teknik Komputer
Fakultas : Teknologi dan Informatika
Jenis Karya : Laporan Tugas Akhir
Judul Karya : **SISTEM KONTROL ROBOT MOBIL MELALUI
DETEKSI BENTUK GESTUR JARI TANGAN
MENGUNAKAN MEDIAPIPE**

Menyatakan dengan sesungguhnya bahwa :

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, Saya menyetujui memberikan kepada **Universitas Dinamika** Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalty Free Right*) atas seluruh isi/sebagian karya ilmiah Saya tersebut diatas untuk disimpan, dialihmediakan, dan dikelola dalam bentuk pangkalan data (*database*) untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama Saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.
2. Karya tersebut diatas adalah hasil karya asli Saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya, atau pendapat orang lain yang ada dalam karya ilmiah ini semata-mata hanya sebagai rujukan yang dicantumkan dalam Daftar Pustaka Saya.
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiasi pada karya ilmiah ini, maka Saya bersedia untuk menerima pencabutan terhadap gelar keserjanaan yang telah diberikan kepada Saya.

Demikian surat pernyataan ini Saya buat dengan sebenar-benarnya.

Surabaya, 19 Juli 2024



Aldi Ramadhani
NIM : 20410200008



The best things are learnt during the worst days of your life. Get up and see what new learning you had on a bad day

UNIVERSITAS
Dinamika



“Dipersembahkan kepada Ayah, Ibu, Keluarga saya dan teman-teman Teknik Komputer atas doa, dukungan, dan motivasi yang telah diberikan. Serta kepada semua pihak yang telah membantu dalam penulisan Laporan Tugas Akhir.”

ABSTRAK

Perkembangan ilmu pengetahuan dan teknologi di era globalisasi saat ini telah menghasilkan berbagai inovasi baru, seperti robot canggih dan media komunikasi yang mempengaruhi cara kita berinteraksi dan bekerja. Dengan seiring perkembangan zaman banyak aktivitas manusia yang dibantu oleh teknologi robot karena dapat diselesaikan dengan cepat, tepat, dan teratur. Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi sistem kontrol Robot Mobil menggunakan MediaPipe untuk deteksi gestur jari tangan lebih stabil dengan komunikasi data melalui WiFi. MediaPipe, sebagai *framework open-source* untuk pemrosesan media dan *machine learning*, digunakan untuk mendeteksi dan mengenali gestur jari tangan secara *real-time*, yang kemudian dikirimkan ke robot melalui koneksi WiFi. Hasil penelitian menunjukkan bahwa sistem ini berhasil mendeteksi gestur jari tangan dan mengontrol Robot Mobil. Hasil pengujian pada penelitian ini mendapatkan nilai akurasi rata-rata sebesar 100% pada jarak terbaik 30cm dan rata-rata *Frame Per Second* dari keseluruhan pengujian 14.12 FPS. Hasil penerapan pengecekan deteksi gestur jari tangan dibandingkan dengan tanpa pengecekan mendapat kesimpulan bahwa metode dengan pengecekan lebih direkomendasikan untuk digunakan dalam pengontrolan robot mobil karena mampu memberikan akurasi deteksi gestur yang sangat tinggi 100% dan konsisten. Peneliti berharap penelitian ini memberikan kontribusi signifikan dalam bidang kontrol robotik dengan menawarkan metode kontrol yang lebih intuitif dan *user-friendly*.

Kata kunci: MediaPipe, Wifi, Robotic, Deteksi, Gestur-jari.

KATA PENGANTAR

Puji syukur penulis haturkan kehadirat Allah SWT. Yang telah melimpahkan nikmat, rahmat, karunia dan juga hidayah-Nya sehingga penulis bisa menyelesaikan Laporan Tugas Akhir yang berjudul “Sistem Kontrol Robot Mobil Melalui Deteksi Gestur Jari Tangan Menggunakan Mediapipe”. Dalam mengerjakan Laporan Tugas Akhir penulis mendapat banyak dukungan dan bantuan dari berbagai pihak. Oleh karena itu penulis mengucapkan terima kasih kepada :

1. Allah SWT, yang telah memberi rahmat nikmat dan keseatan sehingga penulis bisa menyelesaikan Laporan Tugas Akhir
2. Orang tua yang selalu mensupport dan memberi motivasi penulis hingga berhasil menyelesaikan laporan Tugas Akhir
3. Bapak Dr. Anjik Sukmaaji, S.Kom., M.Eng. selaku Dekan Fakultas Teknologi dan Informatika Universitas Dinamika.
4. Bapak Pauladie Susanto, S.Kom., M.T., selaku Ketua Program Studi S1 Teknik Komputer dan selaku Dosen Pembahas. Penulis mengucapkan terimakasih atas bimbingan yang diberikan dan kesempatan serta tuntunan baik secara lisan maupun tertulis, sehingga dapat menyelesaikan Tugas Akhir.
5. Bapak Heri Pratikno, M.T. Selaku dosen pembimbing yang telah memberikan dukungan berupa motivasi, wawasan, dan saran bagi penulis selama pelaksanaan pengerjaan Tugas Akhir dan dalam pembuatan laporan Tugas Akhir.
6. Ibu Musayyanah, S.ST., M.T. selaku dosen pembimbing yang banyak memberikan masukan masukan dan Solusi agar Tugas Akhir ini menjadi lebih baik dan penulis dapat menyelesaikan Tugas Akhir ini.
7. Elsy Anafitria, Terima kasih atas segala dukungan dan semangat yang tidak pernah putus. Terima kasih atas waktu, doa, dan seluruh hal baik yang telah diberikan selama ini.
8. Seluruh rekan-rekan S1 Teknik Komputer Angkatan 2020 yang telah memberikan dukungan dan semangatnya untuk membantu penulis menyelesaikan laporan Tugas Akhir ini.
9. Seluruh pihak yang tidak dapat disebutkan satu per satu yang telah memberikan

dukungan serta bantuan dalam segala bentuk yang akhirnya terselesaikannya laporan Tugas Akhir ini.

10. Penulis berharap semoga laporan ini dapat berguna dan bermanfaat untuk menambah wawasan bagi pembacanya. Penulis juga menyadari dalam penulisan laporan ini banyak terdapat kekurangan. Oleh karena itu, penulis sangat mengharapkan saran dan kritik untuk memperbaiki kekurangan dan berusaha untuk lebih baik lagi.

Surabaya, 7 Juli 2024

Penulis



UNIVERSITAS
Dinamika

DAFTAR ISI

	Halaman
ABSTRAK	vi
KATA PENGANTAR	vii
DAFTAR ISI.....	ix
DAFTAR GAMBAR	xii
DAFTAR TABEL.....	xiii
LAMPIRAN.....	xiv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah	3
1.4 Tujuan	3
1.5 Manfaat	4
BAB II LANDASAN TEORI.....	5
2.1 Robot Mobil.....	5
2.2 Robot Manipulator	5
2.3 Robot Humanoid.....	6
2.4 Robot Terbang	6
2.5 Gestur Jari Tangan	7
2.6 OpenCV	8
2.7 Python	8
2.8 Mediapipe	9
2.9 Mikrokontroler ESP32.....	9
2.10 Arduino IDE.....	10
2.11 Visual Studio Code	11

2.12	<i>User Datagram Protocol (UDP)</i>	11
BAB III METODOLOGI PENELITIAN.....		13
3.1	Perancangan Perangkat Keras.....	13
3.2	Mekanik Robot Mobil.....	16
3.3	Perancangan Perangkat Lunak.....	17
3.4	Instalasi Environment	22
3.5	Dataset.....	22
3.6	Metode Deteksi Dengan Mediapipe.....	23
3.7	<i>Flowchart</i> Python.....	23
3.8	<i>Flowchart</i> Pengiriman Data.....	25
3.9	<i>Flowchart</i> ESP32	25
BAB IV HASIL DAN PEMBAHASAN		27
4.1	Pengujian Pengiriman dan Penerimaan Data menggunakan WiFi	27
4.1.1	Tujuan Uji Pengiriman dan Penerimaan Data Menggunakan WiFi	27
4.1.2	Perlengkapan Untuk Pengujian dan Penerimaan Data Menggunakan WiFi	27
4.1.3	Cara Pengujian	28
4.1.4	Hasil Uji	28
4.2	Pengujian Kontrol Robot Mobil Menggunakan MediaPipe	30
4.2.1	Tujuan Pengujian Kontrol Robot Mobil Menggunakan MediaPipe	30
4.2.2	Perlengkapan Untuk Pengujian Kontrol Robot Mobil Menggunakan MediaPipe	31
4.2.3	Cara Pengujian	31
4.2.4	Hasil Uji Kontrol Robot Mobil Menggunakan MediaPipe.....	31
4.3	Pengujian Perbandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan	34
4.3.1	Tujuan Pengujian Perbandingan Program Pengecekan Deteksi Gestur Jari	

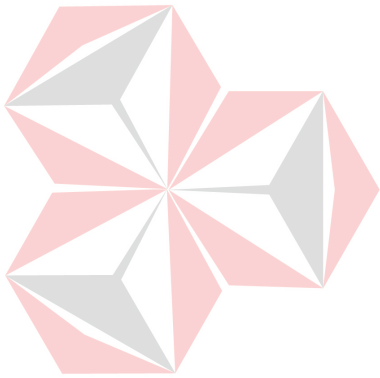
Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan	34
4.3.2 Perlengkapan Pengujian Perbandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan	35
4.3.3 Cara Pengujian	35
4.3.4 Hasil Pengujian Pebandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan	35
4.4 Pengujian Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe	38
4.4.1 Tujuan Pengujian Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe	38
4.4.2 Perlengkapan Untuk Pengujian Jarak Terbaik Mengontrol Robot Mobil Menggunakan MediaPipe	38
4.4.3 Cara Pengujian	38
4.4.4 Hasil Uji Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe	39
BAB V KESIMPULAN DAN SARAN	43
5.1 Kesimplan	43
5.2 Saran	44
DAFTAR PUSTAKA	45
LAMPIRAN	49

DAFTAR GAMBAR

Gambar 2.1 Robot Mobil	5
Gambar 2.2 Robot Manipulator	6
Gambar 2.3 Robot Humanoid	6
Gambar 2.4 Robot Terbang.....	7
Gambar 2.5 Gestur Jari Tangan	7
Gambar 2.6 Logo OpenCV	8
Gambar 2.7 Bahasa Pemograman Python.....	8
Gambar 2.8 Framework Mediapipe	9
Gambar 2.9 Mikrokontroler ESP32	10
Gambar 2.10 Arduino Integrated Development Environment (IDE).....	10
Gambar 2.11 Visual Studio Code	11
Gambar 3.1 Diagram perancangan perangkat keras	13
Gambar 3.2 Perancangan hardware.....	13
Gambar 3.3 Desain hardware	17
Gambar 3.4 source code IP statis	18
Gambar 3.5 Kontrol robot berdasarkan data yang diterima.....	18
Gambar 3.6 Inisialisai variabel pada kestabilan gestur.....	20
Gambar 3.7 Source code pengecekan gestur jari	20
Gambar 3.8 Source code pengiriman data	21
Gambar 3.9 Dataset Gestur Jari Tangan	23
Gambar 3.10 Flowchart Python	24
Gambar 3.11 Flowchart Pengiriman Data.....	25
Gambar 3.12 Flowchart Mikrokontroler ESP32	26
Gambar 4.1 Hasil Deteksi Program Python	28
Gambar 4. 2 Output pada Serial Monitor dan Terminal Program Python	29
Gambar 4.3 Deteksi Gestur Jari Tangan 5	31
Gambar 4.4 Deteksi Gestur Jari Tangan 5 dengan jarak 80 cm.....	39

DAFTAR TABEL

Tabel 3.1 Fungsi dari tiap pin yang terhubung	15
Tabel 4.1 Pengujian komunikasi secara wireless	29
Tabel 4.2 Pengujian Kontrol Robot Mobil Dengan MediaPipe	32
Tabel 4.3 Pengujian tanpa pengecekan deteksi gestur jari tangan	35
Tabel 4.4 pengujian dengan menggunakan pengecekan deteksi gestur jari tangan	36
Tabel 4.5 Pengujian Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe	39



UNIVERSITAS
Dinamika

LAMPIRAN

Lampiran 1 Source code program python dengan pengecekan	49
Lampiran 2 Source code program python tanpa pengecekan.....	54
Lampiran 3 Source Code program arduino	59
Lampiran 4 Form Bimbingan Tugas Akhir.....	64
Lampiran 5 Bukti Originalitas Tugas Akhir	65
Lampiran 6 Biodata Penulis	66



UNIVERSITAS
Dinamika

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan ilmu pengetahuan dan teknologi di era globalisasi saat ini telah menghasilkan berbagai inovasi baru, seperti robot canggih dan media komunikasi yang mempengaruhi cara kita berinteraksi dan bekerja. Dengan seiring perkembangan zaman banyak aktivitas manusia yang dibantu oleh teknologi robot karena dapat diselesaikan dengan cepat, tepat, dan teratur (Waruru, 2021). Dahulu, pengendalian robot memerlukan jarak yang sangat dekat. Namun, beberapa tahun kemudian, pengendalian robot menjadi umum dilakukan dari jarak jauh menggunakan *joystick* atau remote kontrol dengan kabel. Kemudian, dengan perkembangan lebih lanjut, sistem pengendalian dapat dilakukan tanpa kabel, menggunakan teknologi nirkabel. Saat ini, teknologi pengendalian robot sedang dikembangkan untuk dapat bekerja langsung dengan menggunakan gestur tangan manusia (Leksono et al., 2020).

Pengenalan pola gestur tangan telah dimanfaatkan secara luas dalam berbagai bidang, termasuk hiburan, pendidikan, dan keamanan. Perkembangan teknologi baru-baru ini telah memungkinkan penggunaan gestur tangan dalam berbagai aplikasi, seperti kontrol perangkat elektronik, permainan interaktif. Gestur tangan atau hand tracking memungkinkan sistem secara real-time mengidentifikasi arti dari pergerakan tangan manusia (Prananta et al., 2023).

Mikrokontroler berkembang dengan cepat dan semakin populer dalam aplikasi sistem kontrol. ESP32 merupakan mikrokontroler yang diperkenalkan oleh *ESP32ressif Systems* dan merupakan penerus dari mikrokontroler ESP328266. Mikrokontroler ini telah dilengkapi dengan modul WiFi yang terintegrasi dalam chip-nya, sehingga sangat mendukung dalam pengembangan sistem aplikasi Internet of Things (Imran & Rasul, 2020).

Mediapipe adalah platform yang dikembangkan oleh Google yang memungkinkan berbagai proses pemrosesan data persepsi dalam format audio dan video. Berbagai solusi *machine learning*, termasuk pemahaman holistik, segmentasi rambut, deteksi wajah, dan pengenalan gerakan tangan, tersedia dalam

kerangka kerja yang dirilis pada tahun 2019 (Maryamah et al., 2023).

Saat ini ESP32 dapat memenuhi kebutuhan manusia karena banyak digunakan dalam sistem kontrol. Namun, implementasi computer vision dan *deep learning* masih terbatas dalam banyak bidang, meskipun saat ini computer vision sudah dapat terhubung dengan mikrokontroler. Ini merupakan peluang untuk menemukan solusi bagi permasalahan tertentu. Selain itu, teknologi framework MediaPipe dapat efisien digunakan sebagai alat untuk mendeteksi gestur jari yang rumit dengan akurasi yang tinggi. Dikarenakan kemampuannya dalam mengekstraksi landmark, fitur dari MediaPipe sangat cocok untuk mendeteksi gestur tangan. Seperti pada penelitian M. Aldi Fakhrudin tentang Sistem Deteksi Gestur Jari Tangan Menggunakan MediaPipe dan FasterRCNN untuk Mengontrol Kecepatan Kipas Angin (Fakhrudin, 2022). Pada penelitian Tugas Akhir sebelumnya telah dilakukan oleh (Edowai, 2023) adalah Sistem Automatic Feature Selection Berbasis Deteksi Gestur Kedua Jari Tangan Untuk Mengontrol Level Kecepatan Putaran 2 Kipas Angin Menggunakan MediaPipe. Pada penelitian (Wakerkwa, 2023) adalah Kontrol Level Kecepatan Putaran Kipas Angin Melalui Deteksi Bentuk Gestur Jari Tangan Berbasis IoT.

Berdasarkan jurnal (Talakua et al., 2020) yang berjudul “Sistem Kendali Mobile Robot Menggunakan Gestur Tangan Bebas Wireless” metode yang digunakan dengan menggunakan sensor accelerometer Adxl335 yang di letakkan di tangan untuk menggerakkan mobile robot. Untuk mengirim dan menerima data menggunakan NRF24L01, NRF24L01 mengirimkan perintah yang akan menjalankan sebuah Mobil Robot.

Penelitian terdahulu sering kali mengalami masalah dengan ketidakstabilan dalam pendeteksian gestur jari tangan, namun penelitian saat ini telah berhasil mencapai kestabilan yang lebih baik dalam pengenalan gestur jari tangan maka pada Tugas Akhir ini, penulis merancang dan membangun sebuah sistem pengendalian Robot Mobil melalui deteksi gestur jari tangan secara computer vision menggunakan *framework* MediaPipe. Dimana proses pengontrolan Robot Mobil dilakukan secara *wireless*, sehingga pergerakan dari Robot Mobil mempunyai mobilitas yang fleksibel serta berjalan dalam waktu nyata (*realtime*). Adapun arah pergerakan Robot Mobil jari tangan 2 bergerak maju, jari tangan 3

belok kanan, jari tangan 4 belok kiri, jari tangan 5 bergerak mundur, dan jari tangan mengepal untuk berhenti.

1.2 Rumusan Masalah

Dari latar belakang di atas, dapat dirumuskan masalah pada Tugas Akhir ini sebagai berikut:

1. Bagaimana mengontrol arah pergerakan Robot Mobil secara computer vision melalui deteksi gestur jari tangan menggunakan MediaPipe?
2. Bagaimana perbandingan antara program dengan pengecekan dan tanpa pengecekan deteksi gestur jari tangan?
3. Bagaimana analisi komunikasi UDP untuk pengiriman data deteksi bentuk gestur jari tangan?
4. Berapa jarak terbaik antara pengontrol dengan kamera untuk proses deteksi gestur jari tangan terhadap pergerakan Robot Mobil?

1.3 Batasan Masalah

Dalam penelitian ini, terdapat batasan masalah pada beberapa hal sebagai berikut :

1. Robot Mobil hanya dikontrol menggunakan tangan kanan
2. Hanya bisa mendeteksi 5 bentuk gestur jari tangan (2 – 5 jari tangan dan jari menggengam)
3. Hanya bisa mendeteksi telapak tangan
4. Hasil pengujian ini terpengaruh oleh pencahayaan merata pada ruangan

1.4 Tujuan

Berdasarkan latar belakang dan rumusan masalah diatas, yang dapat menjadi tujuan pada Tugas Akhir ini sebagai berikut :

1. Mampu mengontrol arah pergerakan Robot Mobil secara computer vision melalui deteksi gestur jari tangan menggunakan MediaPipe
2. Dapat mengetahui tingkat realtime pada sistem kontrol Robot Mobil melalui

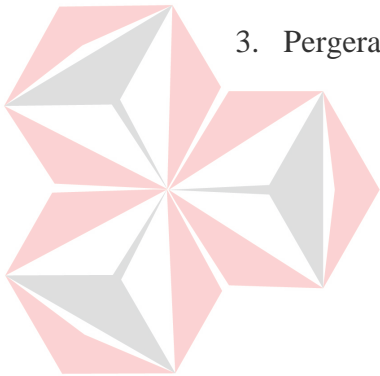
deteksi gestur jari tangan menggunakan MediaPipe

3. Dapat mengetahui besar tingkat akurasi kecocokan antara gestur jari tangan dengan arah pergerakan Robot Mobil
4. Dapat menentukan jarak terbaik antara pengontrol dengan kamera untuk proses deteksi gestur jari tangan terhadap pergerakan Robot Mobil.

1.5 Manfaat

Manfaat dari penelitian ini dapat diperoleh sebagai berikut :

1. Memudahkan pengguna dalam mengontrol Robot Mobil hanya dengan gestur jari tangan.
2. Proses interaksi antara pengguna dan Robot Mobil dapat dilakukan secara natural dan lebih alami karena pengguna tidak perlu kontak fisik.
3. Pergerakan Robot Mobil lebih fleksibel karena dilakukan secara *wireless*



UNIVERSITAS
Dinamika

BAB II LANDASAN TEORI

2.1 Robot Mobil

Mobil robot adalah jenis robot yang dirancang untuk bergerak di lingkungan tertentu dengan menggunakan roda atau kaki sebagai alat pergerakannya. Dengan berbagai desain fisik, sistem penggerak, sensor, kontrol, dan navigasi yang kompleks untuk berpindah dari satu titik ke titik yang lainnya (Khairudin et al., 2020), mobil robot dapat digunakan dalam berbagai aplikasi mulai dari industri hingga layanan rumah tangga. Pengembangan teknologi terus meningkatkan kinerja dan fleksibilitas mobil robot, menjadikannya solusi yang penting dalam memecahkan berbagai tantangan di berbagai bidang kehidupan manusia.



Gambar 2.1 Robot Mobil

2.2 Robot Manipulator

Robot manipulator memiliki lengan mekanis yang memiliki kemampuan bergerak serupa dengan tangan manusia. Bagian lengan ini terhubung oleh sendi, memungkinkan gerakan seperti rotasi dan translasi. Motor menggerakkan sendi-sendi, dan sensor memberikan feedback untuk kontrol yang lebih presisi. Alat di ujung lengan melakukan tugasnya. Berdasarkan input dari sensor dan program yang telah ditetapkan, kontroler mengatur gerakan lengan. Aplikasi industri seperti pengelasan, pengecatan, perakitan, pemindahan material, dan pengemasan adalah

semua contoh aplikasi robot manipulator. Salah satu manfaatnya adalah fleksibilitas, kecepatan, presisi, efisiensi, dan keamanan (Arlean, 2017).



Gambar 2.2 Robot Manipulator

(Sumber: Wibowo, 2020)

2.3 Robot Humanoid

Robot humanoid adalah robot yang memiliki penampilan keseluruhan yang mirip dengan tubuh manusia dan mampu berinteraksi dengan peralatan dan lingkungan yang dibuat untuk manusia. Sebagian besar bentuk robot humanoid memiliki kepala, dua buah lengan, dan dua kaki, tetapi ada juga yang hanya memiliki sebagian dari tubuh manusia, seperti dari pinggang ke atas (Lubis, 2018).



Gambar 2.3 Robot Humanoid

(Sumber: Lubis, 2018)

2.4 Robot Terbang

merupakan teknologi pesawat terbang tak berawak yang dapat beroperasi secara otomatis atau dikelola dari jarak jauh melalui pengendalian jarak jauh.

Dengan menggunakan prinsip aerodinamika, pesawat tanpa awak dapat terbang dan dapat digunakan dalam berbagai misi penerbangan (Yuda & Airlangga, 2023).

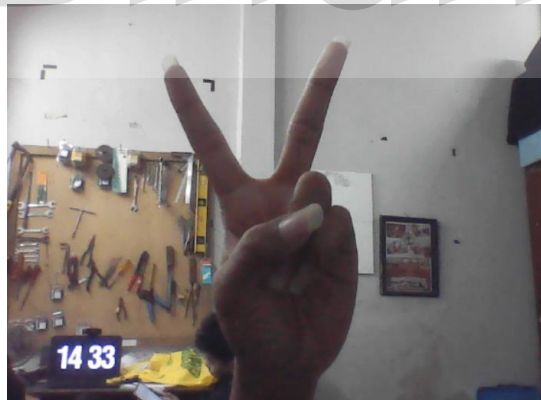


Gambar 2.4 Robot Terbang

(Sumber: Arifin, 2021)

2.5 Gestur Jari Tangan

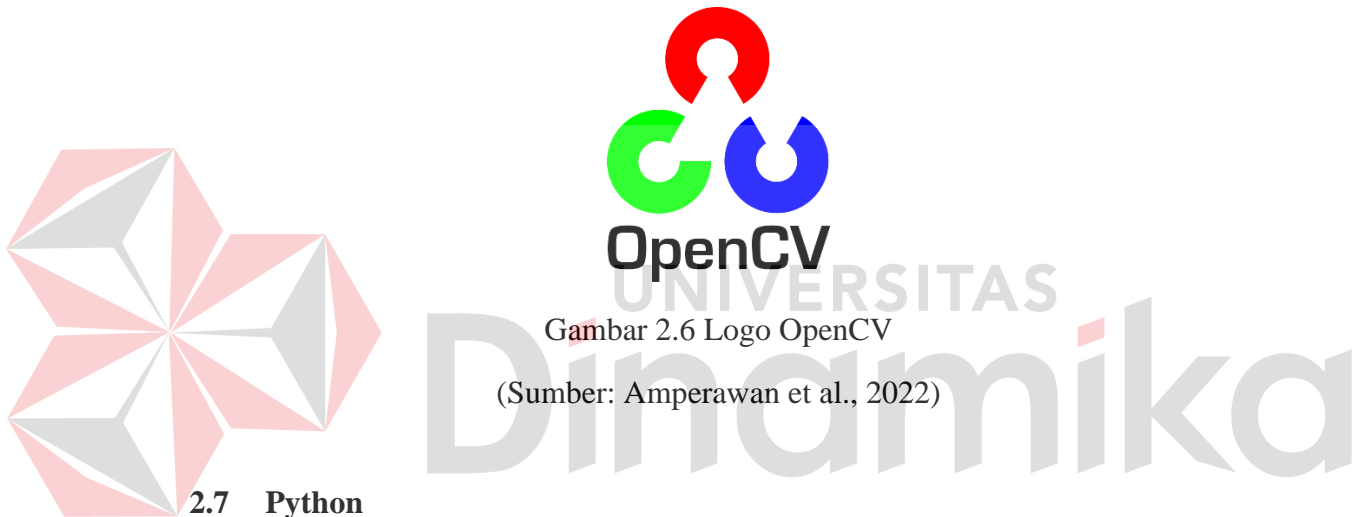
Gestur jari tangan adalah gerakan atau posisi spesifik jari-jari yang digunakan untuk menyampaikan informasi atau perintah tanpa menggunakan kata-kata. Gestur ini bisa digunakan dalam berbagai konteks, seperti komunikasi sehari-hari, bahasa isyarat, pengendalian perangkat, dan interaksi dengan teknologi.



Gambar 2.5 Gestur Jari Tangan

2.6 OpenCV

Open Source Computer Vision Library adalah salah library yang ditujukan untuk pengolahan citra dinamis secara real-time, yang dibuat oleh Intel, dan saat ini didukung oleh Willow Garage dan Itseez. OpenCV tersedia secara bebas di bawah lisensi permisif BSD dan dapat digunakan secara komersial tanpa perlu memberikan kode sumbernya. OpenCV memiliki 2500 jenis algoritma yang dioptimalkan dan dirancang untuk kemudahan komputasi. Berbagai algoritma OpenCV memiliki aplikasinya masing-masing, adapun deteksi gerak tangan, deteksi wajah, dan banyak lagi lainnya (Andri Nugraha Ramdhon & Fadly Febriya, 2021).



Gambar 2.6 Logo OpenCV

(Sumber: Amperawan et al., 2022)

2.7 Python

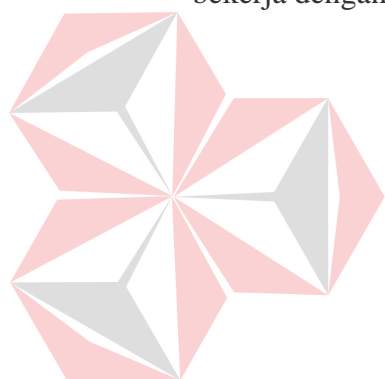
Python adalah bahasa pemrograman *interpretative* yang dapat digunakan untuk memenuhi berbagai kebutuhan programming dengan filosofi rancangan yang berpusat pada tingkat keterbacaan kode. Bahasa ini juga digunakan untuk mengembangkan *machine learning*, *data science*, dan Internet of Things. Pada penelitian ini bahasa pemrograman Python digunakan sebagai bahasa utama untuk melakukan proses training dan proses fitur selection untuk deteksi gestur jari tangan.



Gambar 2.7 Bahasa Pemrograman Python

2.8 Mediapipe

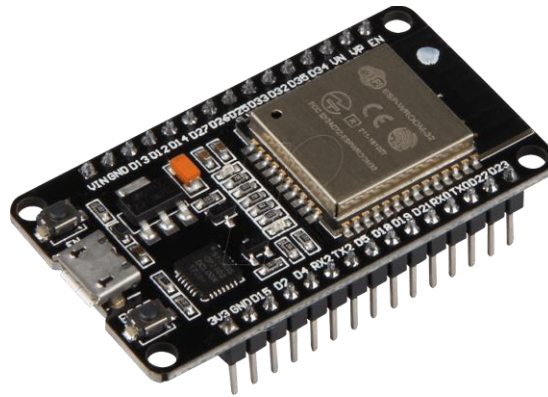
MediaPipe adalah platform yang dikembangkan oleh Google yang memungkinkan pengembangan pipeline untuk mengolah data persepsi dari berbagai format audio dan video. MediaPipe dibuat untuk orang-orang yang ingin membuat aplikasi yang menggunakan AI. Pada tahun 2019, Google membuat MediaPipe menjadi open source dan digunakan untuk tujuan internal pada tahun 2012. MediaPipe menawarkan berbagai solusi pengajaran mesin, seperti deteksi wajah, pembagian rambut holistik, klasifikasi audio, dan lainnya. Solusi yang ditawarkannya kompatibel dengan sistem open source Android dan IOS, dan bekerja dengan bahasa C++, Python, JS, dan Coral (Mediapipe, 2019).



Gambar 2.8 Framework Mediapipe
(Astriani et al., 2022)

2.9 Mikrokontroler ESP32

ESP32 sebagai mikrokontroler yang penulis pakai untuk proyek kali ini. Keunggulan ESP32 adalah tersedia modul WiFi dalam chip, jumlah pin lebih banyak dari mikrokontroler lainnya, memori yang di berikan lebih besar, dan ada juga terdapat Bluetooth 4.0 *low energy*. Modul ini memiliki fitur yang bermanfaat seperti TCP/IP, HTTP, dan FTP, serta pemrosesan sinyal analog, dukungan untuk sensor, dan dukungan untuk perangkat masukan/keluaran digital (I/O). ESP32 juga mendukung konektivitas Bluetooth, sehingga dapat mengontrol perangkat Bluetooth (Rama Akbar, 2020).



Gambar 2.9 Mikrokontroler ESP32

(Sumber: Rama Akbar, 2020)

2.10 Arduino IDE

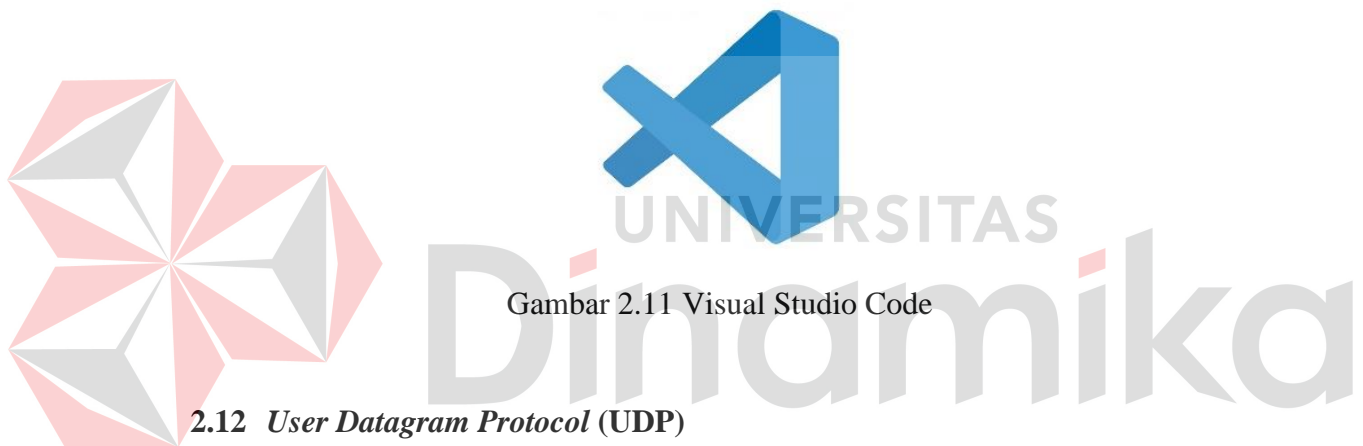
Software Arduino (IDE) adalah kependekan dari *Integrated Development Environment*, atau secara bahasa mudahnya lingkungan pengembangan terintegrasi, yang digunakan untuk pengembangan. disebut sebagai lingkungan karena *software* ini memungkinkan Arduino untuk melakukan tugas yang dibenamkan melalui pemrograman. Arduino memiliki bahasa pemrograman sendiri yang menyerupai bahasa C. Bahasa pemrograman Arduino (*Sketch*) telah dimodifikasi untuk membuatnya lebih mudah bagi pemula untuk mulai menggunakannya dari awal. Sebelum dirilis, mikrokontroler Arduino telah ditanamkan suatu program bernama Bootlader. Ini berfungsi sebagai penghubung antara mikrokontroler dan compiler Arduino. Arduino IDE dibangun dengan menggunakan bahasa pemrograman Java. Selain itu, Arduino IDE dilengkapi dengan library C/C++ yang dikenal sebagai Wiring, yang memudahkan input dan output. Software pengolahan yang diubah menjadi Arduino memberikan inspirasi untuk IDE Arduino ini (Surahman et al., 2021).

Gambar 2.10 Arduino *Integrated Development Environment (IDE)*

(Sumber: Maulana et al., 2022)

2.11 Visual Studio Code

Visual Studio Code adalah aplikasi teks editor sumber terbuka yang dikembangkan oleh Microsoft yang mendukung berbagai jenis bahasa pemrograman, seperti C++, C#, Java, Python, PHP, dan GO. Program dapat membedakan warna berdasarkan fungsi dan mengidentifikasi jenis bahasa pemrograman yang digunakan. Visual Studio Code telah terintegrasi ke Github dan memiliki fitur tambahan seperti kemampuan untuk menambahkan ekstensi bagi Pengembang yang ingin menambah fitur yang tidak ada di Visual Studio Code (nufriani, 2019).



Gambar 2.11 Visual Studio Code

2.12 User Datagram Protocol (UDP)

UDP, singkatan dari *User Datagram Protocol*, adalah salah satu protokol lapisan transport TCP/IP yang mendukung komunikasi yang tidak andal (*unreliable*) yaitu Pesan UDP dikirimkan sebagai datagram tanpa nomor urut atau pesan *acknowledgment*. Tanpa koneksi (*connectionless*) Pesan-pesan UDP akan dikirimkan tanpa harus dilakukan proses negosiasi koneksi antara dua host yang hendak bertukar informasi. Artinya, dalam protokol UDP, tidak ada langkah-langkah seperti pembukaan, pemeliharaan, atau penutupan koneksi seperti yang terjadi dalam protokol TCP (*Transmission Control Protocol*). Sebagai gantinya, setiap pesan dikirimkan secara terpisah, mandiri, dan tidak dijamin untuk tiba dalam urutan yang benar atau tiba sama sekali, karena tidak ada konfirmasi pengiriman atau pengaturan ulang otomatis dalam UDP. Kegunaan UDP adalah Protokol yang "ringan" seperti *Domain Name System* (DNS) memungkinkan pertukaran pesan

spesifik tanpa membebani sumber daya. Protokol yang menyediakan layanan keandalan, seperti *Trivial File Transfer Protocol* (TFTP) dan *Network File System* (NFS), mengurangi ketergantungan pada TCP. Contoh protokol yang tidak memerlukan keandalan adalah *Routing Information Protocol* (RIP). UDP mendukung transmisi broadcast, memungkinkan pengiriman paket data ke banyak tujuan, seperti *NetBIOS Name Service* (Nabilah Humairah, 2017).

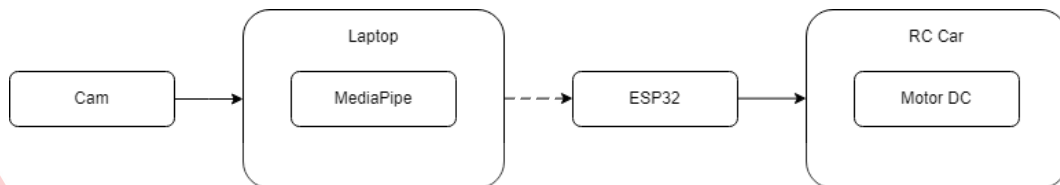


UNIVERSITAS
Dinamika

BAB III METODOLOGI PENELITIAN

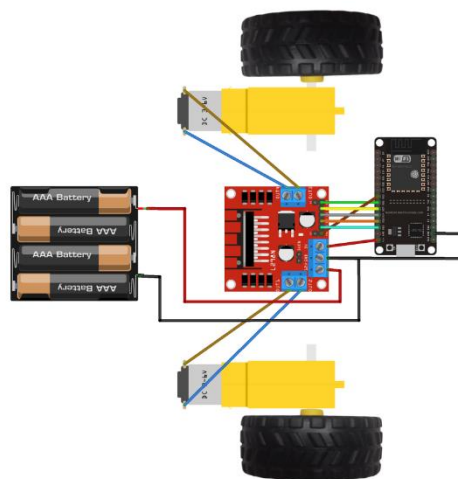
3.1 Perancangan Perangkat Keras

Struktur hardware dapat di lihat pada Gambar dibawah yang inputannya berasal dari cam yang mengenali gestur jari tangan. Selanjutnya inputan tersebut di proses oleh laptop menggunakan sistem computer vision dengan MediaPipe yang outputnya tersebut akan terhubung ke mikrokontroler ESP32 untuk melakukan aksi pada motor DC dengan menggerakkan Robot Mobil berdasarkan gestur jari tangan.



Gambar 3.1 Diagram perancangan perangkat keras

Selain itu pada diagram perancangan perangkat keras mikrokontroler ESP32 dihubungkan ke motor DC. Kemudian motor DC dihubungkan pada ban Robot Mobil untuk menggerakkan Robot Mobil melaju ke kanan, kiri, maju, mundur, dan stop. Pada Gambar 3.2 merupakan model rancangan perangkat keras yang dilakukan pada penelitian Tugas Akhir ini.



Gambar 3.2 Perancangan *hardware*

Pada gambar 3.2 Perancangan *hardware* pada sistem kontrol robot mobil menggunakan mikrokontroler ESP32 dan driver motor L298N melibatkan beberapa komponen penting dan langkah-langkah yang teliti. Komponen utama yang dibutuhkan meliputi mikrokontroler ESP32, dua motor DC, modul driver motor L298N, empat baterai AAA, roda dan sasis untuk struktur fisik robot, serta kabel jumper untuk penyambungan komponen. Langkah pertama adalah menghubungkan motor DC ke output modul driver motor L298N, dengan motor pertama dihubungkan ke OUT1 dan OUT2 dan motor kedua ke OUT3 dan OUT4. Selanjutnya, selanjutnya hubungkan GPIO 13 dari ESP32 ke pin IN1 pada L298N, pin GPIO 12 dari ESP32 ke pin IN2 pada L298N, pin GPIO 14 dari ESP32 ke pin IN3 pada L298N, pin GPIO 27 dari ESP32 ke pin IN4 pada L298N, pin GPIO25 dari ESP32 ke pin EN1, dan pin GPIO26 dari ESP32 ke pin EN2. Untuk menyediakan daya, empat baterai AAA disusun dalam tempat baterai untuk menghasilkan tegangan total sekitar 12V sambungkan kutub positif pada pin 12 volt pada L298N dan kutub negatif ke pin GND setelah itu, hubungkan pin Vin dari ESP32 ke pin 5V pada L298N dan pin GND dari ESP32 ke pin GND pada L298N agar baterai memastikan ESP32 mendapat daya yang cukup. Struktur fisik robot dirancang dengan memasang roda pada motor DC dan menempatkannya pada sasis yang kuat, memastikan motor terpasang kokoh dan roda dapat berputar bebas. Semua komponen seperti baterai, driver motor, dan ESP32 dipasang dengan baik dan aman pada sasis. Koneksi antara komponen dilakukan menggunakan kabel jumper dengan penyambungan yang kuat untuk memastikan kestabilan. Dengan demikian, hardware untuk robot mobil ini siap digunakan, memastikan motor-motor DC dapat dikontrol dengan baik melalui driver motor yang terhubung ke pin-pin digital ESP32, dan seluruh sistem diberi daya yang memadai dari empat baterai AAA. Desain fisik yang kuat memastikan bahwa semua komponen terpasang aman dan dapat berfungsi optimal. Pin-pin yang terhubung memiliki fungsi sebagai berikut:

Tabel 3.1 Fungsi dari tiap pin yang terhubung

Pin yang terhubung	Fungsinya
Pin OUT1 dan OUT2 pada L298N terhubung ke motor pertama.	mengatur arah dan kecepatan motor pertama
Pin OUT3 dan OUT4 pada L298N terhubung ke motor kedua.	mengontrol arah dan kecepatan motor kedua
Pin GPIO 13 pada ESP32 terhubung pada pin IN1 pada L298N	Mengontrol arah motor pertama (misalnya motor kiri). Pin ini mengatur gerakan maju atau mundur motor pertama. Ketika GPIO 13 diberi sinyal tinggi (HIGH), motor akan bergerak maju. Sebaliknya, jika diberi sinyal rendah (LOW), motor akan bergerak mundur.
Pin GPIO 12 pada ESP32 terhubung pada pin IN2 pada L298N	Bekerja bersama dengan IN1 untuk mengontrol arah motor pertama. Jika IN1 dan IN2 dalam kondisi berlawanan (salah satu HIGH, yang lain LOW), motor pertama akan bergerak maju atau mundur sesuai dengan status IN1 dan IN2.
Pin GPIO 14 pada ESP32 terhubung pada pin IN3 pada L298N	Mengontrol arah motor kedua (misalnya motor kanan). Pin ini berfungsi mirip dengan IN1, tetapi untuk motor kedua. Jika GPIO 14 HIGH, motor kedua bergerak maju. Jika LOW, motor kedua bergerak mundur.
Pin GPIO 27 pada ESP32 terhubung ke pin IN4 pada L298N	Bekerja bersama dengan IN3 untuk mengontrol arah motor kedua. Jika IN3 dan IN4 dalam kondisi berlawanan, motor kedua akan bergerak maju atau

	mundur sesuai dengan status IN3 dan IN4.
Pin GPIO 25 (ESP32) terhubung pada EN1 (L298N)	Sinyal ini mengontrol kecepatan motor pertama dan mengaktifkan atau menonaktifkan pin ini juga dapat menyalakan atau mematikan motor pertama.
Pin GPIO 26 (ESP32) terhubung pada EN2 (L298N)	Sinyal ini mengontrol kecepatan motork kedua. seperti EN1, mengaktifkan atau menonaktifkan pin ini juga dapat menyalakan atau mematikan motor kedua.

3.2 Mekanik Robot Mobil

Robot yang ditunjukkan pada Gambar 3.3 adalah Robot Mobil sederhana yang dirancang untuk dikendalikan secara nirkabel melalui WiFi. Struktur utama robot ini terdiri dari rangka yang terbuat dari bahan kayu atau bahan ringan lainnya, yang berfungsi sebagai dasar untuk memasang semua komponen yang di sebut sasis. Rangka ini mendukung 2 roda di belakang dan 1 roda bebas di depan. Roda belakang terhubung ke motor DC, yang menggerakkan robot maju, mundur, dan berbelok kanan kiri. Roda depan tidak terhubung pada motor DC sehingga dapat berputar bebas mengikuti roda bagian belakang.

Di bagian belakang rangka, terdapat beberapa baterai yang tersusun rapi, berfungsi sebagai sumber daya untuk motor dan komponen elektronik lainnya. Di bagian depan, terdapat modul ESP32, sebuah mikrokontroler yang digunakan untuk mengendalikan seluruh sistem robot. Modul ini menerima perintah melalui WiFi dan mengirimkan sinyal kontrol ke driver motor, yang terletak di tengah rangka. Driver motor ini mengatur kecepatan dan arah putaran motor berdasarkan sinyal dari ESP32.

Berbagai kabel penghubung menghubungkan komponen-komponen elektronik ini, memastikan bahwa sinyal kontrol dan daya disalurkan dengan benar. Dengan konfigurasi ini, robot dapat menerima perintah seperti "MAJU",

"MUNDUR", "BELOK KANAN", dan "BELOK KIRI" melalui jaringan WiFi, yang kemudian diproses oleh ESP32 untuk menggerakkan motor sesuai perintah, memungkinkan robot untuk bergerak sesuai dengan instruksi yang diterima.

Cara kerja pada perangkat keras ini pertama-tama pengguna melakukan deteksi jari tangan melalui kamera laptop selanjutnya akan di proses oleh MediaPipe yang memiliki perintah maju, mundur, kanan, kiri, dan stop. Jika kamera mendeteksi jari tangan pengguna menunjukkan angka 2 maka Robot Mobil akan bergerak maju, jika kamera mendeteksi jari tangan pengguna menunjukkan angka 3 maka Robot Mobil akan belok kanan, jika kamera mendeteksi jari tangan pengguna menunjukkan angka 4 maka Robot Mobil akan belok kiri, jika kamera mendeteksi jari tangan pengguna menunjukkan angka 5 maka Robot Mobil akan mundur, jika kamera mendeteksi jari tangan pengguna menunjukkan tangan menggenggam maka Robot Mobil akan stop.



Gambar 3.3 Desain *hardware*

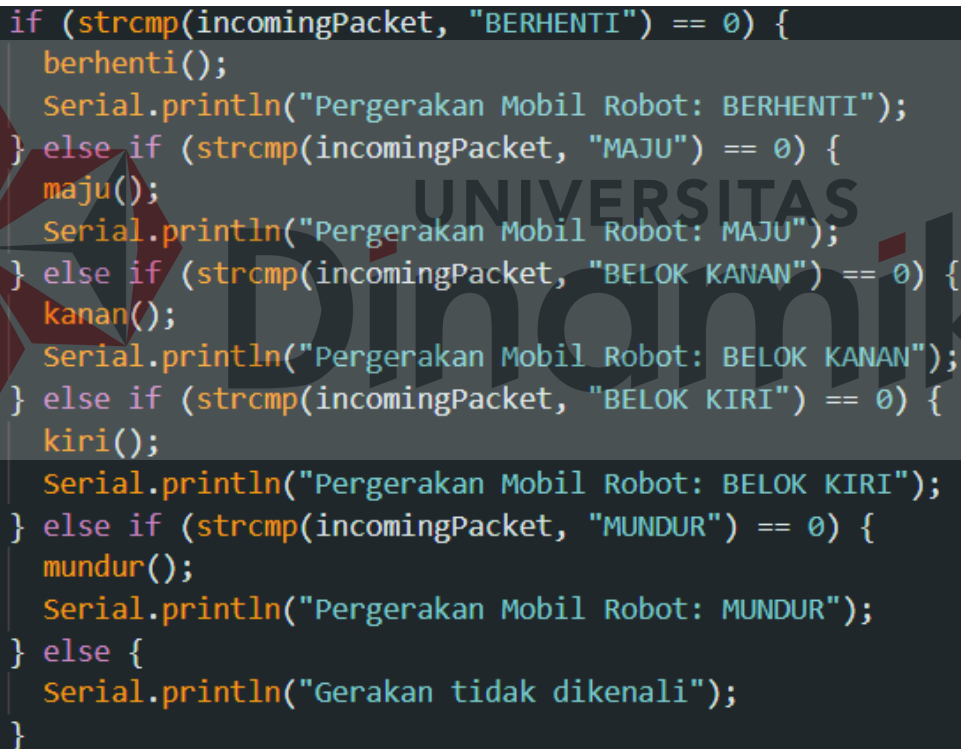
3.3 Perancangan Perangkat Lunak

Perancangan perangkat lunak pada sistem kontrol robot mobil melalui deteksi gestur jari tangan menggunakan MediaPipe melibatkan beberapa tahapan penting, termasuk pemrograman mikrokontroler, pengolahan citra untuk deteksi gestur, dan komunikasi data antara perangkat. Pertama, mikrokontroler ESP32 diprogram menggunakan Arduino IDE untuk mengendalikan motor DC pada robot mobil. Pin-pin yang terhubung ke driver motor diatur sebagai output, dan kode ditulis untuk menerima perintah dari komputer melalui WiFi dan mengatur status pin untuk mengendalikan arah dan kecepatan motor.


```
IPAddress local_IP(192, 168, 80, 100);
IPAddress gateway(192, 168, 80, 1);
IPAddress subnet(255, 255, 255, 0);
```

Gambar 3.4 *source code* IP statis

Pada program diatas untuk menjelaskan konfigurasi jaringan pada ESP32, pertama-tama kita menetapkan alamat IP statis untuk perangkat dengan menggunakan IP Address `local_IP(192, 168, 80, 100);`, yang menetapkan alamat IP 192.168.80.100. Selanjutnya, kita mengatur alamat gateway dan subnet mask dengan menggunakan IP Address `gateway(192, 168, 80, 1);` dan IP Address `subnet(255, 255, 255, 0);`



```
if (strcmp(incomingPacket, "BERHENTI") == 0) {
  berhenti();
  Serial.println("Pergerakan Mobil Robot: BERHENTI");
} else if (strcmp(incomingPacket, "MAJU") == 0) {
  maju();
  Serial.println("Pergerakan Mobil Robot: MAJU");
} else if (strcmp(incomingPacket, "BELOK KANAN") == 0) {
  kanan();
  Serial.println("Pergerakan Mobil Robot: BELOK KANAN");
} else if (strcmp(incomingPacket, "BELOK KIRI") == 0) {
  kiri();
  Serial.println("Pergerakan Mobil Robot: BELOK KIRI");
} else if (strcmp(incomingPacket, "MUNDUR") == 0) {
  mundur();
  Serial.println("Pergerakan Mobil Robot: MUNDUR");
} else {
  Serial.println("Gerakan tidak dikenali");
}
```

Gambar 3.5 Kontrol robot berdasarkan data yang diterima

Program di atas adalah bagian dari kode kontrol untuk sebuah mobil robot yang menerima perintah melalui paket data. Bagian kode ini menggunakan fungsi `strcmp` untuk membandingkan string `incomingPacket` dengan beberapa perintah yang telah ditentukan sebelumnya: "BERHENTI", "MAJU", "BELOK KANAN",

"BELOK KIRI", dan "MUNDUR".

Jika `incomingPacket` sama dengan "BERHENTI", fungsi `berhenti()` akan dipanggil, dan pesan "Pergerakan Mobil Robot: BERHENTI" akan dicetak ke Serial Monitor. Jika `incomingPacket` sama dengan "MAJU", fungsi `maju()` akan dipanggil, dan pesan "Pergerakan Mobil Robot: MAJU" akan dicetak. Jika `incomingPacket` sama dengan "BELOK KANAN", fungsi `kanan()` akan dipanggil, dan pesan "Pergerakan Mobil Robot: BELOK KANAN" akan dicetak. Jika `incomingPacket` sama dengan "BELOK KIRI", fungsi `kiri()` akan dipanggil, dan pesan "Pergerakan Mobil Robot: BELOK KIRI" akan dicetak. Jika `incomingPacket` sama dengan "MUNDUR", fungsi `mundur()` akan dipanggil, dan pesan "Pergerakan Mobil Robot: MUNDUR" akan dicetak. Jika `incomingPacket` tidak cocok dengan salah satu perintah tersebut, pesan "Gerakan tidak dikenali" akan dicetak ke Serial Monitor. Kode ini memastikan bahwa robot dapat menerima dan mengeksekusi perintah untuk mengontrol gerakannya sesuai dengan perintah yang dikirimkan.

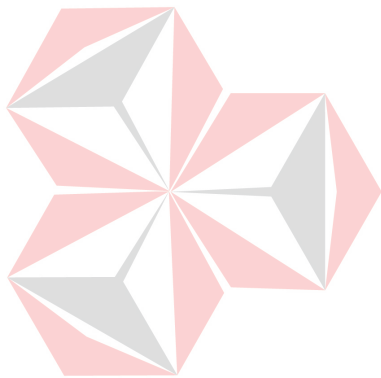
Selanjutnya, deteksi gestur tangan dilakukan menggunakan MediaPipe, sebuah framework dari Google untuk pemrosesan gambar dan video secara real-time. Skrip Python ditulis menggunakan Visual Studio Code untuk menangkap video dari kamera dan menganalisisnya. MediaPipe digunakan untuk mendeteksi posisi jari dan tangan dalam frame video, dan gestur tertentu dipetakan ke perintah yang akan dikirim ke mikrokontroler. Setelah gestur dikenali, perintah yang sesuai dikirimkan ke mikrokontroler ESP32 melalui WiFi, memanfaatkan kemampuan konektivitas WiFi ESP32. Skrip Python di komputer mengirimkan perintah melalui jaringan WiFi ke ESP32, yang kemudian menerima dan mengeksekusi perintah tersebut untuk menggerakkan motor sesuai dengan gestur yang terdeteksi. Alur kerja ini melibatkan inisialisasi ESP32 dan komputer untuk komunikasi WiFi, pengambilan gambar oleh kamera, analisis gestur dengan MediaPipe, pengiriman perintah ke ESP32, dan eksekusi perintah oleh ESP32 untuk mengontrol motor. Dengan integrasi MediaPipe untuk deteksi gestur dan penggunaan mikrokontroler ESP32 untuk kontrol motor, sistem ini memungkinkan robot mobil dikendalikan secara intuitif melalui gerakan tangan, memanfaatkan kombinasi Arduino IDE dan Visual Studio Code untuk fleksibilitas dan efisiensi dalam pengembangan serta pemeliharaan sistem. Berikut adalah program pendeteksian gestur jari tangan agar

jari tangan stabil.

```
gestureBuffer = []
current_gesture = None
previous_gesture = None
gestureCounter = 0
```

Gambar 3.6 Inisialisai variabel pada kestabilan gestur

Program di atas bertujuan untuk mendeteksi dan mengidentifikasi gerakan tangan menggunakan kamera. Program dimulai dengan inisialisasi beberapa variabel: `gestureBuffer` untuk menyimpan data jumlah jari yang terangkat pada setiap frame, `current_gesture` dan `previous_gesture` untuk menyimpan jenis gerakan tangan saat ini dan sebelumnya, serta `gestureCounter` untuk menghitung berapa kali jumlah jari yang terangkat tetap konsisten.



```
while True:
    success, img = cap.read()
    if not success:
        break

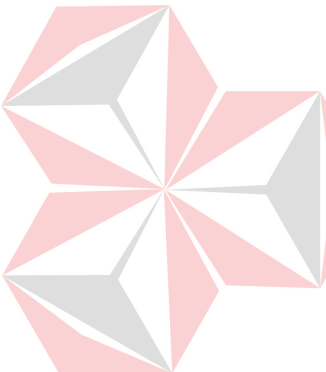
    landmarks = detector.findHandLandmarks(img, draw=True)
    if landmarks:
        fingers = detector.fingersUp(landmarks)
        total = fingers.count(1)

        if gestureCounter == 0:
            gestureBuffer = [total]
            gestureCounter += 1
        else:
            if total == gestureBuffer[-1]:
                gestureBuffer.append(total)
                gestureCounter += 1
                if gestureCounter == 10:
                    current_gesture = gestureBuffer[0]
                    gestureCounter = 0
                    gestureBuffer = []
            else:
                gestureCounter = 1
                gestureBuffer = [total]
```

Gambar 3.7 Source code pengecekan gestur jari

Kemudian, program memasuki loop utama yang akan terus berjalan selama kamera berhasil membaca gambar. Di dalam loop, gambar diambil dari kamera dan diproses untuk mendeteksi landmark tangan dengan metode `findHandLandmarks`. Jika landmark tangan terdeteksi, program akan menghitung jumlah jari yang terangkat.

Jika `gestureCounter` bernilai 0, artinya ini adalah pengecekan pertama, dan `gestureBuffer` diisi dengan jumlah jari yang terangkat saat ini. `gestureCounter` kemudian diinkrementasi. Jika tidak, program memeriksa apakah jumlah jari yang terangkat sama dengan nilai terakhir dalam `gestureBuffer`. Jika ya, nilai tersebut ditambahkan ke `gestureBuffer` dan `gestureCounter` diinkrementasi. Jika `gestureCounter` mencapai 10, berarti telah dilakukan 10 pengecekan berturut-turut dengan jumlah jari yang sama tanpa perubahan, sehingga `current_gesture` diatur ke nilai pertama dari `gestureBuffer`, `gestureCounter` direset ke 0, dan `gestureBuffer` dikosongkan. Jika jumlah jari berubah, `gestureCounter` direset menjadi 1, dan `gestureBuffer` diisi ulang dengan jumlah jari yang baru terdeteksi.



```

if previous_gesture != current_gesture:
    stable_counter = 0
    while stable_counter < 10:
        success, img = cap.read()
        landmarks = detector.findHandLandmarks(img, draw=False)
        if landmarks:
            fingers = detector.fingersUp(landmarks)
            total = fingers.count(1)
            if total == current_gesture:
                stable_counter += 1
            else:
                break
        else:
            break
    if stable_counter == 10:
        sock.sendto(gesture_text.encode(), (esp_ip, esp_port))
        previous_gesture = current_gesture

```

Gambar 3.8 *Source code* pengiriman data

Bagian kode ini melanjutkan proses deteksi gerakan tangan dengan memeriksa kestabilan gerakan yang terdeteksi. Setelah memverifikasi bahwa `current_gesture` berbeda dari `previous_gesture`, kode ini berusaha memastikan bahwa gerakan tersebut stabil sebelum mengirimkan informasi lebih lanjut.

Pertama, jika `previous_gesture` tidak sama dengan `current_gesture`, maka `stable_counter` diinisialisasi ke 0. Program kemudian memasuki loop yang memeriksa kestabilan gerakan selama 10 iterasi. Di setiap iterasi, gambar diambil dari kamera dan landmark tangan dideteksi tanpa menggambar (parameter `draw=False`). Jika landmark terdeteksi, jumlah jari yang terangkat dihitung. Jika jumlah jari yang terangkat sama dengan `current_gesture`, `stable_counter`

diinkrementasi. Jika tidak sama atau jika landmark tidak terdeteksi, loop dihentikan.

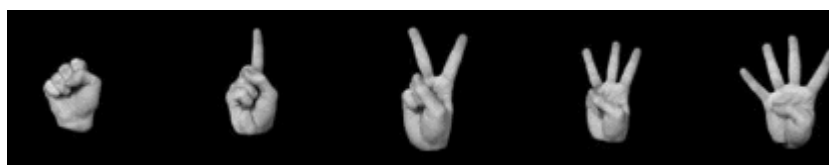
Jika setelah 10 iterasi `stable_counter` mencapai nilai 10, itu berarti `current_gesture` telah stabil dan konsisten selama periode tersebut. Dalam hal ini, informasi mengenai gerakan, yang disimpan dalam variabel `gesture_text`, dikirim melalui soket ke alamat IP dan port yang ditentukan oleh `esp_ip` dan `esp_port`. Kemudian, `previous_gesture` diperbarui menjadi `current_gesture` untuk persiapan deteksi gerakan berikutnya.

3.4 Instalasi Environment

Sebelum melakukan proses pengerjaan lebih lanjut, perlu melakukan instalasi dan pengaturan *environment*. Proses tersebut dilakukan supaya library – library yang dibutuhkan oleh program dapat terpasang dengan baik dan program bisa berjalan tanpa kendala. Pemasangan tersebut dilakukan dengan cara menginstal Python versi 3.7.0 dan *plugin* yang dibutuhkan pada terminal pip Visual Studio Code yaitu MediaPipe, OpenCV, *Socket* dan *Time*. Untuk menyimpan data yang diterima dari komunikasi UDP dalam Python menggunakan tipe data string, Anda bisa menggunakan *socket* UDP untuk menerima data dan kemudian menyimpannya dalam sebuah variabel string.

3.5 Dataset

Dalam penulisan Tugas Akhir ini penulis menggunakan dataset gestur jari tangan yang sudah disediakan oleh mediapipe itu sendiri menggunakan model-model yang sudah siap pakai tanpa memerlukan pelatihan ulang. Dengan demikian, pengguna dapat mengintegrasikan kemampuan deteksi, analisis pose, dan berbagai tugas computer vision lainnya dengan mudah, tanpa perlu menghabiskan waktu dan sumber daya untuk mengumpulkan atau melatih dataset secara mandiri. Serta memastikan implementasi solusi visual seperti deteksi objek atau analisis pose dapat dilakukan dengan akurasi tinggi dan efisiensi yang optimal.



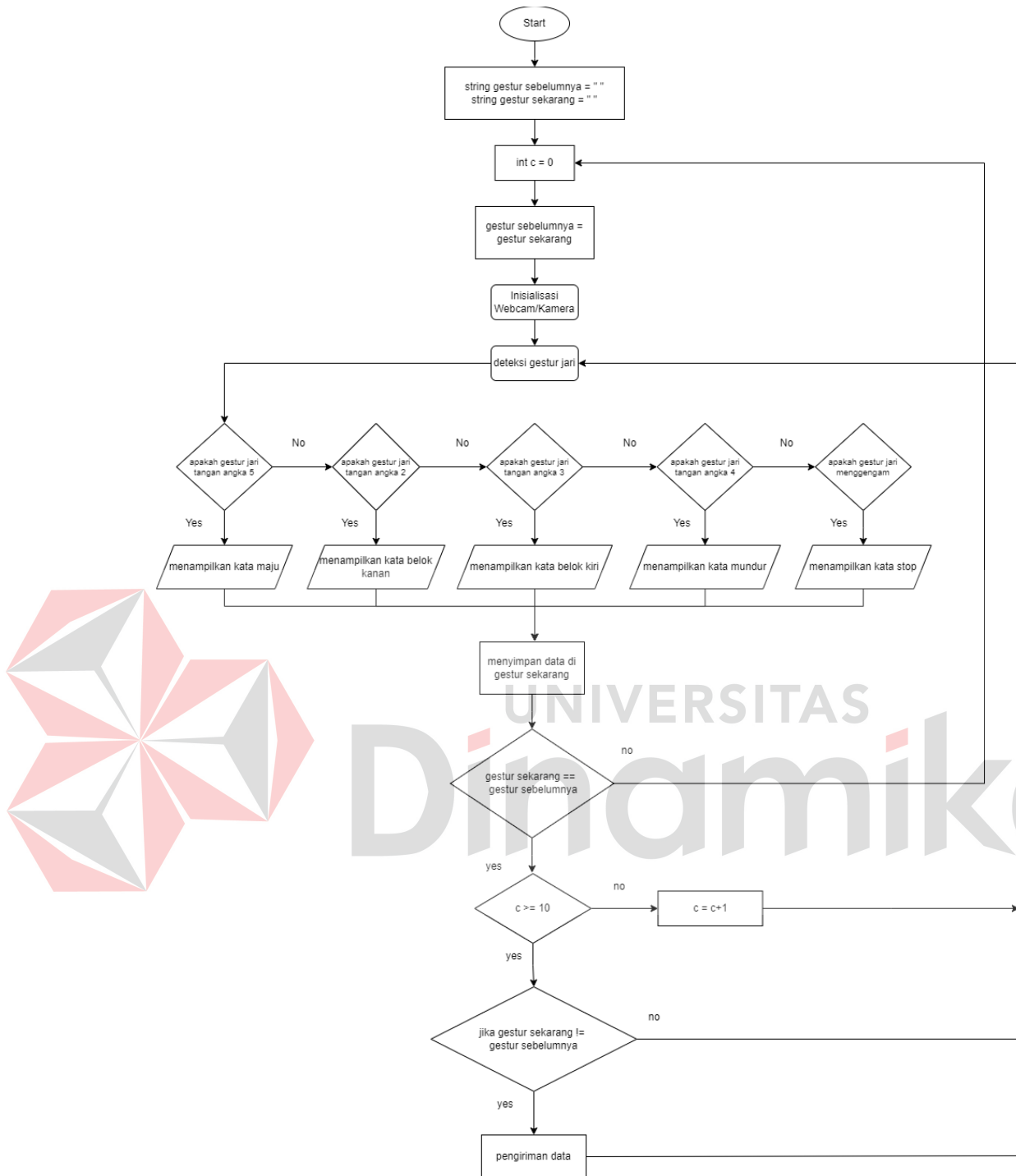
Gambar 3.9 Dataset Gestur Jari Tangan
(Pratama et al., 2023)

3.6 Metode Deteksi Dengan Mediapipe

Metode deteksi dengan menggunakan Mediapipe mencari gestur jari tangan menggunakan Webcam/Kamera. Ketika program mendeteksi gestur jari tangan, program akan menampilkan *landmark* yang digunakan untuk mengidentifikasi gestur jari tangan. Pengimplementasian metode Mediapipe untuk klasifikasi gestur jari tangan dan ekstraksi *landmark* Mediapipe. Ketika gestur dikenali, program menunjukkan jari tangan yang terdeteksi, FPS yang di dapat pada setiap gestur jari tangan. Pengujian jarak dimulai dari 30cm, 80cm, 130cm, dan 180cm. Kemudian dihitung rata-rata akurasi *error* pada setiap pendeteksian.

3.7 Flowchart Python

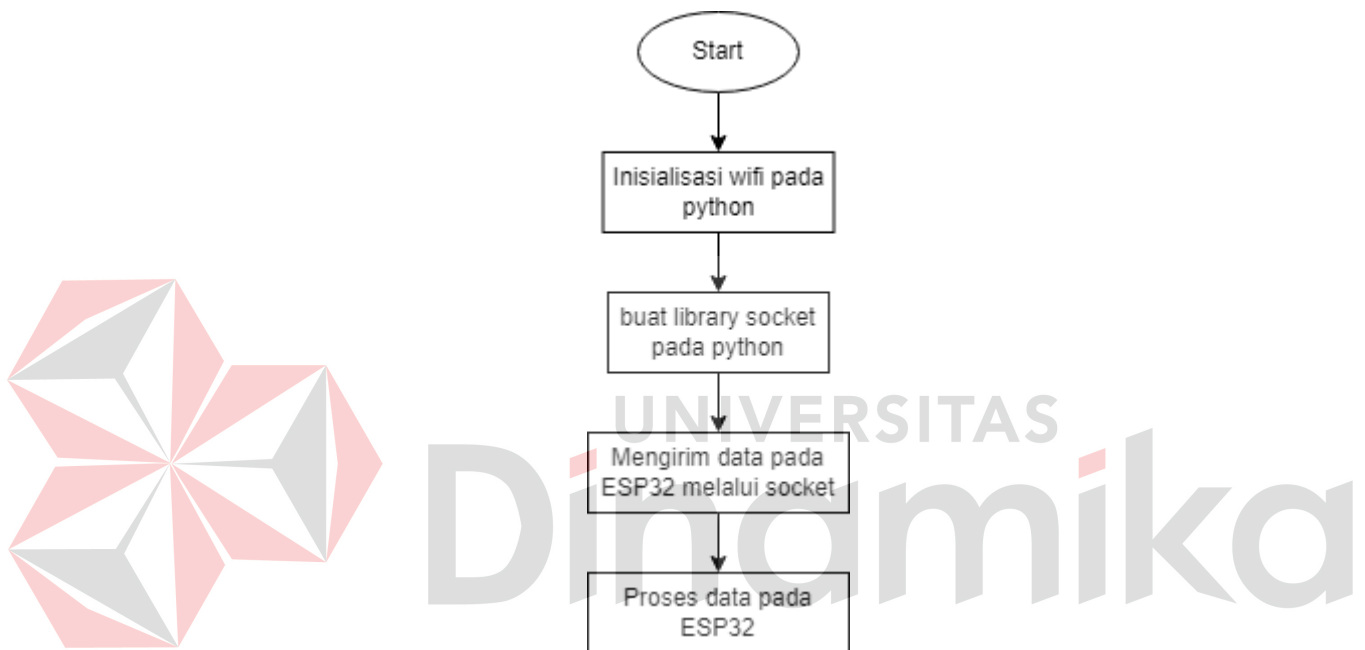
Pada Gambar 10 dijelaskan bahwa inputnya berasal dari webcam yang mendeteksi gestur jari tangan kanan yaitu mendeteksi gestur jari tangan 2, 3, 4, 5 dan jari menggengam sebagai penggerak Robot Mobil. Setelah mendapatkan nilai yang terdeteksi, kemudian ditampilkan hasilnya di layar sebagai indikasi telah terdeteksi gestur jari tangan. Ketika gestur jari tangan menunjukkan angka 2, maka pada layar akan menampilkan kata maju, ketika gestur jari tangan menunjukkan angka 3, maka pada layar akan menampilkan kata belok kanan, ketika gestur jari tangan menunjukkan angka 4, maka pada layar akan menampilkan kata belok kanan, ketika gestur jari tangan menunjukkan angka 5, maka pada layar akan menampilkan kata mundur, dan ketika gestur jari menggengam, maka pada layar akan menampilkan kata stop. Setelah itu, melakukan pengecekan gestur jari selama 10 kali ketika gestur jari tangan sudah stabil maka data gestur jari tangan akan di kirimkan melalui komunikasi UDP, dan jika gestur jari tangan yang terdeteksi sebelumnya sama dengan gestur jari tangan yang sekarang data tidak akan di kirimkan sebaliknya, jika gestur jari tangan sekarang tidak sama dengan gestur jari sebelumnya maka data akan dikirimkan kepada ESP.



Gambar 3.10 Flowchart Python

3.8 Flowchart Pengiriman Data

Langkah pertama dalam proses pengiriman data dari Python ke ESP32 secara *wireless* melalui koneksi Wi-Fi dimulai dengan inialisasi koneksi Wi-Fi pada Python. Setelah itu, Python membuat objek *socket* dan menghubungkannya ke alamat IP dan *port* ESP32. Selanjutnya, Python mengirimkan data melalui *socket* yang telah dibuat kepada ESP32. ESP32 menerima data melalui koneksi Wi-Fi dari Python, dan kemudian melakukan proses terhadap data yang diterima untuk menggerakkan robot mobil.

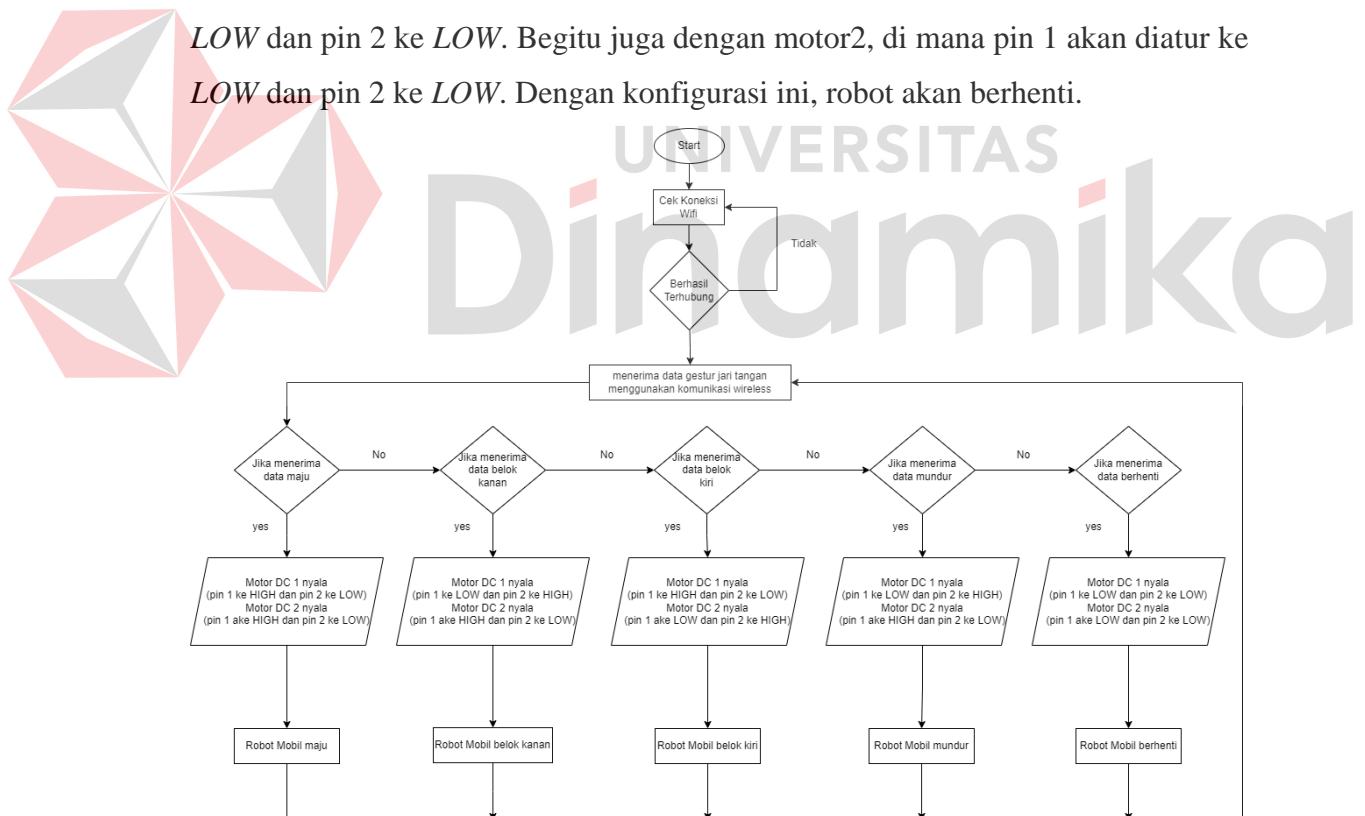


Gambar 3.11 Flowchart Pengiriman Data

3.9 Flowchart ESP32

Berdasarkan Gambar 3.6, mikrokontroler ESP32 menerima data gestur jari tangan, Setelah itu, melakukan pengecekan kondisi, Jika sistem kontrol robot mobil menerima data dengan perintah "maju", maka robot akan mengatur motornya untuk bergerak maju. Konfigurasi pin untuk motor1 akan diatur sebagai berikut: pin 1 ke *HIGH* dan pin 2 ke *LOW*. Begitu juga dengan motor2, di mana pin 1 akan diatur ke *HIGH* dan pin 2 ke *LOW*. Dengan konfigurasi ini, robot akan bergerak maju. Jika sistem kontrol robot mobil menerima data dengan perintah "belok kanan", maka robot akan mengatur motornya untuk bergerak kekanan. Konfigurasi pin untuk motor1 akan diatur sebagai berikut: pin 1 ke *LOW* dan pin 2 ke *HIGH*. Begitu juga

dengan motor2, di mana pin 1 akan diatur ke *HIGH* dan pin 2 ke *LOW*. Dengan konfigurasi ini, robot akan berputar kekanan. Jika sistem kontrol robot mobil menerima data dengan perintah "belok kiri", maka robot akan mengatur motornya untuk bergerak kekiri. Konfigurasi pin untuk motor1 akan diatur sebagai berikut: pin 1 ke *HIGH* dan pin 2 ke *LOW*. Begitu juga dengan motor2, di mana pin 1 akan diatur ke *LOW* dan pin 2 ke *HIGH*. Dengan konfigurasi ini, robot akan berputar kekiri. Jika sistem kontrol robot mobil menerima data dengan perintah "mundur", maka robot akan mengatur motornya untuk bergerak mundur. Konfigurasi pin untuk motor1 akan diatur sebagai berikut: pin 1 ke *LOW* dan pin 2 ke *HIGH*. Begitu juga dengan motor2, di mana pin 1 akan diatur ke *LOW* dan pin 2 ke *HIGH*. Dengan konfigurasi ini, robot akan bergerak mundur. Dan jika sistem kontrol robot mobil menerima data dengan perintah "berhenti", maka robot akan mengatur motornya untuk berhenti. Konfigurasi pin untuk motor1 akan diatur sebagai berikut: pin 1 ke *LOW* dan pin 2 ke *LOW*. Begitu juga dengan motor2, di mana pin 1 akan diatur ke *LOW* dan pin 2 ke *LOW*. Dengan konfigurasi ini, robot akan berhenti.



Gambar 3.12 Flowchart Mikrokontroler ESP32

BAB IV

HASIL DAN PEMBAHASAN

Pada Bab 4 ini, membahas mengenai analisis dan pengujian alat serta perangkat lunak pada sistem Robot Mobil dengan MediaPipe. MediaPipe adalah *framework open-source* yang dirancang untuk memfasilitasi pembangunan aplikasi berbasis pemrosesan media dan *machine learning*. Penggunaan MediaPipe dalam sistem robot mobil memungkinkan deteksi, pelacakan, dan analisis gerakan secara *real-time*, yang sangat penting untuk navigasi dan interaksi robot dengan lingkungannya. Analisis ini akan mencakup evaluasi performa MediaPipe dalam mengidentifikasi objek dan gerakan, serta bagaimana integrasinya dengan perangkat keras robot mobil. Selain itu, pengujian sistem akan dilakukan untuk menilai akurasi MediaPipe dalam kontrol Robot Mobil. Hasil analisis dan pengujian ini diharapkan dapat memberikan wawasan mendalam mengenai efektivitas penggunaan MediaPipe dalam robot mobil.

4.1 Pengujian Pengiriman dan Penerimaan Data menggunakan WiFi

4.1.1 Tujuan Uji Pengiriman dan Penerimaan Data Menggunakan WiFi

Tujuan menggunakan WiFi dalam pengiriman data MediaPipe adalah untuk memastikan transfer data yang cepat dan stabil antara sistem robot mobil dan perangkat pemrosesan atau kontrol. Dengan menggunakan WiFi, data video dan informasi gerakan yang dihasilkan oleh MediaPipe dapat dikirimkan secara *real-time*. Ini penting untuk memastikan bahwa robot mobil dapat merespon lingkungan sekitarnya dengan cepat dan akurat. Selain itu, WiFi memungkinkan komunikasi nirkabel yang fleksibel, memungkinkan robot mobil untuk bergerak bebas tanpa dibatasi oleh kabel.

4.1.2 Perlengkapan Untuk Pengujian dan Penerimaan Data Menggunakan WiFi

Berikut ini adalah perlengkapan untuk melakukan percobaan pengiriman dan penerimaan data menggunakan WiFi:

1. Mikrokontroler ESP32

2. Laptop untuk melakukan *running* program Python dan program ESP32
3. Koneksi WiFi

4.1.3 Cara Pengujian

Berikut ini adalah langkah-langkah melakukan pengujian:

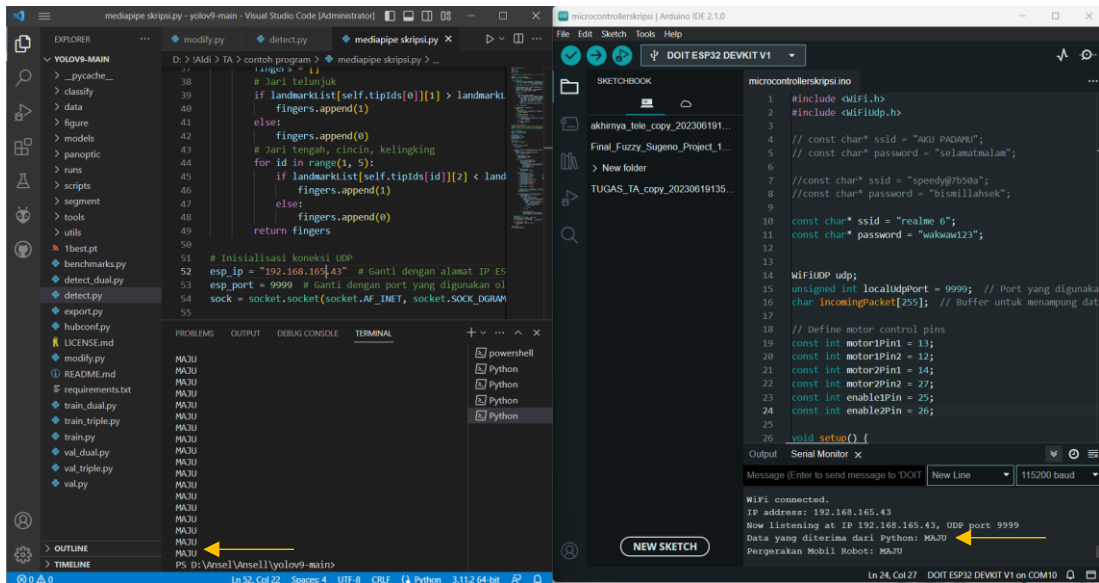
1. Menghubungkan ESP32 dengan laptop .
2. Menjalankan program di Arduino IDE untuk mendapatkan IP ESP32.
3. Menyesuaikan IP ESP32 yang di dapatkan pada Arduino IDE pada program computer vision dan jalankan program.
4. Membandingkan data yang dikirim oleh program Python dengan data yang diterima oleh program Arduino IDE apakah sesuai dengan yang di kirimkan.

4.1.4 Hasil Uji



Gambar 4.1 Hasil Deteksi Program Python

Pada Gambar 4.1 jari tangan terdeteksi akan diproses oleh MediaPipe dan dikirimkan ke ESP32 melalui wifi untuk mengendalikan Robot Mobil. Output yang dikirimkan berupa gestur jari tangan yang terdeteksi seperti Gambar 4.1 diatas.



Gambar 4. 2 Output pada Serial Monitor dan Terminal Program Python

Pada Gambar 4.2 hasil pengujian pada program sistem Robot Mobil dengan IP yang sudah di tentukkan mendapatkan data dari deteksi gestur jari tangan kemudian dikirimkan kepada ESP32. Data yang dikirimkan oleh program Python dan diterima oleh program ESP32 sudah sesuai untuk menjalankan gerakan Robot Mobil.

Tabel 4.1 Pengujian komunikasi secara wireless

Percobaan ke-	Gestur Jari Tangan	Komunikasi
1	2	Berhasil
2	Menggengam	Berhasil
3	5	Berhasil
4	3	Berhasil
5	4	Berhasil
6	Menggengam	Berhasil
7	2	Berhasil
8	Menggengam	Berhasil
9	5	Berhasil
10	3	Berhasil
11	2	Berhasil
12	3	Berhasil
13	Menggengam	Berhasil
14	5	Berhasil
15	3	Berhasil
16	4	Berhasil

17	2	Berhasil
18	5	Berhasil
19	Menggengam	Berhasil
20	4	Berhasil
21	5	Berhasil
22	3	Berhasil
23	2	Berhasil
24	3	Berhasil
25	Menggengam	Berhasil
26	5	Berhasil
27	3	Berhasil
28	4	Berhasil
29	2	Berhasil
30	5	Berhasil

Berdasarkan data percobaan yang tersedia, sistem pengenalan gestur jari tangan menunjukkan kinerja yang sangat baik. Dari total 30 kali percobaan yang dilakukan, semuanya berhasil dalam mengirimkan data, sehingga persentase keberhasilan mencapai 100%. Frekuensi kemunculan setiap gestur yang diuji adalah sebagai berikut: gestur jari tangan 5 dan 3 masing-masing muncul sebanyak 7 kali, gestur menggengam dan 2 diuji 6 kali, sementara gestur jari 4 diuji sebanyak 4 kali tetapi juga berhasil.

Distribusi hasil ini mengindikasikan bahwa sistem dapat mengenali berbagai gestur dengan konsisten tanpa adanya kegagalan komunikasi. Meskipun ada variasi dalam percobaan pengiriman data, sistem dapat memproses semua gestur yang diuji secara akurat. Tidak ada indikasi masalah dalam pengenalan gestur pada data yang ada untuk di kirim ke ESP32.

4.2 Pengujian Kontrol Robot Mobil Menggunakan MediaPipe

4.2.1 Tujuan Pengujian Kontrol Robot Mobil Menggunakan MediaPipe

Tujuannya adalah mengontrol Robot Mobil bergerak maju, belok kanan, belok kiri, mundur, dan berhenti. Pengujian ini dilakukan oleh 3 orang untuk mendeteksi gestur jari tangan, setiap gestur jari tangan akan dilakukan 5 kali percobaan apakah sudah sesuai perintah dengan aksi yang dilakukan Robot Mobil dan apakah program ini sudah *real-time* dan mendapat akurasi yang baik.

4.2.2 Perlengkapan Untuk Pengujian Kontrol Robot Mobil Menggunakan MediaPipe

Berikut ini adalah perlengkapan untuk melakukan percobaan kontrol Robot Mobil dengan menggunakan MediaPipe:

1. Robot Mobil
2. Laptop untuk melakukan *running* program Python dan program ESP32
3. Koneksi WiFi

4.2.3 Cara Pengujian

Berikut adalah langkah-langkah dalam pengujian kontrol Robot Mobil dengan menggunakan MediaPipe:

1. Mendeteksi gestur jari tangan
2. Mengirimkan data gestur jari tangan kepada ESP32
3. Memastikan pergerakan Robot Mobil sesuai dengan data yang sudah di kirimkan program Python

4.2.4 Hasil Uji Kontrol Robot Mobil Menggunakan MediaPipe



Gambar 4.3 Deteksi Gestur Jari Tangan 5

Berdasarkan Gambar pada 4.3 pengujian kontrol Robot Mobil menggunakan MediaPipe berjalan sesuai tujuan peneliti. Gestur jari tangan menunjukkan gestur jari tangan 5 dengan fps sebesar 13.52, Robot Mobil bergerak maju kedepan berdasarkan penerimaan data dari program Python. Hasil MediaPipe sesuai dengan gestur jari tangan yang terdeteksi

Tabel 4.2 Pengujian Kontrol Robot Mobil Dengan MediaPipe

Nama	Gestur Jari Tangan	Percobaan ke-	Status Robot Mobil	FPS	Hasil Deteksi		Akurasi (%)
					Benar	Salah	
Aldi	5	1	Maju	13.52	✓		100%
		2		13.05	✓		
		3		13.15	✓		
		4		13.28	✓		
		5		13.35	✓		
	2	Belok Kanan	1	13.52	✓		100%
			2	13.64	✓		
			3	13.72	✓		
			4	13.74	✓		
			5	13.76	✓		
	3	Belok Kiri	1	13.52	✓		100%
			2	13.61	✓		
			3	13.70	✓		
			4	13.75	✓		
			5	13.74	✓		
	4	Mundur	1	13.80	✓		100%
			2	13.76	✓		
			3	13.73	✓		
			4	13.73	✓		
			5	13.70	✓		
Menggengam	Berhenti	1	13.04	✓		100%	
		2	13.05	✓			
		3	13.12	✓			
		4	13.25	✓			
		5	13.32	✓			
Ibnu	5	1	Maju	14.04	✓		100%
		2		14.10	✓		
		3		14.15	✓		
		4		14.19	✓		
		5		14.22	✓		
	2	Belok Kanan	1	14.31	✓		100%
			2	14.30	✓		

		3		14.33	✓			
		4		14.33	✓			
		5		14.36	✓			
	3		1	Belok Kiri	14.30	✓		100%
			2		14.31	✓		
			3		14.32	✓		
			4		14.33	✓		
			5		14.35	✓		
	4		1	Mundur	14.30	✓		100%
			2		14.32	✓		
			3		14.31	✓		
			4		14.34	✓		
			5		14.34	✓		
	Menggengam		1	Berhenti	14.04	✓		100%
			2		14.09	✓		
3			14.14		✓			
4			14.18		✓			
5			14.21		✓			
Zidha	5	1	Maju	14.57	✓		100%	
		2		14.59	✓			
		3		14.60	✓			
		4		14.61	✓			
		5		14.63	✓			
	2		1	Belok Kanan	14.63	✓		100%
			2		15.11	✓		
			3		15.11	✓		
			4		15.12	✓		
			5		15.11	✓		
	3		1	Belok Kiri	14.63	✓		100%
			2		15.11	✓		
			3		15.12	✓		
			4		15.12	✓		
			5		15.11	✓		
	4		1	Mundur	14.62	✓		100%
			2		15.10	✓		
			3		15.12	✓		
			4		15.11	✓		
			5		15.10	✓		
Menggengam		1	Berhenti	14.56	✓		100%	
		2		14.58	✓			
		3		14.59	✓			
		4		14.61	✓			
		5		14.62	✓			
FPS rata-rata gestur jari 2				14.34	Akurasi rata-rata gestur jari 2	100%		
FPS rata-rata gestur jari 3				14.33	Akurasi rata-rata gestur jari 3	100%		

FPS rata-rata gestur jari 4	14.35	Akurasi rata-rata gestur jari 4	100%
FPS rata-rata gestur jari 5	14.00	Akurasi rata-rata gestur jari 5	100%
FPS rata-rata gestur jari menggenggam	13.96	Akurasi rata-rata gestur jari menggenggam	100%

Berdasarkan Tabel 4.1 analisis hasil percobaan kontrol robot mobil dengan gestur jari tangan menggunakan MediaPipe, sistem ini bekerja sangat baik dan konsisten. Sistem ini dapat mengenali semua gestur (5 jari, 2 jari, 3 jari, 4 jari, dan menggenggam) dengan akurasi 100%, artinya tidak ada kesalahan dalam mengenali gestur. Kecepatan pemrosesan sistem juga tinggi, dengan frame per second (FPS) rata-rata antara 13.96 hingga 14.35, menunjukkan bahwa sistem dapat mendeteksi gestur dengan cepat. Percobaan dilakukan oleh tiga pengguna yang berbeda, yaitu Aldi, Ibnu, dan Zidha, dan hasilnya tetap konsisten. Sistem ini sangat andal, mampu mengenali gestur dengan tepat dan akurasi yang sempurna, sistem ini efektif untuk mengontrol robot mobil, memastikan respons yang akurat terhadap berbagai gestur tangan.

4.3 Pengujian Perbandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan

4.3.1 Tujuan Pengujian Perbandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan

Tujuan pengujian perbandingan program pengecekan deteksi gestur jari tangan dan tanpa pengecekan deteksi gestur jari tangan adalah untuk mengevaluasi perbedaan kinerja dan efisiensi sistem dalam kedua kondisi tersebut. Seberapa pengaruh pada FPS dan hasil deteksi gestur jari tangan untuk menggerakkan Robot Mobil.

4.3.2 Perlengkapan Pengujian Perbandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan

Berikut ini adalah perlengkapan untuk melakukan percobaan perbandingan pengecekan program deteksi gestur jari tangan dan tanpa pengecekan deteksi gestur jari tangan untuk mengontrol Robot Mobil dengan menggunakan MediaPipe:

1. Robot Mobil
2. Laptop untuk melakukan *running* program Python dan program ESP32
3. Koneksi WiFi

4.3.3 Cara Pengujian

Berikut adalah langkah-langkah dalam pengujian perbandingan pengecekan program deteksi gestur jari tangan dan tanpa pengecekan deteksi gestur jari tangan untuk mengontrol Robot Mobil dengan menggunakan MediaPipe:

1. Menyiapkan program deteksi gestur jari tangan tanpa pengecekan dan pengecekan
2. Mendeteksi gestur jari tangan
3. Mengirimkan data gestur jari tangan kepada ESP32
4. Memastikan pergerakan Robot Mobil sesuai dengan data yang sudah di kirimkan program Python

4.3.4 Hasil Pengujian Pebandingan Program Pengecekan Deteksi Gestur Jari Tangan dan Tanpa Pengecekan Deteksi Gestur Jari Tangan

Tabel 4.3 Pengujian tanpa pengecekan deteksi gestur jari tangan

Nama	Gestur Jari Tangan	Percobaan ke-	Status Robot Mobil	FPS	Hasil Deteksi		Akurasi (%)
					Benar	Salah	
Aldi	5	1	Maju	17.01	✓		80%
		2		16.87	✓		
		3		16.56	✓		
		4		16.85		✓	
		5		19.80	✓		
	2	Belok Kanan	1	14.76	✓		100%
			2	17.02	✓		
			3	19.16	✓		
			4	19.69	✓		
			5	16.69	✓		

	3	1	Belok Kiri	19.09	✓		40%
		2		16.99		✓	
		3		16.52		✓	
		4		16.56		✓	
		5		16.69	✓		
	4	1	Mundur	16.78		✓	60%
		2		16.91		✓	
		3		14.68	✓		
		4		15.84	✓		
		5		16.82	✓		
	Menggengam	1	Berhenti	16.90	✓		60%
		2		16.88	✓		
		3		17.05		✓	
		4		14.99		✓	
		5		13.46		✓	

Tabel 4.4 pengujian dengan menggunakan pengecekan deteksi gestur jari tangan

Nama	Gestur Jari Tangan	Percobaan ke-	Status Robot Mobil	FPS	Hasil Deteksi		Akurasi (%)
					Benar	Salah	
Aldi	5	1	Maju	13.52	✓		100%
		2		13.05	✓		
		3		13.15	✓		
		4		13.28	✓		
		5		13.35	✓		
	2	1	Belok Kanan	13.52	✓		100%
		2		13.64	✓		
		3		13.72	✓		
		4		13.74	✓		
		5		13.76	✓		
	3	1	Belok Kiri	13.52	✓		100%
		2		13.61	✓		
		3		13.70	✓		
		4		13.75	✓		
		5		13.74	✓		
	4	1	Mundur	13.80	✓		100%
		2		13.76	✓		
		3		13.73	✓		
		4		13.73	✓		
		5		13.70	✓		
Menggengam	1	Berhenti	13.04	✓		100%	
	2		13.05	✓			
	3		13.12	✓			
	4		13.25	✓			
	5		13.32	✓			

Pengujian deteksi gestur jari tangan untuk mengontrol robot mobil dilakukan dengan dua metode yang berbeda. Metode pertama tanpa pengecekan ulang menunjukkan hasil yang bervariasi dalam hal akurasi dan stabilitas FPS. Pada gestur maju, metode ini menghasilkan akurasi 80%, dengan satu dari lima percobaan menghasilkan deteksi yang salah dan rata-rata FPS sekitar 17.4. Gestur belok kanan memiliki akurasi sempurna 100% dengan rata-rata FPS sekitar 17.26, menunjukkan reliabilitas yang tinggi dalam deteksi gestur ini. Namun, gestur belok kiri hanya mencapai akurasi 40%, dengan tiga dari lima percobaan salah, dan rata-rata FPS sekitar 17.17, menandakan bahwa deteksi pada gestur ini perlu ditingkatkan. Untuk gestur mundur, akurasi mencapai 60%, dengan dua dari lima percobaan salah dan rata-rata FPS sekitar 16.21. Begitu pula dengan gestur menggenggam (berhenti), yang juga memiliki akurasi 60% dan rata-rata FPS sekitar 16.06.

Di sisi lain, metode kedua dengan menggunakan pengecekan ulang menunjukkan hasil yang jauh lebih konsisten dan akurat. Semua gestur (maju, belok kanan, belok kiri, mundur, dan menggenggam) berhasil dideteksi dengan akurasi 100%. Meski FPS pada metode kedua lebih rendah, dengan rata-rata sekitar 13.27 untuk gestur maju, 13.68 untuk belok kanan, 13.66 untuk belok kiri, 13.74 untuk mundur, dan 13.16 untuk menggenggam, stabilitas FPS ini menunjukkan konsistensi yang lebih baik dibandingkan metode pertama.

Kesimpulannya, metode kedua lebih direkomendasikan untuk digunakan dalam pengontrolan robot mobil karena mampu memberikan akurasi deteksi gestur yang sangat tinggi dan konsisten. Meskipun FPS sedikit lebih rendah, stabilitas dan keandalan metode ini jauh lebih baik. Untuk peningkatan lebih lanjut, perlu dievaluasi kemungkinan meningkatkan FPS metode kedua tanpa mengurangi akurasi deteksi. Untuk melihat program python dengan pengecekan terdapat pada lampiran 1, dan untuk melihat program python tanpa pengecekan terdapat pada lampiran 2.

4.4 Pengujian Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe

4.4.1 Tujuan Pengujian Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe

Pengujian ini bertujuan untuk mengetahui jarak terbaik gestur jari tangan. Parameter yang digunakan untuk mengukur menggunakan meteran. Dimana jarak ditentukan dari 30cm, 80cm, 130cm, dan 180cm. Berdasarkan hasil pengukuran tersebut dapat diketahui akurasi sistem dalam merespon aksi yang diberikan.

4.4.2 Perlengkapan Untuk Pengujian Jarak Terbaik Mengontrol Robot Mobil Menggunakan MediaPipe

Berikut ini adalah perlengkapan untuk pengujian jarak terbaik mengontrol Robot Mobil dengan menggunakan MediaPipe:

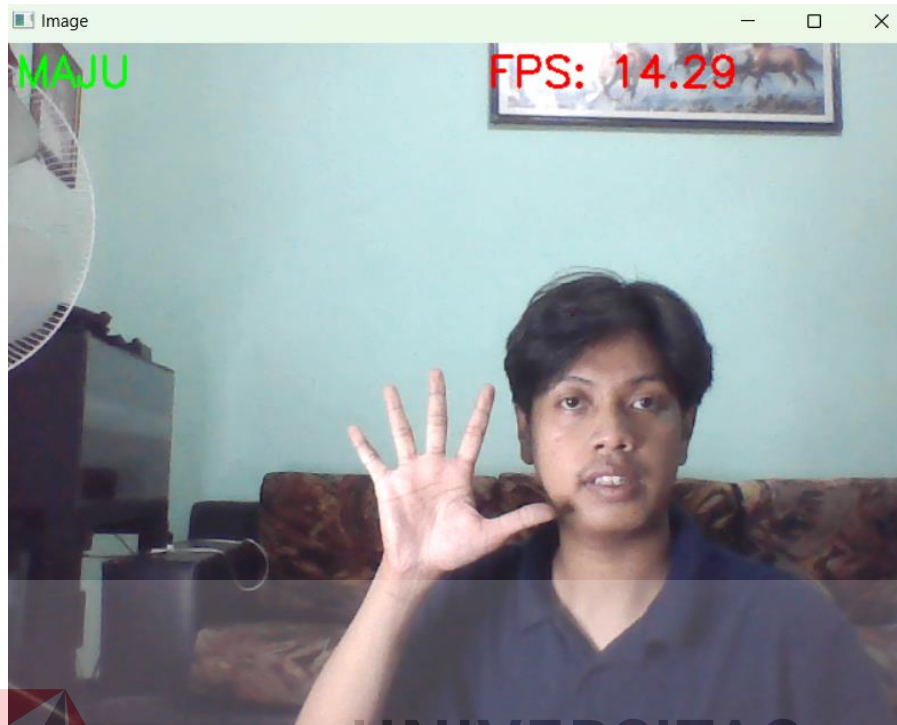
1. Robot Mobil.
2. Laptop untuk melakukan *running* program Python dan program ESP32.
3. Koneksi WiFi.

4.4.3 Cara Pengujian

Berikut ini adalah cara pengujian untuk menentukan jarak terbaik mengontrol Robot Mobil dengan menggunakan MediaPipe:

1. Menentukann jarak sesuai yang akan diuji.
2. Mendeteksi gestur jari tangan
3. Mengirimkan data gestur jari tangan pada ESP32
4. Memastikan pergerakan Robot Mobil sesuai dengan data yang sudah dikirimkan oleh program Python

4.4.4 Hasil Uji Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe



Gambar 4.4 Deteksi Gestur Jari Tangan 5 dengan jarak 80 cm

Berdasarkan pada Gambar 4.4, pengujian kontrol Robot Mobil menggunakan MediaPipe berjalan sesuai tujuan peneliti. Gestur jari tangan 5 yang diukur dengan jarak 80cm dari kamera menunjukkan fps sebesar 14.29 dan Robot Mobil bergerak maju kedepan berdasarkan penerimaan data dari program Python. Hasil MediaPipe sesuai dengan gestur jari tangan yang terdeteksi.

Tabel 4. 5 Pengujian Jarak Terbaik Untuk Mengontrol Robot Mobil Menggunakan MediaPipe

Jarak	Gestur jari tangan	Percobaan ke-	Status Robot Mobil	FPS	Hasil	Akurasi (%)	
30 cm	5	1	Maju	17.22	Benar	100%	
		2		16.88	Benar		
		3		16.40	Benar		
		4		15.77	Benar		
		5		15.26	Benar		
	2		1	Belok kanan	14.86	Benar	100%
			2		14.67	Benar	
			3		14.46	Benar	
			4		14.27	Benar	

		5		14.12	Benar	
	3	1	Belok kiri	14.82	Benar	100%
		2		14.62	Benar	
		3		14.47	Benar	
		4		14.23	Benar	
		5		14.10	Benar	
	4	1	Mundur	14.77	Benar	100%
		2		14.58	Benar	
		3		14.42	Benar	
		4		14.19	Benar	
		5		14.06	Benar	
	Menggengam	1	Berhenti	17.00	Benar	100%
		2		16.70	Benar	
		3		16.28	Benar	
		4		15.72	Benar	
		5		15.17	Benar	
80 cm	5	1	Maju	14.29	Benar	100%
		2		13.71	Benar	
		3		13.56	Benar	
		4		13.45	Benar	
		5		13.43	Benar	
	2	1	Belok kanan	13.65	Benar	100%
		2		13.59	Benar	
		3		13.52	Benar	
		4		13.44	Benar	
		5		13.45	Benar	
	3	1	Belok kiri	13.63	Benar	100%
		2		13.58	Benar	
		3		13.49	Benar	
		4		13.44	Benar	
		5		13.45	Benar	
	4	1	Mundur	13.63	Benar	80%
		2		13.58	Benar	
		3		13.49	Benar	
		4		13.44	Salah	
		5		13.44	Benar	
Menggengam	1	Berhenti	14.32	Benar	100%	
	2		13.70	Benar		
	3		13.47	Benar		
	4		13.43	Benar		
	5		13.43	Benar		
130 cm	5	1	Maju	13.45	Benar	80%
		2		13.42	Benar	
		3		13.33	Benar	
		4		13.28	Benar	
		5		13.24	Benar	

180 cm	2	1	Belok kanan	13.43	Benar	100%
		2		13.42	Benar	
		3		13.32	Benar	
		4		13.28	Benar	
		5		13.23	Benar	
	3	1	Belok kiri	13.43	Benar	100%
		2		13.411	Benar	
		3		13.32	Benar	
		4		13.27	Benar	
		5		13.23	Benar	
	4	1	Mundur	13.43	Benar	80%
		2		13.41	Benar	
		3		13.31	Benar	
		4		13.27	Benar	
		5		13.23	Salah	
	Menggengam	1	Berhenti	13.44	Benar	60%
		2		13.39	Benar	
		3		13.30	Salah	
		4		13.27	Salah	
		5		13.23	Benar	
	5	1	Maju	13.07	Salah	20%
		2		13.05	Salah	
		3		13.00	Benar	
		4		12.93	Salah	
		5		12.91	Salah	
2	1	Belok kanan	12.96	Benar	80%	
	2		12.95	Benar		
	3		12.94	Salah		
	4		12.91	Benar		
	5		12.88	Benar		
3	1	Belok kiri	12.95	Benar	80%	
	2		12.94	Salah		
	3		12.91	Benar		
	4		12.89	Benar		
	5		12.85	Benar		
4	1	Mundur	12.94	Benar	60%	
	2		12.94	Salah		
	3		12.90	Salah		
	4		12.90	Salah		
	5		12.87	Benar		
Menggengam	1	Berhenti	13.02	Benar	60%	
	2		12.98	Benar		
	3		12.93	Benar		
	4		12.95	Salah		
	5		12.92	Salah		

Pada Tabel 4.2 berdasarkan hasil percobaan pengenalan gestur jari tangan dengan MediaPipe untuk mengontrol robot mobil, jarak optimal untuk performa terbaik telah diidentifikasi. Pada jarak 30 cm, sistem menunjukkan kinerja yang sangat baik dengan akurasi 100% untuk semua jenis gestur, seperti maju, belok kanan, belok kiri, mundur, dan berhenti. Kecepatan pemrosesan atau frame per second (FPS) pada jarak ini juga cukup tinggi, berkisar antara 14.06 hingga 17.22, menunjukkan bahwa sistem dapat mendeteksi dan memproses gestur dengan cepat dan tepat.

Pada jarak 80 cm, sistem masih menunjukkan performa yang baik dengan akurasi yang hampir sempurna. Semua gestur dikenali dengan benar kecuali satu kesalahan pada gestur mundur, yang memberikan akurasi 80%. Kecepatan pemrosesan pada jarak ini tetap tinggi, dengan FPS berkisar antara 13.43 hingga 14.32. Meskipun ada sedikit penurunan akurasi pada satu gestur, jarak 80 cm masih dianggap cukup baik untuk pengenalan gestur.

Namun, pada jarak 130 cm, penurunan akurasi mulai terlihat. Beberapa gestur, terutama mundur dan menggenggam, menunjukkan akurasi yang lebih rendah (80% dan 60% masing-masing). FPS pada jarak ini bervariasi antara 13.23 hingga 13.45, yang masih dalam kisaran yang dapat diterima, tetapi penurunan akurasi membuat jarak ini kurang ideal.

Pada jarak 180 cm, penurunan akurasi menjadi lebih signifikan. Gestur maju hanya memiliki akurasi 20%, dan gestur menggenggam memiliki akurasi 60%. FPS pada jarak ini bervariasi antara 12.85 hingga 13.07, menunjukkan bahwa selain penurunan akurasi, kecepatan pemrosesan juga sedikit berkurang. Penurunan kinerja ini menunjukkan bahwa jarak 180 cm tidak ideal untuk pengenalan gestur yang andal dan cepat.

Secara keseluruhan, jarak 30 cm adalah jarak terbaik untuk pengenalan gestur jari dalam mengontrol robot mobil, karena memberikan akurasi yang sempurna dan kecepatan pemrosesan yang tinggi. Jarak 80 cm juga masih dapat diterima dengan performa yang cukup baik. Namun, jarak yang lebih jauh, seperti 130 cm dan 180 cm, menunjukkan penurunan signifikan dalam akurasi dan sedikit penurunan dalam kecepatan pemrosesan, membuat jarak-jarak ini kurang direkomendasikan untuk aplikasi praktis.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pada pengujian sistem kontrol Robot Mobil yang menggunakan MediaPipe pada saat pengiriman data melalui WiFi dan juga pendeteksian gestur jari tangan hingga pengujian jarak terbaik yang dilakukan pada penelitian ini, mendapat kesimpulan sebagai berikut:

1. Pada pendeteksian bentuk gestur jari tangan untuk menggerakkan robot mobil hasil yang didapatkan stabil tidak ada perubahan gestur jari saat dilakukan pengujian sehingga Robot Mobil berjalan sesuai bentuk gestur jari yang terdeteksi.
2. Hasil pengujian pengiriman data dalam 30 kali percobaan dengan gestur yang berbeda-beda menunjukkan hasil yang baik dengan akurasi keberhasilan 100% berhasil dikirim oleh program python dan diterima oleh program ESP32 bahwa sistem komunikasi untuk deteksi gestur jari tangan bekerja dengan efektif dan handal dalam uji coba. Keberhasilan ini menandakan bahwa sistem komunikasi dapat diandalkan untuk mendukung operasi kontrol robot berdasarkan gestur tangan yang terdeteksi.
3. Hasil percobaan mencari akurasi gestur jari tangan 2, 3, 4, 5 dan jari tangan menggengam mendapatkan akurasi 100% dalam 5 kali percobaan oleh 3 pengguna. Gestur jari 2 mendapat rata-rata FPS 14.34, gestur jari 3 mendapat rata-rata FPS 14.33, gestur jari 4 rata-rata FPS 14.35, gestur jari 5 mendapat rata-rata FPS 14.00, dan gestur jari menggengam mendapat rata-rata FPS 13.96. Dengan FPS rata-rata 14.12 sudah bisa dikatakan *real-time* Pada pendeteksian gestur jari tangan.
4. Pada pengujian perbandingan program mendapat kesimpulan bahwa metode kedua dengan pengecekan lebih direkomendasikan untuk digunakan dalam pengontrolan robot mobil karena mampu memberikan akurasi deteksi gestur yang sangat tinggi dan konsisten. Meskipun FPS sedikit lebih rendah, stabilitas dan keandalan metode ini jauh lebih baik. Untuk peningkatan lebih lanjut, perlu dievaluasi kemungkinan meningkatkan FPS metode kedua tanpa mengurangi

akurasi deteksi.

5. Hasil pengujian pada jarak 30 cm adalah jarak terbaik untuk pendeteksian gestur jari tangan mendapat akurasi pendeteksian 100% dan rata-rata FPS 14.57 dan pada jarak 180 pendeteksian menunjukkan penurunan akurasi gestur jari tangan 2 mendapatkan 20% dan gestur jari tangan menggengam mendapatkan 60%, rata-rata FPSnya juga mengalami penurunan sebesar 11.44.

5.2 Saran

Beberapa saran untuk pengembangan lebih lanjut dari sistem kontrol robot mobil yang menggunakan MediaPipe, berdasarkan temuan dan analisis yang telah dilakukan. Saran-saran ini bertujuan untuk meningkatkan kinerja, akurasi, dan fleksibilitas sistem agar dapat berfungsi lebih optimal dalam berbagai kondisi operasional:

1. Disarankan untuk menggunakan protokol komunikasi yang stabil dan handal, seperti MQTT atau *WebSocket*, untuk meningkatkan keandalan pengiriman data antara sistem kontrol dan robot. Protokol ini dirancang untuk komunikasi *real-time* yang efisien, memastikan bahwa data perintah gestur yang dikirim melalui WiFi tiba dengan cepat dan tanpa kehilangan, sehingga robot dapat merespon dengan akurat dan tepat waktu.
2. Mengoptimalkan konsumsi daya pada robot dan perangkat kontrol untuk memastikan operasi yang lebih lama dan efisien, terutama penting untuk aplikasi lapangan yang membutuhkan mobilitas tinggi.
3. Memberikan notifikasi pesan balik kepada program Python agar dapat melakukan tindakan atau memberikan respons yang sesuai berdasarkan hasil deteksi gestur. Dengan adanya notifikasi pesan balik, program Python dapat memproses data deteksi secara *real-time*, melakukan penyesuaian atau perubahan pada kontrol robot sesuai dengan gestur yang terdeteksi, dan memberikan umpan balik yang tepat kepada pengguna. Hal ini memastikan bahwa sistem kontrol robot lebih interaktif, responsif, dan akurat dalam menjalankan perintah yang diberikan melalui gestur tangan.

DAFTAR PUSTAKA

- Fahkrudin, Muhammad Aldi (2023). Sistem Deteksi Gestur Jari Tangan Menggunakan Mediapipe dan Faster-RCNN untuk Mengontrol Kecepatan Kipas Angin. Repository Universitas Dinamika.
- Edowai, Yumerius R.B (2023). Sistem Automatic Feature Selection Berbasis Deteksi Gestur Kedua Jari Tangan untuk Mengontrol Level Kecepatan Putaran 2 Kipas Angin Menggunakan Mediapipe. Repository Universitas Dinamika.
- Wakerkwa, Fredi (2023). Kontrol Level Kecepatan Putaran Kipas Angin Melalui Deteksi Bentuk Gestur Jari Tangan Berbasis IoT. Repository Universitas Dinamika
- Amperawan, A., Andika, D., Anisah, M., & ... (2022). Sistem Deteksi Posisi Dan Pengambilan Bola Pada Robot Sepak Bola. *Jurnal ...*, 7(1), 28–38. <https://jurnal.univpgri-palembang.ac.id/index.php/ampere/article/view/8507%0Ahttps://jurnal.univpgri-palembang.ac.id/index.php/ampere/article/download/8507/5957>
- Andri Nugraha Ramdhon, & Fadly Febriya. (2021). Penerapan Face Recognition Pada Sistem Presensi. *Journal of Applied Computer Science and Technology*, 2(1), 12–17. <https://doi.org/10.52158/jacost.v2i1.121>
- Arifin, A. Y. (2021). Pendeteksi Benda Dengan Metode Color Filtering Hsv Dan Blob Detection Pada Robot Vertical Take Off And Landing (Studi Kasus Kontes Robot Terbang Indonesia *Jurnal Ilmu Teknik*, 1(1), 1–7. <http://ilmuteknik.org/index.php/ilmuteknik/article/view/2%0Ahttp://ilmuteknik.org/index.php/ilmuteknik/article/download/2/2>
- Arlean, T. W. (2017). *Kinematika Balik Manipulator Robot Denso Inverse*

Kinematics Denso Robot Manipulator.

Astriani, Latifah, N., & Lutfi, I. (2022). Monitoring Gerakan Shalat Melalui Kamera Dengan Metode Pose Predict. *Jurnal Qua Teknika*, 12(2), 28–38. <https://doi.org/10.35457/quateknika.v12i2.2324>

Imran, A., & Rasul, M. (2020). Pengembangan Tempat Sampah Pintar Menggunakan Esp32. *Jurnal Media Elektrik*, 17(2), 2721–9100. <https://ojs.unm.ac.id/mediaelektrik/article/view/14193>

Khairudin, M., Refalda, R., Yatmono, S., Pramono, H. S., Triatmaja, A. K., & Shah, A. (2020). The mobile robot control in obstacle avoidance using fuzzy logic controller. *Indonesian Journal of Science and Technology*, 5(3), 334–351. <https://doi.org/10.17509/ijost.v5i3.24889>

Leksono, J. W. ;, Samudra, A. ;, Yannuansa, N., & Fauzi, A. (2020). Kendali Mobil Robot Menggunakan Isyarat Tangan. *Jurnal Electro Luceat*, 6(2).

Lubis, Z. (2018). Metode Baru Robot Pengantar Menu Makanan Menggunakan Android dengan Kendali PID Berbasis Mikrokontroler. *Journal of Electrical Technology*, 3(2), 105–106. www.kelasrobot.com

Maryamah, M., Pratama, M. A., Erfit, M. R., Farhani, N. M., & Hartono, I. A. (2023). Klasifikasi Abjad SIBI (Sistem Bahasa Isyarat Indonesia) menggunakan Mediapipe dengan Metode Deep Learning. *Prosiding Seminar Nasional Sains Data*, 3(1), 134–141.

Maulana, D., Raka Agung, I. G. A. P., & Elba Duta Nugraha, I. P. (2022). Sistem Monitor Budi Daya Sarang Burung Walet Berbasis Esp32-Cam Dilengkapi Aplikasi Telegram. *Jurnal SPEKTRUM*, 9(1), 143. <https://doi.org/10.24843/spektrum.2022.v09.i01.p17>

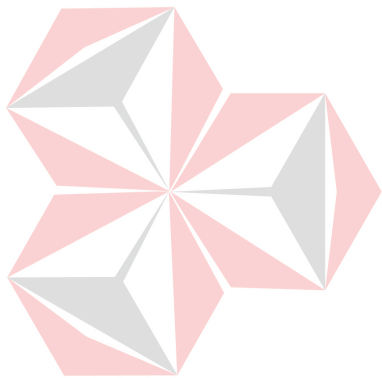
- Nabilah Humairah. (2017). *Penjelasan tentang Layanan pada Protokol TCP dan UDP-Kapan Menggunakan TCP dan UDP?* 1–6.
http://edocs.ilkom.unsri.ac.id/1439/1/NabilahHumairah_%2809011281520109%29.pdf
- nufriani. (2019). Jurnal Ilmiah INTECH: Information Technology Journal of UMUS. *Umus*, 1(02), 1–12.
- Prananta, G. B., Azzikri, H. A., & Rozikin, C. (2023). Jurnal METHODIKA. *Deteksi Dan Pengenalan Gestur Tangan Secara Real-Time menggunakan Jaringan saraf Tiruan Konvolusional*, 9(2), 30–34.
<https://ejurnal.methodist.ac.id/index.php/methodika/article/view/1911/1578>
- Pratama, M. R., Pratikno, H., Triwidyastuti, Y., & Musayyanah. (2023). *Pengenalan Gestur Jari Tangan Sebagai Media Pembelajaran Berhitung Bagi PAUD Berbasis Visi Komputer Dan Deep Learning. Journal of Computer Electronic and Telecommunication*, 4(1).
<https://doi.org/10.52435/complete.v4i1.355>
- Rama Akbar. (2020). *Sistem Kunci Kendaraan Bermotor Menggunakan Radio Frequency Identification (RFID) dan SIM Berbasis NODEMCU ESP32*. 1–74.
- Surahman, A., Aditama, B., Bakri, M., & Rasna, R. (2021). Sistem Pakan Ayam Otomatis Berbasis Internet Of Things. *Jurnal Teknologi Dan Sistem Tertanam*, 2(1), 13. <https://doi.org/10.33365/jtst.v2i1.1025>
- Talakua, E. L., Utama, Y. A. K., & Makruf, A. (2020). Sistem Kendali Mobile Robot. *Seminar Nasional Ilmu Terapan IV 2020*, 1–7.
- Waruru, S. J. B. (2021). Robot Mobil Pencari Target Dalam Robot Mobil Pencari Target Dalam. *Comasie*, 5.

Wibowo, A. (2020). Prototipe Robot Manipulator Sendi Lengan (Joint - Arm)
Berbasis Arduino Uno Pada Sistem Pemilah Barang. In *Universitas Nasional*.

Yuda, M., & Airlangga, P. (2023). *Perencanaan Sistem Gerak Quadcopter Sebagai
Alat Pemantau Kawasan Lingkungan Bencana Untuk Field Triage Korban
Bencana*.

Mediapipe. (2019). Live Machine Learning Anywhere. Von Mediapipe Developer.

<https://mediapipe.dev/abgerufen>.



UNIVERSITAS
Dinamika